Nonlinear Dynamics **29:** 3–22, 2002.

A Predictor-Corrector Approach for the Numerical Solution of Fractional Differential Equations

KAI DIETHELM
Institut für Angewandte Mathematik, Technische Universität Braunschweig, Pockelsstr. 14, D-38106 Braunschweig, Germany

NEVILLE J. FORD
Department of Mathematics, Chester College, Parkgate Road, Chester CH1 4BJ, United Kingdom

ALAN D. FREED
Polymers Branch, MS 49-3, NASA's John H. Glenn Research Center at Lewis Field, 21000 Brookpark Road, Cleveland, OH 44135, U.S.A.

(Received: 3 April 2001; accepted: 7 December 2001)

Abstract. We discuss an Adams-type predictor-corrector method for the numerical solution of fractional differential equations. The method may be used both for linear and for nonlinear problems, and it may be extended to multi-term equations (involving more than one differential operator) too.

Keywords: Fractional differential equation, Caputo derivative, numerical solution, predictor-corrector method, Adams method.

Nomenclature

D_*^α = Caputo type fractional derivative of order α
D^α = Riemann–Liouville type fractional derivative of order α
J^α = Riemann–Liouville type fractional integral of order α

1. Statement of the Problem

In this paper we want to discuss an algorithm for the numerical solution of differential equations of fractional order, equipped with suitable initial conditions. To be precise, we first look at the fractional differential equation

$$D_*^\alpha y(x) = f(x, y(x)), \tag{1}$$

where $\alpha > 0$ (but not necessarily $\alpha \in \mathbb{N}$). We shall determine a solution of this equation on the interval $[0, T]$, say, where T is a suitable positive number. At a later stage we will also investigate a more general problem (Section 4) but for the moment we restrict ourselves to (1) in order to explain the fundamental ideas behind our strategy.

It is well known [1–3] that there are many ways to define a fractional differential operator. The special operator D_*^α that we are using in (1) is defined by

$$D_*^\alpha y(x) = J^{m-\alpha} y^{(m)}(x), \tag{2}$$

where $m := \lceil \alpha \rceil$ is just the value α rounded up to the nearest integer, $y^{(m)}$ is the ordinary mth derivative of y (recall that m is an integer, so we are dealing with the classical situation), and

$$J^\beta z(x) = \frac{1}{\Gamma(\beta)} \int_0^x (x-t)^{\beta-1} z(t)\,\mathrm{d}t \tag{3}$$

is the Riemann–Liouville integral operator of order $\beta > 0$. It is common practice to call the operator D_*^α the *Caputo differential operator* of order α because it has apparently first been used for the solution of practical problems by Caputo [4]. Note however that, according to Butzer and Westphal [1, p. 11], it had already been introduced in a 19th century paper of Liouville.

A typical feature of differential equations (classical or fractional) is that one needs to specify additional conditions to make sure that the solution is unique. In many situations these additional conditions describe certain properties of the solution at the beginning of the process, i.e., at the point $x = 0$. Therefore such a problem is called an initial value problem. It is easily seen that the number of initial conditions that one needs to specify in order to obtain a unique solution is $m = \lceil \alpha \rceil$. In particular, if $0 < \alpha \leq 1$ (which is the case in many applications), we have to specify just one condition. However the precise form of this condition is not arbitrary. Rather, when fractional differential equations are concerned, it turns out that there is a close connection between the type of the initial condition and the type of the fractional derivative. This is actually also the reason for us to choose the Caputo derivative and not the Riemann–Liouville derivative that is more commonly used in pure mathematics: For the Riemann–Liouville case, one would have to specify the values of certain fractional derivatives (and integrals) of the unknown solution at the initial point $x = 0$ (cf. [3, §42]). However, when we are dealing with a concrete physical application then the unknown quantity y will typically have a certain physical meaning (e.g. a dislocation), but it is not clear what the physical meaning of a fractional derivative of y is, and hence it is also not clear how such a quantity can be measured. In other words, the required data simply will not be available in practice. When we deal with Caputo derivatives however, the situation is different. We may specify the initial values $y(0), y'(0), \ldots, y^{(m-1)}(0)$, i.e., the function value itself and integer-order derivatives [5]. These data typically have a well understood physical meaning and can be measured.

Note that Lorenzo and Hartley [6] have discussed the problem of finding the correct form of the initial conditions in a more general setting (not necessarily assuming that the entire history of the process can be observed).

Thus we combine our fractional differential equation (1) with initial conditions

$$y^{(k)}(0) = y_0^{(k)}, \quad k = 0, 1, \ldots, m-1, \tag{4}$$

where once again $m = \lceil \alpha \rceil$ and the real numbers $y_0^{(k)}$, $k = 0, 1, \ldots, m-1$, are assumed to be given. It then turns out that, under some very weak conditions on the function f on the right-hand side of the differential equation, we can indeed say that a solution exists and that this solution is uniquely determined [5]. Specifically these conditions are (a) the continuity of f with respect to both its arguments and (b) a Lipschitz condition with respect to the second argument. We explicitly stress that these are the only conditions on f that we assume; in particular, as we shall see in Section 2, there is no need to impose a linearity assumption on f when we derive and discuss our scheme. Thus we now have a tool that allows a convenient

solution of nonlinear problems. Apparently the development of such nonlinear models has in the past been hampered by the lack of suitable (numerical or analytical) solution methods.

We refer to [5] for a more thorough mathematical analysis of the initial value problem defined by Equations (1) and (4). Some additional properties of the Caputo differential operator D_*^α are discussed in [2, 7, 8].

Many authors formally use Riemann–Liouville fractional derivatives, defined by

$$D^\alpha y(x) = \frac{\mathrm{d}^m}{\mathrm{d}x^m} J^{m-\alpha} y(x),$$

where once again $m = \lceil \alpha \rceil$, instead of Caputo derivatives. Typically those authors then require homogeneous initial conditions. It is known [2] that under those homogeneous conditions the equations with Riemann-Liouville operators are equivalent to those with Caputo operators. We chose the Caputo version because it allows us to specify inhomogeneous initial conditions too if this is desired. As we have described above, for the Riemann–Liouville approach this generalization is connected with major practical difficulties (cf., e.g., [9, 10]).

A few numerical methods for fractional differential equations have been presented in the literature (cf., e.g., [2, 7, 11–26]). However many of these methods are essentially ad hoc methods for very specific types of differential equations (often just linear equations or even smaller classes). It is not clear if they can be generalized and how they behave when they are applied to an equation that does not fit into the class for which the algorithm has originally been derived. Our scheme, on the other hand, has been constructed and analyzed for the fully general set of equations without any special assumptions, and is easy to implement on a computer. We therefore believe that it will be of practical significance and helpful for solving a broad class of problems.

2. The Predictor-Corrector Algorithm

In this section we shall derive the fundamental algorithm that we have developed for the solution of initial value problems with Caputo derivatives. The algorithm is a generalization of the classical Adams–Bashforth–Moulton integrator that is well known for the numerical solution of first-order problems [27, 28].

Our approach is based on the analytical property that the initial value problem (1), (4) is equivalent to the Volterra integral equation

$$y(x) = \sum_{k=0}^{\lceil \alpha \rceil - 1} y_0^{(k)} \frac{x^k}{k!} + \frac{1}{\Gamma(\alpha)} \int_0^x (x-t)^{\alpha-1} f(t, y(t))\, \mathrm{d}t \tag{5}$$

in the sense that a continuous function is a solution of the initial value problem if and only if it is a solution of (5). For a brief derivation of this equivalence we refer to [5, lemma 2.3]. Note that the sum outside the integral on the right-hand side is completely determined by the initial values and hence is known.

