Noname manuscript No.
(will be inserted by the editor)

High-Order Accurate Adaptive Kernel Compression Time-Stepping Schemes for Fractional Differential Equations

Daniel Baffet · Jan S. Hesthaven

February 9, 2017

Abstract High-order adaptive methods for fractional differential equations are proposed. The methods rely on a kernel compression scheme for the approximation and localization of the history term. To avoid complications typical to multistep methods, we focus our study on 1-step methods and approximate the local part of the fractional integral by integral deferred correction to enable high order accuracy. We study the local truncation error of integral deferred correction schemes for Volterra equations and present numerical results obtained with both implicit and the explicit methods applied to different problems.

Keywords fractional differential equations · Volterra equations · kernel compression · high-order numerical methods · integral deferred correction · local schemes.

1 Introduction

The nonlocal nature of the fractional integral and the singularity of its kernel make the numerical treatment of fractional differential equations (FDEs) considerably more difficult than that of standard differential equations. In particular, the design of high-order and adaptive schemes has proved challenging. The most commonly used methods for FDEs are low-order. For example, the L1 scheme has been widely used for the approximation of the time fractional diffusion equation, and has been extended to non-uniform meshes [1,2]. High-order schemes have also been proposed. Convolution quadratures have been proposed in the 1980's [3] and later modified [4] to reduce computational and memory costs. A discontinuous Galerkin time-stepping method has also been proposed for Volterra equations [5,6]. In this paper we propose high-order adaptive methods that are based on an efficient kernel compression [7] approximation of the history term.

The history term of the fractional integral $\mathcal{I}^{\alpha} f(t+\delta)$ of a function f at $t+\delta$ has the form of a Laplace convolution

$$\mathcal{I}_{\delta} f(t) = w_{\delta} * f(t) = \int_0^t w_{\delta}(t-s)\, f(s)\ \mathrm{d}s\ , \tag{1.1}$$

where $\delta > 0$ is the distance between the current time t and the reconstruction time $t+\delta$, $w_{\delta}(t) = w(t+\delta)$, and $w(t) = t^{-1+\alpha}/\Gamma(\alpha)$ is the kernel of the fractional integral. The kernel compression scheme [7] prescribes an approximation to (1.1) given by a linear combination of solutions $\psi_1, \ldots, \psi_J$ to initial value problems for standard ordinary differential equations (ODEs)

$$\psi_j' = \lambda_j \psi_j + f \qquad\qquad \psi(0) = 0\ , \tag{1.2}$$

D. Baffet, E-mail: daniel.baffet@epfl.ch · J. S. Hesthaven, E-mail: jan.hesthaven@epfl.ch
SB-MATHICSE-MCSS, École Polytechnique Fédérale de Lausanne (EPFL), 1015 Lausanne, Switzerland.

where $\lambda_j \in \mathbb{C}$. In the following we refer to $\psi_1, \dots, \psi_J$ as the auxiliary variables of the scheme. This amounts to the approximation, in some sense, of the kernel w by a linear combination of exponentials

$$S(\Lambda, \sigma;\, t) = \sum_{j=1}^{J} \sigma_j \mathrm{e}^{\lambda_j t} \tag{1.3}$$

at some positive distance δ from its singularity.

As the auxiliary variables of the scheme are of the same dimension as f, it is important to reduce their number J as much as possible while maintaing a prescribed error tolerance. The main result of [7] is the following *a priori* estimate on J: for $T > 0$, $\delta > 0$, and an error tolerance $\varepsilon > 0$, the approximation operator $\mathcal{I}_{\delta,r}$ prescribed by the scheme, given by $\mathcal{I}_{\delta,r} f = S * f$, with $J = pm$, where

$$p = O\Big(\log \delta^{-1} T + \log\log \frac{1-\alpha}{\varepsilon}\Big) \ , \qquad m = O\Big(\log \varepsilon^{-1}\Big) \ , \tag{1.4}$$

satisfies

$$\|\mathcal{I}_\delta f - \mathcal{I}_{\delta,r} f\|_{L^2(0,T)} \le \varepsilon \, \|\mathcal{I}_\delta f\|_{L^2(0,T)} \qquad \forall\, f \in L^2(0,T) \ . \tag{1.5}$$

See Theorem 3.16 for a more accurate statement of this result. In practice δ is proportional to the step size h. Employing the kernel compression scheme thus yields schemes for which the effort of evaluating the history term is constant, provided h is constant. The cost at each step is that of $O(dJ)$ operations, where d is the dimension of the problem, and $J = pm$, with p and m satisfying (1.4), where $\delta = h$ the time step size. The idea to reduce the costs of evaluating the history term of the fractional integral [8], or more generally [9,4,10] a convolution $K * f$, by approximating its kernel by a linear combination of exponentials has been explored in the past. See also [11] for approximation of some functions by sums of exponentials. As the kernel compression scheme is not the main focus of this work, we refer the reader to [7] for a qualitative discussion of the different methods. With relevance to the context of this paper, we mention the following. In [10], an adaptive step-size scheme is proposed. The method extends the algorithm proposed in [9] to variable step-size schemes for Volterra equations. In [12] a fast collocation method is proposed.

In addition to the efficient account of the history term, the kernel compression scheme proposed in [7] has other advantages. One is that the theoretical estimates are uniform in $\alpha \in (0, 1)$, and $f \in L^2(0,T)$. Another is the following. As the convolution approximation relies on solutions to standard ODEs, it requires only local information to advance. This is property is shared by the other kernel compression schemes, however the structure of the approximation proposed in [7] has an additional advantage. For a given error tolerance, and $0 < \delta_1 < \delta_2$, the set of poles Λ_2 prescribed for δ_2 is *a subset* of the set of poles Λ_1 prescribed for δ_1. Moreover, the poles that are in Λ_1 and not in Λ_2 lie on a "small" number of circles in the left half of the complex plane $\operatorname{Re}\lambda < 0$. This property is of particular interest when δ is not constant, as is the case when the step-size h is not constant, since it allows us to retain auxiliary variables associated with most of the poles and only add or remove those associated with the "last" circles. Changing δ also requires computing the reconstruction weights $\sigma_1, \dots, \sigma_J$. This is in addition to the cost of $O(dJ)$ operations of evaluating the history term. This added cost of $O(J)$ operations, however, is not great, as J is small compared to the number of time-steps, and the weights are independent of the problem dimension d.

To obtain the local information, the time-stepping schemes also require a method for approximating Volterra equations on short intervals. This may be done in several ways. To avoid complications typical to multistep methods, we focus our study on 1-step methods. Runge-Kutta (RK) methods have been proposed [13] for Volterra equations. However, as the number of order conditions grows to infinity as α tends to zero, their usefulness when α can be any number in $(0, 1)$ is limited. For the approximation of the local part of the fractional integral, we employ the integral deferred correction (IDC) method [14]. In the literature on standard ODEs, the method is called spectral deferred correction (SDC) if Gaussian quadrature nodes are used. We use the Gauss-Lobatto quadrature nodes, due to their good numerical properties. However, since the present application involves a non-trivial weight function, we refer to the method as IDC and reserve the name SDC for methods using the "appropriate" quadrature nodes.

The SDC method, introduced in [14] for standard ODEs, relies on an iterative procedure to recover a prescribed order. In each iteration an inner (low order) scheme is used to compute corrections to the approximation computed previously at the quadrature nodes. Different variations of the IDC approach have been studied for standard ODEs; see e.g. [15,16] and the references therein. To reduce the number of correction iterations required to obtain a prescribed order, RK integrators have been proposed as inner schemes of methods on equidistant quadrature nodes [16]. Unfortunately, this accelerated increase of order does not extend to non-equidistant quadrature nodes [17] when explicit RK integrators are used.

In this work we propose an extension of the IDC approach to Volterra equations. We prove in Theorem 5.1 that the IDC method increases the local order by α each correction iteration [18]. The theorem sets a lower bound on the increase of the order, and leaves room for improved estimates for specific schemes. We test methods that employ the composite left endpoint and trapezoidal integration rules (see §4) as inner schemes – of local orders $1+\alpha$ and $2+\alpha$, respectively. Our numerical results for the left endpoint rule are consistent with the conclusion of Theorem 5.1. The results for the trapezoidal rule, however show an accelerated increase of the order. While for schemes employing the left endpoint rule, we measure an order increase of α per correction iteration, for schemes employing the trapezoidal rule, we measure an order increase of $1+\alpha$. Since we do not have a complete analysis of the methods, we can not determine if the order we measure is indeed the asymptotic order, however, the improvement in accuracy is considerable. This issue is of particular importance since an increase of order α per correction iteration, implies that the number of correction iterations required to obtain a prescribed order tends to infinity as α tends to zero. In contrast, if one can guarantee an increase of the order by $1+\alpha$, a bounded number of correction iterations will suffice to achieve the prescribed order for all $\alpha \in (0,1)$.

The paper is structured as follows. In §2 we provide an overview of the proposed methods. The kernel compression method is described in §3, where we also state some relevant results. In §4 we discuss two standard low order approximations to the fractional integral which are used as inner schemes in the IDC algorithm. In §5 we discuss the IDC method, propose an extension of the method to FDEs, and prove Theorem 5.1 which estimates the local truncation error. We discuss the details of the adaptive error control in §6, and provide further details on the implementation in §7. Some numerical results are presented in §8, and we conclude with some remarks in §9.

2 Overview

In this section we outline the main idea and structure of the methods. For simplicity, we do not discuss the topic of adaptive error control. This topic is discussed in §6. Thus, we only present the basic schemes. To keep the discussion as clear and concise as possible we do not elaborate on the different components of the scheme or state precisely what results are available. Instead we refer the reader to different sections of this paper where more details are provided. We emphasize, however, that at this time, there are still open questions regarding the methods.

2.1 Preliminaries

For $\alpha > 0$, let

$$\mathcal{I}^\alpha f(t) = \frac{1}{\Gamma(\alpha)} \int_0^t (t-s)^{-1+\alpha} f(s) \, \mathrm{d}s \tag{2.1}$$

be the fractional integral of f. Consider $\mathcal{I}^\alpha f(t+\delta)$, for some $t \geq 0$ and $\delta > 0$. We split the fractional integral into a local term

$$\int_0^\delta w(\delta - s)\, f(t+s) \, \mathrm{d}s \tag{2.2}$$

and a history term

$$\mathcal{I}_\delta f(t) = \int_0^t w_\delta(t-s)\, f(s) \, \mathrm{d}s \ , \tag{2.3}$$

where

$$w_\delta(t) = w(t+\delta) \qquad w(t) = \frac{t^{-1+\alpha}}{\Gamma(\alpha)} \ . \tag{2.4}$$

2.2 Basic schemes

Consider the initial value problem

$$D^\alpha u = f(t,u) \qquad u(0) = u_0 \tag{2.5}$$

in $(0,T)$, where $\alpha \in (0,1)$, $f : [0,T] \times \Pi \to \mathbb{R}^d$, $T > 0$, $\Pi \subset \mathbb{R}^d$ open, and D^α is the Caputo α-derivative

$$D^\alpha u = \mathcal{I}^{1-\alpha} u' \ . \tag{2.6}$$

By applying $\mathcal{I}^\alpha$ to (2.5), we get

$$u = u_0 + \mathcal{I}^\alpha (f \circ u) \ , \tag{2.7}$$

where $f \circ u(t) = f(t, u(t))$. In fact (2.5) is equivalent to (2.7), provided f is continuous [19]. If, in addition, f is Lipschitz in u, then (2.7) has a unique solution in a neighborhood of $t = 0$. In the following we assume (2.7) has a unique solution in $[0,T]$.

A standard approach for the derivation of numerical methods for (2.7), and thus for (2.5), is as follows. Fix $t \geq 0$ and $h > 0$, and let $\delta \in (0,h]$. Owing to (2.7), we have

$$u(t+\delta) = \int_0^\delta w(\delta - s)\, f(t+s, u(t+s))\, \mathrm{d}s + H(t,\delta) \tag{2.8a}$$

$$H(t,\delta) = u_0 + \mathcal{I}_\delta (f \circ u)(t) \ . \tag{2.8b}$$

Observing that equation (2.8a) has the form

$$U(\delta) = \mathcal{I}^\alpha (F \circ U)(\delta) + H(\delta) \ , \tag{2.9}$$

where $U(\delta) = u(t+\delta)$, $F(\delta, u) = f(t+\delta, u)$ and $H(\delta)$ substitutes for $H(t,\delta)$, we find that two ingredients are required for the time-stepping scheme. The first is a scheme for the approximation of (2.9) in $(0,h]$, provided H is known.

