Journal of Scientific Computing (2024) 99:48
https://doi.org/10.1007/s10915-024-02517-1

A Spectrally Accurate Step-by-Step Method for the Numerical Solution of Fractional Differential Equations

Luigi Brugnano[1] · **Kevin Burrage**[2] · **Pamela Burrage**[2] · **Felice Iavernaro**[3]

Received: 16 October 2023 / Revised: 25 January 2024 / Accepted: 13 March 2024

Abstract
In this paper we consider the numerical solution of fractional differential equations. In particular, we study a step-by-step procedure, defined over a graded mesh, which is based on a truncated expansion of the vector field along the orthonormal Jacobi polynomial basis. Under mild hypotheses, the proposed procedure is capable of getting spectral accuracy. A few numerical examples are reported to confirm the theoretical findings.

Keywords Fractional differential equations · Fractional integrals · Jacobi polynomials · Hamiltonian Boundary Value Methods · HBVMs · FHBVMs

Mathematics Subject Classification 65L05 · 65L03 · 65L99

1 Introduction

In recent years fractional differential equation modelling has become more and more frequent—see, for example, the two monographs [20, 37]. In fact the use of fractional models has quite a long history and early applications considered their behaviour in describing anomalous diffusion in a randomly hetergeneous porous medium [31] and in water resources modelling [6]. In mathematical biology, a number of researchers have considered fractional reaction diffusion equations—see, for example, Bueno-Orovio et al. [16] who have studied

✉ Luigi Brugnano
luigi.brugnano@unifi.it

Kevin Burrage
kevin.burrage@qut.edu.au

Pamela Burrage
pamela.burrage@qut.edu.au

Felice Iavernaro
felice.iavernaro@uniba.it

[1] Dipartimento di Matematica e Informatica "U. Dini", Università di Firenze, Firenze, Italy

[2] School of Mathematical Sciences, Queensland University of Technology, Brisbane, QLD, Australia

[3] Dipartimento di Matematica, Università di Bari, Bari, Italy

Published online: 08 April 2024

via a spectral approach the interfacial properties of the Allen-Cahn equation with a quartic double well potential, a model often used for alloy kinetics and the dynamics of phase boundaries. Henry, Langlands and Wearne [27] analysed Turing pattern formation in fractional activator-inhibitor systems. In a computational medicine setting, Henry and Langlands [26] developed fractional cable models for spiny neuronal dendrites. Orsingher and Behgin [36] have considered the dynamics of fractional chemical kinetics for unimolecular systems using time change. The authors in [17] have made a series of arguments based on Riesz potential theory and the wide range of heterogeneous cardiac tissues for the use of fractional models in cardiac electrophysiology, while Cusimano et al. [18] have explored the effects of fractional diffusion on the cardiac action potential shape and electrical propagation. In addition, Magin et al. [34], Hori et al. [28], Bueno-Orovio and Burrage [15] have considered the calibration of fractional cardiac models through a fractional Bloch–Torrey equation.

Due to these application advances there is clearly a need to develop advanced numerical methods for fractional differential equations. In a purely time fractional setting the standard approach has been to use a first order approximation to the Caputo derivative. Lubich [32] has extended this and developed higher order approximations based on an underlying linear multistep method that are convergent of the order of this multistep method.

There are two important issues when designing effective numerical methods for Fractional Differential equations: memory and non-locality. In the case of memory, in principle one needs to evaluate the numerical approximations all the way back to the initial time point at each time step and this becomes very computationally intensive over long time intervals. Techniques such as fixed time windowing have been proposed [37] and more recently a number of approaches have been considered to compute the discrete convolutions to approximate the fractional operator [39, 44]. Zeng et al. [43] developed fast memory-saving time stepping methods in which the fractional operator is decoupled into a local part with fixed memory length and the history part is computed by Laguerre–Gauss quadrature, whose error is independent of the stepsize.

In the second situation the solutions to FDEs are often non-smooth and may have a strong singularity at $t = 0$. In these cases the vector field may inherit this singularity behavior and so a constant stepsize would be very inefficient. Thus graded meshes have been proposed by a number of authors [30, 40, 42]. Lubich [33] on the other hand deals with singularities by introducing certain correction terms. Other approaches have been considered such as Adams methods [21], trapezoidal methods [23] and Bernstein splines [38] while Garrappa [24] has given a survey and software tutorial on numerical methods for FDEs.

In this paper, we consider a major improvement of the recent solution approach described in [2, 9], based on previous work on Hamiltonian Boundary value Methods (HBVMs) [7, 8, 11, 14] (also used as spectral methods in time [3, 5, 12, 13]), for solving fractional initial value problems in the form:

$$y^{(\alpha)}(t) = f(y(t)), \quad t \in [0, T], \quad y(0) = y_0 \in \mathbb{R}^m, \tag{1}$$

where, for sake of brevity, we omit t as a formal argument, at the r.h.s. For the same reason, hereafter the dimension m of the state vector will be also omitted, when unnecessary.

Here, for $\alpha \in (0, 1]$, $y^{(\alpha)}(t) \equiv D^\alpha y(t)$ is the Caputo fractional derivative:

$$D^\alpha g(t) = \frac{1}{\Gamma(1-\alpha)} \int_0^t (t-x)^{-\alpha} \left[\frac{\mathrm{d}}{\mathrm{d}x} g(x) \right] \mathrm{d}x. \tag{2}$$

The Riemann–Liouville integral associated to (2) is given by:

$$I^{\alpha} g(t) = \frac{1}{\Gamma(\alpha)} \int_0^t (t-x)^{\alpha-1} g(x) \mathrm{d}x. \tag{3}$$

It is known that [37]:

$$\begin{aligned} D^{\alpha} I^{\alpha} g(t) &= g(t), \quad I^{\alpha} D^{\alpha} g(t) = g(t) - g(0), \\ I^{\alpha} t^{j} &= \frac{j!}{\Gamma(\alpha + j + 1)} t^{j+\alpha}, \quad j = 0, 1, 2, \ldots. \end{aligned} \tag{4}$$

Consequently, the solution of (1) is formally given by:

$$y(t) = y_0 + I^{\alpha} f(y(t)), \quad t \in [0, T]. \tag{5}$$

We shall here generalize the approach given in [2], by defining a step-by-step procedure, whereas the procedure given in [2] was of a global nature. This strategy, when combined with a graded mesh selection, enables better handling of singularities in the derivative of the solution to (1) at the origin, which occur, for example, when f is a suitably regular function. Additionally, our approach partially mitigates issues stemming from the non-local nature of the problem. Indeed, at a given time-step of the integration procedure, it requires the solution of a differential problem within the current time interval along with a finite sequence of pure quadrature problems that account for the contribution brought by the history term.

With this premise, the structure of the paper is as follows: in Sect. 2 we recall some basic facts about Jacobi polynomials, for later use; in Sect. 3 we describe a *piecewise quasi-polynomial approximation* to the solution of (1); in Sect. 4 the analysis of the new method is carried out; in Sect 5 we report a few numerical tests for assessing the theoretical achievements; at last, in Sect. 6 a few concluding remarks are given.

2 Orthonormal Jacobi Polynomials

Jacobi polynomials form an orthogonal set of polynomials w.r.t. a given weighting function. In more detail, for $r, \nu > -1$:

$$\begin{aligned} &\bar{P}_i^{(r,\nu)}(x) \in \Pi_i, \quad \left(P_0^{(r,\nu)}(x) \equiv 1 \right) \\ &\int_{-1}^{1} (1-x)^r (1+x)^{\nu} \bar{P}_i^{(r,\nu)}(x) \bar{P}_j^{(r,\nu)}(x) \mathrm{d}x \\ &= \frac{2^{r+\nu+1}}{2i+r+\nu+1} \frac{\Gamma(i+r+1)\Gamma(i+\nu+1)}{\Gamma(i+r+\nu+1) i!} \delta_{ij}, \quad i, j = 0, 1, \ldots, \end{aligned}$$

where as is usual, Π_i is the vector space of polynomials of degree at most i and δ_{ij} is the Kronecker symbol. Consequently, the shifted and scaled Jacobi polynomials

$$P_i^{(r,\nu)}(c) := \sqrt{(2i+r+\nu+1) \frac{\Gamma(i+r+\nu+1) i!}{\Gamma(i+r+1)\Gamma(i+\nu+1)}} \bar{P}_i^{(r,\nu)}(2c-1), \quad i = 0, 1, \ldots,$$

are orthonormal w.r.t.

$$\int_0^1 (1-c)^r c^{\nu} P_i^{(r,\nu)}(c) P_j^{(r,\nu)}(c) \mathrm{d}c = \delta_{ij}, \quad i, j = 0, 1, \ldots.$$

In particular, hereafter we shall consider the polynomials [1]

$$P_i(c) := \frac{P_i^{(\alpha-1,0)}(c)}{\sqrt{\alpha}} \equiv \frac{\sqrt{2i+\alpha}}{\sqrt{\alpha}} \tilde{P}_i^{(\alpha-1,0)}(2c-1), \quad i = 0, 1, \ldots, \tag{6}$$

with $\alpha \in (0, 1]$ the same parameter in (1), such that:

$$P_0(c) \equiv 1, \quad \alpha \int_0^1 (1-c)^{\alpha-1} P_i(c) P_j(c) \mathrm{d}c = \delta_{ij}, \quad i, j = 0, 1, \ldots, \tag{7}$$

where we have slightly changed the weighting function, in order that it has a unit integral:

$$\alpha \int_0^1 (1-c)^{\alpha-1} \mathrm{d}c = 1. \tag{8}$$

We refer to Gautschi [25] and the accompanying software, for their computation, and for computing the nodes and weights of the Gauss–Jacobi quadrature of order $2k$.

Remark 1 As is clear, when in (6)–(8) $\alpha = 1$, we obtain the scaled and shifted Legendre polynomials, orthonormal in $[0, 1]$.

As is well known, the polynomials (6) form an orthonormal basis for the Hilbert space $L_2[0, 1]$, equipped with the scalar product

$$(f, g) = \alpha \int_0^1 (1-c)^{\alpha-1} f(c) g(c) \mathrm{d}c, \tag{9}$$

and the associated norm

$$\|f\| = \sqrt{(f, f)}. \tag{10}$$

Consequently, from the Parseval identity, it follows that any square summable function can be expanded along such a basis, and the corresponding expansion is convergent to the function itself.

That said, with reference to the vector field in (1), let us consider the interval $[0, h]$, and the expansion

$$f(y(ch)) = \sum_{j \geq 0} \gamma_j(y) P_j(c), \quad c \in [0, 1], \tag{11}$$

with the Fourier coefficients given by:

$$\gamma_j(y) = \alpha \int_0^1 (1-c)^{\alpha-1} P_j(c) f(y(ch)) \mathrm{d}c, \quad j = 0, 1, \ldots. \tag{12}$$

Consequently, on the interval $[0, h]$, (1) can be rewritten as

$$y^{(\alpha)}(ch) = \sum_{j \geq 0} \gamma_j(y) P_j(c), \quad c \in [0, 1], \quad y(0) = y_0, \tag{13}$$

and, by virtue of (3)–(4), one obtains:

$$y(ch) = y_0 + h^\alpha \sum_{j \geq 0} \gamma_j(y) I^\alpha P_j(c), \quad c \in [0, 1]. \tag{14}$$

In particular, due to (7) and (8), one has:

$$y(h) = y_0 + \frac{h^\alpha}{\Gamma(\alpha+1)} \gamma_0(y). \tag{15}$$

[1] Hereafter, for sake of brevity, we shall omit the upper indices for such polynomials.