In order to motivate the construction of our numerical method, we shall first briefly recall the idea behind the classical one-step Adams–Bashforth–Moulton algorithm for first-order equations. So, for a start, we focus our attention on the well-known initial-value problem for the first-order differential equation

$$Dy(x) = f(x, y(x)), \tag{6a}$$

$$y(0) = y_0. \tag{6b}$$

We assume the function f to be such that a unique solution exists on some interval $[0, T]$, say. Following [27, §III.1], we suggest the use of the predictor-corrector technique of Adams where, for the sake of simplicity, we assume that we are working on a uniform grid $\{t_n = nh : n = 0, 1, \ldots, N\}$ with some integer N and $h := T/N$. The basic idea is, assuming that we have already calculated approximations $y_h(t_j) \approx y(t_j)$ $(j = 1, 2, \ldots, n)$, that we try to obtain the approximation $y_h(t_{n+1})$ by means of the equation

$$y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} f(z, y(z))\,\mathrm{d}z. \tag{7}$$

This equation follows upon integration of (6a) on the interval $[t_n, t_{n+1}]$. Of course, we know neither of the expressions on the right-hand side of Equation (7) exactly, but we do have an approximation for $y(t_n)$, namely $y_h(t_n)$, that we can use instead. The integral is then replaced by the two-point trapezoidal quadrature formula

$$\int_a^b g(z)\,\mathrm{d}z \approx \frac{b-a}{2}\left(g(a) + g(b)\right),$$

thus giving an equation for the unknown approximation $y_h(t_{n+1})$, it being

$$y_h(t_{n+1}) = y_h(t_n) + \frac{h}{2}[f(t_n, y(t_n)) + f(t_{n+1}, y(t_{n+1}))],$$

where again we have to replace $y(t_n)$ and $y(t_{n+1})$ by their approximations $y_h(t_n)$ and $y_h(t_{n+1})$, respectively. This yields the equation for the implicit one-step Adams–Moulton method, which is

$$y_h(t_{n+1}) = y_h(t_n) + \frac{h}{2}\left[f(t_n, y_h(t_n)) + f(t_{n+1}, y_h(t_{n+1}))\right]. \tag{8}$$

The problem with this equation is that the unknown quantity $y_h(t_{n+1})$ appears on both sides, and due to the nonlinear nature of the function f, we cannot solve for $y_h(t_{n+1})$ directly in general. Therefore, we may use Equation (8) in an iterative process, inserting a preliminary approximation for $y_h(t_{n+1})$ in the right-hand side in order to determine a better approximation that we can then use.

The required preliminary approximation $y_h^{\mathrm{P}}(t_{n+1})$, the so-called predictor, is obtained in a very similar way, only replacing the trapezoidal quadrature formula by the rectangle rule

$$\int_a^b g(z)\,\mathrm{d}z \approx (b-a)g(a),$$

giving the explicit method

$$y_h^{\mathrm{P}}(t_{n+1}) = y_h(t_n) + hf(t_n, y_h(t_n)), \tag{9}$$

known as the forward Euler or one-step Adams–Bashforth method. It is well known [27, p. 372] that the process defined by Equation (9) and

$$y_h(t_{n+1}) = y_h(t_n) + \frac{h}{2}\left[f(t_n, y_h(t_n)) + f(t_{n+1}, y_h^{\mathrm{P}}(t_{n+1}))\right], \tag{10}$$

known as the one-step Adams–Bashforth–Moulton technique, is convergent of order 2, i.e.,

$$\max_{n=1,2,\ldots,N} |y(t_n) - y_h(t_n)| = O(h^2).$$

Moreover, this method behaves satisfactorily from the point of view of its numerical stability [28, chap. IV]. It is said to be of the PECE (Predict, Evaluate, Correct, Evaluate) type because, in a concrete implementation, we would start by calculating the predictor in Equation (9), then evaluate $f(t_{n+1}, y_h^{\mathrm{P}}(t_{n+1}))$, use this to calculate the corrector in Equation (10), and finally evaluate $f(t_{n+1}, y_h(t_{n+1}))$. This result is stored for future use in the next integration step.

Having introduced this concept, we now try to carry over the essential ideas to the fractional-order problem with some unavoidable modifications. The key is to derive an equation similar to (7). Fortunately, such an equation is available, namely Equation (5). This equation looks somewhat different from Equation (7), because the range of integration now starts at 0 instead of t_n. This is a consequence of the non-local structure of the fractional-order differential operators. This however does not cause major problems in our attempts to generalize the Adams method. We simply use the product trapezoidal quadrature formula to replace the integral, where nodes t_j ($j = 0, 1, \ldots, n+1$) are taken with respect to the weight function $(t_{n+1} - \cdot)^{\alpha-1}$. In other words, we apply the approximation

$$\int_0^{t_{n+1}} (t_{n+1} - z)^{\alpha-1} g(z)\,\mathrm{d}z \approx \int_0^{t_{n+1}} (t_{n+1} - z)^{\alpha-1} \tilde{g}_{n+1}(z)\,\mathrm{d}z, \tag{11}$$

where $\tilde{g}_{n+1}$ is the piecewise linear interpolant for g with nodes and knots chosen at the t_j, $j = 0, 1, 2, \ldots, n+1$. Using standard techniques from quadrature theory (cf. [29]), we find that we can write the integral on the right-hand side of (11) as

$$\int_0^{t_{n+1}} (t_{n+1} - z)^{\alpha-1} \tilde{g}_{n+1}(z)\,\mathrm{d}z = \frac{h^\alpha}{\alpha(\alpha+1)} \sum_{j=0}^{n+1} a_{j,n+1} g(t_j),$$

where

$$a_{j,n+1} = \begin{cases} n^{\alpha+1} - (n-\alpha)(n+1)^\alpha, & \text{if } j = 0, \\ (n-j+2)^{\alpha+1} + (n-j)^{\alpha+1} - 2(n-j+1)^{\alpha+1}, & \text{if } 1 \le j \le n, \\ 1, & \text{if } j = n+1. \end{cases} \tag{12}$$

This then gives us our corrector formula (i.e., the fractional variant of the one-step Adams–Moulton method), which is

$$\begin{aligned} y_h(t_{n+1}) \;=\; & \sum_{k=0}^{\lceil\alpha\rceil-1} \frac{t_{n+1}^k}{k!} y_0^{(k)} + \frac{h^\alpha}{\Gamma(\alpha+2)} f(t_{n+1}, y_h^{\mathrm{P}}(t_{n+1})) \\ & + \frac{h^\alpha}{\Gamma(\alpha+2)} \sum_{j=0}^{n} a_{j,n+1} f(t_j, y_h(t_j)), \end{aligned} \tag{13}$$

where we have used the functional equation of the Gamma function twice (which implies the identity $\Gamma(\alpha)\alpha(\alpha+1) = \Gamma(\alpha+2)$) and the fact that $a_{n+1,n+1} = 1$.