Low-order methods approximate (2.9) by single step of a standard 1-step scheme. Two such schemes are discussed in §4. To obtain high-order we use the IDC method. We postpone the description of the IDC method for Volterra equation to §5. The method requires a standard (low-order) method – the inner scheme – and a finite set of quadrature points in $[0,h]$. We use the Gauss-Lobatto nodes.

For the approximation of (2.9), it is necessary to have $H(t,\cdot)$ at the quadrature points in $(0,h]$. The approximation to $H(t,\cdot)$ is obtained by the second component of the methods: the kernel compression scheme [7] described in §3. This scheme offers an approximation

$$\mathcal{I}_\delta (f \circ u)(t) \approx \Psi(t)\, \sigma(\delta) \tag{2.10}$$

to the history term $\mathcal{I}_\delta (f \circ u)(t)$. Here, σ is a δ dependent complex vector of weights, and $\Psi : [0,T] \to \mathbb{C}^{d \times J}$, the matrix of auxiliary variables, is defined as the solution to an initial value problem

$$\Psi' = \Psi \Lambda + (f \circ u)\, \mathbf{1}_J \qquad \Psi(0) = 0 \tag{2.11}$$

where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_J) \in \mathbb{C}^{J \times J}$ and $\mathbf{1}_J = (1, \ldots, 1) \in \mathbb{R}^J$.

The number J of columns of Ψ is the effective number of auxiliary variables, and $\Psi(t)\,\sigma(\delta)$ is the reconstruction of the history term at $t + \delta$, or the reconstruction at distance of δ. We exploit a symmetry in the distribution of the poles $\lambda_1, \ldots, \lambda_J$, prescribed by the present scheme, around the real line to reduce the number of auxiliary variables in the computation to about half. In the sequel, we denote the reduced number of auxiliary variables by P and refer to it as the number of auxiliary variables. Thus, J – the effective number of auxiliary variables – determines the accuracy of the approximation and P determines the computational effort.

The scheme also requires an approximation to (2.11). For this we employ an A-stable diagonally implicit RK (DIRK) method. Another option is to integrate (2.11) exactly, where $f \circ u$ is replaced by the polynomial interpolating f at the numerical solution at the quadrature nodes. We have not explored this approach, however it is more inline with the discretization methods proposed in [9,4,10]. The reason for requiring the RK method to be A-stable is that some of nodes $\lambda_1, \dots, \lambda_J$ have large negative real parts which makes (2.11) stiff. Equation (2.11), however, is simple to approximate by DIRK methods, provided $f \circ u$ is known. To apply the DIRK method, we use the data computed by the IDC at the Gauss-Lobatto nodes to approximate $f \circ u$ at the RK nodes and treat it as a given function, independent of Ψ. Since Λ is diagonal, a DIRK step does not require us to solve a large algebraic equation and is cheap to perform.

To conclude this section, we provide in Algorithm 1 a short description of the procedure to be performed in a single step of the proposed methods. Let $c_1, \dots, c_\mu$ the RK nodes, $\delta_0, \dots, \delta_{N_{\mathsf{loc}}}$ the Gauss-Lobatto nodes of the interval $[0, h]$, v^n the numerical approximation to $u(t_n)$, V^j the approximation to $U(\delta_j) = u(t_n + \delta_j)$, and Φ^n the approximation to $\Psi(t_n)$.

Algorithm 1 Basic high-order 1-step scheme

```
Input: $v^n$, $\Phi^n$, $t_n$, $h$
Output: $v^{n+1}$, $\Phi^{n+1}$
1: For each $j = 1, \dots, N_{\mathsf{loc}}$, compute approximations $H^j \approx H(t_n, \delta_j)$, where
       $$H^j = u_0 + \Phi^n \sigma(\delta_j) \ .$$
2: Apply the IDC method to (2.9) to compute an approximation $V^j \approx U(\delta^j)$, $j = 1, \dots, N_{\mathsf{loc}}$ at the Gauss-
   Lobatto nodes of the interval $[0, h]$. In this step, use $H^j$ to substitute for $H(\delta_j)$.
3: Evaluate the $N_{\mathsf{loc}}$-degree polynomial interpolating the data
       $$(\delta_j, F(\delta_j, V^j)) \qquad j = 0, \dots, N_{\mathsf{loc}}$$
   at $c_k h$ to obtain an approximation to $f \circ u(t_n + c_k h)$.
4: Advance $\Phi$ by an A-stable DIRK method applied to (2.11), where the values computed in step 3 substitute
   for the values $f \circ u$ at $t_n + c_k h$.
5: Define $v^{n+1} = V^{N_{\mathsf{loc}}}$.
```

3 Kernel compression

The proposed methods employ a kernel compression scheme [7] for the approximation of the history term. In this section we describe that scheme and state the available error estimates. The scheme is an adaptation of the method proposed in [20] to the approximation of the fractional integral. We denote by either $\widehat{f}$ or $\mathcal{L}[f]$ the Laplace transform

$$\widehat{f}(\lambda) = \int_0^\infty \mathrm{e}^{-\lambda t} f(t) \, \mathrm{d}t \tag{3.1}$$

of a function f. Let $\alpha \in (0, 1)$, and $\delta > 0$. For the presentation, let us consider $f : (0, \infty) \to \mathbb{R}^d$. The history term (2.3) of $\mathcal{I}^\alpha f(t + \delta)$, the fractional integral of f at $t+\delta$, has the form of a Laplace convolution (2.3). Thus, its Laplace transform is given by

$$\mathcal{L}[\mathcal{I}_\delta f](\lambda) = \widehat{w}_\delta(\lambda) \, \widehat{f}(\lambda) \ . \tag{3.2}$$

As our goal is to approximate the convolution (2.3), we seek a multipole approximation

$$r(z) = \sum_{j=1}^{J} \frac{\sigma}{z - \lambda_j} \tag{3.3}$$

to $\widehat{w}_\delta$ that satisfies a uniform estimate of the relative error

$$|\widehat{w}_\delta(\lambda) - r(\lambda)\,| \le \varepsilon|\widehat{w}_\delta(\lambda)\,| \qquad \operatorname{Re}\lambda \ge \eta\ . \tag{3.4}$$

The reason for requiring (3.4) is that it yields the following estimate of the relative L^2-error

$$\|\mathcal{I}_\delta f - \mathcal{I}_{\delta,r} f\|_{L^2(0,T)} \le \varepsilon \mathrm{e}^{\eta T}\|\mathcal{I}_\delta f\|_{L^2(0,T)} \qquad \forall f \in L^2(0,T)\ , \tag{3.5}$$

where $\mathcal{I}_{\delta,r} f$ is defined as the inverse Laplace transform of $r\widehat{f}$. Explicitly, $\mathcal{I}_{\delta,r} f$ is given by $\mathcal{I}_{\delta,r} f = S * f$, where

$$S(t) = \sum_{j=1}^{J} \sigma_j \mathrm{e}^{\lambda_j t}\ , \tag{3.6}$$

is the inverse Laplace transform of r.

Given an approximation (3.3) to $\widehat{w}_\delta$, $\mathcal{I}_{\delta,r} f$ may also be expressed in terms of the solution to an initial value problem for a standard ODE system. Suppose r is a multipole approximation (3.3) to $\widehat{w}_\delta$. For each $j = 1, \dots, J$, define

$$\widehat{\psi}_j = (\lambda - \lambda_j)^{-1}\,\widehat{f}\ . \tag{3.7}$$

Inverting the Laplace transform we recover

$$\psi_j(t) = \int_0^t \mathrm{e}^{\lambda_j(t-s)} f(s)\ \mathrm{d}s\ . \tag{3.8}$$

To simplify the notation we organize $\psi_1, \dots, \psi_J$ as the columns of a matrix $\Psi = (\psi_j)$. Thus the approximation $\mathcal{I}_{\delta,r} f$ of $\mathcal{I}_\delta f$ associated with r is

$$\mathcal{I}_{\delta,r} f = \sum_{j=1}^{J} \sigma_j \psi_j = \Psi\sigma\ , \tag{3.9}$$

where $\sigma = (\sigma_1, \dots, \sigma_J)^T$. Observing that each ψ_j is the solution to the ODE $\psi_j' = \lambda_j\psi + f$ satisfying $\psi_j(0) = 0$, we recover

$$\Psi' = \Psi\Lambda + f\mathbf{1}_J \qquad \Psi(0) = 0\ , \tag{3.10}$$

where $\mathbf{1}_J = (1, \dots, 1) \in \mathbb{R}^J$, and $\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_J)$. Below we specify the multipole approximation r to $\widehat{w}_\delta$ and state the error estimate for the approximation $\mathcal{I}_{\delta,r} f$ of the history term $\mathcal{I}_\delta f$.

For a given $\eta > 0$, we require a multipole r that approximates $\widehat{w}_\delta$ uniformly for all $\operatorname{Re} z \ge \eta$. The present approximation relies on the following integral representation of the Laplace transform of w_δ

$$\widehat{w}_\delta(z) = \frac{\sin(\pi\alpha)}{\pi}\int_0^\infty \frac{x^{-\alpha}\mathrm{e}^{-\delta x}}{z + x}\,\mathrm{d}x\ , \tag{3.11}$$

and proceeds as follows. We truncate the integral in (3.11) at a finite x_p, write the integral over the truncated interval $(0, x_p)$ as an integral over a contour $\mathcal{C}$ in the complex plane, and finally approximate the contour integral by a quadrature. For $\alpha \in (0,1)$, $\eta > 0$, and $p, m \in \mathbb{N}$, the procedure yields the approximation

$$r(\lambda) = \sum_{k=1}^{p}\sum_{j=0}^{m-1} \frac{\sigma_{kj}}{\lambda - \lambda_{kj}} \qquad \lambda_{kj} = c_k + r_k\omega^j\ , \tag{3.12a}$$

where $\omega = \mathrm{e}^{2\pi i/m}$, for each $k = 1, \dots, p$ and $j = 0, \dots, m-1$,

$$\sigma_{kj} = \frac{1}{m}\sum_{l=0}^{m-1} \omega^{-jl} Q_{kl} \tag{3.12b}$$

$$Q_{kl} = \frac{\sin(\pi\alpha)}{\pi}\int_{x_{k-1}}^{x_k} x^{-\alpha}\mathrm{e}^{-\delta x}\left(\frac{-x + |c_k|}{r_k}\right)^l \mathrm{d}x \tag{3.12c}$$

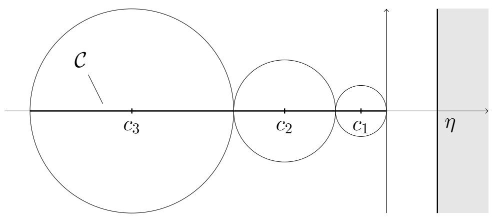

Fig. 3.1 The setup in the complex plane: the half plane $\operatorname{Re} z \geq \eta$ and the contour C covered by disks of radii r_k centered at c_k.

$$r_k = \eta\, 2^{k-2} \qquad c_k = -\eta \left(3 \cdot 2^{k-2} - 1\right) \tag{3.12d}$$

and

$$x_k(\eta) = \eta(2^k - 1) \qquad\qquad k = 0, \ldots, p\ . \tag{3.12e}$$

Thus, for each $k = 0, \ldots, p$, the poles $\lambda_{k0}, \ldots, \lambda_{k,m-1}$ lie on a circle of radius r_k centered at c_k. See Figure 3.1 for an illustration of the setup in the complex plane.

We have the following results [7], estimating the error of the approximation. The first result estimates the error in the approximation of the Laplace transform $\widehat{w}_\delta$ of w_δ.

