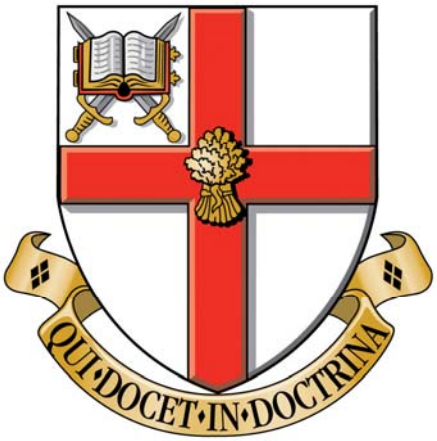

University of Chester

Author(s): Neville J Ford ; Joseph Arthur Connolly

Title: Systems-based decomposition schemes for the approximate solution of multi-term fractional differential equations

Date: 2007

University of Chester

Department of Mathematics

Applied Mathematics Group
Research Report

2007 Series

Systems-based decomposition schemes for the approximate solution of multi-term fractional differential equations

Neville J. Ford Joseph A. Connelly

Report 2007:4

Systems-based decomposition schemes for the approximate solution of multi-term fractional differential equations

Neville J. Ford
and
Joseph A. Connolly

3rd January 2008

Abstract

We give a comparison of the efficiency of three alternative decomposition schemes for the approximate solution of multi-term fractional differential equations using the Caputo form of the fractional derivative. The schemes we compare are based on conversion of the original problem into a system of equations. We review alternative approaches and consider how the most appropriate numerical scheme may be chosen to solve a particular equation.
AMS (2000): 65L20, 35Q72, 34K28, 45E10.
Keywords: Fractional differential equations, numerical methods, multi-term equations.

1 Introduction

This paper concerns the numerical solution of multi-term fractional differential equations which have the general form

$$D^{\alpha_N} y(t) = f(t, y(t), D^{\alpha_1} y(t), \ldots, D^{\alpha_{N-1}} y(t)). \tag{1}$$

$D^\alpha y(t)$ is used to represent the Caputo-type fractional derivative of order $\alpha > 0$ which is defined, for $m \in \mathbb{N}$ and non-integer $\alpha > 0$, by

$$D^\alpha u(t) = \frac{1}{\Gamma(m-\alpha)} \int_0^t (t-\tau)^{m-\alpha-1} u^{(m)}(\tau) d\tau, \quad t > 0, \quad m-1 < \alpha < m \tag{2}$$

where

$$D^m u(t) = u^{(m)}(t). \tag{3}$$

We assume that $\alpha_N > \alpha_{N-1} > \ldots > \alpha_0 > 0$, $\alpha_i - \alpha_{i-1} \leq 1$, $N \in \mathbb{N}$, and $\alpha_i \in \mathbb{Q}$ for all i. An initial value problem consists of (1) equipped with initial conditions

$$y^{(k)}(0) = y_0^{(k)}, k = 0, 1, \ldots, \lceil \alpha_N \rceil - 1. \tag{4}$$

The notation $\lceil \alpha \rceil$ is used to denote the integer closest to and not less than α i.e. the integer lying in the interval $[\alpha, \alpha + 1)$.

Our aim is to establish an effective way to approximate solution(s) of the initial value problem for this equation using a numerical method. To achieve this, we shall reformulate the multi-term equation in an appropriate way and the focus of this paper is to consider three alternative strategies for constructing a system of fractional differential equations

that can be regarded as being equivalent (in a sense that we shall make precise later) to (1). As we shall see, there are alternative approaches that one might consider, and we review these and provide a justification for considering only the systems-based approach here.

Multi-term fractional differential equations have been used to model various types of visco-elastic damping. Model equations proposed so far are almost always linear, so our experiments in this paper will focus on equations of the linear form

$$\left[D^{\alpha_N} + b_{N-1}D^{\alpha_{N-1}} + \ldots + b_1 D^{\alpha_1} + b_0 D^0\right] y(t) = g(t), \tag{5}$$

where $b_i \in \mathbb{R}$, $i = 0, 1, \ldots, N-1$, equipped with initial conditions (4). Our aim is to provide a numerical scheme that is robust, reliable, and reasonably inexpensive in terms of both set-up costs and the time taken to execute. Other things being equal, we would also prefer the use of methods that are easily adapted to future non-linear problems.

Two of the algorithms we consider have appeared previously, the third is introduced in this paper:

1. The earliest algorithm of the type we consider here for the numerical solution of (5) was introduced by Diethelm and Ford [8, 9]. The method is based on the re-expression of the multi-term equation as a system of equations of low fractional order, rather in the same way that one solves a high order ordinary differential equation as a system of first order equations.

2. In [14] Edwards, Ford and Simpson introduced an alternative approach, in which the dimension of the system of equations is reduced, but at the expense of a more complicated formulation. Overall they observed a reduction in computational work compared with the method of Diethelm and Ford.

3. In this paper we introduce a new algorithm, which again produces a system of equations of low dimension, and we compare the methods' computational cost and effectiveness.

2 Rival approaches

We are aware of two approaches to the solution of multi-term equations that do not give rise to a system of fractional equations to solve. These methods involve an analytical stage, in which the original problem is converted into an equivalent form, and a numerical stage where the solution is approximated. Thus any of these approaches may, at least in principle, be combined with a variety of numerical methods to give different algorithms based on the same reformulation.