The remaining problem is the determination of the predictor formula that we require to calculate the value $y_h^{\mathrm{P}}(t_{n+1})$. The idea we use to generalize the one-step Adams–Bashforth

method is the same as the one described above for the Adams–Moulton technique: we replace the integral on the right-hand side of Equation (5) by the product rectangle rule

$$\int_0^{t_{n+1}} (t_{n+1} - z)^{\alpha-1} g(z)\,\mathrm{d}z \approx \sum_{j=0}^{n} b_{j,n+1} g(t_j),$$

where now

$$b_{j,n+1} = \frac{h^\alpha}{\alpha}((n+1-j)^\alpha - (n-j)^\alpha) \tag{14}$$

(see also [29]). Thus, the predicted value $y_h^{\mathrm{P}}(t_{n+1})$ is determined by the fractional Adams–Bashforth method

$$y_h^{\mathrm{P}}(t_{n+1}) = \sum_{k=0}^{\lceil\alpha\rceil-1} \frac{t_{n+1}^k}{k!} y_0^{(k)} + \frac{1}{\Gamma(\alpha)} \sum_{j=0}^{n} b_{j,n+1} f(t_j, y_h(t_j)). \tag{15}$$

Our basic algorithm, the fractional Adams–Bashforth–Moulton method, is fully described now by Equations (15) and (13) with the weights $a_{j,n+1}$ and $b_{j,n+1}$ being defined according to (12) and (14), respectively.

We have thus completed the description of our numerical algorithm. The mathematical analysis of this method in [30] shows that we may expect the error to behave as

$$\max_{j=0,1,\ldots,N} |y(t_j) - y_h(t_j)| = O(h^p), \tag{16a}$$

where

$$p = \min(2, 1+\alpha) \tag{16b}$$

and the quantities h and N are related according to $h = T/N$, and T is the upper bound of the interval on which we are looking for the solution. The reason for this rather special form of the exponent p is that one can prove that p must be the minimum of the order of the corrector (which is 2 in our case) and the order of the predictor method (which is 1 here) plus the order of the differential operator (viz., α). This is a well known fact for PECE algorithms for first-order equations (the case $\alpha = 1$). In view of the fact that $1 < p \leq 2$ we have a satisfactory globally valid error bound. We note that, quite in contrast to the behaviour of the algorithm described in [14], the convergence order p of the Adams–Bashforth–Moulton scheme increases as α, the order of the differential equation, increases.

As far as the stability properties of our algorithm are concerned, we note that it follows (unsing the general methods of Lubich [22]) that the stability properties of this fractional Adams–Bashforth–Moulton scheme are at least as good as the corresponding properties of its counterpart for first-order equations, i.e., the classical second-order Adams–Bashforth–Moulton method. The properties of the latter are described in detail, e.g., in [28, chap. IV].

For a more detailed investigation, including numerical examples, we refer to [30]. The remainder of this paper will be devoted to more practical aspects. Specifically in Section 3 we shall introduce some modifications that can enhance the efficiency of our scheme, and in Section 4 we extend the capabilities of our integrator so that it can handle a more general class of equations. Finally we present some numerical examples in Section 6, and for the

convenience of the reader we also provide an Appendix where the algorithm is stated once again, but in a pseudo-code type of notation.

3. Modifications of the Algorithm

We have completed the description of the basic algorithm and its most important properties. Now we want to address the question of whether it is possible to improve the performance of the algorithm in certain respects. There are indeed a few ways to do this, and we shall mention some here. It is easily seen that most of these modifications are independent of each other, so the user may implement almost any combination of them as required.

3.1. Reduction of the Arithmetic Complexity

First of all, there is a fundamental problem associated with all fractional differential operators (not only the Caputo version that we look at here): In contrast to differential operators of integer order, fractional derivatives are not local operators. This means that we cannot calculate $D_*^\alpha y(x)$, say, solely on the basis of function values of y taken from a neighbourhood of the point x where we work. Rather, we have to take the entire history of y (i.e., all function values $y(t)$ for $0 \le t \le x$) into account. This is clearly exhibited in the definition of the operator, cf. Equation (2). As a matter of fact, this property is highly desirable from the physical point of view because it allows us to model phenomena with memory effects. But when it comes to numerical work we find that this non-locality leads to a significantly higher computational effort: The arithmetic complexity of our algorithm with step size h is $O(h^{-2})$, whereas a comparable algorithm for an integer-order initial value problem (thus involving a local operator) would only give rise to a $O(h^{-1})$ complexity. A number of ways have been suggested to overcome this difficulty. The first idea seems to have been the fixed memory principle of Podlubny [2]. However a close mathematical analysis [29] reveals that the reduced complexity is achieved at the price of a significant loss in the order of accuracy of the method. A more promising approach seems to be the nested memory concept of Ford and Simpson [31] that may well be applied to our algorithm. This concept leads to an $O(h^{-1} \log h^{-1})$ complexity (so almost all of the additional work introduced by the non-locality is removed), and it can be shown that this idea retains the order of accuracy of the underlying algorithm. For more details we refer to [31].

3.2. Additional Corrector Iterations

Recall that in the case of a very stiff equation, we mentioned that the stability properties of the Adams–Bashforth–Moulton integrator may not be sufficient. However, it is well known [27, 28] that the pure one-step Adams–Moulton method (i.e., the trapezoidal method) possesses extremely good stability properties. These are spoiled in the Adams–Bashforth–Moulton approach only by the fact that in Equation (13) we cannot replace the predictor approximation y_{n+1}^{P} on the right-hand side by the corrector value y_{n+1} because, in general, we cannot solve that equation exactly any more. The idea is now to find a better approximation for the exact solution of that equation than the rather simple one obtained by applying just one functional corrector iteration with the predictor as a starting value.

There are two main ways to achieve this goal. The first one is to use the value obtained by the first iteration (the first corrector step) as a new predictor and apply another corrector step.

Obviously, this procedure can be iterated any given number of times, M say; the resulting method is called a P(EC)^{M}E algorithm. In this way, we find a method that is 'closer' to the pure Adams–Moulton technique, and therefore, its stability properties are also closer to the better properties of this method. Taking this idea to the extreme, we could even refrain from stopping after M iterations and continue to iterate until convergence is achieved. This would (theoretically) lead to even better stability, but the computational cost could be prohibitive, and it may even happen that (due to rounding errors) convergence could not be achieved numerically in finite precision. Moreover the iteration may not converge unless the step size h of the method is sufficiently small. It is possible to make the P(EC)^{M}E approach more efficient by not using the same number M of corrector iterations in every step. In regions of higher stiffness, more steps may be taken in order to retain stability and to keep the error under control; whereas, in regions where stiffness is not a problem, high accuracy may be achieved already with a small choice of M, thus speeding up the algorithm.

In practice, one can implement this feature in such a way that the user can supply an upper bound for the number M of corrector steps to be taken. Setting this bound to 1 is then equivalent to switching off this modification completely. Moreover, the user can specify a tolerance $\varepsilon > 0$ to the effect that the iteration is stopped if two consecutive steps give results that differ by less than this tolerance even if the maximum number of iterations has not been reached.

In Section 4 we shall see that there also may exist other reasons for replacing the plain PECE structure by a more complex P(EC)^{M}E version with some suitable M.

As mentioned above, this is not the only possible approach to get a better approximation to the solution of the corrector equation. The second idea that we may use to enhance the stability of the method is to find another way of solving Equation (13) with y_{n+1}^{P} replaced by y_{n+1}. The most obvious way to do this would be to use a Newton iteration. As a starting value, we can still use the Adams–Bashforth predictor. Since it is known that Newton iteration converges faster (locally) than the simple functional iteration of the PECE process, we may expect to come very close to the pure Adams–Moulton method in just a few iterations. However, in order to implement Newton's method, we need to work with the Jacobian of the right-hand side of the differential equation. This can also be a source of numerical problems, and may even lead to very long run times. We have thus not implemented and tested this so far, but a future extension in this way is possible.