Theorem 3.1 *Let $\alpha \in (0, 1)$, $\eta > 0$ and $\delta > 0$. Then, r given by (3.12) satisfies*

$$|\widehat{w}_\delta(\lambda) - r(\lambda)| \leq C_{mp}(\alpha, \delta\eta)\ |\widehat{w}_\delta(\lambda)\,| \tag{3.13}$$

for all $\operatorname{Re} \lambda \geq \eta$, *where*

$$C_{mp}(\alpha, \eta) = C_a A_m + C_b B_p(\alpha, \eta) \tag{3.14a}$$

with

$$A_m = 3^{-m}\ , \qquad B_p(\alpha, \eta) = \frac{\Gamma(1 - \alpha, \eta(2^p - 1))}{\Gamma(1 - \alpha)}\ . \tag{3.14b}$$

The constants C_a and C_b are positive and independent of α, η, δ, p and m.

Note that for a given error tolerance, the number $J = pm$ of terms in (3.12) is bounded uniformly for $\alpha \in (0, 1)$. Also note the rapid convergence of A_m and B_p: A_m decays exponentially, and B_p satisfies an estimate

$$B_p(\alpha, \eta) \leq C\,(1 - \alpha)\, x_p^{-\alpha}(\eta)\ \mathrm{e}^{-x_p(\eta)} \tag{3.15}$$

which captures its asymptotic behavior at the limit $p \to \infty$.

Theorem 3.2 provides an estimate of the error in the approximation of the convolution.

Theorem 3.2 *Let $\alpha \in (0, 1)$, $T > 0$, $\delta > 0$, and $m, p \in \mathbb{N}$. Suppose r is given by (3.12), and $\mathcal{I}_{\delta,r}$ is the approximation operator associated with r, i.e., $\mathcal{I}_{\delta,r} f = S * f$, where $S = \mathcal{L}^{-1}[r]$. Then, the estimate*

$$\|\mathcal{I}_\delta f - \mathcal{I}_{\delta,r} f\|_{L^2(0,T)} \leq \mathrm{e}^{\eta T} C_{mp}(\alpha, \delta\eta)\ \|\mathcal{I}_\delta f\|_{L^2(0,T)} \tag{3.16}$$

holds for every $f \in L^2(0, T)$.

As a corollary to Theorem 3.2, we obtain an estimate of the relative pointwise error of the kernel approximation. An estimate of the type obtained for the methods in [11,21].

Corollary 3.1 *Let $\alpha \in (0, 1)$, $T > 0$, $\delta > 0$, and $m, p \in \mathbb{N}$. Suppose r is given by (3.12) and $S = \mathcal{L}^{-1}[r]$, then the estimate*

$$|w_\delta(t) - S(t)\,| \leq \mathrm{e}^{\eta T} C_{mp}(\alpha, \delta\eta)\ |w_\delta(t)\,| \tag{3.17}$$

holds for all $t \in [0, T]$.

For computations the number of auxiliary variables may be reduced further. We exploit that the poles are distributed symmetrically around the real line and thus remove poles with negative imaginary parts and real duplicate poles to recover

$$\sum_{k=1}^{p}\sum_{j=0}^{m-1}\sigma_{kj}\psi_{kj} = \mathrm{Re}\sum_{j=1}^{P}\theta_j\psi_j \ . \tag{3.18}$$

The reader is referred to [7] for more details on the implementation and proofs.

Due to the exponential term on the right hand side of (3.16) and (3.17), we set $\eta = 1/T$. In the time-stepping schemes, δ is set by a requirement of the discrete scheme, and is hence proportional to the time-step size h. Thus, to control the kernel compression error, we may choose $p, m \in \mathbb{N}$ large, so that A_m and B_p are sufficiently small. The number of circles p is chosen so that $B_p(\alpha, \delta\eta)$ is sufficiently small, and therefore depends on δ. In contrast, as A_m is independent of δ, the number m of poles on each circle depends only on the error tolerance, and thus may be chosen a priori.

That property of the above kernel compression scheme makes it convenient for use within step-size adaptive time stepping schemes. For a given error tolerance, and $0 < \delta_1 < \delta_2$, the set of poles Λ_2 prescribed for δ_2 is *a subset* of the set of poles Λ_1 prescribed for δ_1. Moreover, the poles that are in Λ_1 and not in Λ_2 lie on a "small" number of circles in the left half of the complex plane. This is because the center and radius of each circle is independent of δ, and thus poles λ_{kj} are also independent of δ. This property is of particular interest when the step-size h is not determined a priori, as it allows us to retain most of the poles and only add or remove those associated with the "last" circles.

Changing the step-size h also requires replacing all the weights σ_{kj}. This requires $O(J)$ operations (independent of the dimension d of the problem), which is relatively cheap, however creates an overhead which should be compared to the cost of maintaing a constant step-size.

4 Inner schemes

Let us now discuss the two methods used as inner schemes for the IDC method. For that purpose we employ low-order standard approximations of the fractional integral. Following the procedure described in §2, at $t_n \geq 0$ we consider (2.9) in $(0, h]$, where $H(\delta)$, understood as $H(t, \delta)$, is given. The analysis of the accuracy of the schemes discussed in this section relies on the regularity of F and U in the interval of interest $(0, h]$. As the typical singularity is at the initial time, assuming F and U to be smooth in $\delta \geq 0$, for $t = t_n > 0$ is not overly restrictive. To simplify the notation, we present the methods for a Volterra equation (2.9) in $(0, h)$, disregarding how it is obtained, and thus suppose H and $U(0)$ are given. Introduce the grid $\mathcal{P} = (\delta_0, \ldots, \delta_{N_{\mathsf{loc}}})$ to $[0, h]$, where $0 = \delta_0 < \cdots < \delta_{N_{\mathsf{loc}}} \leq h$, and let V be the numerical approximation to U. Suppose we have the numerical solution V^k at δ_k, for each $k = 0, \ldots, j-1$, and let $\mathcal{P}_j = (\delta_0, \ldots, \delta_j)$. We compute V^j by solving

$$V^j = \sum_{k=0}^{j}\omega_{jk}\, F\big(\delta_k, V^k\big) + H^j \tag{4.1}$$

for V^j, where $H^j = H(\delta_j)$, and $\omega_{jk} = \omega_{jk}(\mathcal{P}_j)$ are the weights characterizing the scheme. For the two schemes of interest to us, the weights ω_{jk} are chosen so the equality

$$\sum_{k=0}^{j}\omega_{jk}\, g^k = \mathcal{I}^{\alpha} g_{\mathcal{P}_j}(\delta_j) \tag{4.2a}$$

holds for every grid function $g = (g^0, \ldots, g^j)$ on $\mathcal{P}_j = (\delta_0, \ldots, \delta_j)$, where

$$g_{\mathcal{P}_j}(\delta) = \sum_{k=0}^{j-1}\chi_k(\delta)\, g_k\left(\frac{\delta - \delta_k}{\tau_k}\right) \ , \tag{4.2b}$$

is either piecewise constant or piecewise linear. Here

$$\tau_k = \delta_{k+1} - \delta_k \qquad k = 0, \dots, j-1 \ , \tag{4.2c}$$

for each k, χ_k is the indicator function of the interval (δ_{k-1}, δ_k), and the functions $g_1, \dots, g_n$ are given by either

$$g_k(s) = g^k \qquad k = 0, \dots, j-1 \tag{4.3}$$

or

$$g_k(s) = (1-s)g^k + sg^{k+1} \qquad k = 0, \dots, j-1 \ . \tag{4.4}$$

For simplicity, we refer to (4.2) with (4.3) and (4.4) as the composite left endpoint and trapezoidal integration rules, respectively, and similarly to the corresponding schemes (4.1). Next we derive the weights ω_{jk} for the two schemes. The weights ω_{jk} are obtained by substituting (4.2b) into (4.2a) and comparing coefficients. Applying the α-integral to (4.2a) and evaluating at δ_j, we obtain

$$\begin{aligned} \mathcal{I}^\alpha g_{\mathcal{P}_j}(\delta_j) &= \sum_{k=0}^{j-1} \int_{\delta_k}^{\delta_{k+1}} w(\delta_j - s)\, g_k\left(\frac{s-\delta_k}{\tau_k}\right) \mathrm{d}s \\ &= \sum_{k=0}^{j-1} \frac{\tau_k^\alpha}{\Gamma(\alpha)} \int_0^1 \left(\xi_{jk} - s\right)^{-1+\alpha} g_k(s)\ \mathrm{d}s \ . \end{aligned} \tag{4.5}$$

where

$$\xi_{jk} = \frac{\delta_j - \delta_k}{\tau_k} \ . \tag{4.6}$$

To obtain the wights ω_{jk} for the composite left endpoint rule, we substitute (4.3) into (4.5), and compare the coefficients of g^k. Thus, we reover

$$\omega_{jk}(\mathcal{P}_j) = \begin{cases} \tau_k^\alpha\, a(\xi_{jk}) & k = 0, \dots, j-1 \\ 0 & k = j \end{cases} \ , \tag{4.7a}$$

where

$$a(\xi) = \frac{1}{\Gamma(\alpha)} \int_0^1 (\xi - s)^{-1+\alpha}\ \mathrm{d}s = \frac{\xi^\alpha - (\xi-1)^\alpha}{\Gamma(1+\alpha)} \qquad \xi \ge 1 \ . \tag{4.7b}$$

Similarly, we obtain the weights ω_{jk} for the composite trapezoidal rule by substituting (4.4) into (4.5). We obtain an expression of the form

$$\begin{aligned} \mathcal{I}^\alpha g_{\mathcal{P}_j}(\delta_j) &= \sum_{k=0}^{j-1} \tau_k^\alpha \left(a_{jk}^L g^k + a_{jk}^R g^{k+1}\right) \\ &= \sum_{k=0}^{j-1} \tau_k^\alpha a_{jk}^L g^k + \sum_{k=1}^{j} \tau_{k-1}^\alpha a_{j,k-1}^R g^k \ , \end{aligned} \tag{4.8}$$

and thus find

$$\omega_{jk}(\mathcal{P}_j) = \begin{cases} \tau_0^\alpha a_{j0}^L & k = 0 \\ \tau_k^\alpha a_{jk}^L + h_{k-1}^\alpha a_{j,k-1}^R & k = 1, \dots, j-1 \\ \tau_{j-1}^\alpha a_{j,j-1}^R & k = j \end{cases} \ , \tag{4.9a}$$

where

$$a_{jk}^L = a^L(\xi_{jk}) \qquad k = 0, \dots, j-1 \tag{4.9b}$$

$$a_{jk}^R = a^R(\xi_{jk}) \qquad k = 1, \dots, j \ , \tag{4.9c}$$

and for $\xi \geq 1$, $a^L(\xi)$ and $a^R(\xi)$ are given by

$$a^L(\xi) = \frac{1}{\Gamma(\alpha)} \int_0^1 (\xi - s)^{-1+\alpha} (1-s)\, \mathrm{d}s = \frac{(1+\alpha-\xi)\xi^\alpha + (\xi-1)^{1+\alpha}}{\Gamma(2+\alpha)} \ , \tag{4.9d}$$

$$a^R(\xi) = \frac{1}{\Gamma(\alpha)} \int_0^1 (\xi - s)^{-1+\alpha} s\, \mathrm{d}s = \frac{\xi^{1+\alpha} - (\xi+\alpha)(\xi-1)^\alpha}{\Gamma(2+\alpha)} \ . \tag{4.9e}$$

The accuracy of the schemes can be obtained by a direct computation. It is convenient to introduce the operator

$$I^\alpha_{\mathcal{P}_j} g = \sum_{k=0}^{j} \omega_{jk}\, g^k \tag{4.10}$$

where $g = (g^0, \dots, g^j)$ is a grid function on $\mathcal{P}_j$. The extension of $I^\alpha_{\mathcal{P}_j}$ to continuous functions is obvious. It can be easily verified that for each $g \in C^1[0, \delta_j]$, (4.7) yields

$$|\mathcal{I}^\alpha g(\delta_j) - I^\alpha_{\mathcal{P}_j} g| \leq \frac{\|g'\|_\infty}{\Gamma(1+\alpha)}\, \delta_j^\alpha \max_k \tau_k \ , \tag{4.11}$$

and for each $g \in C^2[0, \delta_j]$, (4.9) yields

$$|\mathcal{I}^\alpha g(\delta_j) - I^\alpha_{\mathcal{P}_j} g| \leq \frac{\|g''\|_\infty}{2\Gamma(1+\alpha)}\, \delta_j^\alpha \max_k \tau_k^2 \ . \tag{4.12}$$

The schemes presented in this section are used as inner schemes for the IDC scheme. As such, at each step, the grid $\mathcal{P}$ is of the form $\mathcal{P} = h\widehat{\mathcal{P}}$, where h is the current step-size, and $\widehat{\mathcal{P}}$ is some fixed (small) set of quadrature nodes in $[0,1]$. In this paper, $\widehat{\mathcal{P}}$ is the set of Gauss-Lobatto nodes. Using the following property

$$\omega_{kj}\big(h\widehat{\mathcal{P}}_j\big) = h^\alpha \omega_{kj}\big(\widehat{\mathcal{P}}_j\big) \qquad\qquad h > 0 \ , \tag{4.13}$$

one may compute the weights ω_{kj} associated with each $I^\alpha_{\mathcal{P}_j}$ at each step with little computational effort.

