Trapezoidal methods for fractional differential equations: theoretical and computational aspects ☆

Roberto Garrappa

Università degli Studi di Bari, Dipartimento di Matematica
Via E. Orabona n.4, 70125 Bari, Italy

Abstract

The paper describes different approaches to generalize the trapezoidal method to fractional differential equations. We analyze the main theoretical properties and we discuss computational aspects to implement efficient algorithms. Numerical experiments are provided to illustrate potential and limitations of the different methods under investigation.

Keywords: fractional differential equations, multistep methods, trapezoidal method, convergence, stability, computational aspects
2010 MSC: 34A08, 65L05, 65L20

1. Introduction

It is nowadays well established that several real–life phenomena are better described by fractional differential equations (FDEs), where the term *fractional*, used for historical reasons, refers to derivative operators of any real positive order. Applications of FDEs are commonly found in bioengineering, chemistry, control theory, electronic circuit theory, mechanics, physics, seismology, signal processing and so on (e.g., [3, 7, 5, 33, 22, 41, 50]). We refer to [40] for an historical perspective on fractional calculus.

As a consequence of the growing interest for fractional order models, the analysis and the development of numerical methods for FDEs and partial differential equations with time or space fractional derivatives have become an active area of research (see, for instance, [4, 18, 17, 13, 25, 26, 32, 35, 38, 44, 45, 47, 48, 46, 53, 54]).

☆This work was supported by the GNCS-INdAM 2013 project "Metodi numerici per equazioni differenziali alle derivate frazionarie".

Email address: roberto.garrappa@uniba.it (Roberto Garrappa)

Some of the methods for FDEs are directly derived from methods for integral equations; this is the case, for instance, of product integration (PI) rules. A completely different approach, named as fractional linear multistep methods (FLMMs) [38], has instead been specifically tailored to provide a solid theoretical background for the numerical treatment of FDEs.

Interestingly, these two approaches (PI and FLMM) lead to the same methods when the fractional order α converges to the nearest integer; indeed, different methods for FDEs are often generalizations of the same corresponding method for ordinary differential equations (ODEs).

Some important differences are however observed in the pure fractional case. For instance, the PI and FLMM generalizations of the implicit Euler method behave in a different way when applied to problems with discontinuous right–hand side [10, 11, 12] and only the method devised in the framework of FLMMs succeeds in stabilizing the solution on the switching surface, as expected from theoretical considerations, without generating unwanted spurious oscillations [24].

The presence of distinguishing characteristics in methods for FDEs developed from the same method for ODEs deserves to be more thoroughly investigated. Some differences are of theoretical interest (convergence, stability, etc.); nevertheless, the two approaches also present distinctive features of computational nature that affect the efficiency of the solution process.

The main aim of this paper is to describe and compare some methods for FDEs, highlight strengths and weaknesses and investigate the circumstances under which a certain method is preferable to some others.

The analysis will be confined to those PI rules and FLMMs that generalize, in some respects, the implicit second–order trapezoidal rule. This A–stable one–step method has the smallest possible error constant [34] and is appreciated since provides a satisfactory balance between accuracy, stability properties and computational complexity. Our interest is in exploring when these strengths are inherited by the corresponding methods for FDEs.

There are also some more specific motivations for restricting the attention to second–order methods: a) PI rules very seldom converge with an order equal or greater than 2; b) the implementation of higher order FLMMs is usually computationally demanding; c) as for multistep methods for ODEs, order 2 represents a barrier for stability.

Despite this apparently limited extent, we will however be able to include in our analysis a sufficiently large number of methods.

The choice of focusing on implicit methods is also motivated by the fact that they have been so far very seldom investigated (very often, implicit methods are taken into account in the framework of predictor–corrector

algorithms in which the implicit scheme is actually implemented as an explicit one) and explicit methods are unduly preferred in most of the works on FDEs, regardless of stability issues which discourage their use.

The paper is organized as follows. In Section 2 we present some basic definitions and properties concerning FDEs. Section 3 is devoted to describe PI rules and discuss different implementations on uniform and graded grids. In Section 4 we present FLMMs and we describe three different generalizations of the implicit trapezoidal rule; the use of an effective formula for the evaluation of the weights is also proposed. In Section 5 linear stability is investigated and some implementation details are discussed in Section 6. Finally, in Section 7 the results of some numerical tests are presented to compare the methods under investigation and some concluding remarks are presented at the end of the paper.

2. Fractional differential equations

Let us consider the initial value problem for a system of FDEs in the form

$$\begin{cases} {}^{C}D^{\alpha}_{t_0} y(t) = f(t, y(t)) \\ y^{(k)}(t_0) = y_{0,k}, \quad k = 0, \dots m-1 \end{cases}, \tag{1}$$

where $m = \lceil \alpha \rceil$ is the smallest integer such that $m > \alpha$, $f : [t_0, T] \times \mathbb{R}^q \to \mathbb{R}^q$ is assumed to be sufficiently smooth, $y : [t_0, T] \to \mathbb{R}^q$ is the unknown solution and $y^{(k)}$ denotes classical derivative of integer order k.

The fractional derivative operator ${}^{C}D^{\alpha}_{t_0}$ is introduced in this paper according to the Caputo's definition [14, 42, 49]

$${}^{C}D^{\alpha}_{t_0} y(t) = I^{m-\alpha}_{t_0} y^{(m)}(t),$$

with $I^{\beta}_{t_0}$ the *Riemann-Liouville* (RL) integral of order β on the interval $[t_0, t]$

$$I^{\beta}_{t_0} g(t) = \frac{1}{\Gamma(\beta)} \int_{t_0}^{t} (t-s)^{\beta-1} g(s) ds. \tag{2}$$

A classical result in fractional calculus [14] allows to compute, for any $\alpha \in \mathbb{R}$, the RL integral of the power function as

$$I^{\alpha}_{t_0} (t-t_0)^{\nu} = \frac{\Gamma(\nu+1)}{\Gamma(\alpha+\nu+1)} (t-t_0)^{\alpha+\nu}, \quad \nu > -1, \quad t \geq t_0. \tag{3}$$

It is a well known result that (1) can be equivalently written in terms of the integral equation

$$y(t) = T_{m-1}(t) + I^{\alpha}_{t_0} f(t, y(t)), \tag{4}$$

where $T_{m-1}(t)$ is the Taylor expansion of $y(t)$ centered at t_0

$$T_{m-1}(t) = \sum_{k=0}^{m-1} \frac{(t-t_0)^k}{k!} y_0^{(k)}.$$

One of the major difficulties in the treatment of FDEs derives from the lack of smoothness of the true solution on the whole interval of integration. Indeed, as proved in [36], the true solution of (1) possesses an expansion in mixed (integer and real) powers of the form

$$y(t) = T_{m-1}(t) + \sum_{\nu \in \mathcal{A}_{p,m}} (t-t_0)^\nu Y_\nu + \mathcal{O}((t-t_0)^p), \quad t \to t_0, \tag{5}$$

with $p > m$, $\mathcal{A}_{p,m} = \mathcal{A}_p - \{0, \ldots, m-1\}$ and

$$\mathcal{A}_p = \{\nu \in \mathbb{R} \,|\, \nu = i + j\alpha,\, i, j \in \mathbb{N},\, \nu < p\},$$

with the consequence that the derivatives of $y(t)$ are unbounded at t_0.

3. Product integration rules

The main idea behind PI rules [52] is to evaluate the integral in (4) by approximating the vector field f with suitable polynomials. For the integer–order case this is the way in which Adams multistep methods are devised.