Note that, since we keep the Adams–Moulton formula as the basis of Equation (13), the convergence order of these modified algorithms remains unchanged. Moreover, this modification of the algorithm does not alter the order of the arithmetic complexity in terms of the step size h. Only the stability is affected by these modifications. Further, we stress that this modification (unlike the P(EC)^{M}E scheme) does not avoid the problems that we shall encounter in certain of the situations mentioned in Section 4.

3.3. Richardson Extrapolation

Whereas the suggestions above were constructed in order to improve the run time or the stability of the algorithm without changing the convergence behaviour, we now turn our attention towards a possibile way of speeding up the convergence. As noted in [30], numerical evidence suggets that the error of our Adams scheme, taken to approximate the exact solution at a fixed point $T > 0$, possesses an asymptotic expansion of the form

$$y(T) - y_h(T) = \sum_{j=1}^{k_1} c_j h^{2j} + \sum_{j=1}^{k_2} d_j h^{j+\alpha} + O(h^{k_3}), \tag{17}$$

where k_1, k_2 and k_3 are certain constants depending only on the solution y and satisfying $k_3 > \max(2k_1, k_2+\alpha)$. In practice it is unlikely that these constants can be determined explicitly, but as a rule of thumb one may often work with small or moderate values like $k_1 = 3$ and $k_2 = 5$. We shall see below that knowledge of the value of k_3 is only necessary in order to derive an error bound but not in order to actually implement the convergence speedup procedure.

Notice that the asymptotic expansion begins with an h^2 term and continues with $h^{1+\alpha}$ for $1 < \alpha < 3$, whereas it begins with $h^{1+\alpha}$, followed by h^2, for $0 < \alpha < 1$.

Our belief in the truth of this conjecture is not only supported by numerical results but also by the theoretical analysis of de Hoog and Weiss [32, §5] who show that asymptotic expansions of this form hold if we use the fractional Adams–Moulton method (i.e., if we solve the corrector equation exactly) and that a similar expansion can be derived for the fractional Adams–Bashforth method (using the predictor as the final approximation rather than correcting once with the Adams–Moulton formula). Assuming that the conjecture is correct, we may use the relation (17) to eliminate the leading terms and thus obtain an approximation that converges faster than the original one. According to the usual constructions (cf., e.g., [33]), this results in a Richardson extrapolation scheme that has the following form. We begin by sorting the exponents of h in Equation (17) in increasing order, thus obtaining a sequence $j_1, j_2, j_3, \ldots$ where, in the case $0 < \alpha < 1$, we have $j_1 = 1 + \alpha$, $j_2 = 2$, $j_3 = 2 + \alpha$, $j_4 = 3 + \alpha$, $j_5 = 4$, $j_6 = 4 + \alpha$ etc., whereas for $1 < \alpha < 2$ we have $j_1 = 2$, $j_2 = 1 + \alpha$, $j_3 = 2 + \alpha$, $j_4 = 4$, $j_5 = 3 + \alpha$, $j_6 = 4 + \alpha$, etc. Obviously, we can proceed in a similar way for the case $\alpha > 2$, but since this latter case seems to be of minor practical interest, we omit the details. Using this newly introduced terminology we can rewrite the asymptotic expansion (17) for the error in the form

$$y(T) - y_h(T) = \sum_{k=1}^{K-1} c_k^* h^{j_k} + O(h^{j_K}) \tag{18}$$

with certain coefficients c_k^*, some $K \in \mathbb{N}$ and some positive real numbers $j_1 < j_2 < \ldots < j_K$. Note that in practice the j_k are known by the considerations above whereas the coefficients c_k^* are in general unknown.

The key observation is that we can use Equation (18) to improve the accuracy of our approximation even though we do not know those coefficients c_k^*. We proceed by setting up a triangular array (a so-called Romberg tableau) of approximation values for $y(T)$ of the form

$$\begin{array}{cccccc}
y_h^{(0)} & & & & & \\
y_{h/2}^{(0)} & y_{h/2}^{(1)} & & & & \\
y_{h/4}^{(0)} & y_{h/4}^{(1)} & y_{h/4}^{(2)} & & & \\
y_{h/8}^{(0)} & y_{h/8}^{(1)} & y_{h/8}^{(2)} & y_{h/8}^{(3)} & & \\
\vdots & \vdots & \vdots & \vdots & \ddots &
\end{array} \tag{19}$$

The leftmost column of this array consists of the approximations determined by our Adams algorithm with the step sizes h, $h/2$, $h/4$, etc.; i.e., $y_h^{(0)} = y_h(T)$. We may terminate the

succesive step size halving at any time we wish; this then determines the number of rows in the tableau. Having set up this (the zeroth) column, we turn our attention towards the next column. To determine these values, we use the asymptotic expansion and see that we may calculate a linear combination

$$y_{h/2^\mu}^{(1)} := \frac{2^{j_1} y_{h/2^\mu}^{(0)} - y_{h/2^{\mu-1}}^{(0)}}{2^{j_1} - 1}.$$

Then, in view of (18),

$$\begin{aligned}
y(T) - y_{h/2^\mu}^{(1)} &= \frac{2^{j_1} y(T) - y(T)}{2^{j_1} - 1} - \frac{2^{j_1} y_{h/2^\mu}^{(0)} - y_{h/2^{\mu-1}}^{(0)}}{2^{j_1} - 1} \\
&= \frac{1}{2^{j_1} - 1}\left[2^{j_1}(y(T) - y_{h/2^\mu}(T)) - (y(T) - y_{h/2^{\mu-1}}(T))\right] \\
&= \frac{2^{j_1}}{2^{j_1} - 1}\left(\sum_{k=1}^{K-1} c_k^*(2^{-\mu}h)^{j_k} + O((2^{-\mu}h)^{j_K})\right) \\
&\quad - \frac{1}{2^{j_1} - 1}\left(\sum_{k=1}^{K-1} c_k^*(2^{1-\mu}h)^{j_k} + O((2^{1-\mu}h)^{j_K})\right) \\
&= \frac{2^{j_1}}{2^{j_1} - 1} c_1^*(2^{-\mu}h)^{j_1} - \frac{1}{2^{j_1} - 1} c_1^*(2^{1-\mu}h)^{j_1} \\
&\quad + \sum_{k=2}^{K-1} c_k^{**}(2^{-\mu}h)^{j_k} + O((2^{-\mu}h)^{j_K}) \\
&= \sum_{k=2}^{K-1} c_k^{**}(2^{-\mu}h)^{j_k} + O((2^{-\mu}h)^{j_K}).
\end{aligned}$$

We can thus see that the h^{j_1} term has been eliminated, and therefore the first column converges towards the true solution as $O(h^{j_2})$ which is faster than the zeroth column. Moreover the error in the first column has got an asymptotic expansion that formally coincides with that of the zeroth column, except that the leading term is missing. Thus we may apply the same scheme again and again, and then we find that the νth column is determined by the relation

$$y_{h/2^\mu}^{(\nu)} = \frac{2^{j_\nu} y_{h/2^\mu}^{(\nu-1)} - y_{h/2^{\mu-1}}^{(\nu-1)}}{2^{j_\nu} - 1}.$$

There is almost no computational work involved in the calculation of the right part of the table: only a few elementary operations need to be done. This is completely negligible when compared to the effort required for the zeroth column (where the basic Adams algorithm really needs to be carried out). It then turns out that the νth column converges to $y(T)$ with an error of the order $O(h^{j_{\nu+1}})$.