It is useful to unify (4.11) and (4.12) into a single estimate

$$|\mathcal{I}^\alpha g(\delta_j) - I^\alpha_{\mathcal{P}_j} g| \leq C \max \left|g^{(q)}\right| h^{q+\alpha} \ , \tag{4.14}$$

where $q = 1, 2$, and $C > 0$. Here we also make use of the fact that the schemes are only used for a bounded number of steps each time; i.e., to improve the accuracy we take smaller h, and maintain the number of steps N_{loc} of the inner scheme fixed. This estimate also holds for other schemes, where q may be grater than two. Schemes like this, however, may require more initial conditions and more complicated treatment of left boundary of the interval.

5 Integral deferred correction

Consider the ODE

$$u' = f(t,u) \ . \tag{5.1}$$

The general notion of SDC, introduced in [14], leaves room for many variations to be explored; see, e.g., [15, 16] and the references therein. For simplicity, we forgo the generality of the presentation and restrict ourselves to a single method for standard ODEs and the two methods proposed above for FDEs. Theorem 5.1 of §5.3 provides an estimate of the local truncation error for the IDC scheme for Volterra equations (2.9).

5.1 Standard ordinary differential equations

Fix some t and $h > 0$, and suppose u is a solution of (5.1) in $[t, t+h]$. For $\delta \in [0, h]$, let $U(\delta) = u(t+\delta)$, and $F(\delta, u) = f(t+\delta, u)$. Then,

$$U(\delta) = U(0) + \int_0^\delta F(s, U(s))\ \mathrm{d}s\ . \tag{5.2}$$

For an approximation $V = V(\delta)$ of U we have the following identity

$$U(\delta) = U(0) + \int_0^\delta (F \circ U(s) - F \circ V(s))\ \mathrm{d}s + R(\delta)\ \ , \tag{5.3}$$

where

$$R(\delta) = \int_0^\delta F \circ V(s)\ \mathrm{d}s\ . \tag{5.4}$$

The approximation is computed on a finite set of points in the interval $[0, h]$. Let $\mathcal{P} = (\delta_0, \dots, \delta_{N_{\mathsf{loc}}})$ be the Gauss-Lobatto quadrature node of $[0, h]$. For $j = 1, \dots, N_{\mathsf{loc}}$, we subtract (5.3) at $\delta = \delta_{j-1}$ from (5.3) at $\delta = \delta_j$ to obtain

$$U(\delta_j) = U(\delta_{j-1}) + \int_{\delta_{j-1}}^{\delta_j} \Big(F \circ U(s) - F \circ V(s)\Big)\ \mathrm{d}s + R(\delta_j) - R(\delta_{j-1})\ \ . \tag{5.5}$$

The approximation procedure is iterative. At each iteration the approximation computed at the previous step is used to compute an improved approximation. In the literature, the method is usually formulated using the error equation, and, at each iteration, the correction is computed as the discrete approximation to the error and then approximation is updated. Below we take a slightly different approach and compute the approximation directly. This is equivalent to simply defining the updated approximation at the present iteration as the sum of the previous approximation and the correction.

The scheme is obtained by formally replacing $F \circ V$ in (5.4) by an interpolating polynomial and the integral in (5.5) by a discrete approximation. For simplicity of the presentation we use the forward Euler scheme as the inner scheme, i.e., for the discrete approximation of the integral in (5.5). The approximation V_k at the kth iteration is obtained by solving

$$V_k^j = V_k^{j-1} + \tau_j \left(F_k^{j-1} - F_{k-1}^{j-1}\right) + R_k^j - R_k^{j-1} \tag{5.6}$$

for V_k^j, for each $j = 1, \dots, N_{\mathsf{loc}}$, where $\tau_j = \delta_j - \delta_{j-1}$, V_k^j is the approximation to $U(\delta_j)$, $F_k^j = F(\delta_j, V_k^j)$, and

$$R_k^j = \int_0^{\delta_j} L[\mathcal{P}, F_{k-1}](s)\ \mathrm{d}s \tag{5.7}$$

is the approximation of $R(\delta_j)$. The approximation to u at $t+h$ is given by $V_{N_{\mathsf{iter}}}^{N_{\mathsf{loc}}}$. Notice that setting $F_{-1}^j = 0$, for $j = 0, \dots, N_{\mathsf{loc}}$, yields $R_0^0 = \cdots = R_0^{N_{\mathsf{loc}}} = 0$ and thus (5.6) with $k = 0$ may be used to obtain the initial approximation $V_0^0, \dots, V_0^{N_{\mathsf{loc}}}$.

5.2 Voterra equations

Let $t \geq 0$, and $h > 0$. Consider the Volterra equation (2.9) in $[0, h]$. In practice we have $U(\delta) = u(t+\delta)$, $F(\delta, u) = f(t+\delta, u)$, and $H = H(t, \delta)$, which accounts for the history term, is not known exactly, and we only have its approximation at a finite number of points $\delta \in (0, h]$. For the presentation of the method, however, this does not matter and thus we suppose $H = H(\delta)$ is given. Suppose $V = V(\delta)$ is an approximation of U. Clearly, it holds

$$U(\delta) = \mathcal{I}^\alpha(F \circ U - F \circ V) + \widetilde{H}(\delta) \tag{5.8}$$

where

$$\widetilde{H}(\delta) = H(\delta) + \mathcal{I}^\alpha(F \circ V) \ . \tag{5.9}$$

The scheme is obtained by formally replacing the fractional integral in (5.8) by a discrete approximation, and $F \circ V$ in (5.9) by an interpolating polynomial. In our implementation we use the Gauss-Lobatto nodes $\mathcal{P} = (\delta_0, \ldots, \delta_{N_{\mathsf{loc}}})$ of $[0, h]$. Thus the approximation V_k at the kth iteration is obtained by solving

$$V_k^j = \sum_{s=0}^{j} \omega_{js}(F_k^s - F_{k-1}^s) + H_k^j \tag{5.10a}$$

$$H_k^j = H(\delta_j) + S_{\mathcal{P}}^\alpha F_{k-1}^j \tag{5.10b}$$

for V_k^j, where, for each $j = 0, \ldots, N_{\mathsf{loc}}$, V_k^j is the approximation to $U(\delta_j)$ at the kth correction iteration, $F_k^j = F(\delta_j, V_k^j)$, the coefficients $\omega_{js} = \omega_{js}(\mathcal{P}_j)$, with $\mathcal{P}_j = (\delta_0, \ldots, \delta_j)$, are given by either (4.7) or (4.9), and S^α is the operator defined by

$$S_{\mathcal{P}}^\alpha g^j = \mathcal{I}^\alpha L[\mathcal{P}, g](\delta_j) \qquad\qquad j = 0, \ldots, N_{\mathsf{loc}} \ , \tag{5.11}$$

where $L[\mathcal{P}, g]$ is the polynomial interpolating $g = (g^0, \ldots, g^{N_{\mathsf{loc}}})$ at $\mathcal{P}$. The operator $S_{\mathcal{P}}^\alpha$ may be computed exactly by mapping the Lagrange basis to the Legendre basis and using the formula

$$\frac{1}{\Gamma(\alpha)} \int_{-1}^{x} (x-y)^{-1+\alpha} \widetilde{P}_n^{(0,0)}(y)\, \mathrm{d}y = \sqrt{\frac{\Gamma(n-\alpha+1)}{\Gamma(n+\alpha+1)}}\, (1+x)^\alpha \widetilde{P}_n^{(-\alpha,\alpha)}(x) \tag{5.12}$$

valid for $x \in [-1, 1]$, where $\widetilde{P}_n^{(a,b)}$ is the nth degree normalized Jacobi polynomial associated with the weight $(1-y)^a(1+y)^b$, [22]. Note that setting $F_{-1} = 0$ yields $H_0 = H$ and thus (5.10) with $k = 0$ may be used to obtain the initial approximation V_0.

5.3 The local truncation error

We use the notation F_k for both the grid function $F_k = (F_k^0, \ldots, F_k^{N_{\mathsf{loc}}})$ on $\mathcal{P} = (\delta_0, \ldots, \delta_{N_{\mathsf{loc}}})$ and the polynomial interpolating the data (δ_j, F_k^j), $j = 0, \ldots, N_{\mathsf{loc}}$. Let $\mathcal{I}^\alpha|_t g = \mathcal{I}^\alpha g(t)$, and $L = L(\delta) = L[\mathcal{P}, F \circ U](\delta)$ the polynomial interpolating $F \circ U$ at $\mathcal{P}$. The following theorem gives an estimate of the local truncation error of the IDC method.

Theorem 5.1 *Let $\alpha \in (0, 1)$ and $\widehat{\mathcal{P}} \subset [0, 1]$ a set of $N_{\mathsf{loc}} + 1$ distinct quadrature nodes. For $h > 0$, let $\mathcal{P} = h\widehat{\mathcal{P}} = (\delta_0, \ldots, \delta_{N_{\mathsf{loc}}}) \subset [0, h]$, for each $j = 0, \ldots, N_{\mathsf{loc}}$, $\mathcal{P}_j = (\delta_0, \ldots, \delta_j)$ and $I_{\mathcal{P}_j}^\alpha$ a discrete operator (4.10) satisfying (4.14). Suppose U a solution of (2.9) such that $F \circ U$ is smooth, and V_k the numerical solution obtained by the IDC scheme (5.10) at the kth correction iteration, with appropriate initial conditions, then*

$$U(\delta_j) - V_k^j = O(h^\nu) \qquad\qquad h \to 0^+ \ , \tag{5.13}$$

where $\nu = \min(N_{\mathsf{loc}} + 1 + \alpha,\ q + (k+1)\alpha)$.

Note that in practice we apply the method to (2.9) which we obtain from (2.8) by substituting $U(\delta) = u(t + \delta)$, $F(\delta, U) = f(t + \delta, U)$ and $H(\delta)$ for $H(t, \delta)$. Thus, for $t > 0$ the hypothesis that U and F are smooth in $[0, h]$ is not very restrictive.