Remark 2 By considering that $P_0(c) \equiv 1$, from (12) one derives that

$$\gamma_0(y) = \alpha \int_0^1 (1-c)^{\alpha-1} f(y(ch))\mathrm{d}c.$$

Consequently, because of (3), (15) becomes

$$y(h) = y_0 + I^\alpha f(y(h)),$$

i.e., (5) with $t = h$. This clearly explains the use of the Jacobi basis for the expansion (11).

Further, we observe that, when $\alpha = 1$, one retrieves the approach described in [11] for ODE-IVPs.

We also need the following preliminary results.

Lemma 1 *Let $G : [0, h] \to V$, with V a vector space, admit a Taylor expansion at $t = 0$. Then,*

$$\alpha \int_0^1 (1-c)^{\alpha-1} P_j(c) G(ch)\mathrm{d}c \;=\; O(h^j), \quad j = 0, 1, \ldots.$$

Proof By the hypothesis and (7), one has:

$$\begin{aligned}
&\alpha \int_0^1 (1-c)^{\alpha-1} P_j(c) G(ch)\mathrm{d}c \\
&= \alpha \int_0^1 (1-c)^{\alpha-1} P_j(c) \sum_{\ell\geq 0} \frac{G^{(\ell)}(0)}{\ell!}(ch)^\ell \mathrm{d}c \\
&= \alpha \sum_{\ell\geq 0} \frac{G^{(\ell)}(0)}{\ell!} h^\ell \int_0^1 (1-c)^{\alpha-1} P_j(c) c^\ell \mathrm{d}c \\
&= \alpha \sum_{\ell\geq j} \frac{G^{(\ell)}(0)}{\ell!} h^\ell \int_0^1 (1-c)^{\alpha-1} P_j(c) c^\ell \mathrm{d}c = O(h^j).
\end{aligned}$$

□

Remark 3 Concerning the above result, we observe that, by considering (9)–(10), from the Cauchy–Schwarz theorem one derives that

$$\left|\alpha \int_0^1 (1-c)^{\alpha-1} P_j(c) c^\ell \mathrm{d}c\right| = |(P_j, c^\ell)| \leq \|P_j\| \cdot \|c^\ell\| \leq 1, \quad \forall \ell \geq j \geq 0.$$

From Lemma 1, one has the following:

Corollary 1 *Assume that the r.h.s. of problem (1) admits a Taylor expansion at $t = 0$, then the coefficients defined at (12) satisfy:*

$$\gamma_j(y) = O(h^j).$$

The result of the previous lemma can be weakened as follows (the proof is similar and, therefore, omitted).

Lemma 2 *Let $G : [0, h] \to V$, with V a vector space, admit a Taylor polynomial expansion of degree k with remainder at $t = 0$. Then,*

$$\alpha \int_0^1 (1-c)^{\alpha-1} P_j(c) G(ch)\mathrm{d}c \;=\; O(h^{\min(j,k)}), \quad j = 0, 1, \ldots.$$

In such a case, the result of Corollary 1 is weakened accordingly.

Corollary 2 *Assume that the r.h.s. of problem (1) admits a Taylor polynomial expansion of degree k with remainder at $t = 0$. Then, the coefficients defined at (12) satisfy:*

$$\gamma_j(y) = O(h^{\min(j,k)}).$$

However, in general, at $t = 0$ the solution may be not regular, since the derivative may be singular (this is, indeed, the case, when f is a regular function). In such a case, we shall resort to the following result.

Lemma 3 *Assume that the r.h.s. in (1) is continuous for all $t \in [0, h]$. Then, for a convenient $\xi_t \in (0, t)^m$, one has[2]:*

$$y(t) = y_0 + \frac{y^{(\alpha)}(\xi_t)}{\Gamma(\alpha+1)} t^\alpha, \quad t \in [0, h].$$

Proof In fact, from (4), and by using the weighted mean-value theorem for integrals, one has:

$$\begin{aligned} y(t) = y_0 + I^\alpha y^{(\alpha)}(t) &= y_0 + \frac{1}{\Gamma(\alpha)} \int_0^t (t-x)^{\alpha-1} y^{(\alpha)}(x) \mathrm{d}x \\ &= y_0 + \frac{y^{(\alpha)}(\xi_t)}{\Gamma(\alpha)} \int_0^t (t-x)^{\alpha-1} \mathrm{d}x = y_0 + \frac{y^{(\alpha)}(\xi_t)}{\Gamma(\alpha+1)} t^\alpha. \end{aligned}$$

□

As a consequence, we derive the following weaker results concerning the Fourier coefficients (12).

Corollary 3 *In the hypotheses of Lemma 3, and assuming f is continuously differentiable in a neighborhood of y_0, one has:*

$$\gamma_0(y) = f(y_0) + O(h^\alpha), \quad \gamma_j(y) = O(h^\alpha), \quad j = 1, 2, \ldots.$$

Proof In fact, from Lemma 3, one has:

$$f(y(ch)) = f\left(y_0 + \frac{y^{(\alpha)}(\xi_{ch})}{\Gamma(\alpha+1)} (ch)^\alpha \right) = f(y_0) + f'(y_0) \frac{y^{(\alpha)}(\xi_{ch})}{\Gamma(\alpha+1)} (ch)^\alpha + o((ch)^\alpha).$$

From this relation, and from (12), the statement then follows. □

For later use, we also need to discuss the quadrature error in approximating the first s Fourier coefficients (12) by means of the Gauss–Jacobi formula of order $2k$, for a convenient $k \geq s$, whose abscissae are the zeros of P_k, as defined in (7), and with weights:

$$P_k(c_i) = 0, b_i = \alpha \int_0^1 (1-c)^{\alpha-1} \ell_i(c) \mathrm{d}c, \quad \ell_i(c) = \prod_{j=1,\, j\neq i}^{k} \frac{c - c_j}{c_i - c_j}, \quad i = 1, \ldots, k. \tag{16}$$

[2] Hereafter, $y^{(\alpha)}(\xi_t)$ means that each component of the function is evaluated at the corresponding entry of the argument.

That is,

$$\gamma_j(y) \;\approx\; \sum_{i=1}^{k} b_i P_j(c_j) f(y(c_i h)) \;=:\; \hat{\gamma}_j(y). \tag{17}$$

Concerning the quadrature errors

$$\Sigma_j^{\alpha}(y,h) \;:=\; \hat{\gamma}_j(y) - \gamma_j(y), \quad j = 0, \ldots, s-1, \tag{18}$$

the following result holds true.

Theorem 1 *Assume the function $G_{j,h}(c) := P_j(c) f(y(ch))$ be of class $C^{(2k)}[0,h]$. Then, with reference to (18), one has, for a suitable $\xi = (0,1)^m$:*

$$\Sigma_j^{\alpha}(y,h) \;=\; K_j \frac{G_{j,h}^{(2k)}(\xi)}{(2k)!} \;=\; O(h^{2k-j}), \quad j = 0, \ldots, s-1, \tag{19}$$

with the constants K_j independent of both $G_{j,h}$ and h.

Proof The statement follows by considering that the quadrature is exact for polynomials in Π_{2k-1}. □

However, as recalled above, when f is a smooth function, the derivative of the solution of problem (1) may have a singularity at $t = 0$. In such a case, estimates can be also derived (see, e.g., [19, 41] and [35, Theorem 5.1.8]). However, for our purposes, because of (7), for $j = 0, \ldots, s-1$, from Corollary 3 we can easily derive the following one, assuming that $k \geq s$ and (13) hold true:

$$\begin{aligned}
\hat{\gamma}_j(y) = \sum_{i=1}^{k} b_i P_j(c_i) f(y(c_i h)) \;&=\; \sum_{i=1}^{k} b_i P_j(c_i) \sum_{\ell \geq 0} P_\ell(c_i) \gamma_\ell(y) \\
&= \sum_{\ell=0}^{s-1} \underbrace{\left(\sum_{i=1}^{k} b_i P_j(c_i) P_\ell(c_i) \right)}_{=\delta_{j\ell}} \gamma_\ell(y) \;+\; O(h^{\alpha}) \\
&\equiv \gamma_j(y) + \Sigma_j(y,h), \quad j = 0, \ldots, s-1.
\end{aligned}$$

Consequently,

$$\Sigma_j(y,h) \;=\; O(h^{\alpha}), \quad j = 0, \ldots, s-1. \tag{20}$$

In order to derive a polynomial approximation to (13), it is enough to truncate the infinite series in (11) to a finite sum, thus obtaining:

$$\sigma^{(\alpha)}(ch) \;=\; \sum_{j=0}^{s-1} \gamma_j(\sigma) P_j(c), \quad c \in [0,1], \quad \sigma(0) = y_0, \tag{21}$$

with $\gamma_j(\sigma)$ formally given by (12) upon replacing y by σ. In so doing, (14) and (15) respectively become:

$$\sigma(ch) \;=\; y_0 + h^{\alpha} \sum_{j=0}^{s-1} \gamma_j(\sigma) I^{\alpha} P_j(c), \quad c \in [0,1], \tag{22}$$

and

$$\sigma(h) \;=\; y_0 + \frac{h^{\alpha}}{\Gamma(\alpha+1)} \gamma_0(\sigma). \tag{23}$$

It can be shown that Corollary 1, Corollary 2, Lemma 3, and Corollary 3 continue formally to hold for σ. Further, by considering the approximation of the Fourier coefficients obtained by using the Gauss–Jacobi quadrature of order $2k$,

$$\hat{\gamma}_j(\sigma) \;=\; \sum_{i=1}^{k} b_i P_j(c_j) f(\sigma(c_i h)) \;\equiv\; \gamma_j(\sigma) + \Sigma_j^{\alpha}(\sigma, h), \tag{24}$$

the result of Theorem 1 continues to hold. Moreover, we shall assume that the expansion

$$f(\sigma(ch)) = \sum_{j\geq 0} P_j(c)\gamma_j(\sigma), \quad c \in [0,1], \tag{25}$$

holds true, similarly as (11), from which also (20) follows for the quadrature error $\Sigma_j(\sigma, h)$, when σ has a singular derivative at 0. In the next sections, we shall better detail, generalize, and analyze this approach.