For the fractional differential equation algorithm presented in [14] (that is less suitable for nonlinear problems) a similar extrapolation strategy has been used successfully [16]. Note however that the exponents of the h-terms in the asymptotic expansions for that algorithm are different from the exponents for our Adams scheme.

We note that, according to Ford and Simpson [31], the extrapolation procedure may also be applied (with the same exponents) if we replace our original Adams scheme by the modification mentioned in Section 3.1.

3.4. General Meshes

Up to this point we have assumed the mesh points $t_0, t_1, \ldots, t_N$ to be equispaced. In some practical applications it may be useful to relax this requirement. Indeed it is possible to do so. Of course we then find that the predictor weights $b_{j,n+1}$ and the corrector weights $a_{j,n+1}$ need to be changed. The corresponding expressions have been derived in [10, 29]. In this case the validity of the asymptotic expansion (and hence the possibility of working with Richardson extrapolation) will be destroyed, and in the error estimates the parameter h will then represent the maximum of the distances of consecutive mesh points.

4. Multi-Term Fractional Differential Equations

We now want to extend our algorithm to a more general class of equations. To this end we introduce the multi-term initial value problem

$$D_*^{\alpha_n} y(x) = f(x, y(x), D_*^{\alpha_1} y(x), \ldots, D_*^{\alpha_{n-1}} y(x)), \tag{20a}$$

$$y^{(k)}(0) = y_0^{(k)}, \quad k = 0, 1, \ldots, m-1, \tag{20b}$$

where now $m = \lceil \alpha_n \rceil$, and where we assume that the orders of the differential operators are sorted in such a way that $0 < \alpha_1 < \alpha_2 < \ldots < \alpha_n$. Since we have a total of n differential operators in (20a), we say that this is an n-term fractional differential equation. Obviously, we may recover the class of problems that we have considered so far upon setting $n = 1$ and $\alpha := \alpha_n$. There are a number of instances where such multi-term problems arise; the earliest example seems to be the Bagley-Torvik equation

$$D_*^2 y(x) = b D_*^{3/2} y(x) + c y(x) + f(x)$$

that describes the motion of a rigid plate immersed into a Newtonian viscous fluid (cf. [9, 34]). Other special cases are Babenko's model

$$D_*^1 y(x) = F(x) D_*^{1/2} y(x) + G(x) y(x) + H(x)$$

of a gas dissolved in a liquid [2, §8.3.3], or Koeller's model

$$D_*^{2\alpha} y(x) = p_1 D_*^{\alpha} y(x) + p_0 y(x) + f(x)$$

for a copolymer [35], where α is some fixed real number.

Our strategy for the numerical solution of such initial value problems is based on the classical approach for higher-order differential equations: We want to rewrite the given problem as an equivalent system of equations, where each of those new equations contains only one differential operator and where all these differential operators are of the same order. It is easily oberved [36] however that this is possible only under some additional number-theoretic assumptions on the parameters $\alpha_1, \alpha_2, \ldots, \alpha_n$. To be precise, we must assume them

– either to be rational numbers, or

– to be commensurate real numbers (i.e., all the ratios α_j/α_k must be rational) contained in the interval $(0, 1)$.

Therefore we need to perform some preliminary manipulations before we can follow this path in the general case. Our algorithm thus consists of two stages.

In the first stage we replace the given initial value problem (20) by the related problem

$$D_*^{\tilde{\alpha}_n} \tilde{y}(x) = f(x, \tilde{y}(x), D_*^{\tilde{\alpha}_1} \tilde{y}(x), \ldots, D_*^{\tilde{\alpha}_{n-1}} \tilde{y}(x)), \tag{21a}$$

with initial conditions

$$\tilde{y}^{(k)}(0) = y_0^{(k)}, \quad k = 0, 1, \ldots, m-1, \tag{21b}$$

with $m = \lceil \alpha_n \rceil$. That is, we perturb the orders of the differential operators in the given equation, but all the other given data (the function f on the right-hand side and the initial values) remain unchanged. The perturbed orders $\tilde{\alpha}_1, \ldots, \tilde{\alpha}_n$ are chosen according to the following criteria:

(a) $\tilde{\alpha}_1, \ldots, \tilde{\alpha}_n$ must be rational numbers,
(b) $\lceil \alpha_n \rceil = \lceil \tilde{\alpha}_n \rceil$,
(c) $\gcd(1, \tilde{\alpha}_1, \ldots, \tilde{\alpha}_n)$ should be as large as possible,
(d) $|\alpha_j - \tilde{\alpha}_j|$ should be as small as possible for all j.

One can interpret this approach as a careful refinement of an idea sketched in [2, §9.2] in connection with applications in control theory.

The essence of condition (a) is that it allows us to rewrite the newly constructed initial value problem as the system of simple equations that we want to have.

Condition (b) asserts that both the original problem (20) and the perturbed problem (21) require the same number of initial conditions.

In condition (c), we denote by gcd the greatest common divisor of its arguments. This concept is to be understood in the generalized sense here because the α_j are not natural numbers. This means that

$$\gcd(z_1, \ldots, z_n) = \max\{q \in \mathbb{Q} : z_j/q \in \mathbb{N} \quad \forall j = 1, 2, \ldots, n\},$$

so that the greatest common divisor is the largest rational number that divides all the given numbers, and by "divides" we mean that the division gives a natural number as a result, without any remainder. As we shall see below, the size of this greatest common divisor essentially determines the size of the resulting system of equations and hence the computational complexity.

Condition (d) assures $\tilde{y} \approx y$ because, as shown in [36],

$$\|y - \tilde{y}\|_\infty = O\left(\max_{j=1,2,\ldots,n} |\alpha_j - \tilde{\alpha}_j|\right).$$

We can thus make sure that the solution $\tilde{y}$ of the new problem (21) does not differ too much from the solution y of the original problem (20).

It is obvious that the parameters $\tilde{\alpha}_1, \ldots, \tilde{\alpha}_n$ are not determined uniquely by the four conditions, so there is some freedom for the user to trade off accuracy against speed.

Note here that, to some extent, conditions (c) and (d) contradict each other: one can typically make the difference $|\tilde{\alpha}_j - \alpha_j|$ smaller by choosing $\tilde{\alpha}_j$ as a rational number with a larger denominator, but this at the same time makes the greatest common divisor smaller. In a practical application one thus needs to balance these two conditions and find a useful compromise.