To generalize the 1–step trapezoidal rule consider a (not necessarily uniform) grid $\{t_0, t_1, \ldots, t_N\}$ on the whole interval of integration $[t_0, T]$ and, in the piecewise decomposition of (4)

$$y(t) = T_{m-1}(t) + \frac{1}{\Gamma(\alpha)} \sum_{j=0}^{n-1} \int_{t_j}^{t_{j+1}} (t-s)^{\alpha-1} f(s, y(s)) ds, \quad t \geq t_n,$$

replace f in each subinterval by the first–degree polynomial interpolant

$$p_j(s) = f_{j+1} + \frac{s - t_{j+1}}{h_j}(f_{j+1} - f_j), \quad s \in [t_j, t_{j+1}],$$

where $h_j = t_{j+1} - t_j$, $f_j = f(t_j, y_j)$ and y_j is the numerical approximation to the solution $y(t_j)$. After exactly solving the integrals

$$\begin{aligned} \frac{1}{\Gamma(\alpha)} \int_{t_j}^{t_{j+1}} (t_n - s)^{\alpha-1} ds &= I_{n,j}^{(0)} - I_{n,j+1}^{(0)}, \\ \frac{1}{\Gamma(\alpha)} \int_{t_j}^{t_{j+1}} (t_n - s)^{\alpha-1} \frac{(s - t_{j+1})}{h_j} ds &= \frac{I_{n,j}^{(1)}}{h_j} - I_{n,j}^{(0)} - \frac{I_{n,j+1}^{(1)}}{h_j}, \end{aligned}$$

where, for shortness, we denoted

$$I_{n,j}^{(k)} = \frac{1}{\Gamma(\alpha)} \int_{t_j}^{t_n} (t_n - s)^{\alpha-1} (s - t_j)^k ds = \frac{(t_n - t_j)^{\alpha+k}}{\Gamma(\alpha + k + 1)},$$

it is elementary, after a simple change of the summation index, to obtain the numerical approximation of the solution $y(t)$ at $t = t_n$ as

$$y_n = T_{m-1}(t_n) + w_n f_0 + \sum_{j=1}^{n} b_{n,j} f_j, \tag{6}$$

where

$$\begin{cases} w_n = I_{n,0}^{(0)} - \dfrac{I_{n,0}^{(1)}}{h_0} + \dfrac{I_{n,1}^{(1)}}{h_0}, \\ b_{n,j} = \dfrac{I_{n,j-1}^{(1)} - I_{n,j}^{(1)}}{h_{j-1}} - \dfrac{I_{n,j}^{(1)} - I_{n,j+1}^{(1)}}{h_j}, \quad j = 1, \ldots, n-1, \quad b_{n,n} = \dfrac{I_{n,n-1}^{(1)}}{h_{n-1}}. \end{cases}$$

According to the analysis presented in [6], this scheme is convergent of second order whenever the true solution is assumed sufficiently smooth. Unfortunately, this is a rather optimistic result compared to the expansion (5) of $y(t)$: due to the presence of unbounded derivatives at t_0 the convergence rate of (6) is actually lower than the theoretical value 2, as it can be formally stated by a direct application of the main result by Dixon in [19].

Theorem 1. *Let f be Lipschitz continuous with respect to the second variable and y_n the numerical approximation obtained by applying the PI trapezoidal rule (6) on the interval $[t_0, T]$. There exists a constant $C = C(T - t_0)$, which does not depend on h, such that*

$$\|y(t_n) - y_n\| \leq C(t_n^{\alpha-1} h^{1+\alpha} + h^2), \quad h = \max_{j=0,\ldots,n-1} h_j.$$

Unless very restrictive (and usually unrealistic) assumptions are made to ensure the smoothness of $f(t, y(t))$ also at the left endpoint of $[t_0, T]$, the rate by which the error reduces to 0 is therefore, away from t_0, proportional to $h^{1+\alpha}$ when $0 < \alpha < 1$. The same drop in the convergence order occurs also when higher degree polynomials are employed; for this reason PI rules of higher order are never taken into account for FDEs when $0 < \alpha < 1$. When $\alpha > 1$ order 2 of convergence is instead obtained.

3.1. Uniform meshes

Using (6) on uniform grids $t_n = t_0 + nh$, with $h > 0$ a constant step–size, offers some advantages of computational type. The evaluation of the weights is indeed less expensive and just involves the computation of real powers of integer numbers. Furthermore, (6) presents a convolution structure, in which each weight $b_{n,j}$ depends just on $n - j$, and it can be simplified as

$$y(t_n) = T_{m-1}(t_n) + \frac{h^\alpha}{\Gamma(\alpha+2)} \left(\tilde{w}_n f_0 + \sum_{j=1}^{n} \tilde{b}_{n-j} f_j \right), \tag{7}$$

where now

$$\begin{cases} \tilde{w}_n = (\alpha + 1 - n)n^\alpha + (n-1)^{\alpha+1} \\ \tilde{b}_0 = 1, \quad \tilde{b}_n = (n-1)^{\alpha+1} - 2n^{\alpha+1} + (n+1)^{\alpha+1}, \quad n = 1, 2, \dots \end{cases}.$$

Formulation (7) of the trapezoidal PI rule is well–known since it is a building component of a widely used predictor–corrector algorithm [18].

The main advantage of the convolution structure is however that (7) can be implemented in a fast way by means of FFT algorithms [28].

3.2. Graded meshes

Unlike FLMMs, PI rules can be implemented on non uniform grids, with the aim of overcoming the order reduction stated in Theorem 1. This issue has been investigated, in the context of weakly singular integral equations, by several authors [1, 9] and graded grids are highly praised to this purpose.

By clustering the grid–points at the left boundary of the integration interval, graded grids are able to cope with the lack of regularity and counteract the inefficiency of polynomials to satisfactorily approximate real powers.

A graded mesh on the interval $[t_0, T]$ is defined according to

$$t_n = t_0 + \left(\frac{n}{N}\right)^r (T - t_0), \quad n = 0, 1, \dots, N,$$

where $r > 1$ is a suitable grading exponent. After denoting $h_0 = t_1 - t_0 = (T - t_0)/N^r$ and observing that $h_j = t_{j+1} - t_j = h_0((j+1)^r - j^r)$, a simple manipulation allows to reformulate (6) according to

$$y(t_n) = T_{m-1}(t_n) + \frac{h_0^\alpha}{\Gamma(\alpha+2)} \left(\hat{w}_n f_0 + \sum_{j=1}^{n} \hat{b}_{n,j} f_j \right),$$

where

$$\begin{cases} \hat{w}_n = (n^r-1)^{\alpha+1} - n^{r\alpha}(n^r-\alpha-1), \\ \hat{b}_{n,j} = \phi_{n,j}(\alpha,r) - \phi_{n,j+1}(\alpha,r), \quad j=1,\dots,n-1, \quad \hat{b}_{n,n} = (n^r-(n-1)^r)^\alpha \end{cases}$$

and

$$\phi_{n,j}(\alpha,r) = \frac{(n^r-(j-1)^r)^{(\alpha+1)} - (n^r-j^r)^{(\alpha+1)}}{j^r-(j-1)^r}.$$

The following result [1, 2] on the second order convergence of graded grids for linear FDEs can be readily extended to more general equations.

Theorem 2. *Let $f(t,y(t)) = K(t)y(t)$, $K \in \mathcal{C}^2([t_0,T])$. For $r = 2/\alpha$ it is*

$$\|y(t_n) - y_n\| \le C_\alpha h^2, \quad h = \max_{j=0,\dots,n-1} h_j = h_{n-1}.$$

The use of PI rules on nonuniform grids is more demanding; not only the weights must be recomputed at each step but, since the absence of a convolution structure, it is not possible to employ fast algorithms. By means of some numerical experiments we will verify, at the end of the paper, whether or not this further amount of computation is, in some way, balanced by the enhancement in the accuracy.

4. Fractional linear multistep methods

In the pioneering work on discretized fractional calculus [38], Lubich proposed an elegant and effective strategy for classical linear multistep methods (LMMs) originally devised for integer–order ODEs.

Although the potentials of FLMMs have been recognized in several papers (see, for instance, the review article by Gorenflo [27]), their use for practical computation has not captured so far widespread attention, perhaps due to some difficulties in explicitly formulating FLMMs.

The key aspect in FLMMs is the approximation of the RL integral (2) by means of the convolution quadrature

$$I_h^\beta g(t_n) = h^\beta \sum_{j=0}^{n} \omega_{n-j} g(t_j) + h^\beta \sum_{j=0}^{s} w_{n,j} g(t_j) \tag{8}$$

on uniform grids $t_n = t_0 + nh$, $h > 0$, and where convolution and starting quadrature weights ω_n and $w_{n,j}$ do not depend on h.

Starting weights $w_{n,j}$ play an important role, especially in the first part of the integration interval, in order to cope with the possibly singular character

of the integrand function at t_0; we will discuss the problem of their evaluation later on.