3 Piecewise Quasi-Polynomial Approximation

To begin with, in order to obtain a piecewise quasi-polynomial approximation to the solution of (1), we consider a partition of the integration interval in the form:

$$t_n = t_{n-1} + h_n, \quad n = 1, \ldots, N, \tag{26}$$

where

$$t_0 = 0, \quad \sum_{n=1}^{N} h_n = T. \tag{27}$$

Then, according to (21)–(23), on the first subinterval $[0, h_1]$ we can derive a polynomial approximation of degree $s-1$ to (11), thus getting the fractional initial value problem

$$\sigma_1^{(\alpha)}(ch_1) = \sum_{j=0}^{s-1} \gamma_j(\sigma_1) P_j(c), \quad c \in [0,1], \quad \sigma_1(0) = y_0, \tag{28}$$

where $\gamma_j(\sigma_1)$ is formally given by (12), upon replacing y by σ_1 at the right-hand side:

$$\gamma_j(\sigma_1) = \alpha \int_0^1 (1-c)^{\alpha-1} P_j(c) f(\sigma_1(ch_1))\mathrm{d}c, \quad j = 0, \ldots, s-1. \tag{29}$$

The solution of (28) is a *quasi-polynomial* of degree $s - 1 + \alpha$, formally given by:

$$\sigma_1(ch_1) = y_0 + h_1^{\alpha} \sum_{j=0}^{s-1} \gamma_j(\sigma_1) I^{\alpha} P_j(c), \quad c \in [0,1]. \tag{30}$$

However, in order to actually compute the Fourier coefficients $\{\gamma_j(\sigma_1)\}$, one needs to approximate them by using the Gauss–Jacobi quadrature (16)[3]:

$$\gamma_j^1 \;:=\; \hat{\gamma}_j(\sigma_1) \;=\; \sum_{i=1}^{k} b_i P_j(c_i) f(\sigma_1(c_i h_1)) \;\equiv\; \gamma_j(\sigma_1) + \Sigma_j^{\alpha}(\sigma_1, h_1).$$

[3] Hereafter, as a notational convention, $\gamma_j^i := \hat{\gamma}_j(\sigma_i)$, $i = 1, \ldots, N$.

In so doing, (30) now formally becomes

$$\sigma_1(ch_1) = y_0 + h_1^\alpha \sum_{j=0}^{s-1} \gamma_j^1 I^\alpha P_j(c), \quad c \in [0,1]. \tag{31}$$

Moreover, one solves the system of equations, *having (block) dimension s independently of k*:

$$\gamma_j^1 = \sum_{i=1}^{k} b_i P_j(c_i) f\left(y_0 + h^\alpha \sum_{\nu=0}^{s-1} \gamma_\nu^1 I^\alpha P_\nu(c_i) \right), \quad j = 0, \dots, s-1. \tag{32}$$

This kind of procedure is typical of HBVM(k,s) methods, in the case of ODE-IVPs [7]: the main difference, here, derives from the non-locality of the operator. As is clear [compare with (23)], the approximation at $t = h_1$ will be now given by:[4]

$$\bar{y}_1 := \sigma_1(h_1) = y_0 + \frac{h_1^\alpha}{\Gamma(\alpha+1)} \gamma_0^1. \tag{33}$$

Remark 4 For an efficient and stable evaluation of the integrals

$$I^\alpha P_0(c_i), \; I^\alpha P_1(c_i), \; \dots, \; I^\alpha P_{s-1}(c_i), \quad i = 1, \dots, k, \tag{34}$$

we refer to the procedure described in [4].

3.1 The General Procedure

We now generalize the previous procedure for the subsequent integration steps. For later use, we shall denote:

$$y_n(ch_n) := y(t_{n-1} + ch_n), \quad c \in [0,1], \quad n = 1, \dots, N, \tag{35}$$

the restriction of the solution on the interval $[t_{n-1}, t_n]$. Similarly, we shall denote by $\sigma(t) \approx y(t)$ the piecewise quasi-polynomial approximation such that:

$$\sigma|_{[t_{n-1},t_n]}(t_{n-1} + ch_n) =: \sigma_n(ch_n) \approx y_n(ch_n), \quad c \in [0,1], \quad n = 1, \dots, N. \tag{36}$$

Then, by using an induction argument, let us suppose one knows the quasi-polynomial approximations

$$\sigma_i(ch_i) = \phi_{i-1}^\alpha(c,\sigma) + h_i^\alpha \sum_{j=0}^{s-1} \gamma_j^i I^\alpha P_j(c) \approx y_i(ch_i), \quad c \in [0,1], \quad i = 1, \dots, n, \tag{37}$$

where $\phi_{i-1}^\alpha(c,\sigma)$ denotes a *history term*, to be defined later, such that:

- in the first subinterval, according to (31), $\phi_0^\alpha(c,\sigma) \equiv y_0$, $c \in [0,1]$;
- for $i > 1$, $\phi_{i-1}^\alpha(c,\sigma)$ only depends on $\sigma_1, \dots, \sigma_{i-1}$.

The corresponding approximations at the grid-points are given by

$$\bar{y}_i := \sigma_i(h_i) = \phi_{i-1}^\alpha(1,\sigma) + \frac{h_i^\alpha}{\Gamma(\alpha+1)} \gamma_0^i \approx y_i(h_i) \equiv y(t_i), \quad i = 1, \dots, n, \tag{38}$$

[4] Hereafter, we shall in general denote by $\bar{y}_n$ the approximation to $y(t_n)$, since $y_n(t)$ will be later used to denote the restriction of $y(t)$ to the subinterval $[t_{n-1}, t_n]$, $n = 1, \dots, N$.

and assume we want to compute

$$\sigma_{n+1}(ch_{n+1}) := \phi_n^{\alpha}(c,\sigma) + h_{n+1}^{\alpha}\sum_{j=0}^{s-1}\gamma_j^{n+1} I^{\alpha}P_j(c) \;\approx\; y_{n+1}(ch_{n+1}), \quad c\in[0,1]. \tag{39}$$

Hereafter, we shall assume that the time-steps in (26) define a *graded mesh*. In more detail, for a suitable $r > 1$:

$$h_n \;=\; rh_{n-1} \;\equiv\; r^{n-1}h_1, \quad n = 1,\dots,N. \tag{40}$$

Remark 5 As is clear, in the limit case where $r = 1$, (40) reduces to a uniform mesh with a constant time-step $h = T/N$.

In order to derive the approximation (39), we start computing the solution of the problem in the subinterval $[t_n, t_{n+1}]$. Then, for $t \equiv t_n + ch_{n+1}$, $c \in [0,1]$, one has:

$$\begin{aligned}
y(t) &\equiv y_{n+1}(ch_{n+1})\\
&= y_0 \;+\; \frac{1}{\Gamma(\alpha)}\int_0^{t_n+ch_{n+1}} (t_n+ch_{n+1}-x)^{\alpha-1} f(y(x))\mathrm{d}x\\
&= y_0 \;+\; \frac{1}{\Gamma(\alpha)}\int_0^{t_n} (t_n+ch_{n+1}-x)^{\alpha-1} f(y(x))\mathrm{d}x\\
&\quad + \frac{1}{\Gamma(\alpha)}\int_{t_n}^{t_n+ch_{n+1}} (t_n+ch_{n+1}-x)^{\alpha-1} f(y(x))\mathrm{d}x\\
&= y_0 \;+\; \frac{1}{\Gamma(\alpha)}\sum_{\nu=1}^{n}\int_{t_{\nu-1}}^{t_\nu} (t_n+ch_{n+1}-x)^{\alpha-1} f(y(x))\mathrm{d}x\\
&\quad + \frac{1}{\Gamma(\alpha)}\int_{t_n}^{t_n+ch_{n+1}} (t_n+ch_{n+1}-x)^{\alpha-1} f(y(x))\mathrm{d}x\\
&= y_0 \;+\; \frac{1}{\Gamma(\alpha)}\sum_{\nu=1}^{n}\int_{0}^{h_\nu} (t_n-t_{\nu-1}+ch_{n+1}-x)^{\alpha-1} f(y_\nu(x))\mathrm{d}x\\
&\quad + \frac{1}{\Gamma(\alpha)}\int_{0}^{ch_{n+1}} (ch_{n+1}-x)^{\alpha-1} f(y_{n+1}(x))\mathrm{d}x\\
&= y_0 \;+\; \frac{1}{\Gamma(\alpha)}\sum_{\nu=1}^{n} h_\nu^{\alpha}\int_0^1 \left(\frac{r^{n-\nu+1}-1}{r-1} + cr^{n-\nu+1} - \tau\right)^{\alpha-1} f(y_\nu(\tau h_\nu))\mathrm{d}\tau\\
&\quad + \frac{h_{n+1}^{\alpha}}{\Gamma(\alpha)}\int_0^c (c-\tau)^{\alpha-1} f(y_{n+1}(\tau h_{n+1}))\mathrm{d}\tau\\
&\equiv G_n^{\alpha}(c,y) \;+\; I^{\alpha} f(y_{n+1}(ch_{n+1})),
\end{aligned}$$

having set the *history term*

$$\begin{aligned}
&G_n^{\alpha}(c,y)\\
&:= y_0 \;+\; \frac{1}{\Gamma(\alpha)}\sum_{\nu=1}^{n} h_\nu^{\alpha}\int_0^1 \left(\frac{r^{n-\nu+1}-1}{r-1} + cr^{n-\nu+1} - \tau\right)^{\alpha-1} f(y_\nu(\tau h_\nu))\mathrm{d}\tau,
\end{aligned} \tag{41}$$

which is a known quantity, since it only depends on $y_1,\dots,y_n$ [see (35)]. Consequently, we have obtained

$$y_{n+1}(ch_{n+1}) = G_n^{\alpha}(c,y) + I^{\alpha} f(y_{n+1}(ch_{n+1})), \quad c\in[0,1], \tag{42}$$

which reduces to (5), when $n = 0$ and $t \in [0, h]$. Further, by considering the expansion

$$f(y_{n+1}(ch_{n+1})) = \sum_{j\geq 0} \gamma_j(y_{n+1}) P_j(c), \quad c \in [0,1], \tag{43}$$

with the Fourier coefficients given by

$$\gamma_j(y_{n+1}) = \alpha \int_0^1 (1-c)^{\alpha-1} P_j(c) f(y_{n+1}(ch_{n+1}))\mathrm{d}c, \quad j \geq 0, \tag{44}$$

one obtains that (42) can be rewritten as

$$y_{n+1}(ch_{n+1}) = G_n^\alpha(c, y) + \frac{h_{n+1}^\alpha}{\Gamma(\alpha)} \sum_{j\geq 0} \gamma_j(y_{n+1}) I^\alpha P_j(c), \quad c \in [0,1], \tag{45}$$

with the value at $t = h$ given by:

$$y(t_{n+1}) \equiv y_{n+1}(h_{n+1}) = G_n^\alpha(1, y) + \frac{h_{n+1}^\alpha}{\Gamma(\alpha+1)} \gamma_0(y_{n+1}). \tag{46}$$

The corresponding approximation is obtained by truncating the series in (42) after s terms, and approximating the corresponding Fourier coefficients via the Gauss–Jacobi quadrature of order $2k$,

$$\sigma_{n+1}(ch_{n+1}) = \phi_n^\alpha(c, \sigma) + \frac{h_{n+1}^\alpha}{\Gamma(\alpha)} \sum_{j=0}^{s-1} \gamma_j^{n+1} I^\alpha P_j(c), \quad c \in [0,1], \tag{47}$$

and

$$\bar{y}_{n+1} := \sigma_{n+1}(h_{n+1}) = \phi_n^\alpha(1, \sigma) + \frac{h_{n+1}^\alpha}{\Gamma(\alpha+1)} \gamma_0^{n+1}, \tag{48}$$

with $\phi_n^\alpha(c, \sigma)$ an approximation of $G_n^\alpha(c, y)$ in (41), defined as follows[5]:

$$\begin{aligned}
&\phi_n^\alpha(c, \sigma) \\
&:= y_0 + \frac{1}{\Gamma(\alpha)} \sum_{\nu=1}^{n} h_\nu^\alpha \int_0^1 \left(\frac{r^{n-\nu+1}-1}{r-1} + cr^{n-\nu+1} - \tau \right)^{\alpha-1} \sum_{j=0}^{s-1} P_j(\tau) \gamma_j^\nu \mathrm{d}\tau \\
&= y_0 + \frac{1}{\Gamma(\alpha)} \sum_{\nu=1}^{n} h_\nu^\alpha \sum_{j=0}^{s-1} \int_0^1 \left(\frac{r^{n-\nu+1}-1}{r-1} + cr^{n-\nu+1} - \tau \right)^{\alpha-1} P_j(\tau) \mathrm{d}\tau \gamma_j^\nu \\
&\equiv y_0 + \frac{1}{\Gamma(\alpha)} \sum_{\nu=1}^{n} h_\nu^\alpha \sum_{j=0}^{s-1} J_j^\alpha \left(\frac{r^{n-\nu+1}-1}{r-1} + cr^{n-\nu+1} \right) \gamma_j^\nu,
\end{aligned} \tag{49}$$

having set

$$J_j^\alpha(x) := \int_0^1 (x-\tau)^{\alpha-1} P_j(\tau)\mathrm{d}\tau, \quad j = 0, \ldots, s-1, \quad x > 1, \tag{50}$$

which, as we shall see in Sect. 3.3, can be accurately and efficiently computed.