In the second stage of our approach we want to solve the new initial value problem (21) numerically. In order to describe this second stage we introduce some notation. By

$$\gamma := \gcd(1, \tilde{\alpha}_1, \ldots, \tilde{\alpha}_n)$$

we denote the greatest common divisor (already encountered in condition (c)). Moreover we set

$$\tilde{N} := \frac{\tilde{\alpha}_n}{\gamma}.$$

Then we set up the system

$$\begin{aligned}
D_*^\gamma y_0(x) &= y_1(x), \\
D_*^\gamma y_1(x) &= y_2(x), \\
&\vdots \quad \vdots \\
D_*^\gamma y_{\tilde{N}-2}(x) &= y_{\tilde{N}-1}(x), \\
D_*^\gamma y_{\tilde{N}-1}(x) &= f(x, y_0(x), y_{\tilde{\alpha}_1/\gamma}(x), \ldots, y_{\tilde{\alpha}_{n-1}/\gamma}(x))
\end{aligned} \tag{22a}$$

with initial conditions

$$y_j(0) = \begin{cases} y_0^{(j\gamma)}, & \text{for } j\gamma \in \mathbb{N}_0, \\ 0, & \text{else,} \end{cases} \tag{22b}$$

and find [36] that this is equivalent to the perturbed initial value problem in the following sense. Assume that the solution of the system is the vector-valued function $(y_0, \ldots, y_{\tilde{N}-1})$, and that $\tilde{y}$ is the solution of (21). Then $y_0 = \tilde{y}$.

We can immediately see that the dimension of the system is $\tilde{N}$, and by definition of this quantity we see that this is large if γ is small and vice versa. This justifies condition (c) above.

The initial conditions (22b) correspond to the given initial conditions in the sense that those components of $(y_0, \ldots, y_{\tilde{N}-1})$ that correspond to an integer-order derivative of $\tilde{y}$ are assigned that initial value, whereas all other initial values are set to zero. A mathematical justification of this choice is given in [9]; a physical argument can be found in [37].

Then we note that we can rewrite this $\tilde{N}$-dimensional problem in more compact vector notation as

$$D_*^\gamma \mathbf{Y}(x) = \mathbf{F}(x, \mathbf{Y}(x)) \tag{23a}$$

with initial condition

$$\mathbf{Y}(0) = \mathbf{Y}_0, \tag{23b}$$

where now $\mathbf{Y} = (y_0, y_1, \ldots, y_{\tilde{N}-1})^{\mathrm{T}}$ is a function of $\tilde{N}$ variables that maps to $\mathbb{R}^{\tilde{N}}$,

$$\mathbf{F}(x, \mathbf{Y}(x)) = \begin{pmatrix} y_1(x) \\ y_2(x) \\ \vdots \\ y_{\tilde{N}-1}(x) \\ f(x, y_0(x), y_{\tilde{\alpha}_1/\gamma}(x), \ldots, y_{\tilde{\alpha}_{n-1}/\gamma}(x)) \end{pmatrix},$$

and $\mathbf{Y}_0 = (y_0(0), y_1(0), \ldots, y_{\tilde{N}-1}(0))^{\mathrm{T}}$ with the components of the vector being as in (22b). Formally the initial value problem (23) is identical with the problem consisting of Equations (1) and (4) that we have considered in Sections 1, 2 and 3, the only difference being that we now consider vectors $\mathbf{Y}$ and $\mathbf{F}$ instead of scalars y and f. This difference produces no problems at all since we can simply apply the scalar scheme in a componentwise fashion to each component of the vector problem, and we find that all considerations made above (like the details of the implementation, the error analysis, and the possible modifications) remain valid. So we can now apply the Adams scheme to the vector problem and use the first component of the numerical solution $\mathbf{Y}_h$ say, viz. the approximation for the function $y_0(x)$, as an approximation for the desired solution y of the original multi-term fractional differential equation (20).

Obviously, the error of this two-stage approximation scheme can be decomposed additively into two parts, each of which can be attributed to one stage of the scheme, as

$$y(x) - y_{0,h}(x) = (y(x) - \tilde{y}(x)) + (\tilde{y}(x) - y_{0,h}(x)) = \epsilon_1(x) + \epsilon_2(x),$$

say, where ϵ_1 is the error introduced in stage 1 by the perturbation of the given equation, and ϵ_2 is the error of stage 2 that is due to the fact that we can only solve the perturbed equation approximately and not exactly. Based on this decomposition it may seem tempting to reduce the overall error simply by reducing the contribution from ϵ_1 and leaving stage 2 unchanged. In view of what we said above, this can easily be achieved by choosing the $\tilde{\alpha}_j$ values closer to the original α_j. However, we repeat that this is likely to increase the dimension of the system, and the very structure of the system (to be precise: the large number of zeros in the initial condition (22b)) has the unwanted effect of making this more precise system much more difficult to handle for a numerical integration scheme. Thus it will often be observed that the cavalier approach of improving in stage 1 and not changing stage 2 will lead to a small improvement in ϵ_1 at the price of a major deterioration of ϵ_2 and hence a worse overall result.

5. Further Work

One area that needs further investigation is this last dilemma. We have identified two possible ways forward, but the question remains open as to which of these (or some other approach) would be more appropriate.

The first option we have identified is to decrease the step size h used in stage 2 as the dimension $\tilde{N}$ of the system increases (a useful rule of thumb would be to use $h = c/\tilde{N}$ with some constant c).

Our second approach would be to remove those zeros (that do not contribute to the overall solution, so that the numerical solution gets stuck at the initial value for the first approximately

$1/(2\gamma)$ steps) by replacing the plain PECE structure by a P(EC)^{M}E version with a sufficiently large M (e.g. $M = 1/\gamma$).

One can combine these two ideas or use different values of M in different regions. For the moment we can say that both these approaches have the disadvantage of a significant increase in the computational effort, so one may often be better off with a quite crude approximation in the first stage followed by a simple version of the approximation algorithm in the second stage. This is particularly true if the given orders $\alpha_1, \ldots, \alpha_n$ of the differential operators in (20a) are something like material constants that are known only up to a limited accuracy.

6. Numerical Examples

Now we present some numerical examples to illustrate the error bounds stated above. We only looked at examples with $0 < \alpha < 2$ since the case $\alpha \geq 2$ does not seem to be of major practical interest. All computations were performed in double precision arithmetic on a Pentium-based PC.

Our first example deals with the equation

$$\begin{aligned} D_*^\alpha y(x) \;=\; & \frac{40320}{\Gamma(9-\alpha)} x^{8-\alpha} - 3\frac{\Gamma(5+\alpha/2)}{\Gamma(5-\alpha/2)} x^{4-\alpha/2} + \frac{9}{4}\Gamma(\alpha+1) \\ & + \left(\frac{3}{2}x^{\alpha/2} - x^4\right)^3 - [y(x)]^{3/2}. \end{aligned}$$

The initial conditions were chosen to be homogeneous ($y(0) = 0$, $y'(0) = 0$; the latter only in the case $\alpha > 1$). The exact solution of this initial value problem is

$$y(x) = x^8 - 3x^{4+\alpha/2} + \frac{9}{4}x^\alpha.$$

We display some of the results in Tables 1 and 2. In particular, we have not only used our basic algorithm here, but we have also tried to increase the accuracy of the numerical solution by using the Richardson extrapolation idea of Section 3.3. In each case, the leftmost column shows the step size used; the following column gives the error of our scheme at $x = 1$, and the columns after that give the extrapolated values. The bottom line (marked 'EOC') states the experimentally determined order of convergence for each of the columns on the right of the table. According to our theoretical considerations, these values should be $1 + \alpha$, 2, $2 + \alpha$, $3 + \alpha$, 4, $4 + \alpha$, ... in the case $0 < \alpha < 1$ and 2, $1 + \alpha$, $2 + \alpha$, 4, $3 + \alpha$, $4 + \alpha$, ... for $1 < \alpha < 2$. The numerical data in the following tables show that these values are reproduced approximately at least for $\alpha > 1$ (Table 1). In the case $0 < \alpha < 1$, displayed in Table 2, the situation seems to be less obvious. Apparently, we need to use much smaller values for h than in the case $\alpha > 1$ before we can see that the asymptotic behaviour really sets in. This would normally correspond to the situation that the coefficients of the leading terms are small in magnitude compared to the coefficients of the higher-order terms.