Convolution quadrature weights ω_n are the main components of the quadrature rule and characterize the specific FLMM. They are obtained starting from any LMM for ODEs

$$\sum_{j=0}^{k} \rho_j y_{n-j} = \sum_{j=0}^{k} \sigma_j f(t_{n-j}, y_{n-j}),$$

being $\rho(z) = \rho_0 z^k + \rho_1 z^{k-1} + \ldots + \rho_k$ and $\sigma(z) = \sigma_0 z^k + \sigma_1 z^{k-1} + \ldots + \sigma_k$ the characteristic polynomials. As shown in [51], LMMs can be equivalently reformulated as (8) (but with $\beta = 1$ for ODEs) with weights ω_n obtained as the coefficients of the formal power series (FPS)

$$\omega(\xi) = \sum_{n=0}^{\infty} \omega_n \xi^n, \quad \omega(\xi) = \frac{\sigma(1/\xi)}{\rho(1/\xi)}$$

and the function $\omega(\xi)$ goes therefore under the name of *generating function* of the LMM. The idea proposed in [38] is to generate a quadrature rule for fractional problems (1) by evaluating the convolution weights as the coefficients in the FPS of the fractional–order power of the generating function

$$\omega_\beta(\xi) = \sum_{n=0}^{\infty} \omega_n \xi^n, \quad \omega_\beta(\xi) = \left(\frac{\sigma(1/\xi)}{\rho(1/\xi)}\right)^\beta. \tag{9}$$

Methods of this kind, named as FLMMs, when applied to (1) read as

$$y_n = T_{m-1}(t_n) + h^\alpha \sum_{j=0}^{s} w_{n,j} f_j + h^\alpha \sum_{j=0}^{n} \omega_{n-j} f_j \tag{10}$$

and their convergence properties are stated in the following result [37, 38].

Theorem 3. *Let (ρ, σ) be a stable and consistent of order p implicit LMM with the zeros of $\sigma(\xi)$ having absolute value ≤ 1. The FLMM (10) is convergent of order p.*

One of the major difficulties in FLMMs (10) is in evaluating the weights ω_n as the coefficients in the FPS (9). Although some sophisticated algorithms are available for manipulating FPS [31], for most of the methods an efficient tool is the J.C.P. Miller formula stated by the following theorem [30, Theorem 1.6c].

Theorem 4. *Let $\varphi(\xi) = 1 + \sum_{n=1}^{\infty} a_n \xi^n$ be a FPS. Then for any $\beta \in \mathbb{C}$,*

$$(\varphi(\xi))^\beta = \sum_{n=0}^{\infty} v_n^{(\beta)} \xi^n,$$

where coefficients $v_n^{(\beta)}$ can be recursively evaluated as

$$v_0^{(\beta)} = 1, \quad v_n^{(\beta)} = \sum_{j=1}^{n} \left(\frac{(\beta+1)j}{n} - 1 \right) a_j v_{n-j}^{(\beta)}.$$

The Miller formula allows to evaluate, in the most general case, the first N coefficients of $(\varphi(\xi))^\beta$ with a number of operations proportional to N^2. In most cases of practical interest, this computational effort is actually reduced by an order of magnitude. This is the case of the evaluation of the coefficients $\{\omega_n^{(\beta)}\}_{n\in\mathbb{N}}$ in the FPS of $(1 \pm \xi)^\beta$; indeed, since $a_1 = \pm 1$ and $a_2 = a_3 = \ldots = 0$, it is elementary to see that the application of Theorem 4 leads to

$$\omega_0^{(\beta)} = 1, \quad \omega_n^{(\beta)} = \pm \left(\frac{\beta+1}{n} - 1 \right) \omega_{n-1}, \tag{11}$$

involving just $2N$ multiplications and N additions.

4.1. Fractional trapezoidal rule

In order to generalize to FDEs the classical trapezoidal rule, let us first consider its classical formulation for ODEs

$$y_{n+1} - y_n = \frac{h}{2}(f_n + f_{n+1}), \tag{12}$$

with characteristic polynomials $\rho(z) = z - 1$ and $\sigma(z) = (z+1)/2$ and generating function

$$\omega(\xi) = \frac{\sigma(1/\xi)}{\rho(1/\xi)} = \frac{(1+\xi)}{2(1-\xi)} = \frac{1}{2}\left(1 + 2\sum_{n=1}^{\infty} \xi^n\right).$$

To practically compute the first N weights in the FPS of $(\omega(\xi))^\alpha$ we advise against the direct application of the Miller formula to the infinite series in $\omega(\xi)$ since it would involve a number of operations proportional to N^2. It is instead preferable to first apply twice Theorem 4 to $(1+\xi)^\alpha$ and $(1-\xi)^{-\alpha}$, by means of (11), and hence evaluate the coefficient of their product by an FFT algorithm, with a number of operations proportional to $3N \log_2 4N$ when N is a power of 2 [31]. This last task can be efficiently performed by means of the very few lines of Matlab code

```
x = fft([omega1,zeros(size(omega1))]) ;
y = fft([omega2,zeros(size(omega2))]) ;
omega = ifft(x.*y) ; omega = omega(1:N) ;
```

where `omega1` and `omega2` are the row arrays with coefficients of $(1+\xi)^\alpha$ and $2^{-\alpha}(1-\xi)^{-\alpha}$ respectively, evaluated in a fast way thanks to (11).

4.2. Newton–Gregory formula

The trapezoidal rule (12) belongs to the more general family of k–step Adams–Moulton methods

$$y_{n+1} - y_n = h \sum_{i=0}^{k} \gamma_i \nabla^i f_{n+1},$$

with $\nabla^i f_{n+1}$ the classical backward differences (i.e., $\nabla^0 f_{n+1} = f_{n+1}$ and $\nabla^{i+1} f_{n+1} = \nabla^i f_{n+1} - \nabla^i f_n$); the coefficients γ_i are the first $k+1$ terms in the truncation

$$[G(\xi)]_k = \gamma_0 + \gamma_1(1-\xi) + \ldots + \gamma_k(1-\xi)^k$$

of the power series expansion, with respect to $1-\xi$, of $G(\xi) = (\xi-1)/\ln\xi$ and the resulting generating function is $\omega(\xi) = [G(\xi)]_k/(1-\xi)$.

It is easy to see that $\gamma_0 = 1$ and $\gamma_1 = -\frac{1}{2}$, thus immediately leading to (12) when $k = 1$. Therefore, a straightforward generalization to fractional–order problems by means of the α–power of $\omega(\xi)$ leads, in the case of the 1–step method, to the trapezoidal rule discussed in the previous subsection.

It has been shown in [36] that alternative methods, named as *Newton–Gregory formulas*, can be devised by first evaluating the α power of $G(\xi)$ and successively truncating to the first $k+1$ terms as

$$\omega_\alpha(\xi) = \frac{[(G(\xi))^\alpha]_k}{(1-\xi)^\alpha}.$$

The interchange between the truncation series symbol $[\,\cdot\,]_k$ and the α–power is important and generates a different family of methods. By operating for simplicity the change of variable $\tau = 1-\xi$,

$$(G(\xi))^\alpha = (G(1-\tau))^\alpha = \left(-\frac{\ln(1-\tau)}{\tau}\right)^{-\alpha} = \left(1 + \sum_{n=1} \frac{\tau^n}{n+1}\right)^{-\alpha} = \sum_{n=0}^{\infty} \bar{\gamma}_n^\alpha \tau^n.$$

The first coefficients $\bar{\gamma}_n^\alpha$ can be easily evaluated thanks again to Theorem 4 (see [21] for the explicit formulation of the first coefficients).

Since $\bar{\gamma}_0^\alpha = 1$ and $\bar{\gamma}_1^\alpha = -\frac{\alpha}{2}$, the generating function of the FLLM obtained by the 1–step Adams method is therefore

$$\omega_\alpha(\xi) = \frac{1 - \frac{\alpha}{2}(1-\xi)}{(1-\xi)^\alpha} = (1-\xi)^{-\alpha}\left(1 - \frac{\alpha}{2}(1-\xi)\right)$$

and, after a simple manipulation, it is possible to evaluate the first N coefficients in $\omega_\alpha(\xi)$, with a cost proportional to N, as

$$\omega_0 = 1 - \frac{\alpha}{2}, \quad \omega_n = \left(1 - \frac{\alpha}{2}\right)\omega_n^{(-\alpha)} + \frac{\alpha}{2}\omega_{n-1}^{(-\alpha)},$$

with $\omega_n^{(-\alpha)}$ the coefficients in the FPS $(1-\xi)^{-\alpha}$ evaluated by means of (11).