[5] As anticipated above, from (49) it follows that $\phi_0^\alpha(c, \sigma) \equiv y_0$, $c \in [0,1]$, so that (47) reduces to (31), whereas $\phi_n^\alpha(c, \sigma)$, $n > 1$, only depends on $\sigma_1, \ldots, \sigma_n$.

As is clear, in (49) we have used the approximation

$$f(\sigma_\nu(\tau h)) \;=\; \sum_{j=0}^{s-1} P_j(\tau)\gamma_j^\nu \;+\; R_\nu^\alpha(\tau), \tag{51}$$

with the error term given by [see (17)–(18) and (25)]:

$$\begin{aligned} R_\nu^\alpha(\tau) &:= \sum_{j=0}^{s-1} P_j(\tau)\left[\gamma_j(\sigma_\nu) - \gamma_j^\nu\right] \;+\; \overbrace{\sum_{j\geq s} P_j(\tau)\gamma_j(\sigma_\nu)}^{=:\,E_\nu^\alpha(\tau)} \\ &\equiv \underbrace{\sum_{j=0}^{s-1} P_j(\tau)\Sigma_j^\alpha(\sigma_\nu,h_\nu)}_{=:\,S_\nu^\alpha(\tau)} \;+\; E_\nu^\alpha(\tau) \;\equiv\; S_\nu^\alpha(\tau) + E_\nu^\alpha(\tau). \end{aligned} \tag{52}$$

Because of the results of Sect. 2, we shall hereafter assume that

$$\left|\Sigma_j^\alpha(\sigma_\nu,h_\nu)\right| \leq \begin{cases} O(h_1^\alpha), & \nu=1,\\ O(h_\nu^{2k-j}), & \nu>1,\end{cases} \quad \left|E_\nu^\alpha(\tau)\right| \leq \begin{cases} O(h_1^\alpha), & \nu=1,\\ O(h_\nu^s), & \nu>1.\end{cases} \tag{53}$$

Consequently, considering that $k \geq s$, $\alpha \in (0,1)$, and $j = 0,\ldots,s-1$, one deduces:

$$\left|S_\nu^\alpha(\tau)\right| \leq \begin{cases} O(h_1^\alpha), & \nu=1,\\ O(h_\nu^{s+1}), & \nu>1,\end{cases} \quad \left|R_\nu^\alpha(\tau)\right| \leq \begin{cases} O(h_1^\alpha), & \nu=1,\\ O(h_\nu^s), & \nu>1.\end{cases} \tag{54}$$

In so doing, see (47), the Fourier coefficients satisfy the system of equations:

$$\gamma_j^{n+1} = \sum_{\ell=1}^{k} b_\ell P_j(c_\ell) f_{n+1}\left(\phi_n^\alpha(c_\ell,\sigma) + h_{n+1}^\alpha \sum_{i=0}^{s-1}\gamma_i^{n+1} I^\alpha P_i(c_\ell h)\right), \quad j=0,\ldots,s-1, \tag{55}$$

having (block)-size s, independently of the considered value of k.

Clearly,

$$\gamma_j^{n+1} \;=\; \gamma_j(\sigma_{n+1}) \;+\; \Sigma_j^\alpha(\sigma_{n+1},h_{n+1}), \quad j=0,\ldots,s-1,$$

with $\gamma_j(\sigma_{n+1})$ formally defined as in (44), upon replacing y_{n+1} with σ_{n+1}, and

$$\Sigma_j^\alpha(\sigma_{n+1},h_{n+1}) = O(h_{n+1}^{2k-j}), \quad j=0,\ldots,s-1,$$

the corresponding quadrature errors.

3.2 Solving the Discrete Problems

The discrete problem (55), to be solved at each integration step, can be better cast in vector form by introducing the matrices

$$\mathcal{P}_s = \begin{pmatrix} P_0(c_1) & \ldots & P_{s-1}(c_1)\\ \vdots & & \vdots \\ P_0(c_k) & \ldots & P_{s-1}(c_k)\end{pmatrix}, \quad \mathcal{I}_s^\alpha = \begin{pmatrix} I^\alpha P_0(c_1) & \ldots & I^\alpha P_{s-1}(c_1)\\ \vdots & & \vdots \\ I^\alpha P_c(c_k) & \ldots & I^\alpha P_{s-1}(c_k)\end{pmatrix} \in \mathbb{R}^{k\times s},$$

$$\Omega = \begin{pmatrix} b_1 & & \\ & \ddots & \\ & & b_k \end{pmatrix},$$

and the (block) vectors:

$$\boldsymbol{\gamma}^{n+1} = \begin{pmatrix} \gamma_0^{n+1} \\ \vdots \\ \gamma_{s-1}^{n+1} \end{pmatrix} \in \mathbb{R}^{sm}, \quad \boldsymbol{\phi}_n^\alpha = \begin{pmatrix} \phi_n^\alpha(c_1, \sigma) \\ \vdots \\ \phi_n^\alpha(c_k, \sigma) \end{pmatrix} \in \mathbb{R}^{km}.$$

In fact, in so doing (55) can be rewritten as[6]:

$$\boldsymbol{\gamma}^{n+1} = \mathcal{P}_s^\top \Omega \otimes I_m f\left(\boldsymbol{\phi}_n^\alpha + h_{n+1}^\alpha \mathcal{I}_s^\alpha \otimes I_m \boldsymbol{\gamma}^{n+1}\right), \tag{56}$$

This formulation is very similar to that used for HBVMs in the case $\alpha = 1$ [10]. The formulation (56) has a twofold use:

- it shows that, assuming, for example, f Lipschitz continuous, then the solution exists and is unique, for all sufficiently small timesteps h_{n+1};
- it induces a straightforward fixed-point iteration:

$$\begin{aligned} \boldsymbol{\gamma}^{n+1,0} &= \mathbf{0}, \\ \boldsymbol{\gamma}^{n+1,\ell} &= \mathcal{P}_s^\top \Omega \otimes I_m f\left(\boldsymbol{\phi}_n^\alpha + h_{n+1}^\alpha \mathcal{I}_s^\alpha \otimes I_m \boldsymbol{\gamma}^{n+1,\ell-1}\right), \quad \ell = 1, 2, \ldots, \end{aligned} \tag{57}$$

  which we shall use for the numerical tests.

Concerning the first point, the following result holds true.

Theorem 2 *Assume f be Lipschitz with constant L in in the interval $[t_n, t_{n+1}]$. Then, the iteration (57) is convergent for all timesteps h_{n+1} such that*

$$h_{n+1}^\alpha L \|\mathcal{P}_s^\top \Omega\| \|\mathcal{I}_s^\alpha\| < 1. \tag{58}$$

Proof In fact, one has:

$$\begin{aligned} &\|\boldsymbol{\gamma}^{n+1,\ell+1} - \boldsymbol{\gamma}^{n+1,\ell}\| \\ &= \|\mathcal{P}_s^\top \Omega \otimes I_m \left[f\left(\boldsymbol{\phi}_n^\alpha + h_{n+1}^\alpha \mathcal{I}_s^\alpha \otimes I_m \boldsymbol{\gamma}^{n+1,\ell}\right) - f\left(\boldsymbol{\phi}_n^\alpha + h_{n+1}^\alpha \mathcal{I}_s^\alpha \otimes I_m \boldsymbol{\gamma}^{n+1,\ell-1}\right)\right] \| \\ &\le h_{n+1}^\alpha L \|\mathcal{P}_s^\top \Omega\| \|\mathcal{I}_s^\alpha\| \cdot \|\boldsymbol{\gamma}^{n+1,\ell} - \boldsymbol{\gamma}^{n+1,\ell-1}\|, \end{aligned}$$

hence the iteration function defined at (57) is a contraction, when (58) holds true. □

A simplified Newton-type iteration, akin to that defined in [10] for HBVMs, will be the subject of future investigations.

Remark 6 We observe that the discrete problem (56) can be cast in a Runge-Kutta type form. In fact, the vector

$$Y^{n+1} := \boldsymbol{\phi}_n^\alpha + h_{n+1}^\alpha \mathcal{I}_s^\alpha \otimes I_m \boldsymbol{\gamma}^{n+1}, \tag{59}$$

in argument to the function f at the r.h.s. in (56), can be regarded as the stage vector of a Runge–Kutta method, tailored for the problem at hand. Substituting (56) into (59), and using Y^{n+1} as argument of f, then gives

$$Y^{n+1} = \phi_n^\alpha + h_{n+1}^\alpha \mathcal{I}_s^\alpha \mathcal{P}_s^\top \Omega \otimes I_m f(Y^{n+1}). \tag{60}$$

[6] As is usual, the function f, here evaluated in a (block) vector of dimension k, denotes the (block) vector made up by f evaluated in each (block) entry of the input argument.

It is worth noticing that (60) has (block) dimension k, instead of s. However, by considering that usually $k > s$ (see Sect. 5), it follows that solving (60) is less efficient than solving (56).

This fact is akin to what happens for a HBVM(k, s) method [7, 8, 11], which is the k-stage Runge–Kutta method obtained when $\alpha = 1$. In fact, for such a method, the discrete problem can also be cast in the form (56), then having (block) dimension s, independently of k (which is usually much larger than s).

From (59), one easily derives that the Butcher tableau of the Runge–Kutta type method is given by:

$$\begin{array}{c|c} \boldsymbol{c} & \mathcal{I}_s^\alpha \mathcal{P}_s^\top \Omega \\ \hline & \boldsymbol{b}^\top \end{array} \tag{61}$$

where $\boldsymbol{b}, \boldsymbol{c} \in \mathbb{R}^k$ are the vectors with the Gauss–Jacobi weights and abscissae, respectively. As anticipated above, when $\alpha = 1$, (61) becomes the Butcher tableau of a HBVM(k, s) method [7, 8, 11].

Because of what exposed in the previous remark, we give the following definition.

Definition 1 The Runge–Kutta type method defined by (61), with $\alpha \in (0, 1)$, will be referred to as a *Fractional HBVM*(k, s) *method* or, in short, *FHBVM*(k, s).