Proof Let the local error $e_k^j = U(\delta_j) - V_k^j$ at the kth correction iteration. The proof is by induction on k. By (4.14), clearly (5.13) holds for $k = 0$. Suppose (5.13) holds with $k - 1$ substituting for k and $\nu' = \min(N_{\mathsf{loc}} + 1 + \alpha,\ q + k\alpha)$ substituting for ν. We have

$$e_k^j = \mathcal{I}^\alpha|_{\delta_j}(F \circ U) - I_{\mathcal{P}_j}^\alpha(F_k - F_{k-1}) - \mathcal{I}^\alpha|_{\delta_j} F_{k-1} \ . \tag{5.14}$$

Due to

$$\begin{aligned} e_k^j - I^\alpha_{\mathcal{P}_j}\left(F\circ U - F_k\right) &= \mathcal{I}^\alpha|_{\delta_j}(F\circ U - F_{k-1}) - I^\alpha_{\mathcal{P}_j}(F\circ U - F_{k-1}) \\ &= \mathcal{I}^\alpha|_{\delta_j}(F\circ U - L) + \mathcal{I}^\alpha|_{\delta_j}(L - F_{k-1}) - I^\alpha_{\mathcal{P}_j}(L - F_{k-1}) \end{aligned} \tag{5.15}$$

we obtain

$$e_k^j = \sum_{s=0}^{j} \omega_{js}\Big(F(\delta_s, U(\delta_s)) - F_k^s\Big) + \rho_k^j \tag{5.16}$$

where

$$\rho_k^j = \mathcal{I}^\alpha|_{\delta_j}(F\circ U - L) + \big(\mathcal{I}^\alpha|_{\delta_j} - I^\alpha_{\mathcal{P}_j}\big)(L - F_{k-1}) \ . \tag{5.17}$$

Thus we recover

$$|e_k^j| \le C_0 \sum_{s=0}^{j} \omega_{js}|e_k^s| + |\rho_k^j| \ , \tag{5.18}$$

which yields

$$|e_k^j| \le C_1 \max_s |\rho_k^s| \ , \tag{5.19}$$

by a simple induction on j, recalling that $j \le N_{\mathsf{loc}}$ is bounded as $h \to 0^+$. It is left to estimate ρ_k^s. We do this by estimating each of the terms separately. Due to standard results regarding polynomial interpolation, the first term on the right hand side of (5.17) satisfies

$$\left|\mathcal{I}^\alpha|_{\delta_j}\left(F\circ U - L\right)\right| \le \frac{h^{N_{\mathsf{loc}}+1+\alpha}}{(N_{\mathsf{loc}}+1)!\,\Gamma(1+\alpha)} \left\|(F\circ U)^{(N_{\mathsf{loc}}+1)}\right\|_{L^\infty(0,h)} \tag{5.20}$$

and thus

$$\mathcal{I}^\alpha|_{\delta_j}\left(F\circ U - L\right) = O\big(h^{N_{\mathsf{loc}}+1+\alpha}\big) \ . \tag{5.21}$$

To estimate the second term, we use (4.14) to recover the following estimate

$$\left|\big(\mathcal{I}^\alpha|_{\delta_j} - I^\alpha_{\mathcal{P}_j}\big)(L - F_{k-1})\right| \le C_2 h^{q+\alpha} \max\left|\frac{\mathrm{d}^q}{\mathrm{d}\delta^q}\Big(L - F_{k-1}\Big)\right| \tag{5.22}$$

for each j. Note that $L - F_{k-1}$ is the N_{loc}-degree polynomial interpolating the error e_{k-1} at $\mathcal{P}$. Due to the Lagrange form of this polynomial, we have

$$\left|\frac{\mathrm{d}^q}{\mathrm{d}\delta^q}\Big(L - F_{k-1}\Big)\right| \le C_3 h^{-q} \max_s |e_{k-1}^s| \tag{5.23}$$

which, by the induction hypothesis, yields

$$\big(\mathcal{I}^\alpha|_{\delta_j} - I^\alpha_{\mathcal{P}_j}\big)(L - F_{k-1}) = O\big(h^{\nu'+\alpha}\big) \ . \tag{5.24}$$

Substituting (5.21) and (5.24) into (5.17), we recover $\rho_k^j = O(h^\nu)$ with $\nu = \min(N_{\mathsf{loc}} + 1 + \alpha,\, q + (k+1)\alpha)$. Thus, due to (5.19) we have (5.13), and the proof is complete. $\square$

6 Adaptive error control

6.1 Adaptive control of the step-size

As the proposed schemes employ a DIRK method to advance the auxiliary variables, it is reasonable to consider the use of an embedded scheme to estimate the local error. A sightly different approach is to control the error of the reconstruction of the history term at the nearest point δ_1 of the inner scheme. For the first approach, we define

$$\Delta = |\Phi^{n+1} - \widetilde{\Phi}^{n+1}|_\infty \tag{6.1}$$

and for the second

$$\Delta = |(\Phi^{n+1} - \widetilde{\Phi}^{n+1})\sigma(\delta_1)\,|_\infty \ , \tag{6.2}$$

where Φ^{n+1}, and $\widetilde{\Phi}^{n+1}$ are the approximations to $\Psi(t_n + h)$ obtained by the main and the embedded schemes, respectively, and $|\cdot|_\infty$ denotes the entry-wise maximum norm. For an error tolerance $\varepsilon_h > 0$, we define

$$q = \left(\frac{\varepsilon_h}{\Delta}\right)^{1/(\widetilde{\nu}_{\mathsf{RK}}+1)} \tag{6.3}$$

where $\widetilde{\nu}_{\mathsf{RK}}$ is the order of the embedded scheme. Let $q_u > q_l > 0$. The step size is chosen as follows. If $q < p_l$, the approximation is discarded and the step-size h is halved. If $q \geq q_l$ the approximation is accepted and the scheme advances; in that case the new value for the auxiliary variable is set to Φ^{n+1}. If in addition $q > q_u$, the step-size is doubled.

In our experience, (6.2) performs better than (6.1) at small tolerances near the singularity at the initial condition. When using (6.1), the error decreases as the tolerance is reduced, however near the initial condition, it decreases at a lower rate, i.e., the indicator does not control the error adequately near the singularity. With (6.2), this issue seems to be resolved. The price for high accuracy is, obviously, advancing with a very small step-size near the singularity at the initial condition.

6.2 Adaptive control of P

The kernel compression scheme used for the approximation of the history term offers an a priori estimate on the number of auxiliary variables required to satisfy an error tolerance. Thus, to control the kernel compression error, we use A_m and B_p given by (3.14b) to estimate the number of poles per circle m and the total number of circles p. The kernel compression approximation, however, is not uniform in δ – the distance of the reconstruction time from the present time. Since δ is proportional to the step-size h, the number of auxiliary variables may need to increase when h decreases. Also, when δ increases, maintaining the same number of auxiliary variables may be unnecessary. As the approximation is costly, it is desirable to adapt the number of auxiliary variables to keep it close to the optimal value for a given tolerance ε_P.

Estimate (3.16) and the multipole approximation (3.12) associated with the approximation operator $\mathcal{I}_{\delta,r}$ offer a possible approach to this end. Note that the number of poles on each circle m, required to satisfy an error tolerance is independent of δ. Therefore it may remain fixed throughout the computation. In fact, the radii r_k and the centers c_k of the circles are also independent of δ. Hence, the poles λ_{kj} and therefore the auxiliary variables are likewise independent of δ (see discussion in §3).

Thus, changing δ requires only that we change the weights σ and perhaps the number of circles. This raises the question of what values should be assigned to new auxiliary variables. In our approach, new auxiliary variables are set to zero at the current time as their expected contribution to the reconstruction was below the error tolerance previously, and redundant auxiliary variables are discarded as their expected contribution drops below the error tolerance.

This approach relies on the following argument. The error committed by neglecting information from the interval $[0, t_n]$ in the auxiliary variable associated with λ is given by

$$\rho(t) = \int_0^{t_n} e^{\lambda(t-s)} f(s) \, \mathrm{d}s = e^{\lambda(t-t_n)} \int_0^{t_n} e^{\lambda(t_n-s)} f(s) \, \mathrm{d}s \; . \tag{6.4}$$

Note that this expression decays exponentially in t. Also note that λ is typically on one of the last circles, and therefore has a large negative real part, which implies that the decay is very rapid.

7 Further details on the implementation

The results below are obtained with two high-order methods – one explicit and one implicit. The two methods differ only by their inner schemes. The explicit scheme is based on the composite left endpoint rule (4.7), while the implicit utilizes the composite trapezoidal rule (4.9). For simplicity, we denote the methods employing the composite trapezoidal and left endpoint rules as inner schemes by TR-IDC and LER-IDC, respectively. Both methods employ the same DIRK method for the approximation of the auxiliary variables, (2.11). The DIRK method – denoted ARK4(3)6L[2]SA - ESDIRK in [23] – is 6-stage, L-stable with main and embedded methods of orders 4 and 3, respectively. For the LER-IDC and TR-IDC schemes, we use

$$N_{\mathsf{iter}} = \left\lceil \frac{3}{\alpha} - 1 \right\rceil \tag{7.1a}$$

and

$$N_{\mathsf{iter}} = \left\lceil \frac{3}{1+\alpha} - 1 \right\rceil \tag{7.1b}$$

correction iterations, respectively. This choice corresponds to an increase of the order of the LER-IDC and TR-IDC schemes by α and $1+\alpha$, respectively, each iteration and aims to achieve local order of 4. By Theorem 5.1, (7.1a) correction iterations guarantee the LER-IDC scheme has local order of 4. In contrast, (7.1b) is due to numerical evidence and not a complete analysis. The number of correction iterations which guaranties local order of 4 for the TR-IDC scheme is

$$N_{\mathsf{iter}} = \left\lceil \frac{2}{\alpha} - 1 \right\rceil \; , \tag{7.2}$$

by Theorem 5.1. For both schemes, and independently of N_{iter}, we take $N_{\mathsf{loc}} = 5$. The reason for this choice is that it yields six quadrature points, the same number as the RK scheme. The tolerance for the kernel compression error indicator is $\varepsilon_P = 10^{-1}\varepsilon_h$, where ε_h is the tolerance for the step-size error indicator.

8 Numerical tests

In the tests below we set $\eta = 1/T$, ε_h denotes the tolerance for the step-size error indicator, $\varepsilon_P = 10^{-1}\varepsilon_h$ is the tolerance for the error indicator controlling the history term. Unless mentioned otherwise, the error indicator (6.2) is used to control the step-size, $q_l = 1$, $q_u = 10$, the number of correction iterations N_{iter} is given by (7.1a), and (7.1b) for the LER-IDC and TR-IDC schemes, respectively, and the initial step size $h_0 = 2^{-5}$. A version of the code used in the numerical tests is available at [24].

8.1 Number of correction iterations

Consider the initial value problem

$$D^\alpha u = -u \qquad u(0) = 1 \ . \tag{8.1}$$

in $(0,T)$. Its solution is given by

$$u(t) = E_\alpha(-t^\alpha) \qquad E_\alpha(t) = \sum_{k=0}^{\infty} \frac{t^k}{\Gamma(\alpha k+1)} \ , \tag{8.2}$$

where E_α is the Mittag-Leffler function [25]. Unless mentioned otherwise, the reference solution is computed using [26]. The results are obtained for $\alpha = 0.5$, and $T = 5$.

In this test we look at the influence of the number of correction iterations N_{iter} on the accuracy of the scheme. We apply the two methods to (8.1) and measure the global error for different N_{iter}. Figure 8.1 shows E_1/T, where E_1 is the global error

$$E_1 = \sum_{n=1}^{N} h_n |v^n - u(t_n)\,| \qquad h_n = t_n - t_{n-1} \ , \tag{8.3}$$

as a function of the average step-size

$$h_{\mathsf{avg}} = \frac{1}{N} \sum_{n=1}^{N} h_n \ . \tag{8.4}$$

The data for each graph is obtained by setting the error tolerance ε_h to $\varepsilon_h = 10^{-k}$, with $k = 2,3,4,5,6$. The results on the left, obtained with the LER-IDC method, show the following behavior. When no correction iterations are performed, i.e. $N_{\mathsf{iter}} = 0$, the global error behaves as $O(h^{3/2})$, which coincides the local order $1+\alpha = 3/2$ of the method approximating (2.9). For $N_{\mathsf{iter}} \geq 1$, we measure global orders $1 + (N_{\mathsf{iter}}+1)\alpha$ which is in agreement with the expected local order of the IDC method applied to (2.9). The figure on the right, obtained with the TR-IDC method, shows the graphs of the global error E_1/T of the methods with $N_{\mathsf{iter}} = 0, 1$. Here we measure order 4 after one correction iteration. This is in agreement with the local order $1 + (N_{\mathsf{iter}}+1)(1+\alpha) = 4$ expected of the IDC method provided its order increases by $1+\alpha$ each iteration, while Theorem 5.1 only guarantees local order $2 + (N_{\mathsf{iter}}+1)\alpha = 3$, i.e., an increase of α per iteration. Figure 8.2 shows the results of the same test for $\alpha = 0.8$ and the LER-IDC scheme. The figure shows a similar behavior. Note that in all these tests, the global order measured is the minimum between the local order of the approximation of (2.9) and the global order of the DIRK method applied to (2.11).