4.3. Fractional BDF formula

The second order BDF formula for ODEs is given by

$$y_{n+2} - \frac{4}{3}y_{n+1} + \frac{1}{3}y_n = \frac{2h}{3}f_{n+2}, \tag{13}$$

with characteristic polynomials $\rho(z) = (z^2 - 4/3z + 1/3)$ and $\sigma(z) = 2/3z^2$ and generating function

$$\omega(\xi) = \frac{\sigma(1/\xi)}{\rho(1/\xi)} = \frac{2}{3(1 - 4\xi/3 + \xi^2/3)}.$$

BDF method (13) can be considered in no way as coming from the trapezoidal rule; it is indeed obtained by differentiating a second–degree interpolant polynomial. However, it is equally of interest in our analysis due to its second order of convergence as the trapezoidal rule and the excellent stability properties that we are interested in verifying also for FDEs.

The coefficients in the FPS of $(\omega(\xi))^\alpha$ can be profitably evaluated again by the Miller formula applied to $1 - 4\xi/3 + \xi^2/3$, with negative power $-\alpha$, to first produce

$$\tilde{\omega}_0 = 1, \quad \tilde{\omega}_1 = \frac{4}{3}\alpha\tilde{\omega}_0, \quad \tilde{\omega}_n = \frac{4}{3}\left(1 + \frac{\alpha - 1}{n}\right)\tilde{\omega}_{n-1} + \frac{4}{3}\left(\frac{2(1-\alpha)}{n} - 1\right)\tilde{\omega}_{n-2}$$

and hence by computing $\omega_n = 2^\alpha\tilde{\omega}_n/3^\alpha$ with an overall number of operations that still remains proportional to N.

5. Linear stability analysis

The analysis of stability is of primary importance to understand the qualitative behavior of a numerical method. By assuming the existence of a stable steady state, stability analysis aims to verify whether or not approximated trajectories move towards the steady state.

As usual, stability issues for FDEs can be addressed by studying the scalar linear test equation

$$ {}^{C}D^{\alpha}_{t_0} y(t) = \lambda y(t), \quad \lambda \in \mathbb{C}, \tag{14} $$

whose steady–state is at $y = 0$. The generalization of a well–known result in the theory of ODEs can be done by means of the following result [37, 43].

Theorem 5. *Let $\alpha > 0$. The steady–state $y = 0$ of (14) is stable if and only if $\lambda \in \Sigma_\alpha$, where $\Sigma_\alpha = \{s \in \mathbb{C} \ : \ |\arg(s)| > \alpha\pi/2\}$.*

In Figure 1 we present the sector of stability Σ_α for two exemplifying cases. It is immediate to verify that Σ_α coincides with the left complex semi–plane $\mathbb{C}^-$ when $\alpha = 1$, thus generalizing the classical result for ODEs.

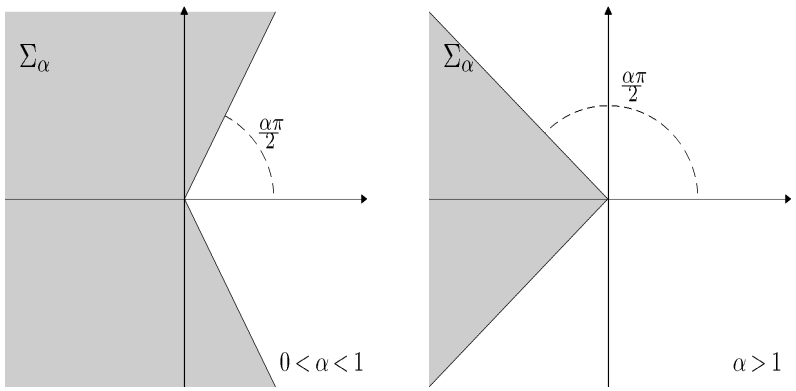

Figure 1: Stability sectors Σ_α for $0 < \alpha < 1$ (left) and $\alpha > 1$ (right)

When PI rules or FLMMs are applied to (14), the numerical solution $\{y_n\}_{n\in\mathbb{N}}$ is evaluated by means of a convolution quadrature

$$ y_n = g_n + z_\alpha \sum_{j=0}^{n} \omega_{n-j} y_j, \quad z_\alpha = h^\alpha \lambda. \tag{15} $$

The problem of determining the stability region $S_\alpha \subseteq \mathbb{C}$ such that $y_n \to 0$ whenever $z_\alpha \in S_\alpha$ has been discussed in [37, 39] and the main result can be summarized in the following theorem.

Acronym	Formula	Subsection
PI U	PI rule on uniform grid	3.1
PI G	PI rule on graded grid	3.2
FT	fractional trapezoidal rule	4.1
NG	Newton–Gregory formula	4.2
FBDF	fractional BDF formula	4.3

Table 1: Acronyms used to denote methods

Theorem 6. *Let $\alpha > 0$. Under the assumption that the sequence $\{g_n\}_n$ is convergent and that the quadrature weights ω_n satisfy*

$$\omega_n = \frac{n^{\alpha-1}}{\Gamma(\alpha+1)} + u_n \quad \text{with} \quad \sum_{n=1}^{\infty} |u_n| < \infty, \quad n \geq 1,$$

the stability region of the convolution quadrature (15) is

$$S_\alpha = \mathbb{C} \setminus \left\{ \frac{1}{\omega_\alpha(\xi)} \, : \, |\xi| \leq 1 \right\}, \quad \omega_\alpha(\xi) = \sum_{n=0}^{\infty} \omega_n \xi^n.$$

We refer to [23] and [38] to verify that the PI rule and the FLMMs investigated in this paper fulfill the assumptions of Theorem 6.

In the numerical treatment of ODEs A–stability means that the whole complex semi–plane is included in the stability region of the method. A–stable methods are therefore particularly attractive since they ensure that the numerical solution of stable linear problems converges to 0 regardless of the size of the step h. Because of the result in Theorem 5, for FDEs it is more useful to use the concept of $A(\alpha\frac{\pi}{2})$–stability, which means that the whole sector Σ_α must be included in the stability region.

While $\omega_\alpha(\xi)$ is a priori known for FLMMs, the generating function of PI rules does not possess an analytical representation and, therefore, it can be just numerically approximated.

In Figure 2 we present a comparison of the stability regions (the gray areas) of the methods discussed in the previous sections for various values $0 < \alpha < 1$ (dotted lines denote the corresponding sectors of stability Σ_α). To identify each method, here and throughout the following section we make use of the acronyms listed in Table 1.

We can distinctly observe that, as α increases from 0 to 1, the stability region of each method decreases but always includes Σ_α; we can draw the conclusion that all the methods are $A(\alpha\frac{\pi}{2})$–stable for $0 < \alpha < 1$.