3.3 Computing the Integrals $J_j^\alpha(\boldsymbol{x})$

From (55) and (49), it follows that one needs to compute the integrals

$$J_j^\alpha\left(\frac{r^{n-\nu+1}-1}{r-1} + c_i r^{n-\nu+1}\right), \quad i = 1, \dots, k, \quad \nu = 1, \dots, n, \tag{62}$$

with $\{c_1, \dots, c_k\}$ the abscissae of the Gauss–Jacobi quadrature (16). As an example, in Fig. 1 we plot the Gauss–Jacobi abscissae for $\alpha = 0.5$ and $k = 5, 10, 15, 20, 25, 30$.

Further, there is numerical evidence that a sufficiently high-order Gauss–Legendre formula can compute the integrals (50) up to round-off, when $x \geq 1.5$. Namely,

$$J_j^\alpha\left(\frac{r^{n-\nu+1}-1}{r-1} + c_i r^{n-\nu+1}\right), \quad i = 1, \dots, k, \quad \nu = 1, \dots, n-1, \tag{63}$$

and

$$J_j^\alpha\left(1 + c_i r\right), \quad \text{s.t.} \quad c_i r \geq 0.5, \quad j = 0, \dots, s-1. \tag{64}$$

However, this is no more the case, when computing:

$$J_j^\alpha\left(1 + c_i r\right), \quad \text{s.t.} \quad c_i r < 0.5, \quad j = 0, \dots, s-1. \tag{65}$$

Consequently, there is the problem of accurately and efficiently computing these latter integrals.

Remark 7 We observe that the evaluation of the integrals (63)–(64), $j = 0, \dots, s-1$, via a $2p$-order Gauss–Legendre formula is inexpensive. As matter of fact, only the values $P_j(\xi_i)$ need to be computed, $j = 0, \dots, s-1$, with

$$\{\xi_1, \dots, \xi_p\} \tag{66}$$

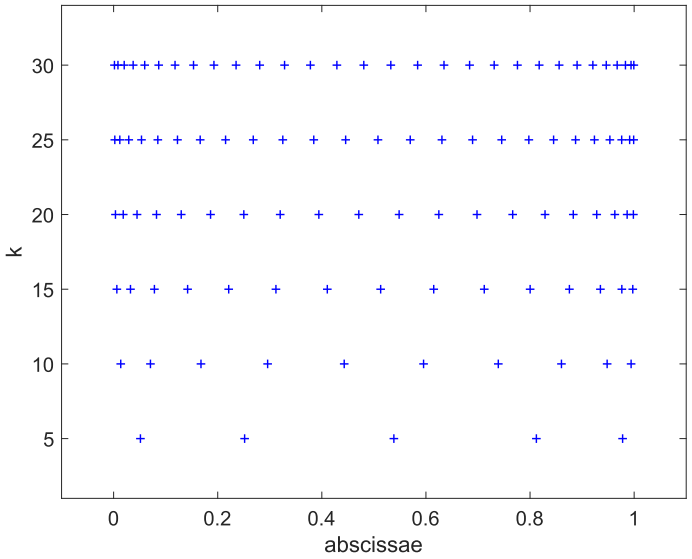

Fig. 1 Gauss–Jacobi abscissae for $\alpha = 0.5$

the Gauss–Legendre abscissae of the considered formula. Further, we observe that, with reference to (62), only the integrals

$$J_j^\alpha\left(\frac{r^n-1}{r-1}+c_i r^n\right),\quad j=0,\ldots,s-1,\quad i=1,\ldots,k,$$

need to be actually computed: as matter of fact, the remaining integrals,

$$J_j^\alpha\left(\frac{r^\nu-1}{r-1}+c_i r^\nu\right),\quad j=0,\ldots,s-1,\quad i=1,\ldots,k,\quad \nu=1,\ldots,n-1,$$

are inherited from the previous steps.

The starting point to derive an efficient algorithm for computing the integrals (65), is that the Jacobi polynomials (6) satisfy, as does any other family of orthogonal polynomials, a three-term recurrence:

$$\begin{aligned} &P_{j+1}(c)=(a_j c-b_j)P_j(c)-d_j P_{j-1}(c),\quad j=0,1,\ldots,\\ &P_0(c)\equiv 1,\quad P_{-1}(c)\equiv 0, \end{aligned} \tag{67}$$

with prescribed a_j, b_j, d_j, $j \geq 0$. In fact, by setting $\phi(c) = c$, and using the scalar product (9), to enforce (7), for $P_0, \ldots, P_{s-1}$, one obtains:

$$a_j=\frac{1}{(P_{j+1},\phi\cdot P_j)},b_j=\frac{(P_j,\phi\cdot P_j)}{(P_{j+1},\phi\cdot P_j)},\quad d_j=\frac{(P_j,\phi\cdot P_{j-1})}{(P_{j+1},\phi\cdot P_j)},\quad j=0,\ldots,s-2. \tag{68}$$

Consequently, by recalling the definition (50), from (67) one has:

$$J_0^\alpha(x)=\frac{x^\alpha-(x-1)^\alpha}{\alpha},\quad J_{-1}^\alpha(x)=0,\quad \alpha>0,\quad x>1,$$

and

$$\begin{aligned} J^{\alpha}_{j+1}(x) &= \int_0^1 (x-\tau)^{\alpha-1} P_{j+1}(\tau)\mathrm{d}\tau \\ &= a_j \int_0^1 (x-\tau)^{\alpha-1}\tau P_j(\tau)\mathrm{d}\tau - b_j J^{\alpha}_j(x) - d_j J^{\alpha}_{j-1}(x) \\ &= -a_j \int_0^1 (x-\tau)^{\alpha} P_j(\tau)\mathrm{d}\tau + [a_j x - b_j] J^{\alpha}_j(x) - d_j J^{\alpha}_{j-1}(x) \\ &= -a_j J^{\alpha+1}_j(x) + [a_j x - b_j] J^{\alpha}_j(x) - d_j J^{\alpha}_{j-1}(x), \quad j = 0, \ldots, j-2. \end{aligned}$$

From the last two formulae, one derives that `Jjalfa(a,b,d,alfa,1+ci*r)` computes all the integrals in (65) corresponding to the abscissa `ci`, with `Jjalfa` the Matlab© function listed in Table 1, and the vectors `a,b,d` containing the scalars (68). An implementation of the previous function, using the standard variable precision arithmetic (i.e., using `vpa` in Matlab©), allows handling values of s up to 20, at least.

4 Analysis of the Method

From (45) and (47), and with reference to (51)–(54), one derives:

$$\begin{aligned} & y_{n+1}(ch_{n+1}) - \sigma_{n+1}(ch_{n+1}) \\ &= \left[G^{\alpha}_n(c,y) - \phi^{\alpha}_n(c,\sigma)\right] + \frac{h^{\alpha}_{n+1}}{\Gamma(\alpha)} \sum_{j=0}^{s-1} \left[\gamma_j(y_{n+1}) - \gamma^{n+1}_j\right] I^{\alpha} P_j(c) \\ &\quad + \frac{h^{\alpha}_{n+1}}{\Gamma(\alpha)} \sum_{j\geq s} \gamma_j(y_{n+1}) I^{\alpha} P_j(c) \\ &= \frac{1}{\Gamma(\alpha)} \sum_{\nu=1}^{n} h^{\alpha}_{\nu} \int_0^1 \left(\frac{r^{n-\nu+1}-1}{r-1} + c r^{n-\nu+1} - \tau\right)^{\alpha-1} \\ &\quad \left[f(y_{\nu}(\tau h_{\nu})) - f(\sigma_{\nu}(\tau h_{\nu}) + R^{\alpha}_{\nu}(\tau)\right] \mathrm{d}\tau \\ &\quad + \frac{h^{\alpha}_{n+1}}{\Gamma(\alpha)} \sum_{j=0}^{s-1} \left[\gamma_j(y_{n+1}) - \gamma_j(\sigma_{n+1}) + \Sigma_j(\sigma_{n+1}, h_{n+1})\right] I^{\alpha} P_j(c) \\ &\quad + \frac{h^{\alpha}_{n+1}}{\Gamma(\alpha)} E^{\alpha}_{n+1}(c). \end{aligned}$$

Assuming f Lipschitz with constant L in a suitable neighborhood of the solution, and setting

$$\begin{aligned} R^{\alpha}_{\nu} &:= \max_{\tau\in[0,1]} |R^{\alpha}_{\nu}(\tau)|, \quad \nu = 1, \ldots, N-1, \\ E^{\alpha}_{n+1} &:= \max_{c\in[0,1]} |E^{\alpha}_{n+1}(c)|, \quad n = 1, \ldots, N-1, \\ e_n &:= \max_{c\in[0,1]} |y_n(ch_n) - \sigma_n(ch_n)|, \quad n = 1, 2, \ldots, N, \end{aligned}$$

Table 1 Matlab© function `Jjalfa`

```
function Ialfa = Jjalfa( a, b, d, alfa, x )
%
% Matlab function for computing the integrals J_j^alfa(x), j=0...s-1.
%
s1    = length(a); % s1 == s-1
Ialfa = zeros(s1+1,1);
valfa = alfa+(0:s1);
I1    = ( x.^valfa -(x-1).^valfa )./valfa;
Ialfa(1) = I1(1);
if s1>=1
  I2 = ( a(1)*x -b(1) )*I1(1:s1) -a(1)*I1(2:s1+1);
  Ialfa(2) = I2(1);
  for j = 2:s1
    I0 = I1; I1 = I2;
    I2 = ( a(j)*x -b(j) )*I1(1:s1-j+1) -a(j)*I1(2:s1-j+2) -d(j)*I0(1:s1-j+1);
    Ialfa(j+1) = I2(1);
  end
end
return
```

one then obtains that, for h_{n+1}^{α} sufficiently small, and a suitable constant $K_1 > 0$:

$$e_{n+1} \le \frac{K_1 L}{\Gamma(\alpha)} \sum_{\nu=1}^{n} h_\nu^\alpha \int_0^1 \left(\frac{r^{n-\nu+1}-1}{r-1} - \tau \right)^{\alpha-1} \mathrm{d}\tau \, e_\nu + \frac{K_1}{\Gamma(\alpha)} \sum_{\nu=1}^{n} h_\nu^\alpha \int_0^1 \left(\frac{r^{n-\nu+1}-1}{r-1} - \tau \right)^{\alpha-1} \mathrm{d}\tau \, R_\nu^\alpha + g_{n+1}^\alpha, \quad n = 1, \dots, N-1,$$

with

$$g_{n+1}^{\alpha} = O(h_{n+1}^{s+\alpha}). \tag{69}$$

Considering that

$$\frac{1}{\Gamma(\alpha)} \int_0^1 \left(\frac{r^{n-\nu+1}-1}{r-1} - \tau \right)^{\alpha-1} \mathrm{d}\tau = \frac{1}{\Gamma(\alpha+1)} \left[\left(\frac{r^{n-\nu+1}-1}{r-1} \right)^{\alpha} - \left(\frac{r^{n-\nu+1}-r}{r-1} \right)^{\alpha} \right] \le \frac{1}{\Gamma(\alpha+1)}, \quad \nu = 1, \dots, n, \tag{70}$$

and [see (54)]

$$h_1^\alpha R_1^\alpha \le O(h_1^{2\alpha}), \quad h_\nu^\alpha R_\nu^\alpha \le O(h_\nu^{s+\alpha}), \quad \nu > 1,$$

so that [see (40)], for a suitable constant $K_2 > 0$,

$$\begin{aligned} \frac{K_1}{\Gamma(\alpha+1)} \sum_{\nu=1}^{n} h_\nu^\alpha R_\nu^\alpha &\le K_2 \left(h_1^{2\alpha} + \sum_{\nu=2}^{n} h_\nu^{s+\alpha} \right) \\ &= K_2 \left(h_1^{2\alpha} + h_n^{s+\alpha} \sum_{\nu=0}^{n-2} (r^{s+\alpha})^{-\nu} \right) \\ &\le K_2 \left(h_1^{2\alpha} + \frac{h_n^{s+\alpha}}{1 - r^{-(s+\alpha)}} \right), \end{aligned}$$

one eventually obtains:

$$e_{n+1} \le \frac{K_1 L}{\Gamma(\alpha)} \sum_{\nu=1}^{n} h_\nu^\alpha \int_0^1 \left(\frac{r^{n-\nu+1}-1}{r-1} - \tau \right)^{\alpha-1} \mathrm{d}\tau \, e_\nu + \psi_n^\alpha, \quad n = 1, \dots, N-1,$$

with [see (40) and (69)]

$$\psi_n^\alpha := K_2 \left(h_1^{2\alpha} + \frac{h_n^{s+\alpha}}{1 - r^{-(s+\alpha)}} \right) + g_{n+1}^\alpha = O(h_1^{2\alpha} + h_{n+1}^{s+\alpha}), \quad n = 1, \dots, N-1.$$