Next we apply the TR-IDC method to (8.1) with $\alpha = 0.2$. The results, obtained with $\varepsilon_h = 10^{-k}$, for each $k = 3,4,5,6,7$, are in Figure 8.3. In this test, the order of convergence measured is less than 4, and increasing the number of iterations seems to have no significant effect. The number of iterations $N_{\mathsf{iter}} = 2$ is chosen according to (7.1b), and $N_{\mathsf{iter}} = 9$ chosen by (7.2) corresponds to an order increase of α per correction iteration. This order reduction could be caused by the more severe singularity at the initial time. Since adding correction iterations does not seem to improve the accuracy or the order of the method, the order reduction is likely caused by either the polynomial interpolation (of the data at the Gauss-Lobatto nodes) or the RK scheme. Such order reduction is known to occur in the application of RK methods to very stiff problems [27, 28].

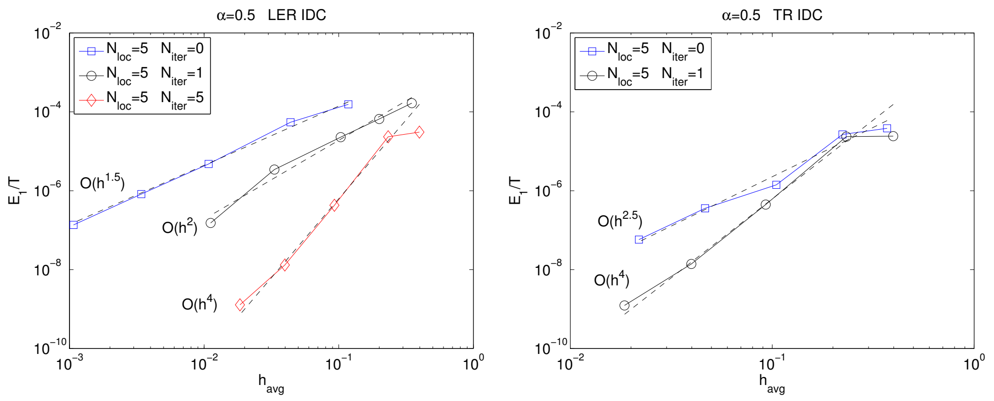

Fig. 8.1 The global error E_1/T for (8.1) with $\alpha = 0.5$ plotted as a function of the average step size h_{avg}. The different graphs correspond to different numbers N_{iter} of correction iterations. The results on the left and right are obtained with the LER-IDC and TR-IDC methods, respectively.

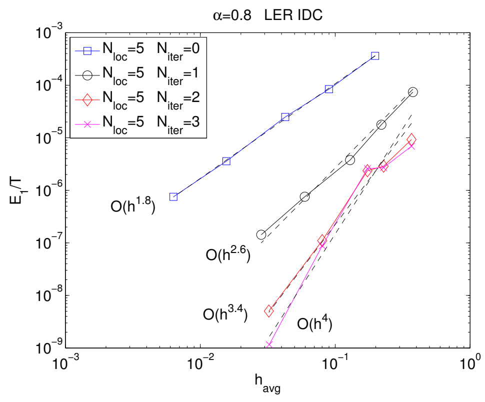

Fig. 8.2 The global error E_1/T for (8.1) with $\alpha = 0.8$ plotted as a function of the average step size h_{avg}. The different graphs correspond to different numbers N_{iter} of correction iterations.

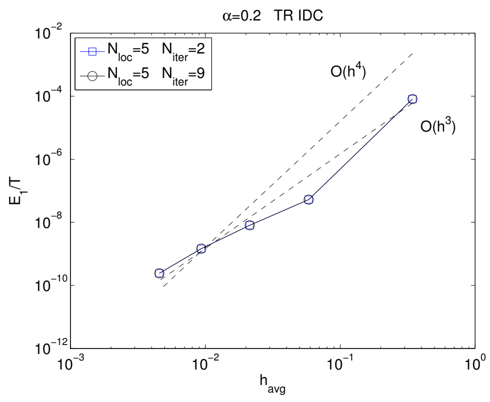

Fig. 8.3 Global error E_1/T for (8.1) with $\alpha = 0.2$ plotted as a function of the average step size h_{avg}. The different graphs correspond to different numbers N_{iter} of correction iterations.

8.2 Comparison of error indicators

We consider (8.1) in $(0, T)$, with $\alpha = 0.5$, and $T = 5$. The purpose of this test is to demonstrate an issue that may arise when using (6.1) as an error indicator. The results are in Figure 8.4. Results in the left column are obtained with (6.1) and results in the right column with (6.2). The first row shows the local error

$$e_h^n = |v^n - u(t_n)| \tag{8.5}$$

of the LER-IDC method as a function of time. In this test we are specifically interested in the behavior of the error near the singularity at the initial condition. Thus the second row shows the same results, but on a log-log scale. The third row shows the step-size chosen by the program as a function of time, also on a log-log scale.

As can be seen at the left figure of the second row, obtained with (6.1), as the tolerance decreases, the error is reduced. However, the reduction is not uniform in time. Notice that near the initial condition the error decreases at a lower rate. Although in this example, this is observable at very small scales, if left untreated, it may ruin the accuracy of the approximation. The right figure of the second row, obtained with (6.2), does not show this behavior. While the jump near the initial time is still visible, the relative error at the peak, compared to the error elsewhere does not seem to grow as the tolerance decreases. The price is, of course, that the starting step-size is considerably smaller. Regardless, with both error indicators we measure 4th order convergence for E_1. This is shown in Figure 8.5, where the graphs of Ind. 1 and Ind. 2 are obtained with (6.1) and (6.2), respectively.

8.3 Fractional Van der Pol equation

Consider the nonlinear fractional differential equation

$$\left(D^\alpha\right)^2 x - \varepsilon \left(1 - x^2\right) D^\alpha x + x = 0 \tag{8.6a}$$

in $(0, T)$, with initial conditions

$$x(0) = x_0 \qquad D^\alpha x(0) = y_0 \ . \tag{8.6b}$$

Here ε is a non-negative constant, and x_0, $y_0 \in \mathbb{R}$. For $\alpha = 1$, (8.6a) is reduced to the classical Van der Pol equation which can be shown to have a stable periodic solution. To apply the scheme we write (8.6a) as a system by substituting $y = D^\alpha x$. Thus, we have

$$D^\alpha x = y \tag{8.7a}$$

$$D^\alpha y = \varepsilon \left(1 - x^2\right) y - x \ , \tag{8.7b}$$

in $(0, T)$ subject to the initial condition

$$x(0) = x_0 \qquad y(0) = y_0 \ . \tag{8.7c}$$

In the tests, we fix $\alpha = 0.8$, $\varepsilon = 4$, $x_0 = 2$, $y_0 = 0$, and $T = 25$. Figure 8.6 shows the reference solution computed with the TR-IDC method and $\varepsilon_h = 10^{-10}$. Figure 8.7 shows the global error E_1/T as a function of the average step size (8.4). The numerical solutions are compared to the spline interpolation of the reference numerical solution. In this example we measure 4-th order convergence of both methods.

Figure 8.8 shows the local error e_h (top), the step size (middle), and number of auxiliary variables P (bottom) as functions of time. Comparing Figures 8.6 and 8.8, we observe that the error indicator detects the changes in the solution and adapts the step size h accordingly. Based on the error indicator for the history term and the step size h, the number P of auxiliary variables is also modified.

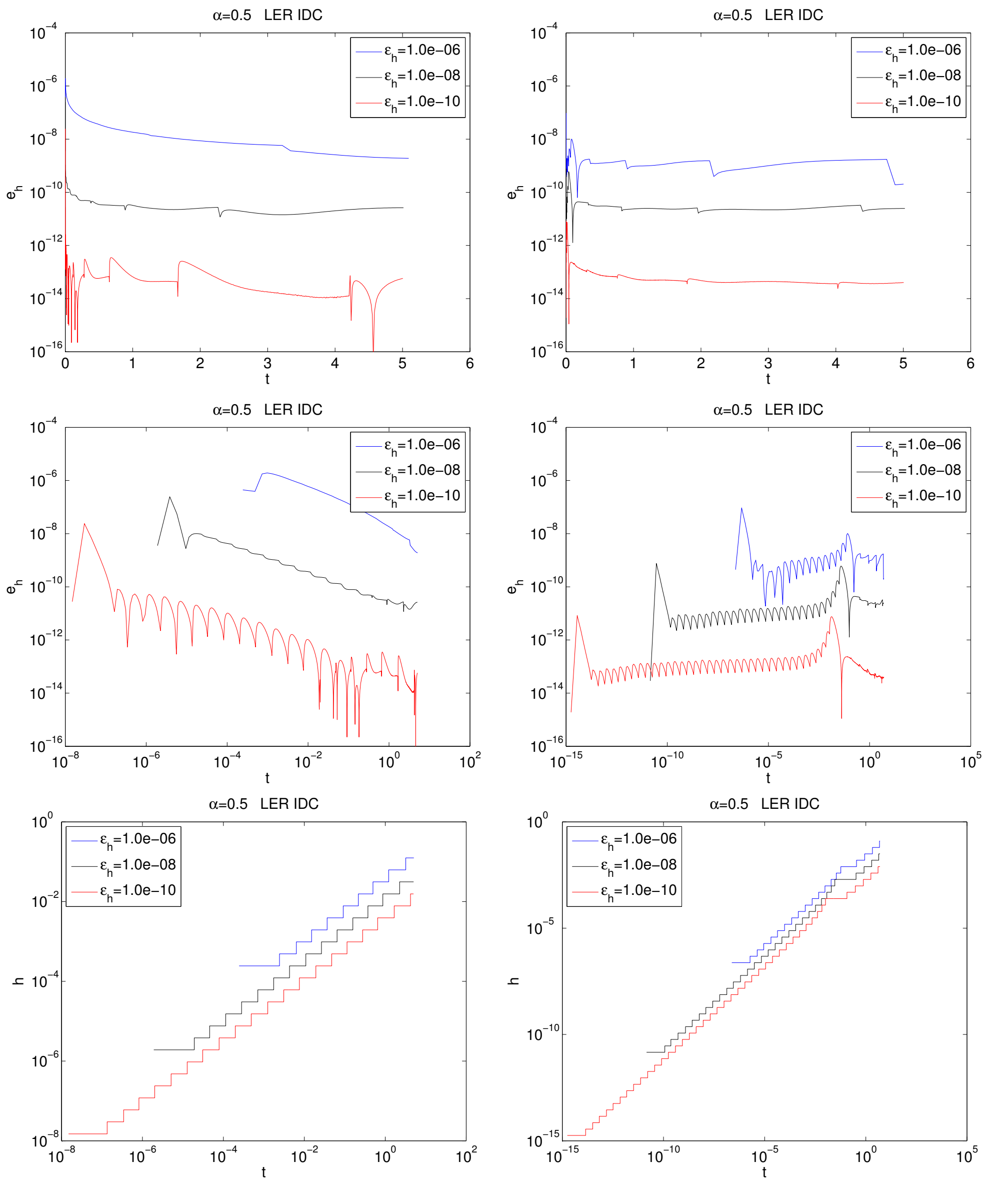

Fig. 8.4 Comparison of error indicator performance – problem (8.1): The figures on the left and right show the results obtained with (6.1) and (6.2), respectively. Top: the local error e_h as a function of time; middle: the local error as a function of time on log-log scale; bottom: the step-size as a function of time on log-log scale.

The following test provides an example to a situation where the error indicator does not perform as well. When α becomes smaller, (8.7) becomes harder to approximate. Figure 8.9 shows a numerical solution computed for (8.7) with $\alpha = 0.5$, and $\varepsilon = 4$, and error tolerance $\varepsilon_h = 10^{-10}$. Problem (8.7) with these parameters seems more difficult for treatment than the the problem with $\alpha = 0.8$. In Table 8.1 we show the number of steps and the number of rejected steps in the TR-IDC method, for different error tolerances. Here, q_u is the parameter which determines when the step-size is increased. The table shows that when $q_u = 10$, the method rejects a significant number of steps. While the method completed the tests, rejecting a significant number of steps should be avoided if possible. When q_u is increased to $q_u = 20$, the method rejects very few steps, however it requires more steps to cover the time interval, as, on average, a smaller step-size is used.