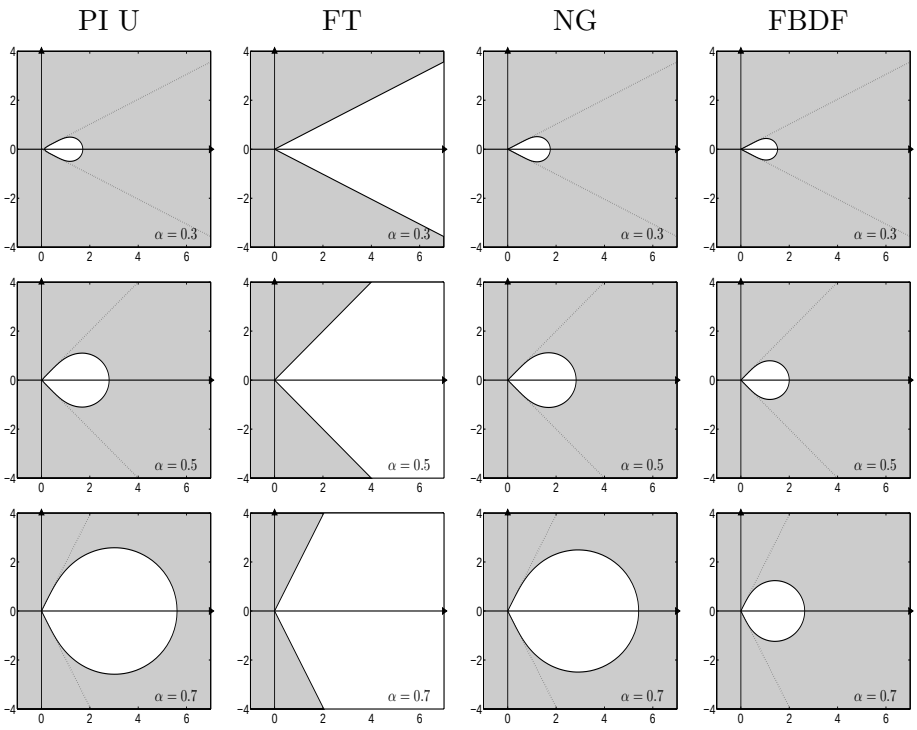

Figure 2: Stability regions (gray areas) for some orders $0 < \alpha < 1$

Within ODEs, the stability region of the trapezoidal rule is the negative semi–plane $\mathbb{C}^-$ and exactly corresponds to the region in which the true solution converges to the steady–state. Among the methods presented in this paper, just FT preserves this feature and indeed its stability regions coincide with Σ_α for any value of α. NG and PI have regions of almost the same size but larger with respect to FT; as in the ODE case, FBDF possesses the largest regions of stability.

Interestingly, the situation is completely different for $1 < \alpha < 2$ as shown in Figure 3. Whilst the stability regions of FT still coincide with the stability sectors Σ_α and the stability properties of FBDF still remain the most remarkable ones, we observe that NG and PI completely lose the $A(\alpha\frac{\pi}{2})$–stability. This is an important aspect to be taken into account: despite the implicit nature of these methods, numerical instability can arise when a not sufficiently small step–size is used and, therefore, they seem not suited to solve stiff problems when $1 < \alpha < 2$.

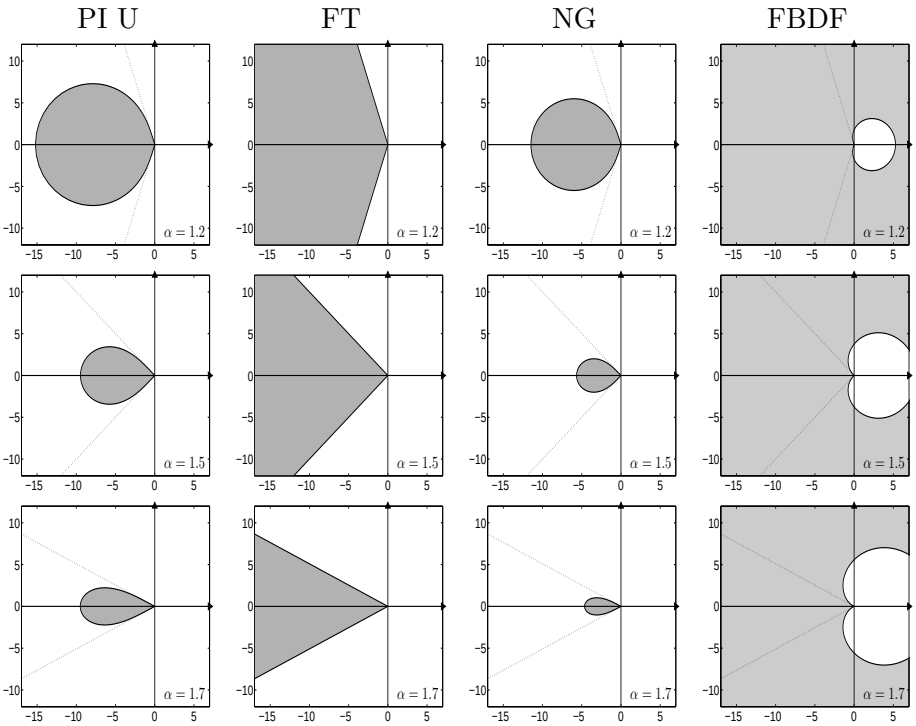

Figure 3: Stability regions (gray areas) for some orders $\alpha > 1$

6. Implementation details and Matlab codes

In this section we discuss some implementation issues that are essential for the development of efficient codes.

6.1. Evaluation of the lag term

The evaluation of the (usually long) lag term is one of the most expensive tasks in convolution quadratures like (7) or (10). The integration along a large number N of steps is usually a critical bottleneck and, if not well designed, it can be an impracticable task involving a number of operations proportional to N^2.

Several authors have discussed this problem (see, for instance, [8]); in [15] a strategy based on parallel algorithms has been described.

In our codes we implement the FFT algorithm described in [28] by which the computational effort downgrades from N^2 to $N \log(N)^2$. We do not further discuss technical details of this algorithm, for which we just refer to the above mentioned paper; we simply observe that, without efficient algorithms

of this kind, the numerical solution of FDEs very quickly degenerates in an unworkable task as the number N of grid–points increases.

6.2. Starting weights

Starting weights $w_{n,j}$ are introduced in (10) to cope with the singular behavior of the solution close to the left endpoint of the integration interval. They are evaluated after imposing that (8) is exact for $g(t) = t^\nu$ with $\nu \in \mathcal{A}_{p-1} \cup \{p-1\}$ and p the order of convergence of the FLMM [38].

Because of (3), and since the methods investigated in this paper have order $p = 2$, it is immediate to see that weights $w_{n,j}$ must satisfy the relation

$$\sum_{j=0}^{s} w_{n,j} j^\nu = -\sum_{j=0}^{n} \omega_{n-j} j^\nu + \frac{\Gamma(\nu+1)}{\Gamma(1+\nu+\alpha)} n^{\nu+\alpha}, \quad \nu \in \mathcal{A}_1 \cup \{1\} \tag{16}$$

and, hence, they are evaluated by solving, at each step n, a system of $s+1$ linear equations, with s the cardinality of $\mathcal{A}_1$.

This problem has been in–depth discussed in [16], where the authors stressed the attention on the mildly ill–conditioned nature of the coefficient matrix V of (16). Nevertheless, for the methods under investigation this is not a serious issue since the size of V is small, being $s = \lceil 1/\alpha \rceil + 1$ (in most cases s will be 3 or 4 and the inversion of V can be done analytically). The only exception is for very small values of α which, however, are extremely uncommon in applications.

6.3. Starting the computation

The initialization of (10) requires the knowledge of the first $s+1$ approximations $y_0, y_1, \dots y_s$ of the solution. Since just y_0 is given by the problem, the remaining values must be evaluated in some other way, for instance by using some different methods.

To avoid mixing methods of different nature, we prefer to evaluate all the approximations $y_1, \dots y_s$ at the same time by setting the single implicit system

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_s \end{pmatrix} = \begin{pmatrix} T_{m-1}(t_1) \\ T_{m-1}(t_2) \\ \vdots \\ T_{m-1}(t_s) \end{pmatrix} + h^\alpha \begin{pmatrix} (\omega_1 + w_{1,0}) f_0 \\ (\omega_2 + w_{2,0}) f_0 \\ \vdots \\ (\omega_s + w_{s,0}) f_0 \end{pmatrix} + h^\alpha \mathcal{B} \begin{pmatrix} f(t_1, y_1) \\ f(t_2, y_2) \\ \vdots \\ f(t_s, y_s) \end{pmatrix},$$

where

$$\mathcal{B} = \begin{pmatrix} \omega_0 I & & & \\ \omega_1 I & \omega_0 I & & \\ \vdots & \vdots & \ddots & \vdots \\ \omega_{s-1} I & \omega_{s-2} I & \dots & \omega_0 I \end{pmatrix} + \begin{pmatrix} w_{1,1} I & w_{1,2} I & \dots & w_{1,s} I \\ w_{2,1} I & w_{2,2} I & \dots & w_{2,s} I \\ \vdots & \vdots & \ddots & \vdots \\ w_{s,1} I & w_{s,2} I & \dots & w_{s,s} I \end{pmatrix}.$$

This nonlinear system has size $s \cdot q$, where q is the size of the system of FDEs we want to solve. No particular problems arise when the size q of the problem is small; otherwise some memory allocation problems could occur in presence of very small values of α, involving a very large number s of initial approximations to compute.