At last, by considering that, for $n = 1, \dots, N-1$,

$$\begin{aligned} &\frac{1}{\Gamma(\alpha)} \sum_{\nu=1}^{n} h_\nu^\alpha \int_0^1 \left(\frac{r^{n-\nu+1}-1}{r-1} - \tau \right)^{\alpha-1} \mathrm{d}\tau \\ &= \frac{1}{\Gamma(\alpha+1)} \sum_{\nu=1}^{n} \left[h_\nu^\alpha \left(\frac{r^{n-\nu+1}-1}{r-1} \right)^{\alpha} - h_{\nu+1}^\alpha \left(\frac{r^{n-\nu}-1}{r-1} \right)^{\alpha} \right] \\ &= \frac{1}{\Gamma(\alpha+1)} \left[\left(h_1 \frac{r^n-1}{r-1} \right)^{\alpha} - h_{n+1}^\alpha \right] \le \frac{T^\alpha}{\Gamma(\alpha+1)}, \end{aligned} \tag{71}$$

setting

$$\rho = \exp\left(\frac{K_1 L T^\alpha}{\Gamma(\alpha+1)}\right),$$

and considering that $e_1 \le O(h_1^{2\alpha})$, from the Gronwall lemma (see, e.g., [29]) one eventually obtains, for a suitable $K > 0$:

$$\begin{aligned} e_{n+1} &\le K\rho\left(h_1^{2\alpha} + \sum_{\nu=1}^{n} \psi_n^\alpha\right) = K\rho\left((n+1)h_1^{2\alpha} + \frac{h_{n+1}^{s+\alpha}}{1-r^{-(s+\alpha)}}\right) \\ &\le K\rho\left(Nh_1^{2\alpha} + \frac{h_N^{s+\alpha}}{1-r^{-(s+\alpha)}}\right), \quad n = 1, \ldots, N-1. \end{aligned} \tag{72}$$

Considering that [see (27) and (40)] $N = \log_r(1 + T(r-1)/h_1)$, the error is optimal when $h_1^{2\alpha} \sim h_N^{s+\alpha}$.

Remark 8 In the case where the vector field of problem (1) is suitably smooth at $t = 0$, so that a constant timestep $h = T/N$ can be used, the estimate

$$e_n \le O(nh^{s+\alpha}), \quad n = 1, \ldots, N, \tag{73}$$

can be derived for the error, by using similar arguments as above.

Remark 9 From (72) [or (73)], one deduces that, for a given prescribed accuracy, it is possible to decrease the value of the number N of the time steps, by increasing the degree s of the polynomial approximating the vector field. By considering that, as anticipated in the introduction, one main difficulty in solving the problem (1) stems from the evaluation of the memory terms, reducing N has a welcome impact on the overall complexity of the method.

5 Numerical Tests

We here report a few numerical tests to illustrate the theoretical findings. For all tests, when not otherwise specified, we use $k = 30$ for the Gauss–Jacobi quadrature (16), and $p = 30$ for the Gauss–Legendre quadrature formula (66). Also, the following values of s will be considered:

$$s \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20\}.$$

All the numerical tests have been performed in Matlab© (Rel. 2023b) on a Silicon M2 laptop with 16GB of shared memory. As anticipated, we use the fixed-point iteration (57) to solve the generated discrete problems (56). The iteration is carried out until full machine accuracy is gained.

5.1 Example 1

The first problem, taken from Garrappa [24], is given by:

$$y^{(0.6)} = -10y, \quad t \in [0, 5], \quad y(0) = 1, \tag{74}$$

Table 2 Maximum error for Problem (74), $r = 1.01$ and $k = 30$

h_1	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}
s \ N	626	857	1088	1320	1551	1783
1	***	***	***	***	***	***
2	3.73e−06	2.43e−07	5.40e−08	5.38e−08	5.37e−08	5.37e−08
3	5.81e−07	3.78e−08	2.40e−09	1.52e−10	4.64e−11	4.64e−11
4	1.47e−07	9.60e−09	6.11e−10	3.86e−11	2.44e−12	1.56e−13
5	5.29e−08	3.46e−09	2.20e−10	1.39e−11	8.79e−13	5.37e−14
6	2.04e−08	1.33e−09	8.49e−11	5.37e−12	3.41e−13	2.32e−14
7	1.19e−08	7.81e−10	4.98e−11	3.15e−12	1.97e−13	1.08e−14
8	4.26e−09	2.77e−10	1.76e−11	1.11e−12	7.18e−14	7.91e−15
9	4.56e−09	3.03e−10	1.94e−11	1.22e−12	7.57e−14	7.91e−15
10	1.89e−09	1.22e−10	7.79e−12	4.90e−13	2.96e−14	7.91e−15
20	1.84e−09	1.22e−10	7.77e−12	4.90e−13	2.96e−14	7.91e−15

whose solution is given by the following Mittag–Leffler function[7]:

$$E_{0.6}(-10t^{0.6}) = \sum_{j\geq 0} \frac{(-10t^{0.6})^j}{\Gamma(0.6j+1)}.$$

In Table 2, we list the obtained results, in terms of maximum error, when using different values of s and h_1, with h_N fixed, in order to ascertain the contribution of the first term of the error in the bound (72). In order to satisfy the requirements:

$$t_N = h_1 \frac{r^N - 1}{r - 1} \approx 5, \quad h_N \approx 0.05,$$

for all combinations of h_1 and N, we use the value $r = 1.01$ in (40), and the number of timesteps N is chosen in order that t_N is the closest point to the end of the integration interval. The ***, in the line corresponding to $s = 1$, means that the solution is not properly evaluated: this is due to the failure of the fixed-point iteration (57) in the last integration steps (the same notation will be used in the subsequent tables). As is expected from (72), the error decreases as s increases and the initial timestep h_1 decreases. Further, having a large value of s is not effective, if h_1 is not suitably small, and vice versa (again from the bound (72)). Remarkably, by choosing s large enough and h_1 suitably small, full machine accuracy is gained (cf. the last 4 entries in the last column of the table, having the same error).

In Table 3 we list the results obtained by using $k = s$, which is the minimum value allowed for k. In such a case, the accuracy is generally slightly worse, due to the fact that the quadrature error is of the same order as the truncation error. It is, however, enough choosing k only slightly larger than s, to achieve a comparable accuracy, as is shown in Table 4 for $k = s + 5$. Nevertheless, it must be stressed that choosing larger values of k is not an issue, since the discrete problem (56), to be solved at each integration step, has (block) size s, independently of k.

[7] We refer to [22] and the accompanying software, for its efficient Matlab© implementation.

Table 3 Maximum error for Problem (74), $r = 1.01$ and $k = s$

h_1	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}
s \ N	626	857	1088	1320	1551	1783
1	***	***	***	***	***	***
2	7.52e−06	4.99e−07	5.78e−08	5.76e−08	5.76e−08	5.76e−08
3	2.27e−06	1.51e−07	9.68e−09	6.13e−10	4.55e−11	4.55e−11
4	9.67e−07	6.41e−08	4.09e−09	2.59e−10	1.64e−11	1.03e−12
5	4.91e−07	3.26e−08	2.08e−09	1.32e−10	8.33e−12	5.23e−13
6	2.82e−07	1.87e−08	1.19e−09	7.56e−11	4.77e−12	2.99e−13
7	1.76e−07	1.16e−08	7.44e−10	4.71e−11	2.97e−12	1.85e−13
8	1.16e−07	7.71e−09	4.92e−10	3.12e−11	1.97e−12	1.22e−13
9	8.07e−08	5.35e−09	3.42e−10	2.16e−11	1.36e−12	8.42e−14
10	5.82e−08	3.86e−09	2.46e−10	1.56e−11	9.83e−13	6.01e−14
20	6.63e−09	4.39e−10	2.80e−11	1.77e−12	1.11e−13	7.80e−15

Table 4 Maximum error for Problem (74), $r = 1.01$ and $k = s + 5$

h_1	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}
s \ N	626	857	1088	1320	1551	1783
1	***	***	***	***	***	***
2	3.58e−06	2.33e−07	5.40e−08	5.38e−08	5.38e−08	5.37e−08
3	6.56e−07	4.29e−08	2.73e−09	1.73e−10	4.63e−11	4.63e−11
4	8.57e−08	5.43e−09	3.43e−10	2.16e−11	1.36e−12	8.76e−14
5	9.20e−08	6.12e−09	3.92e−10	2.48e−11	1.56e−12	9.70e−14
6	4.38e−08	2.87e−09	1.83e−10	1.16e−11	7.30e−13	4.41e−14
7	3.39e−08	2.28e−09	1.46e−10	9.25e−12	5.83e−13	3.51e−14
8	2.58e−08	1.70e−09	1.09e−10	6.87e−12	4.32e−13	2.53e−14
9	2.03e−08	1.35e−09	8.60e−11	5.44e−12	3.43e−13	1.97e−14
10	1.64e−08	1.09e−09	6.93e−11	4.38e−12	2.76e−13	1.54e−14
20	3.28e−09	2.17e−10	1.38e−11	8.74e−13	5.40e−14	8.19e−15

5.2 Example 2

The next problem that we consider is from Diethelm et al. [21] (see also [24]):

$$y^{(0.5)} = -|y|^{1.5} + \frac{40320}{\Gamma(8.5)}t^{7.5} - 3\frac{\Gamma(5.25)}{\Gamma(4.75)}t^{3.75} + \left(\frac{3}{2}t^{0.25} - t^4\right)^3 + \frac{9}{4}\Gamma(1.5),$$
$$t \in [0, 1], \quad y(0) = 0, \tag{75}$$

whose solution is

$$y(t) = t^8 - 3\,t^{4.25} + \frac{9}{4}\,t^{0.5}. \tag{76}$$

According to Garrappa [24], "this problem is surely of interest because, unlike several other problems often proposed in the literature, it does not present an artificial smooth solution, which is indeed not realistic in most of the fractional-order applications." Despite this, a