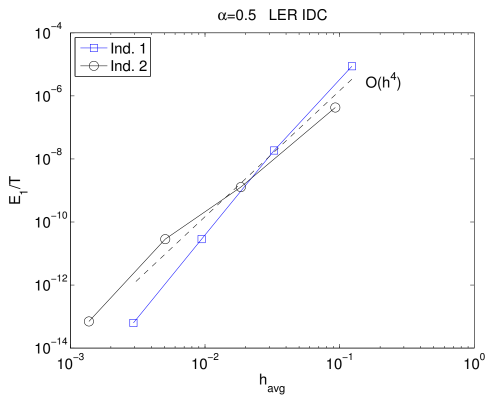

Fig. 8.5 Comparison of error indicator performance – problem (8.1): the global error E_1/T is plotted as a function of the average step-size; the graphs Ind. 1 and Ind. 2 are obtained with (6.1) and (6.2), respectively.

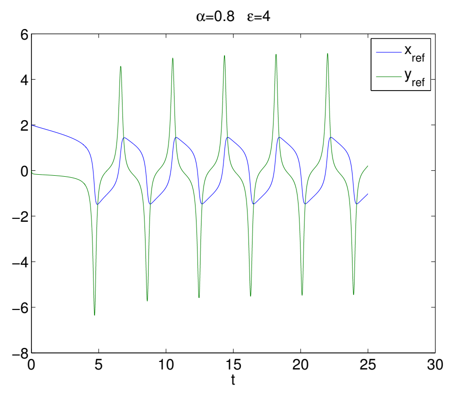

Fig. 8.6 The reference solution to (8.7), with $\alpha = 0.8$, and $\varepsilon = 4$ obtained with the TR-IDC and error tolerance $\varepsilon_h = 10^{-10}$.

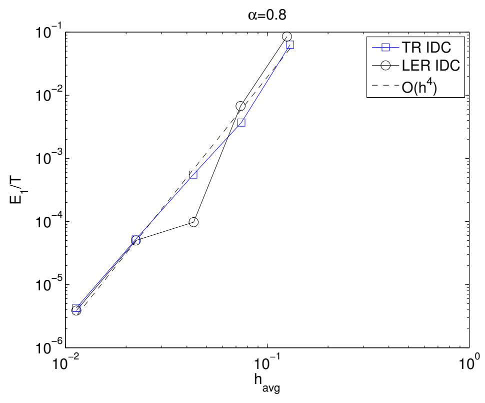

Fig. 8.7 The global error E_1/T for (8.7) with $\alpha = 0.8$, and $\varepsilon = 4$. The numerical solutions are compared to a reference solution computed as the spline interpolation of the numerical solution shown in Figure 8.6.

	$q_u = 10$					$q_u = 20$				
ε_h	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}
N	651	665	731	905	1242	1056	1119	1722	2632	3776
Rejected	549	269	564	291	192	4	4	6	16	23

Table 8.1 Problem (8.7) with $\alpha = 0.5$ and $\varepsilon = 4$. Results obtained with the TR-IDC method. The number of steps the method performed and rejected for different error tolerances and q_u.

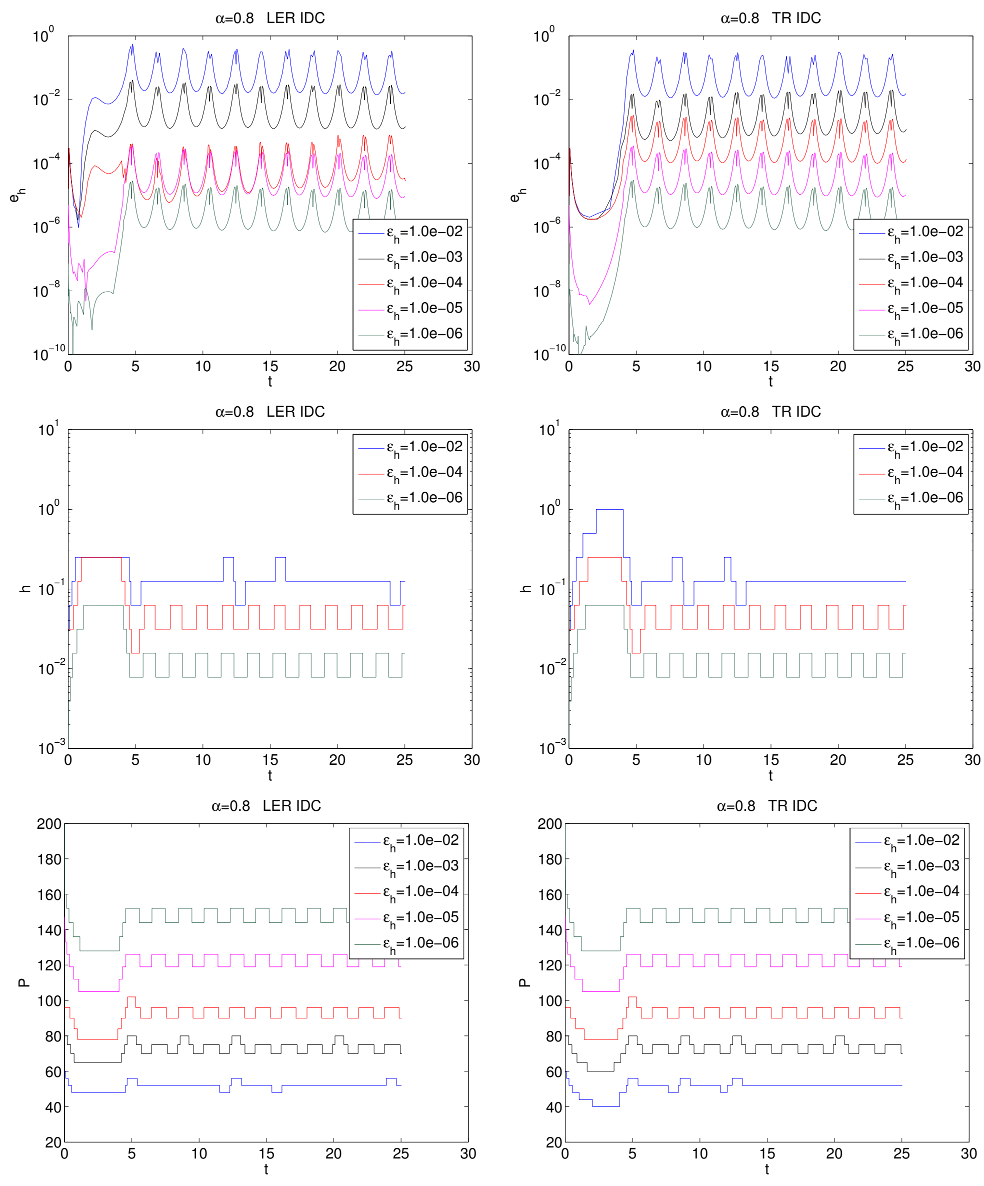

Fig. 8.8 Problem (8.7): top – the local 1-norm of the error; middle – h as a function of t; bottom – P as a function of t. The results on the left and right are obtained with the LER-IDC and TR-IDC methods, respectively.

8.4 Fractional telegraph equation

Consider the fractional telegraph equation

$$\left(D^{\alpha}\right)^2 q - c^2 \frac{\partial^2 q}{\partial x^2} + aD^{\alpha} q + bq = 0 \ , \tag{8.8a}$$

in $(0,1) \times (0,T)$, subject to Dirichlet boundary conditions

$$q|_{x=0} = q_E(t) \qquad\qquad q|_{x=1} = q_W(t) \tag{8.8b}$$

and initial conditions

$$q(x,0) = q_0(x) \qquad\qquad D^{\alpha} q(x,0) = q_1(x) \ \ . \tag{8.8c}$$

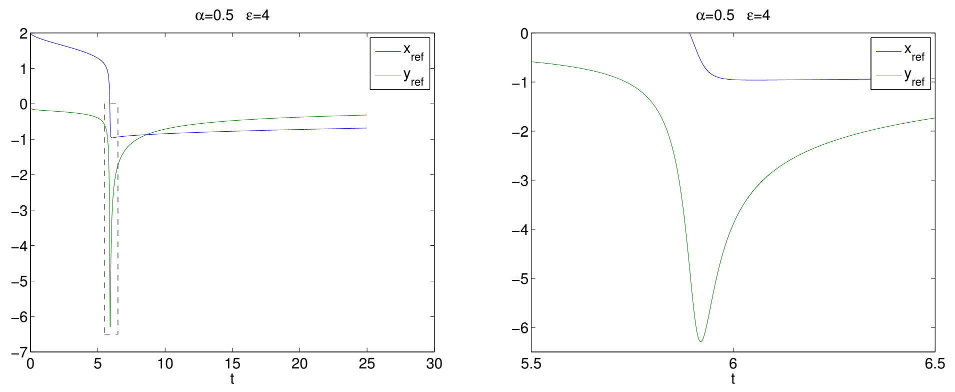

Fig. 8.9 A numerical solution to (8.7), with $\alpha = 0.5$, and $\varepsilon = 4$ obtained with the TR-IDC and error tolerance $\varepsilon_h = 10^{-10}$. The figure on the right is an enlargement of the dashed frame on the left.

To apply the method we define $r = D^\alpha q$ and transform (8.8a) into

$$D^\alpha q = r \tag{8.9a}$$

$$D^\alpha r = c^2 \frac{\partial^2 q}{\partial x^2} - bq - ar \ . \tag{8.9b}$$

To discretize the spacial operator and thus obtain a system (2.5), we substitute

$$Q(x,t) = \sum_{j=0}^{N_x+1} Q_j(t)\,\ell_j(x) \qquad\qquad R(x,t) = \sum_{j=1}^{N_x} R_j(t)\,\bar{\ell}_j(x) \tag{8.10}$$

for q, and r in (8.9), impose the boundary conditions and evaluate at x_i, with $i = 1, \dots, N_x$, where ℓ_j are the Lagrange polynomials associated with the Gauss-Lobatto nodes $x_0, \dots, x_{N_x+1}$ of the spacial domain $[0, 1]$, and $\bar{\ell}_j$ are the Lagrange polynomials associated with $x_1, \dots, x_{N_x}$. Thus, we obtain

$$D^\alpha Q_i = R_i \tag{8.11}$$

$$D^\alpha R_i = c^2 \sum_{j=0}^{N_x+1} \ell_j''(x_i)\,Q_j - bQ_i - aR_i \tag{8.12}$$

with $i = 1, \dots, N_x$, where

$$Q_0 = q_W \qquad\qquad Q_{N_x+1} = q_E \ . \tag{8.13}$$

We set $\alpha = 0.5$, $a = b = 0$, $c = 1/\pi$, $q_W(t) = \sin(t)$, $q_E(t) = 0$, $q_0 = q_1 = 0$, $T = 2\pi$, and $N_x = 11$. All tests are performed with the LER-IDC method. In the following $q^n(x)$ denotes the approximation to $q(x, t_n)$. Figure 8.10 shows the reference solution q_{ref} computed with $\varepsilon_h = 10^{-8}$. We compare the numerical solutions to the reference solution at $x = x_6 = 1/2$. The results are in Figure 8.11. In this figure,

$$E_1 = \sum_n h_n e_h^n \ , \tag{8.14}$$

and

$$e_h^n = |q^n(1/2) - q_{\mathsf{ref}}(1/2, t_n)| \ , \tag{8.15}$$

where $q_{\mathsf{ref}}(1/2, \cdot)$ is the spline interpolation in time of q_{ref} at $x = 1/2$. The top and bottom figures on the right show the global-in-time error E_1/T at $x = 1/2$ as a function of the average step size, and the step size as a function of time, respectively. Note that when the error tolerance ε_h is reduced from 10^{-2} to 10^{-3} or 10^{-4}, the average step size does not change significantly. The error, however, decreases from $E_1/T \approx 10^{-5}$, for $\varepsilon_h = 10^{-2}$, to $E_1/T \approx 10^{-6}$, for $\varepsilon_h = 10^{-3}$, and

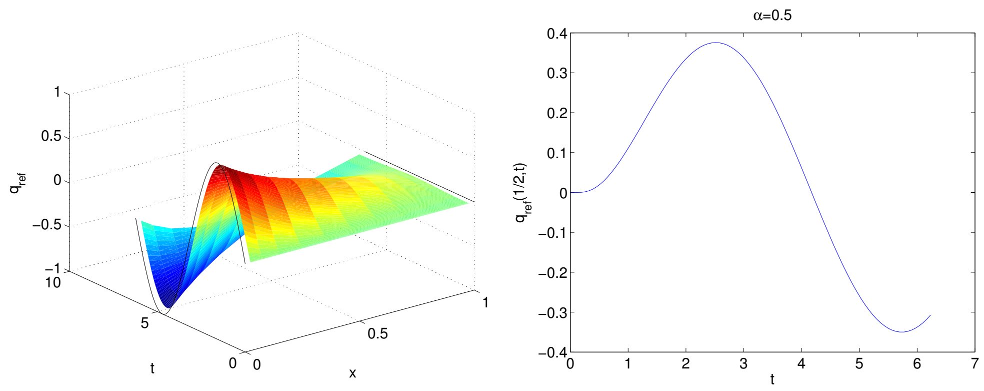

Fig. 8.10 The reference solution q_{ref} to (8.9). The figure on the left shows the solution surface in the x-t plane, and the figure on the right shows $q_{\mathsf{ref}}(1/2,\cdot)$ as a function of t.