6.4. Solution of the nonlinear systems

All the implicit methods analyzed in Sections 3 and 4 require, at each step, the solution of an algebraic nonlinear system in the form

$$y_n = g_n + h^\alpha \bar{\omega}_0 f(t_n, y_n),$$

where g_n and $\bar{\omega}_0$ do not depend on y_n. To preserve the good stability properties, it is convenient to use Newton iterations

$$y_n^{(k+1)} = y_n^{(k)} + \Delta_n^{(k)},$$

where $\Delta_n^{(k)}$ is solution of the linear system

$$A\Delta_n^{(k)} = b, \quad A = I - h^\alpha \bar{\omega}_0 J_f(y_n^{(k)}), \quad b = -y_n^{(k)} + g_n + h^\alpha \bar{\omega}_0 f(t_n, y_n^{(k)})$$

and, usually, $y_n^{(0)} = y_{n-1}$. We refer to [29] for more insights on this issue and for a discussion on the stopping criteria.

We just highlight here that the convergence can depend on h^α, instead of h as in ODEs. Whilst this is not an issue when $1 < \alpha < 2$, some unexpected problems could arise whenever α is small. For this reason it is necessary to use a quite small step–size h with low values of α.

7. Numerical experiments

By means of numerical experiments we compare the methods discussed in the paper. We perform all the experiments in Matlab, version 7.9.0.529, on a computer equipped with the Intel Dual Core E5400 processor running at 2.70 GHz under the Windows XP operating system; all the Matlab codes have

been optimized, to get the best performance, according to the discussion reported in the previous sections.

As a first test problem we consider the linear FDE (14) for $\lambda = -2$. For each method, and for an increasing number N of grid–points, we report the error $E(N)$ at $T = 2$, with respect to a reference solution, and we provide an estimation of the order of convergence (EOC) as $\log_2(E(N)/E(2N))$.

	PI U		PI G		FT		NG		FBDF	
N	Error	EOC	Error	EOC	Error	EOC	Error	EOC	Error	EOC
32	3.29(−4)		1.45(−4)		1.71(−5)		3.92(−5)		1.10(−4)	
64	1.15(−4)	1.524	3.65(−5)	1.987	5.65(−6)	1.602	1.20(−5)	1.707	3.16(−5)	1.798
128	4.00(−5)	1.516	9.17(−6)	1.991	1.74(−6)	1.698	3.50(−6)	1.780	8.83(−6)	1.842
256	1.40(−5)	1.511	2.30(−6)	1.993	5.07(−7)	1.779	9.78(−7)	1.838	2.40(−6)	1.880
512	4.94(−6)	1.508	5.78(−7)	1.994	1.41(−7)	1.844	2.65(−7)	1.883	6.37(−7)	1.912
1024	1.74(−6)	1.505	1.45(−7)	1.992	3.77(−8)	1.903	6.98(−8)	1.924	1.66(−7)	1.939
2048	6.14(−7)	1.503	3.67(−8)	1.987	9.49(−9)	1.991	1.77(−8)	1.978	4.25(−8)	1.969

Table 2: Errors and EOC at $T = 2.0$ for the linear FDE (14) with $\alpha = 0.5$ and $\lambda = -2.0$

The numerical results in Table 2 for $\alpha = 0.5$ clearly confirm theoretical findings on the order of convergence. As expected from Theorem 1, the PI rule on uniform grids fails to obtain order 2 and converges with order $1 + \alpha$; on the other hand, the PI rule on graded grids converges with full order 2.

As in the ODE case, FT produces the smallest error. Although also NG and PI are generalizations of the trapezoidal rule, it seems that only FT inherits this feature; by taking into account, from the analysis of Section 5, that FT preserves the other peculiar character of the trapezoidal rule of having the region of stability exactly equal to the region in which the solution converges towards the steady state, it is tempting to conclude that FT is the most natural generalization of the trapezoidal rule.

Unfortunately, the situation is more complicate since when $1 < \alpha < 2$ it is NG that provides the lowest error, as shown in Table 3 for $\alpha = 1.5$.

This phenomenon can be more clearly observed in Figure 4, where we show the errors as a function of α (we used the same values for λ and T from the previous experiment and a fixed number $N = 1024$ of grid points).

We must highlight that this is a not completely unexpected result. Stability and accuracy are indeed conflicting requirements in most of numerical methods; thus, FT shows the smallest regions of stability but the lowest error when $0 < \alpha < 1$, exactly as it happens with NG for $1 < \alpha < 2$. What we observe is a kind of roles swap between FT and NG when α moves from the left to the right of 1.

	PI U		PI G		FT		NG		FBDF	
N	Error	EOC	Error	EOC	Error	EOC	Error	EOC	Error	EOC
64	$3.71(-5)$		$6.16(-5)$		$5.50(-5)$		$1.55(-5)$		$1.95(-4)$	
128	$9.31(-6)$	1.993	$1.54(-5)$	2.000	$1.39(-5)$	1.988	$3.73(-6)$	2.053	$5.22(-5)$	1.902
256	$2.33(-6)$	1.997	$3.85(-6)$	2.001	$3.48(-6)$	1.993	$9.10(-7)$	2.035	$1.35(-5)$	1.951
512	$5.82(-7)$	2.003	$9.59(-7)$	2.004	$8.71(-7)$	1.999	$2.22(-7)$	2.035	$3.43(-6)$	1.975
1024	$1.43(-7)$	2.024	$2.37(-7)$	2.016	$2.16(-7)$	2.013	$5.25(-8)$	2.081	$8.65(-7)$	1.989
2048	$3.32(-8)$	2.106	$5.67(-8)$	2.064	$5.15(-8)$	2.068	$1.05(-8)$	2.325	$2.15(-7)$	2.008

Table 3: Errors and EOC at $T = 2.0$ for the linear FDE (14) with $\alpha = 1.5$ and $\lambda = -2.0$

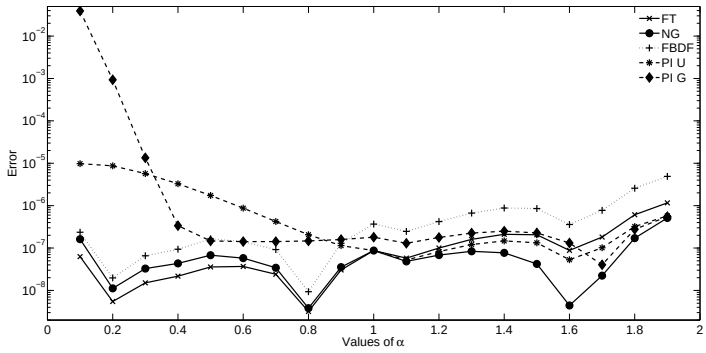

Figure 4: Errors at $T = 2.0$ for the linear FDE (14) with $\lambda = -2.0$ and $N = 1024$.

The very bad performance of PI on graded grids for low values of α is due to the fact that the grids generated by the graded exponent $r = 2\alpha$ are very coarse close to right endpoint of the integration interval. Furthermore, graded grids do not add any improvement for $\alpha > 1$, since the maximum expected order 2 is already achieved by using uniform grids. We can draw the conclusion that the utility of graded grids is confined only to $\frac{1}{2} < \alpha < 1$.

It is also interesting to analyze the computational effort in relation to the accuracy. To this purpose, the plots in Figure 5, which are related to the experiments presented in Tables 2 and 3, show that FT, NG and FBDF behave in a more efficient way with respect to PI when $0 < \alpha < 1$; indeed, PI U pays for the reduction in the accuracy whilst PI G pays for the absence of a convolution structure which precludes the application of FFT algorithms. The minor efficiency of FBDF with respect to FT and NG is due to the major complexity in the evaluation of weights. It is worthwhile to note that the preeminence of FLMMs with respect to PI rules holds despite the need of solving linear systems for evaluating the starting weights. For $1 < \alpha < 2$ the absence of order reduction increases the efficiency of PI on uniform nodes

but however it is below that of NG which instead has lower errors.