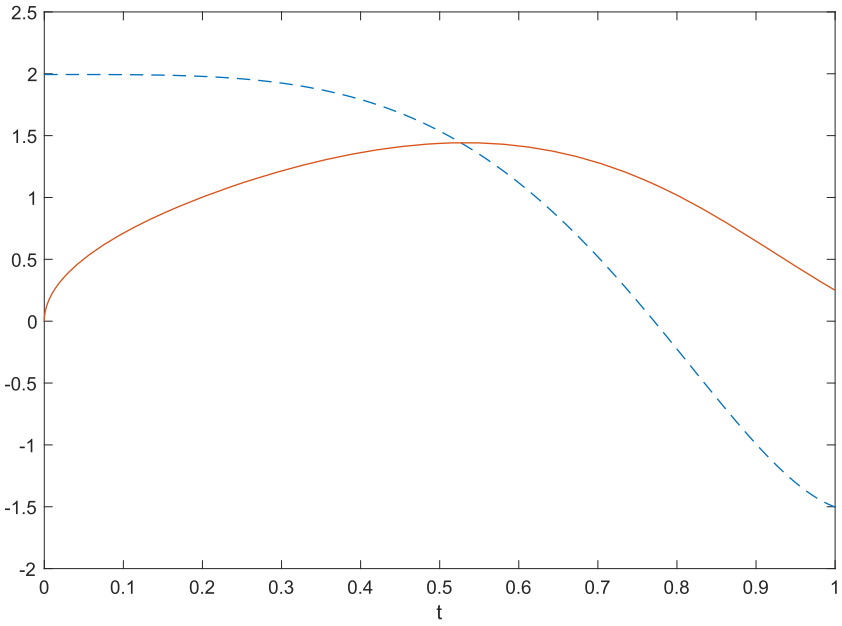

Fig. 2 solution (continuous line) and vector field (dashed line) for problem (75)

Table 5 Maximum error for Problem (75), constant timestep $h = 1/N$

s \ N	2	4	8	16	32
1	9.22e−01	5.65e−02	1.28e−02	1.35e−02	9.12e−03
2	7.48e−03	2.68e−03	5.15e−04	8.02e−05	1.91e−05
3	2.02e−03	1.96e−04	1.23e−05	2.04e−06	5.07e−07
4	2.29e−04	8.42e−06	2.72e−07	3.55e−08	3.70e−09
5	1.63e−05	3.52e−07	4.43e−09	3.44e−10	1.62e−11
6	7.61e−07	9.80e−09	6.57e−11	2.26e−12	1.47e−13
7	4.11e−08	3.71e−10	9.02e−12	3.46e−13	2.18e−14
8	1.24e−09	6.02e−11	1.87e−12	6.54e−14	4.22e−15
9	4.56e−10	1.44e−11	4.27e−13	1.65e−14	1.11e−15
10	1.40e−10	4.40e−12	1.33e−13	4.77e−15	8.88e−16
20	4.93e−14	1.33e−15	6.66e−16	6.66e−16	8.88e−16

constant timestep $h = 1/N$ turns out to be appropriate since, unlike the solution (76), the vector field (75) is very smooth at the origin, as one may see in Fig. 2. In Table 5 we list the maximum error by using different values of N: as is clear, by using $s \geq 8$, only 32 steps are needed to gain full machine accuracy for the computed solution. Further, in Fig. 3 is the *work-precision diagram* (i.e., execution time[8] vs. computed accuracy) for the following methods, used on a uniform grid with $N + 1$ points, with $N = 2^\nu$, as below specified:

– FHBVM(30,1), $\nu = 5, 6, \ldots, 12$;

[8] The execution time is measured in seconds.

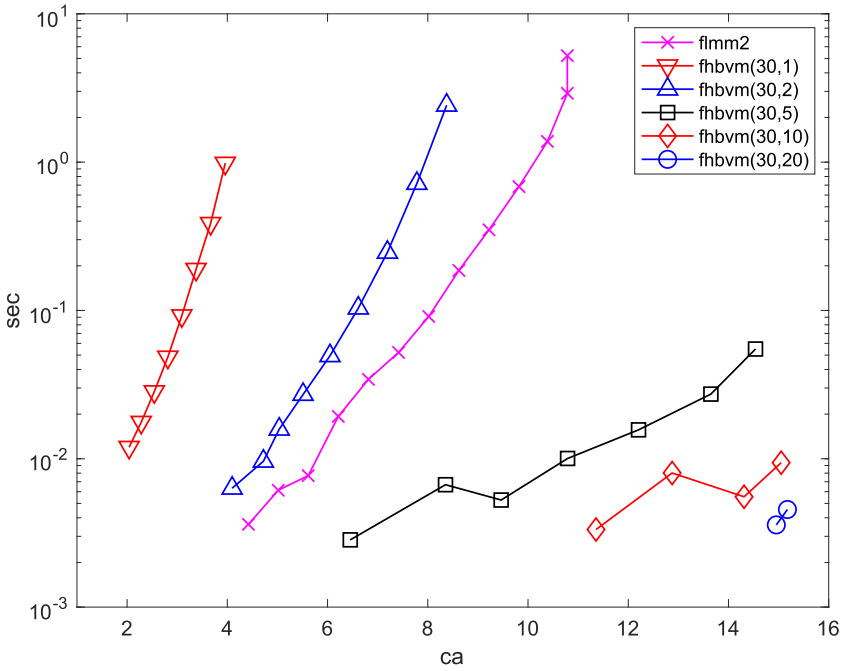

Fig. 3 Work-precision diagram for Problem (75)

- FHBVM(30,2), $\nu = 4, 5, \ldots, 12$;
- FHBVM(30,5), $\nu = 2, 3, \ldots, 8$;
- HFBVM(30,10), $\nu = 2, 3, 4, 5$;
- FHBVM(30,20), $\nu = 2, 3$;
- the BDF-2 method implemented in the Matlab code `flmm2` [23],[9] $\nu = 6, \ldots, 20$.

The rationale for this choice is to use FHBVMs both as spectral methods and not, with a further comparison with a state-of-art numerical code. In the plots, the *computed accuracy* (`ca`) is measured as follows: if e_i is the absolute error at the i-th grid point, then

$$\texttt{ca} = -\log_{10} \max_{i=1,\ldots,N} |e_i|.$$

From the obtained results, one concludes that:

- low-order FHBVMs are less competitive than high-order ones;
- high-order FHBVMs require less time steps and are, therefore, able to obtain a fully accurate solution;
- when used as spectrally accurate methods in time, FHBVMs are very effective, and competitive with existing methods.

[9] The code is called with parameters `tol=1e−15` and `itmax=1000`.

Table 6 Maximum error for Problems (77) and (79), $r = 1.2$ and $h_1 = 10^{-11}$

s	Error (77)	Error (79)
1	3.25e−02	***
2	8.86e−05	5.13e−04
3	8.36e−07	4.21e−06
4	1.41e−08	7.55e−08
5	3.03e−10	1.63e−09
6	7.54e−12	3.95e−11
7	3.46e−13	1.06e−12
8	2.09e−13	2.09e−13
9	2.09e−13	2.09e−13
10	2.09e−13	2.09e−13
20	2.09e−13	2.09e−13

5.3 Example 3

We now consider the following problem taken from Satmari [38],

$$y^{(1/3)} = \frac{t}{10}\left[y^3 - (t^{2/3}+1)^3\right] + \frac{\Gamma(5/3)}{\Gamma(4/3)}t^{1/3}, \quad t \in [0,1], \quad y(0) = 1, \tag{77}$$

whose solution is $y(t) = t^{2/3} + 1$. We solve it by using $h_1 = 10^{-11}$ and $r = 1.2$. In such a case, $N = 130$ timesteps are needed to cover approximately the integration interval, with $h_N \leq 0.2$. The obtained results, by considering increasing values of s, are listed in Table 6. Also in this case, we obtain full accuracy starting from $s = 8$.

Remark 10 Concerning the choice of the parameters h_1, r, and N for the graded mesh, one has to consider that, in principle [see (72)]:

$$T \equiv t_N = h_1 \frac{r^N - 1}{r - 1}, \quad h_N = h_1 r^{N-1}, \quad h_1^{2\alpha} \sim h_N^{s+\alpha}.$$

It is then important to properly choose h_1, and arrange the other parameters accordingly. In such a case, one may reduce the number N of time steps by increasing s, as observed in Remark 9. The optimal choice of such parameters, however, deserves to be further investigated.

5.4 Example 4

Next, we consider the following problem, again taken from [38],

$$y^{(1/3)} = \frac{1}{3}\left(y^3 - t^4\right) + \Gamma(7/3)t, \quad t \in [0,1], \quad y(0) = 0, \tag{78}$$

whose solution is $y(t) = t^{4/3}$. We solve this problem by using a constant timestep $h = 1/N$: in fact, the vector field can be seen to be a polynomial of degree 1. The obtained results are listed in Table 7: an order 1 convergence can be observed for $s = 1$ [which is consistent with (73)], whereas full machine accuracy is obtained for $s \geq 2$, due to the fact that, as anticipated above, the vector field of problem (78) is a polynomial of degree 1 in t and, consequently, (13) and (21) coincide, for all $s \geq 2$.

Table 7 Maximum error for Problem (78), constant timestep $h = 1/N$

s \ N	2	4	8	16	32	64
1	***	1.56e−01	7.01e−02	3.59e−02	1.87e−02	9.75e−03
2	8.88e−16	1.33e−15	8.88e−16	8.88e−16	8.88e−16	8.88e−16
3	4.44e−16	6.66e−16	4.44e−16	4.44e−16	3.33e−16	4.44e−16
4	6.66e−16	6.66e−16	5.55e−16	2.22e−16	3.33e−16	5.55e−16
5	9.99e−16	9.99e−16	6.66e−16	5.55e−16	2.22e−16	5.55e−16
6	1.33e−15	7.77e−16	8.88e−16	6.66e−16	3.33e−16	5.55e−16
7	1.33e−15	8.88e−16	7.77e−16	4.44e−16	4.44e−16	7.77e−16
8	1.78e−15	1.11e−15	7.77e−16	6.66e−16	4.44e−16	6.66e−16
9	2.00e−15	1.22e−15	8.88e−16	8.88e−16	4.44e−16	6.66e−16
10	2.00e−15	1.22e−15	8.88e−16	8.88e−16	4.44e−16	7.77e−16
20	2.78e−15	1.89e−15	1.44e−15	1.11e−15	6.66e−16	8.88e−16

5.5 Example 5

Finally, we reformulate the two problems (77) and (78) as a system of two equations, having the same solutions as above, as follows:

$$\begin{aligned} y_1^{(1/3)} &= \frac{t}{10}\left[y_1^3 - (y_2^{1/2}+1)^3\right] + \frac{\Gamma(5/3)}{\Gamma(4/3)}t^{1/3}, \quad y_1(0) = 1, \\ y_2^{(1/3)} &= \frac{1}{3}\left(y_2^3 - (y_1-1)^6\right) + \Gamma(7/3)t, \qquad y_2(0) = 0, \quad t \in [0,1]. \end{aligned} \tag{79}$$

We solve Problem (79) by using the same parameters used for Problem (77): $h_1 = 10^{-11}$, $r = 1.2$, and 130 timesteps. The obtained results are again listed in Table 6: it turns out that they are similar to those obtained for Problem (77) and, also in this case, we obtain full accuracy starting from $s = 8$.