$E_1/T \approx 10^{-7}$, for $\varepsilon_h = 10^{-4}$. It is likely that at these moderate tolerances, the main restriction on the step size is imposed by the stability of the discrete scheme and not its accuracy. This explains why the average step size does not decrease significantly. The reason the error is reduced so quickly could be that when $\varepsilon_h = 10^{-2}$, the approximation of the history term contributes a significant error compared with the error due to the step size picked by the program. Thus, when ε_h is reduced, the tolerance for the history term is also reduced – as they are linked by $\varepsilon_P = 10^{-1}\varepsilon_h$. As a result, more auxiliary variables are retained, as is shown in the bottom left figure, and hence the approximation of the history term improves. When ε_h is decreased further, the graph of the error continues to decrease at a lower rate, closer to the expected rate.

9 Concluding remarks

We have developed fully discrete, high-order, adaptive time-stepping methods for FDEs. The methods are based on the kernel compression scheme proposed in [7] for the approximation of the history term and on the IDC method for the approximation of the local term. The analysis shows that the IDC method increases the order of the local approximation by α each correction iteration. We have presented numerical results obtained with two methods for a number of problems, illustrating the performance. The results demonstrate the capability to yield high order accuracy and the ability of the error indicators to control the error, detect changes in the solution, and adapt the step size accordingly. A version of the code used in the numerical tests is available at [24].

At this point there are some interesting theoretical issues left open. One issue is the local order of the IDC method, and its increase at each correction iteration. Theorem 5.1 states that the local order of the IDC method increases by at least α each correction iteration. In our tests we measure the global errors of the schemes and observe an increase of the order by α and $1+\alpha$ for each iteration with the composite left endpoint and trapezoidal rules, respectively. Furthermore, our results show a considerable improvement of the accuracy when using the trapezoidal rule compared to the left endpoint rule. This issue is of particular importance since an increase of the order by α per correction iteration implies the number of correction iterations required to obtain a prescribed order tends to infinity as α tends to zero. In contrast, if one can guarantee an increase of the order by $1+\alpha$, a bounded number of correction iterations will suffice to achieve the prescribed order for all $\alpha \in (0,1)$. Some variations of the approach could also be explored. Different quadratures nodes may yield improved accuracy and stability, or allow for more efficient algorithms. Other types of schemes such as convolution quadratures may be used as inner schemes. We leave the treatment of these issues to future work.

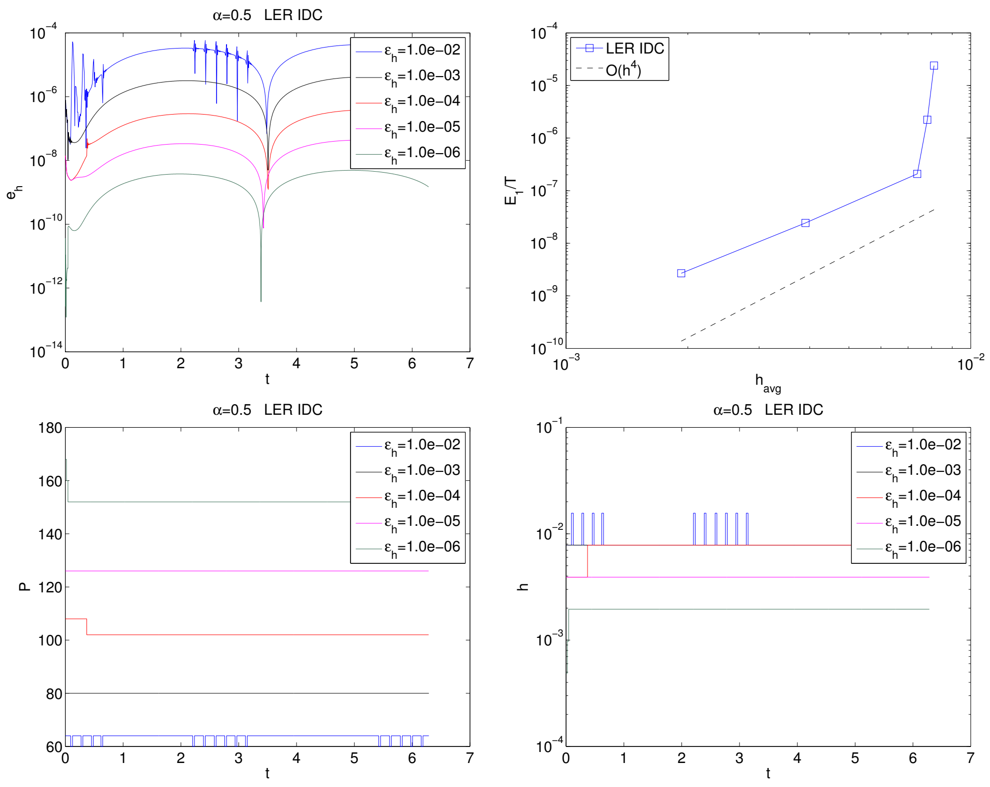

Fig. 8.11 Problem (8.9); numerical results. Starting from the top left and continuing clockwise, the figures show the local error e_h at $x = 0.5$ as a function of time, the global in time error E_1/T at $x = 0.5$ as a function of the average step size h_{avg}, the step size h as a function of time, and P as a function of time.

Acknowledgements The authors are grateful to Stefano Guarino for his contribution.

References

1. Y. Zhang, Z. Sun, H. Liao, *Finite Difference Methods for the Time Fractional Diffusion Equation on Non-Uniform Meshes*, J. Comput. Phys. 256 (2014), 195-210.
2. B. Jin, R. Lazarov, Z. Zhou, *An Analysis of the L1 Scheme for the Subdiffusion Equation with Nonsmooth Data*, IMA J. Numer. Anal., DOI: 10.1093/imanum/dru063.
3. C. Lubich, *Convolution Quadrature and Discretized Operational Calculus I*, Numer. Math. 52 (1988), 129-145.
4. A. Schädle, M. López-Fernández, C. Lubich, *Fast and Oblivious Convolution Quadrature*, SIAM J. Sci. Comput., Vol. 28, No. 2 (2006), pp. 421-438.
5. H. Brunner, D. Schötazau, *hp Discontinuous Galerkin Time-Stepping for Volterra Integrodifferential Equations*, SIAM J. Numer. Anal., Vol. 44, No. 1, pp. 224-245.
6. K. Mustapha, H. Brunner, H. Mustapha, D. Schötazau, *An hp-Version Discontinuous Galerkin Method for Integro-Differential Equations of Parabolic Type*, SIAM J. Numer. Anal., Vol. 49, No. 4, pp. 1369-1396.
7. D. Baffet, J. S. Hesthaven, *A Kernel Compression Scheme for Fractional Differential Equations*, SIAM J. Numer. Anal., (accepted) 2016.
8. J.R. Li, *A Fast Time Stepping Method for Evaluating Fractional Integrals* SIAM J. Sci. Comput., Vol. 31, No. 6 (2010), pp. 4696-4714.
9. C. Lubich, A. Schädle, *Fast Convolution for Nonreflecting Boundary Conditions*, SIAM J. Sci. Comput., Vol. 24, No. 1 (2002), pp. 161-182.
10. M. López-Fernández, C. Lubich, A. Schädle, *Adaptive Fast and Oblivious Convolution in Evolution Equations with Memory*, SIAM J. Sci. Comput., Vol. 30, No. 2 (2008), pp. 1015-1037.
11. G. Beylkin, L. Monzón, *Approximation by Exponential Sums Revisited*, Appl. Comput. Harmon. Anal. 28 (2010), 131-149.
12. D. Conte, I. Del Prete, *Fast Collocation Methods for Volterra Integral Equations of Convolution Type*, J. Comput. Appl. Math., 196 (2006), 652-663.

13. C. Lubich, *Runge-Kutta Theory for Volterra and Abel Integral Equations of the Second Kind*, Math. Comp., Vol. 41, No. 163 (1983), pp. 87-102.
14. A. Dutt, L. Greengard, V. Rokhlin, *Spectral Deferred Correction Methods for Ordinary Differential Equations*, BIT Vol. 40, No. 2, 241-266 (2000).
15. T. Hagstrom, R. Zhou, *On the Spectral Deferred Correction of Splitting Methods for Initial Value Problems*, Comm. App. Math. and Comp. Sci., Col. 1, No. 1, 2006.
16. A. Christlieb, B. Ong, J.M. Qiu, *Integral Deferred Correction Methods Constructed with High Order Runge-Kutta Integrators*, Math. Comp., Vol. 79, No. 270, 761-783 (2010).
17. A. Christlieb, B. Ong, J.M. Qiu, *Comments on High-Order Integrators Embedded Within Integral Deferred Correction Methods*, Comm. App. Math. and Comp. Sci., Vol. 4, No. 1, 2009.
18. S. Guarino, *Spectral Deferred Correction Methods for Differential Integral Equations*, Master Dissertation, EPFL, 2016.
19. K. Diethelm, N. J. Ford, *Analysis of Fractional Differential Equations*, J. Math. Anal. Appl., Vol. 265, 229-248 (2002).
20. B. Alpert, L. Greengard, T. Hagstrom, *Rapid Evaluation of Nonreflecting Boundary Kernels for Time-Domain Wave Propagation*, SIAM J. Numer. Anal., Vol. 37, No. 4, pp. 1138-1164.
21. M. López-Fernández, C. Palencia, A. Schädle, *A Spectral Order Method for Inverting Sectorial Laplace Transforms*, SIAM J. Numer. Anal., Vol. 44, No. 3 (2006), pp. 1332-1350.
22. R. Askey, J. Fitch, *Integral Representations for Jacobi Polynomials and Some Applications*, Journal of Mathematical Analysis and Applications, 26 (1969), 411-437.
23. C. A. Kennedy, M. H. Carpenter, *Additive Runge-Kutta Schemes for Convection-Diffusion-Reaction Equations*, Appl. Numer. Math., Vol. 44 (2003), Issue 1-2, pp. 139-181.
24. D. Baffet, *Kernel Compression Schemes for Fractional Differential Equations*, MATLAB Central File Exchange, 2017, file ID: 61024.
25. I. Podlubny, *Fractional Differential Equations*, Academic Press, San Diego, 1999.
26. R. Grrappa, *The Mittag-Leffler Function*, MATLAB Central File Exchange, 2014, file ID: 48154.
27. J. C. Butcher, *Numerical Methods for Ordinary Differential Equations*, Wiley, 2003.
28. A. Prothero, A. Robinson, *On the Stability and Accuracy of One-Step Methods for Solving Stiff Systems of Ordinary Differential Equations*, Math. Comp., Vol. 28, No. 125, 145-162 (1974).