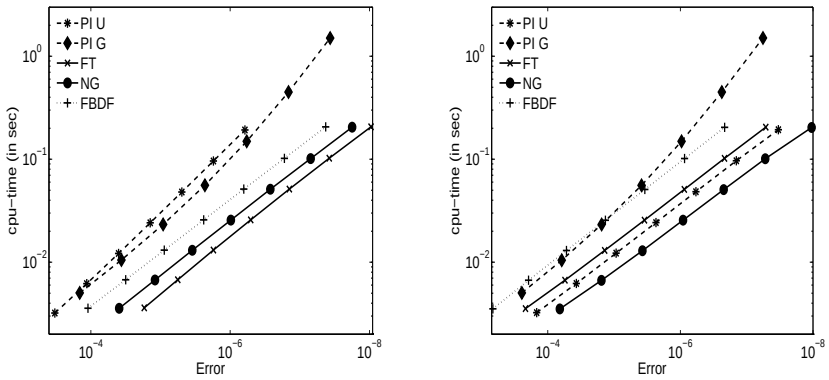

Figure 5: Errors versus time of execution for $\alpha = 0.5$ (left) and $\alpha = 1.5$ (right)

To observe the behavior of the methods under investigation for nonlinear problems, we consider the fractional reaction diffusion system of Brusselator type

$$\begin{cases} {}^{C}D^{\alpha}_{t_0} x_1(t) = a - (\mu+1)x_1(t) + x_1(t)^2 x_2(t) \\ {}^{C}D^{\alpha}_{t_0} x_2(t) = \mu x_1(t) - x_1(t)^2 x_2(t) \end{cases} . \tag{17}$$

It has been shown [20] that there exists a marginal value $\bar{\alpha}$, depending on a and μ, such that system (17) approaches a stable limit cycle, as shown in Figure 6 for $\alpha = 0.8$ and $(a, \mu) = (1, 4)$.

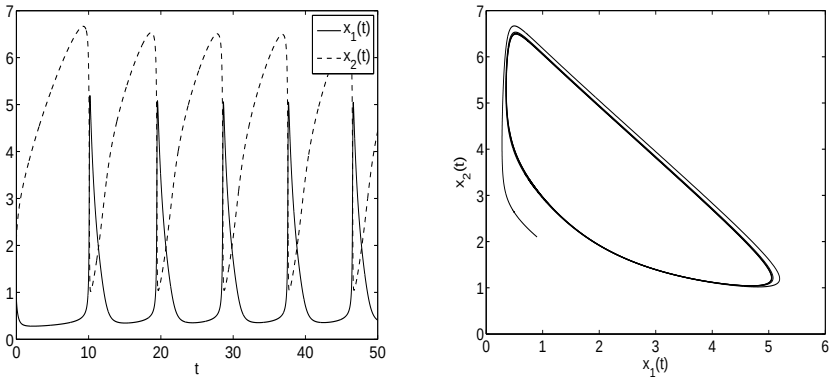

Figure 6: Solution of the fractional Brusselator system (17) for $\alpha = 0.8$ and $(a, \mu) = (1, 4)$ in the (t, x)-plane (left) and in the phase plane (right)

From Table 4 the good results provided by PI U stand out. This is an indirect evidence that the findings of Theorem 1 can be quite pessimistic

	PI U		PI G		FT		NG		FBDF	
N	Error	EOC	Error	EOC	Error	EOC	Error	EOC	Error	EOC
400	$3.66(-1)$		$7.33(-1)$		$2.88(-1)$		$3.67(-1)$		$5.23(-1)$	
800	$6.04(-2)$	2.601	$2.55(-1)$	1.523	$5.12(-2)$	2.490	$6.32(-2)$	2.537	$1.50(-1)$	1.800
1600	$1.55(-2)$	1.965	$5.66(-2)$	2.173	$1.28(-2)$	2.005	$1.63(-2)$	1.956	$4.31(-2)$	1.798
3200	$4.00(-3)$	1.953	$1.35(-2)$	2.064	$3.27(-3)$	1.964	$4.21(-3)$	1.951	$1.19(-2)$	1.864
6400	$1.01(-3)$	1.982	$3.37(-3)$	2.006	$8.27(-4)$	1.982	$1.07(-3)$	1.982	$3.12(-3)$	1.926
12800	$2.41(-4)$	2.071	$8.33(-4)$	2.015	$1.94(-4)$	2.095	$2.55(-4)$	2.065	$7.92(-4)$	1.977

Table 4: Errors and EOC at $T = 50.0$ for the Brusselator system (17) with $\alpha = 0.8$

in some cases; indeed, the drop in the convergence order depends on the expansion of the vector field which in some cases could not involve real powers of low degree. However, also in this case FT presents the smallest error.

The relationship between errors and execution times in Figure 7 shows however that the efficiency of PI U, also in this favourable case, can be at least comparable but not higher with respect to FT and NG; it is hence quite difficult to justify the major attention that PI rules usually attract in the literature on the numerical solution of FDEs.

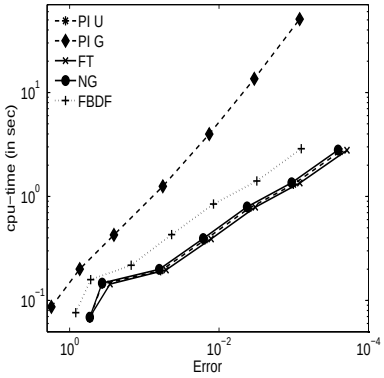

Figure 7: Errors versus time of execution for the Brusselator system (17) with $\alpha = 0.8$

8. Concluding remarks

In this paper we have discussed some implicit second order methods for FDEs and investigated some of the main aspects related to their efficient implementation.

We have observed that stability properties change according to the value of the fractional order α and whenever $1 < \alpha < 2$ the NG and PI are no longer $A(\alpha\frac{\pi}{2})$–stable. As in the ODE case, the method obtained starting from BDF shows the largest stability regions.

Whilst FT provides the smallest error for $0 < \alpha < 1$, the performance of NG stands out for $1 < \alpha < 2$. PI rules with graded grids appear effective just for α between $\frac{1}{2}$ and 1; anyway, it seems that in general the overall efficiency of FLMMs is superior to PI, despite the major interest that PI rules usually attract.

References

[1] H. Brunner, The numerical solution of weakly singular Volterra integral equations by collocation on graded meshes, Math. Comp. 45 (172) (1985) 417–437.

[2] H. Brunner, Collocation methods for Volterra integral and related functional differential equations, Cambridge University Press, Cambridge, 2004.

[3] A. Bueno-Orovio, D. Kay, V. Grau, B. Rodriguez, K. Burrage, Fractional diffusion models of cardiac electrical propagation : role of structural heterogeneity in dispersion of repolarization, Tech. Rep. OCCAM 13/35, Oxford Centre for Collaborative Applied Mathematics, Oxford (UK) (2013).

[4] K. Burrage, N. Hale, D. Kay, An efficient implicit FEM scheme for fractional-in-space reaction-diffusion equations, SIAM J. Sci. Comput. 34 (4) (2012) A2145–A2172.

[5] D. Cafagna, G. Grassi, Observer-based projective synchronization of fractional systems via a scalar signal: Application to hyperchaotic Rössler systems, Nonlinear Dynamics 68 (1-2) (2012) 117–128.

[6] R. F. Cameron, S. McKee, The analysis of product integration methods for Abel's equation using discrete fractional differentiation, IMA J. Numer. Anal. 5 (1) (1985) 339–353.

[7] R. Caponetto, G. Maione, A. Pisano, M. M. R. Rapaić, E. Usai, Analysis and shaping of the self-sustained oscillations in relay controlled fractional-order systems, Fractional Calculus and Applied Analysis 16 (1) (2013) 93–108.

[8] D. Conte, I. D. Prete, Fast collocation methods for Volterra integral equations of convolution type, J. Comput. Appl. Math. 196 (2) (2006) 652–663.

[9] F. de Hoog, R. Weiss, High order methods for a class of Volterra integral equations with weakly singular kernels, SIAM J. Numer. Anal. 11 (1974) 1166–1180.