6 Conclusions

In this paper we have devised a novel step-by-step procedure for solving fractional differential equations. The procedure, which generalizes that given in [2], relies on the expansion of the vector field along a suitable orthonormal basis, here chosen as the shifted and orthonormal Jacobi polynomial basis. The analysis of the method has been given, along with its implementation details. These latter details show that the method can be very efficiently implemented. A few numerical tests confirm the theoretical findings.

It is worth mentioning that systems with FDEs of different orders can be also solved by slightly adapting the previous framework: as matter of fact, it suffices using different Jacobi polynomials, corresponding to the different orders of the FDEs. This, in turn, allows easily managing fractional differential equations of order $\alpha > 1$, by casting them as a system of $\lfloor\alpha\rfloor$ ODEs, coupled with a fractional equation of order $\beta := \alpha - \lfloor\alpha\rfloor \in (0,1)$.

As anticipated in Sect. 3.2, a future direction of investigation will concern the efficient numerical solution of the generated discrete problems, which is crucial when using larger stepsizes. Also the optimal choice of the parameters for the methods will be further investigated, in particular those for generating the graded mesh, as well as the possibility of

adaptively defining it. Further, the parallel implementation of the methods could be also considered, following an approach similar to [1].

Acknowledgements The authors wish to thank two anonymous referees for their useful comments. L. Brugnano and F. Iavernaro are members of GNCS-INdAM.

Funding Open access funding provided by Università degli Studi di Firenze within the CRUI-CARE Agreement. No funding was received for conducting this study.

Data Availability All data reported in the manuscript have been obtained by a Matlab© implementation of the methods presented. They can be made available on request.

Declarations

Conflict of interest The authors declare no Conflict of interest, nor Conflict of interest.

References

1. Amodio, P., Brugnano, L.: Parallel implementation of block boundary value methods for ODEs. J. Comput. Appl. Math. **78**, 197–211 (1997). https://doi.org/10.1016/S0377-0427(96)00112-4
2. Amodio, P., Brugnano, L., Iavernaro, F.: Spectrally accurate solutions of nonlinear fractional initial value problems. AIP Confer. Proc. **2116**, 140005 (2019). https://doi.org/10.1063/1.5114132
3. Amodio, P., Brugnano, L., Iavernaro, F.: Analysis of Spectral Hamiltonian Boundary Value Methods (SHBVMs) for the numerical solution of ODE problems. Numer. Algorithms **83**, 1489–1508 (2020). https://doi.org/10.1007/s11075-019-00733-7
4. Amodio, P., Brugnano, L., Iavernaro, F.: A note on a stable algorithm for computing the fractional integrals of orthogonal polynomials. Appl. Math. Lett. **134**, 108338 (2022). https://doi.org/10.1016/j.aml.2022.108338
5. Amodio, P., Brugnano, L., Iavernaro, F.: (Spectral) Chebyshev collocation methods for solving differential equations. Numer. Algoritms **93**, 1613–1638 (2023). https://doi.org/10.1007/s11075-022-01482-w
6. Benson, D.A., Wheatcraft, S.W., Meerschaert, M.M.: Applications of a fractional advection-dispersion equation. Water Resour. Res. **36**(6), 1403–1412 (2000)
7. Brugnano, L., Iavernaro, F.: Line Integral Methods for Conservative Problems. Chapman et Hall/CRC, Boca Raton (2016)
8. Brugnano, L., Iavernaro, F.: Line integral solution of differential problems. Axioms **7**(2), 36 (2018). https://doi.org/10.3390/axioms7020036
9. Brugnano, L., Iavernaro, F.: A general framework for solving differential equations. Ann. Univer. Ferrara Sez. VII Sci. Mat. **68**, 243–258 (2022). https://doi.org/10.1007/s11565-022-00409-6
10. Brugnano, L., Iavernaro, F., Trigiante, D.: A note on the efficient implementation of Hamiltonian BVMs. J. Comput. Appl. Math. **236**, 375–383 (2011). https://doi.org/10.1016/j.cam.2011.07.022
11. Brugnano, L., Iavernaro, F., Trigiante, D.: A simple framework for the derivation and analysis of effective one-step methods for ODEs. Appl. Math. Comput. **218**, 8475–8485 (2012). https://doi.org/10.1016/j.amc.2012.01.074
12. Brugnano, L., Montijano, J.I., Iavernaro, F., Randéz, L.: Spectrally accurate space-time solution of Hamiltonian PDEs. Numer. Algorithms **81**, 1183–1202 (2019). https://doi.org/10.1007/s11075-018-0586-z

13. Brugnano, L., Montijano, J.I., Iavernaro, F., Randéz, L.: On the effectiveness of spectral methods for the numerical solution of multi-frequency highly-oscillatory Hamiltonian problems. Numer. Algorithms **81**, 345–376 (2019). https://doi.org/10.1007/s11075-018-0552-9
14. Brugnano, L., Frasca-Caccia, G., Iavernaro, F., Vespri, V.: A new framework for polynomial approximation to differential equations. Adv. Comput. Math. **48**, 76 (2022). https://doi.org/10.1007/s10444-022-09992-w
15. Bueno-Orovio, A., Burrage, K.: Exact solutions to the fractional time-space Bloch–Torrey equation for magnetic resonance imaging. Commun. Nonlinear Sci. Numer. Simul. **52**, 91–109 (2017)
16. Bueno-Orovio, A., Kay, D., Burrage, K.: Fourier-spectral methods for fractional in space reaction diffusion equations. BIT **54**, 937–954 (2014)
17. Bueno-Orovio, A., Kay, D., Grau, V., Rodriguez, B., Burrage, K.: Fractional diffusion models of cardiac electrical propagation: role of structural heterogeneity in dispersion of repolarization. J. R. Soc. Interface **11**(97), 20140352 (2014)
18. Cusimano, N., Bueno-Orovio, A., Turner, I., Burrage, K.: On the order of the fractional Laplacian in determining the spatio-temporal evolution of a space fractional model of cardiac electrophysiology. PLoS One **10**(12), e0143938 (2015)
19. De Vore, R., Scott, L.R.: Error bounds for Gaussian quadrature and weighted-L^1 polynomial approximation. SIAM J. Numer. Anal. **21**(2), 400–412 (1984). https://doi.org/10.1137/0721030
20. Diethelm, K.: The Analysis of Fractional Differential Equations. An Application-oriented Exposition using Differential Operators of Caputo Type. Lecture Notes in Math. Springer, Berlin (2010)
21. Diethelm, K., Ford, N.J., Freed, A.D.: Detailed error analysis for a fractional Adams method. Numer. Algorithms **36**, 31–52 (2004). https://doi.org/10.1023/B:NUMA.0000027736.85078.be
22. Garrappa, R.: Numerical evaluation of two and three parameter Mittag–Leffler functions. SIAM J. Numer. Anal. **53**(3), 1350–1369 (2015). https://doi.org/10.1137/140971191
23. Garrappa, R.: Trapezoidal methods for fractional differential equations: theoretical and computational aspects. Math. Comput. Simul. **110**, 96–112 (2015). https://doi.org/10.1016/j.matcom.2013.09.012
24. Garrappa, R.: Numerical solution of fractional differential equations: a survey and a software tutorial. Mathematics **6**(2), 16 (2018). https://doi.org/10.3390/math6020016
25. Gautschi, W.: Orthogonal Polynomials Computation and Approximation. Oxford University Press (2004)
26. Henry, B.I., Langlands, T.A.M.: Fractional cable models for spiny neuronal dendrites. Phys. Rev. Letts. **100**(12), 128103 (2008)
27. Henry, B.I., Langlands, T., Wearne, S.: Turing pattern formation in fractional activator-inhibitor systems. Phys. Rev. E **72**(2), 026101 (2005)
28. Hori, M., Fukunaga, I., Masutani, V., Taoka, T., Kamagata, K., Suzuki, Y., et al.: Visualising non Gaussian diffusion—clinical application of q-space imaging and diffusional kurtosis imaging of the brain, and spine. Magn. Reson. Med. Sc. **11**, 221–233 (2012)
29. Lakshmikantham, V., Trigiante, D.: Theory of Difference Equations. Academic Press Inc, Boston (1988)
30. Li, C., Yi, Q., Chen, A.: Finite difference methods with non-uniform meshes for nonlinear fractional differential equations. J. Comput. Phys. **316**, 614–631 (2016)
31. Lindenberg, K., Yuste, S.B.: Properties of the reaction front in a reaction-subdiffusion process. Noise Complex Syst. Stoch. Dyn. **II**(5471), 20–28 (2004)
32. Lubich, Ch.: Fractional linear multistep methods for Abel–Volterra integral equations of the second kind. Math. Comput. **45**(172), 463–469 (1985). https://doi.org/10.1090/S0025-5718-1985-0804935-7
33. Lubich, Ch.: Discretized fractional calculus. SIAM J. Math. Anal. **17**, 704–719 (1986)
34. Magin, R., Feng, X., Baleanu, D.: Solving the fractional order Bloch equation. Concepts Magn. Res., Part A **34A**, 16–23 (2009)
35. Mastroianni, G., Milovanovic, G.: Interpolation processes. In: Basic Theory and Applications. Springer Monogr. Math. Springer, Berlin (2008)
36. Orsingher, E., Beghin, L.: Fractional diffusion equations and processes with randomly varying time. Ann. Probab. **37**(1), 206–249 (2009)
37. Podlubny, I.: Fractional Differential Equations. An Introduction to Fractional Derivatives, Fractional Differential Equations, to Methods of their Solution and Some of their Applications. Academic Press, Inc., San Diego (1999)
38. Satmari, Z.: Iterative Bernstein splines technique applied to fractional order differential equations. Math. Found. Comput. **6**, 41–53 (2023). https://doi.org/10.3934/mfc.2021039
39. Schädle, A., Lopez-Fernandez, M., Lubich, Ch.: Fast and oblivious convolution quadrature. SIAM J. Sci. Comput. **28**, 421–438 (2006)
40. Stynes, M., O'Riordan, E., Gracia, J.L.: Error analysis of a finite difference method on graded meshes for a time-fractional diffusion equation. SIAM J. Numer. Anal. **55**, 1057–1079 (2017)

41. Themistoclakis, W.: Some error bounds for Gauss–Jacobi quadrature rules. Appl. Numer. Math. **116**, 286–293 (2017). https://doi.org/10.1016/j.apnum.2017.02.009
42. Zeng, F., Zhang, Z., Karniadakis, G.E.: Second order numerical methods for multi-term fractional differential equations. Comput. Methods Appl. Mech. Eng. **327**, 478–502 (2017)
43. Zeng, F., Turner, I., Burrage, K.: A stable fast time-stepping method for fractional integral and derivative operators. J. Sci. Comput. **77**, 283–307 (2018)
44. Zeng, F., Turner, I., Burrage, K., Karniadakis, G.: A new class of semi-implicit methods with linear complexity for nonlinear fractional differential equations. SIAM J. Sci. Comput. **40**(5), A2986–A3011 (2018). https://doi.org/10.1137/18M1168169