As usual, the notation $-5.53(-3)$ stands for $-5.53 \cdot 10^{-3}$, etc.

Our second example covers the case where the given function f (the right-hand side of the differential equation) is smooth. We look at the homogeneous linear equation

$$D_*^\alpha y(x) = -y(x), \quad y(0) = 1,\ y'(0) = 0$$

Table 1. Errors for $\alpha = 1.25$.

Step size	Error of Adams scheme	Extrapolated values			
1/10	−5.53(−3)				
1/20	−1.59(−3)	−2.80(−4)			
1/40	−4.33(−4)	−4.60(−5)	1.63(−5)		
1/80	−1.14(−4)	−8.17(−6)	1.90(−6)	2.13(−7)	
1/160	−2.97(−5)	−1.54(−6)	2.24(−7)	2.71(−8)	1.47(−8)
1/320	−7.66(−6)	−3.04(−7)	2.56(−8)	2.28(−9)	6.24(−10)
1/640	−1.96(−6)	−6.16(−8)	2.85(−9)	1.73(−10)	3.25(−11)
EOC	1.97	2.30	3.17	3.72	4.26

Table 2. Errors for $\alpha = 0.25$.

Step size	Error of Adams scheme	Extrapolated values			
1/10	2.50(−1)				
1/20	1.81(−2)	−1.50(−1)			
1/40	3.61(−3)	−6.91(−3)	4.09(−2)		
1/80	1.45(−3)	−1.10(−4)	2.16(−3)	−8.15(−3)	
1/160	6.58(−4)	8.19(−5)	1.46(−4)	−3.89(−4)	1.28(−4)
1/320	2.97(−4)	3.49(−5)	1.92(−5)	−1.45(−5)	1.05(−5)
1/640	1.31(−4)	1.12(−5)	3.37(−6)	−8.50(−7)	6.01(−8)
EOC	1.18	1.63	2.51	4.09	7.44

(the second of the initial conditions only for $\alpha > 1$ of course). It is well known that the exact solution is

$$y(x) = E_\alpha(-x^\alpha),$$

where

$$E_\alpha(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\alpha k + 1)}$$

is the Mittag–Leffler function of order α.

In Table 3 we state some numerical results for this problem in the case $\alpha < 1$. The data given in the tables is the error of the Adams scheme at the point $x = 1$. We can see from the last line that the order of convergence is always close to $1 + \alpha$. In contrast, Table 4 displays the case $\alpha > 1$; here the results confirm the $O(h^2)$ behaviour. This reflects the statement of Equation (16).

Table 3. Errors for $\alpha < 1$.

h	$\alpha = 0.1$	$\alpha = 0.3$	$\alpha = 0.5$	$\alpha = 0.7$	$\alpha = 0.9$
1/10	−5.42(−3)	−1.86(−3)	−1.30(−3)	−9.91(−4)	−7.51(−4)
1/20	−1.22(−3)	−5.85(−4)	−3.93(−4)	−2.81(−4)	−1.91(−4)
1/40	−4.40(−4)	−1.97(−4)	−1.26(−4)	−8.28(−5)	−4.99(−5)
1/80	−1.68(−4)	−6.90(−5)	−4.18(−5)	−2.50(−5)	−1.32(−5)
1/160	−6.65(−5)	−2.49(−5)	−1.42(−5)	−7.63(−6)	−3.54(−6)
1/320	−2.68(−5)	−9.18(−6)	−4.86(−6)	−2.35(−6)	−9.48(−7)
EOC	1.31	1.44	1.54	1.70	1.90

Table 4. Errors for $\alpha > 1$.

h	$\alpha = 1.25$	$\alpha = 1.5$	$\alpha = 1.85$
1/10	−5.61(−4)	−5.46(−4)	−4.40(−4)
1/20	−1.27(−4)	−1.28(−4)	−1.07(−4)
1/40	−2.90(−5)	−3.04(−5)	−2.65(−5)
1/80	−6.68(−6)	−7.33(−6)	−6.57(−6)
1/160	−1.55(−6)	−1.78(−6)	−1.63(−6)
1/320	−3.63(−7)	−4.37(−7)	−4.07(−7)
EOC	2.09	2.03	2.00

Appendix: Pseudo-Code of the Algorithm

Following the derivation and description of the Adams–Bashforth-Moulton algorithm in mathematical terms given in Section 2 we now state this algorithm in a pseudo-code type notation. In this way the reader interested in implementing the method can easily do so in the language of his or her preference.

INPUT VARIABLES

f = the real-valued function of two real variables that defines the right-hand side of the differential equation (1)
α = the order of the differential equation (a positive real number)
y_0 = an array of $\lceil \alpha \rceil$ real numbers that contains the initial values $y(0)$, $y'(0)$, ..., $y^{(\lceil \alpha \rceil - 1)}(0)$
T = the upper bound of the interval where the solution is to be approximated (a positive real number)
N = the number of time steps that the algorithm is to take (a positive integer)

OUTPUT VARIABLES

y = an array of $N+1$ real numbers that contains the approximate solutions $y_{T/N}(jT/N)$, $j = 0, 1, \ldots, N$. We have $y_{T/N}(jT/N) \approx y(T/N)$ where y is the exact solution of the given fractional differential equation

INTERNAL VARIABLES

h = the step size of the algorithm (a positive real number)
m = the number of initial conditions specified (a positive integer)
j, k = integer variables used as indices
a, b = arrays of $N+1$ real numbers that contain the weights of the corrector and predictor formulae, respectively
p = the predicted value (a real variable)

BODY OF THE PROCEDURE

$h := T/N$
$m := \lceil \alpha \rceil$
FOR $k := 1$ TO N DO BEGIN
  $b[k] := k^\alpha - (k-1)^\alpha$
  $a[k] := (k+1)^{\alpha+1} - 2k^{\alpha+1} + (k-1)^{\alpha+1}$
END
$y[0] := y_0[0]$
FOR $j := 1$ TO N DO BEGIN

$$p := \sum_{k=0}^{m-1} \frac{(jh)^k}{k!} y_0[k] + \frac{h^\alpha}{\Gamma(\alpha+1)} \sum_{k=0}^{j-1} b[j-k] f(kh, y[k])$$

$$y[j] := \sum_{k=0}^{m-1} \frac{(jh)^k}{k!} y_0[k] + \frac{h^\alpha}{\Gamma(\alpha+2)} \left(f(jh, p) + \left((j-1)^{\alpha+1} - (j-1-\alpha) j^\alpha\right) f(0, y[0]) + \sum_{k=1}^{j-1} a[j-k] f(kh, y[k]) \right)$$

END

COMMENT

The notation in Section 2 indicates that the arrays a and b ought to be two-dimensional. However, those weights (whose values $a_{j,k}$ and $b_{j,k}$ are defined in Equations (12) and (14), respectively) essentially have a convolution structure, i.e., they only depend on the difference $k-j$ of the two indices. This structure allows us to set up the arrays a and b as one-dimensional and not two-dimensional, leading to a significant reduction in the memory requirements.