[10] L. Dieci, L. Lopez, Sliding motion in Filippov differential systems: theoretical results and a computational approach, SIAM J. Numer. Anal. 47 (3) (2009) 2023–2051.

[11] L. Dieci, L. Lopez, Sliding motion on discontinuity surfaces of high codimension. A construction for selecting a Filippov vector field, Numer. Math. 117 (4) (2011) 779–811.

[12] L. Dieci, L. Lopez, A survey of numerical methods for IVPs of ODEs with discontinuous right-hand side, J. Comput. Appl. Math. 236 (16) (2012) 3967–3991.

[13] K. Diethelm, An investigation of some nonclassical methods for the numerical approximation of Caputo-type fractional derivatives, Numer. Algorithms 47 (4) (2008) 361–390.

[14] K. Diethelm, The analysis of fractional differential equations, vol. 2004 of Lecture Notes in Mathematics, Springer-Verlag, Berlin, 2010.

[15] K. Diethelm, An efficient parallel algorithm for the numerical solution of fractional differential equations, Fract. Calc. Appl. Anal. 14 (3) (2011) 475–490.

[16] K. Diethelm, J. M. Ford, N. J. Ford, M. Weilbeer, Pitfalls in fast numerical solvers for fractional differential equations, J. Comput. Appl. Math. 186 (2) (2006) 482–503.

[17] K. Diethelm, N. J. Ford, A. D. Freed, Y. Luchko, Algorithms for the fractional calculus: a selection of numerical methods, Comput. Methods Appl. Mech. Engrg. 194 (6-8) (2005) 743–773.

[18] K. Diethelm, A. D. Freed, The FracPECE subroutine for the numerical solution of differential equations of fractional order, in: S.Heinzel, T.Plesser (eds.), Forschung und wissenschaftliches Rechnen 1998, 1999, pp. 57–71.

[19] J. Dixon, On the order of the error in discretization methods for weakly singular second kind Volterra integral equations with nonsmooth solutions, BIT 25 (4) (1985) 624–634.

[20] V. V. Gafiychuk, B. Datsko, Stability analysis and limit cycle in fractional system with Brusselator nonlinearities, Physics Letters A 372 (29) (2008) 4902 – 4904.

[21] L. Galeone, R. Garrappa, Fractional Adams-Moulton methods, Math. Comput. Simulation 79 (4) (2008) 1358–1367.

[22] R. Garra, Fractional-calculus model for temperature and pressure waves in fluid-saturated porous rocks, Phys. Rev. E 84 (2011) 036605.

[23] R. Garrappa, On linear stability of predictor–corrector algorithms for fractional differential equations, Int. J. Comput. Math. 87 (10) (2010) 2281–2290.

[24] R. Garrappa, On some generalizations of the implicit Euler method for discontinuous fractional differential equations, Math. Comput. Simulat. URL http://dx.doi.org/10.1016/j.matcom.2012.04.009

[25] R. Garrappa, M. Popolizio, On accurate product integration rules for linear fractional differential equations, J. Comput. Appl. Math. 235 (5) (2011) 1085–1097.

[26] R. Garrappa, M. Popolizio, Evaluation of generalized Mittag–Leffler functions on the real line, Adv. Comput. Math. 39 (1) (2013) 205–225.

[27] R. Gorenflo, Fractional calculus: some numerical methods, in: A. Carpinteri, F. Mainardi (eds.), Fractals and Fractional Calculus in Continuum Mechanics, vol. 278 of CISM Courses and Lectures, Springer Verlag, Wien and New York, 1997, pp. 277–290.

[28] E. Hairer, C. Lubich, M. Schlichte, Fast numerical solution of nonlinear Volterra convolution equations, SIAM J. Sci. Statist. Comput. 6 (3) (1985) 532–541.

[29] E. Hairer, G. Wanner, Solving ordinary differential equations. II, 2nd ed., Springer-Verlag, Berlin, 1996.

[30] P. Henrici, Applied and computational complex analysis, vol. 1, John Wiley & Sons, New York, 1974.

[31] P. Henrici, Fast Fourier methods in computational complex analysis, SIAM Rev. 21 (4) (1979) 481–527.

[32] A. Kadem, Y. Luchko, D. Baleanu, Spectral method for solution of the fractional transport equation, Rep. Math. Phys. 66 (1) (2010) 103–115.

[33] A. A. Kilbas, H. M. Srivastava, J. J. Trujillo, Theory and applications of fractional differential equations, vol. 204 of North-Holland Mathematics Studies, Elsevier Science B.V., Amsterdam, 2006.

[34] J. D. Lambert, Numerical methods for ordinary differential systems, John Wiley & Sons Ltd., Chichester, 1991.

[35] C. Li, F. Zeng, The finite difference methods for fractional ordinary differential equations, Numer. Funct. Anal. Optim. 34 (2) (2013) 149–179.

[36] C. Lubich, Runge-Kutta theory for Volterra and Abel integral equations of the second kind, Math. Comp. 41 (163) (1983) 87–102.

[37] C. Lubich, Fractional linear multistep methods for Abel–Volterra integral equations of the second kind, Math. Comp. 45 (172) (1985) 463–469.

[38] C. Lubich, Discretized fractional calculus, SIAM J. Math. Anal. 17 (3) (1986) 704–719.

[39] C. Lubich, A stability analysis of convolution quadratures for Abel-Volterra integral equations, IMA J. Numer. Anal. 6 (1) (1986) 87–101.

[40] J. T. Machado, V. Kiryakova, F. Mainardi, Recent history of fractional calculus, Commun. Nonlinear Sci. Numer. Simul. 16 (3) (2011) 1140–1153.

[41] R. L. Magin, Fractional calculus models of complex dynamics in biological tissues, Computers & Mathematics with Applications 59 (5) (2010) 1586 – 1593.

[42] F. Mainardi, Fractional calculus and waves in linear viscoelasticity, Imperial College Press, London, 2010.

[43] D. Matignon, Stability properties for generalized fractional differential systems, in: Systèmes différentiels fractionnaires (Paris, 1998), vol. 5 of ESAIM Proc., Soc. Math. Appl. Indust., Paris, 1998, pp. 145–158.

[44] M. M. Meerschaert, C. Tadjeran, Finite difference approximations for fractional advection-dispersion flow equations, J. Comput. Appl. Math. 172 (1) (2004) 65–77.

[45] M. M. Meerschaert, C. Tadjeran, Finite difference approximations for two-sided space-fractional partial differential equations, Appl. Numer. Math. 56 (1) (2006) 80–90.

[46] I. Moret, A note on Krylov methods for fractional evolution problems, Numer. Func. Anal. Opt. 34 (5) (2013) 539–556.

[47] I. Moret, P. Novati, On the convergence of Krylov subspace methods for matrix Mittag-Leffler functions, SIAM J. Numer. Anal. 49 (5) (2011) 2144–2164.

[48] I. Moret, M. Popolizio, The restarted shift-and-invert Krylov method for matrix functions, Numerical Linear Algebra with Applications.

[49] I. Podlubny, Fractional differential equations, vol. 198 of Mathematics in Science and Engineering, Academic Press Inc., San Diego, CA, 1999.

[50] J. Tenreiro Machado, P. Stefanescu, O. Tintareanu, D. Baleanu, Fractional calculus analysis of the cosmic microwave background, Romanian Reports in Physics 65 (1) (2013) 316–323.

[51] P. H. M. Wolkenfelt, Linear mulstistep methods and the construction of quadrature formulae for volterra integral and integro-differential equations, Tech. Rep. NW 76/79, Mathematisch Centrum,, Amsterdam (Netherlands) (1979).

[52] A. Young, Approximate product-integration, Proc. Roy. Soc. London Ser. A. 224 (1954) 552–561.

[53] S. B. Yuste, L. Acedo, An explicit finite difference method and a new von Neumann-type stability analysis for fractional diffusion equations, SIAM J. Numer. Anal. 42 (5) (2005) 1862–1874.

[54] S. B. Yuste, J. Quintana-Murillo, A finite difference method with non-uniform timesteps for fractional diffusion equations, Comput. Phys. Commun. 183 (12) (2012) 2594–2600.