References

1. Butzer, P. L. and Westphal, U., 'An introduction to fractional calculus,' in *Applications of Fractional Calculus in Physics*, R. Hilfer (ed.), World Scientific, Singapore, 2000, pp. 1–85.
2. Podlubny, I., *Fractional Differential Equations*, Academic Press, San Diego, CA, 1999.
3. Samko, S. G., Kilbas, A. A., and Marichev, O. I., *Fractional Integrals and Derivatives: Theory and Applications*, Gordon and Breach, Yverdon, 1993.
4. Caputo, M., 'Linear models of dissipation whose Q is almost frequency independent, II', *The Geophysical Journal of the Royal Astronomical Society* **13**, 1967, 529–539.
5. Diethelm, K. and Ford, N. J., 'Analysis of fractional differential equations', *Journal of Mathematical Analysis and Applications* **265**, 2002, 229–248.
6. Lorenzo, C. F. and Hartley, T. T., 'Initialized fractional calculus', *International Journal of Applied Mathematics* **3**, 2000, 249–265.
7. Gorenflo, R., 'Fractional calculus: Some numerical methods', in *Fractals and Fractional Calculus in Continuum Mechanics*, A. Carpinteri and F. Mainardi (eds.), Springer-Verlag, Wien, 1997, pp. 277–290.
8. Gorenflo, R. and Mainardi, F., 'Fractional calculus: Integral and differential equations of fractional order', in *Fractals and Fractional Calculus in Continuum Mechanics*, A. Carpinteri and F. Mainardi (eds.), Springer-Verlag, Wien, 1997, pp. 223–276.
9. Diethelm, K. and Ford, N. J., 'Numerical solution of the Bagley–Torvik equation', *BIT*, to appear.
10. Diethelm, K. and Freed, A. D., 'On the solution of nonlinear fractional differential equations used in the modeling of viscoplasticity', in *Scientific Computing in Chemical Engineering II — Computational Fluid Dynamics, Reaction Engineering, and Molecular Properties*, F. Keil, W. Mackens, H. Voß, and J. Werther (eds.), Springer-Verlag, Heidelberg, 1999, pp. 217–224.
11. Benson, D. A., 'The fractional advection-dispersion equation: Development and application', Ph.D. Thesis, University of Nevada Reno, 1998.
12. Blank, L., 'Numerical treatment of differential equations of fractional order', Numerical Analysis Report 287, Manchester Centre for Computational Mathematics, Manchester, 1996.
13. Chern, J.-T., 'Finite element modeling of viscoelastic materials on the theory of fractional calculus', Ph.D. Thesis, Pennsylvania State University, 1993.
14. Diethelm, K., 'An algorithm for the numerical solution of differential equations of fractional order', *Electronic Transactions on Numerical Analysis* **5**, 1997, 1–6.
15. Diethelm, K. and Luchko, Y., 'Numerical solution of linear multi-term initial value problems of fractional order', *Journal of Computational Analysis and Applications*, to appear.
16. Diethelm, K. and Walz, G., 'Numerical solution of fractional order differential equations by extrapolation', *Numerical Algorithms* **16**, 1997, 231–253.
17. Enelund, M., Fenander, Å., and Olsson, P., 'Fractional integral formulation of constitutive equations of viscoelasticity', *AIAA Journal* **35**, 1997, 1356–1362.
18. Enelund, M. and Josefson, B. L., 'Time-domain finite element analysis of viscoelastic structures with fractional derivative constitutive relations', *AIAA Journal* **35**, 1997, 1630–1637.
19. Enelund, M. and Lesieutre, G. A., 'Time domain modeling of damping using anelastic displacement fields and fractional calculus', *International Journal of Solids and Structures* **36**, 1999, 4447–4472.
20. Enelund, M. and Olsson, P., 'Damping described by fading memory – Analysis and application to fractional derivative models', *International Journal of Solids and Structures* **36**, 1998, 939–970.
21. Lubich, C., 'Runge–Kutta theory for Volterra and Abel integral equations of the second kind', *Mathematics of Computation* **41**, 1983, 87–102.
22. Lubich, C., 'Fractional linear multistep methods for Abel-Volterra integral equations of the second kind', *Mathematics of Computation* **45**, 1985, 463–469.
23. Lubich, C., 'Discretized fractional calculus', *SIAM Journal on Mathematical Analysis* **17**, 1986, 704–719.
24. Ruge, P. and Wagner, N., 'Time-domain solutions for vibration systems with fading memory', in *Proceedings of the European Conference on Computational Mechanics 1999*, W. Wunderlich (ed.), CD-ROM, Lehrstuhl für Statik, Technische Universität München, 1999 (http://www.isd.uni-stuttgart.de/∼nwagner/ eccm99.ps).
25. Shokooh, A. and Suarez, L. E., 'A comparison of numerical methods applied to a fractional derivative model of damping materials', *Journal of Vibration and Control* **5**, 1999, 331–354.
26. Yuan, L. and Agrawal, O. P., 'A numerical scheme for dynamic systems containing fractional derivatives', in *Proceedings of the 1998 ASME Design Engineering Technical Confer-*

ences, Atlanta, GA, September 13–16, ASME International, New York, 1998, CD-ROM publication (http://heera.engr.siu.edu/mech/faculty/agrawal/mech5857.pdf).
27. Hairer, E., Nørsett, S. P., and Wanner, G., *Solving Ordinary Differential Equations I: Nonstiff Problems*, 2nd revised edition, Springer-Verlag, Berlin, 1993.
28. Hairer, E. and Wanner, G., *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, Springer-Verlag, Berlin, 1991.
29. Diethelm, K. and Freed, A. D., 'The FracPECE subroutine for the numerical solution of differential equations of fractional order', in *Forschung und wissenschaftliches Rechnen 1998*, S. Heinzel and T. Plesser (eds.), Gesellschaft für wissenschaftliche Datenverarbeitung, Göttingen, 1999, pp. 57–71.
30. Diethelm, K., Ford, N. J., and Freed, A. D., 'Detailed error analysis for a fractional Adams method', Berichte der Mathematischen Institut der TU Braunschweig O2/02, Braunschweig, submitted for publication (http://www.tu-bs.de/∼diethelm/publications/adams.ps).
31. Ford, N. J. and Simpson, A. C., 'The numerical solution of fractional differential equations: Speed versus accuracy', *Numerical Algorithms* **26**, 2001, 333–346.
32. de Hoog, F. and Weiss, R., 'Asymptotic expansions for product integration', *Mathematics of Computation* **27**, 1973, 295–306.
33. Walz, G., *Asymptotics and Extrapolation*, Akademie-Verlag, Berlin, 1996.
34. Torvik, P. J. and Bagley, R. L., 'On the appearance of the fractional derivative in the behavior of real materials', *Journal of Applied Mechanics* **51**, 1984, 294–298.
35. Koeller, R. C., 'Polynomial operators, Stieltjes convolution, and fractional calculus in hereditary mechanics', *Acta Mechanica* **58**, 1986, 251–264.
36. Diethelm, K. and Ford, N. J., 'Numerical solution of linear and non-linear fractional differential equations involving fractional derivatives of several orders', Numerical Analysis Report 379, Manchester Centre for Computational Mathematics, Manchester, 2001, submitted for publication (http://www.maths.man.ac.uk/∼nareports/narep379.ps.gz).
37. Bagley, R. L. and Calico, R. A., 'Fractional order state equations for the control of viscoelastically damped structures', *Journal of Guidance, Control, and Dynamics* **14**, 1991, 304–311.