One approach was proposed by Diethelm and Luchko in the paper [13]. Here the authors build on an idea proposed by Luchko and Gorenflo in [21]. The approach is to reformulate the problem (assumed linear) through the use of Laplace transforms to provide a representation of the solution in terms of a sum of Mittag-Lefler functions. Essentially, the approach leads to an expression for the solution in terms of a Green's function. One needs some method for approximating the solution and the approach proposed in [13] is to apply the discretised operational calculus analysed in [17, 18] to evaluate the solution.

In fact, one could use alternative approximations such as one based, for example, on the BDF method of Diethelm [3].

The second approach was proposed in the PhD thesis of Nkamnang [23]. This time, the problem is reformulated as a Volterra integral (or integro-differential) equation which is then solved using a suitable quadrature method that is designed to take care of the singularities. Again, the discretised operational calculus of Lubich [17, 18] is proposed for the numerical scheme. Other alternative numerical quadratures are possible.

Both these schemes have been shown to perform well for simple test problems. They inherit the order properties of the fractional multistep methods on which the discretised operational calculus approach is based. They also appear to be very versatile, since they do not impose a requirement (as the systems based approach does) that the orders of derivatives are all rational numbers.

However, despite this theoretical advantage, it turns out to be desirable nevertheless to restrict ourselves to problems where all the orders of derivative are rational numbers. In the case of the schemes of Diethelm and Luchko, and of Nkamnang, it is not necessary to impose this restriction during the reformulation stage. However, as has been pointed out elsewhere, the numerical schemes cannot be implemented exactly for irrational indices, because the computer cannot store or calculate with irrational numbers. Therefore, while there may be some attraction in the slightly more general form of reformulation available, there is no practical benefit.

Further, as has been remarked by Diethelm and Ford in [12], these alternative formulations are actually mathematically equivalent to the use of Method 1 (described below), combined with the use of a fractional multistep method. This observation is not so surprising, since in the case of ordinary differential equations, the conversion to an integral equation form and solution by a reducible quadrature method is mathematically equivalent to the application of a multistep method directly to the differential equation.

Finally, we observe that there is also a potential difficulty in the use of the discretised operational calculus: the calculation of the starting weights. This problem has been considered in detail in the recent paper [5] and the conclusions of that paper were that the weights cannot be calculated reliably and the resulting method may become unstable and inaccurate if the fractional orders do not produce a reasonably small integer when divided into unity. We also note that the numerical examples given in the paper [13] and the thesis [23] all have convenient derivative orders which are (small) integer divisors of unity. For these problems, the starting weights can be calculated reasonably reliably and accurately. Furthermore, fractional multistep methods did not turn out to be the most efficient solvers for single term problems in our experiments in [15].

As we already remarked in the introduction, we shall assume throughout this paper that all derivative orders appearing are rational numbers.

3 Choice of numerical algorithm

Systems of fractional differential equations may be solved approximately by applying a numerical method for a scalar problem of the appropriate order to each component of the system. Therefore our starting point in the selection of numerical methods is the paper [11] in which a full range of approaches is presented. We do not reproduce the details here for

the sake of space, since [11] provides a full description of each algorithm. For single term equations, the efficiency of these algorithms was compared in our paper [15] and the results of that paper help us to focus on appropriate numerical methods in this work. However, as we have noted previously, the main focus here is on the different decompositions from multi-term problem into a system that are available and, in principle, one could implement each of these decompositions with a full range of numerical algorithms.

The results of our previous paper can be summarised as follows:

1. an iterative scheme, based on an Adams-type predictor corrector (PECE) pair, was frequently the most efficient method to adopt

2. the BDF method described by Diethelm in [3] was the only other method that seemed to provide a competitive alternative to the PECE scheme

3. even when the predictor-corrector scheme was not the most efficient, it was always a close competitor with the best scheme tested. Therefore we proposed this scheme as a good universal choice in general

4 The representations as systems

4.1 Method 1

Diethelm and Ford [9] introduced the following scheme for the solution of multi-term FDEs of the form (5).

Let $\alpha_N = vq$, where $q \in \mathbb{Q}$ is the largest rational number for which each order α_n appearing in (5) is an integer multiple of q. A suitable value for q exists since we assumed all the orders were rational.

We can now write the multi-term equation (5) in the form

$$\left[D^{vq} + a_{v-1}D^{(v-1)q} + \ldots + a_1 D^q + a_0 D^0\right] y(t) = g(t), \tag{6}$$

with initial conditions

$$y^{(k)}(0) = y_0^{(k)}, k = 0, 1, \ldots, \lceil vq \rceil - 1. \tag{7}$$

Notice that this equation has, in general, additional terms included (each with co-efficient zero) compared with (5).

Remark 4.1 *We have assumed all the orders are rational. In the case where any order α_k is irrational this approach cannot be applied exactly. However, it may be appropriate (see [7]) to approximate the irrational order by a nearby rational value so that an approximate solution may be obtained.*

Diethelm and Ford [9] proved that equation (6) can be written as a system of equations, with appropriate initial conditions:

Theorem 4.1 *The equation (6) with initial conditions (7) (or, equivalently, (5) equipped with (4)) is equivalent to the system of equations*

$$\begin{aligned}
&D^q\ {}^0Y(t) = {}^1Y(t),\\
&D^q\ {}^1Y(t) = {}^2Y(t),\\
&D^q\ {}^2Y(t) = {}^3Y(t),\\
&\vdots\\
&D^q\ {}^{v-2}Y(t) = {}^{v-1}Y(t),\\
&D^q\ {}^{v-1}Y(t) = -b_0\ {}^0Y(t) - b_1\ {}^{\alpha_1/q}Y(t) - \ldots - b_{N-1}\ {}^{\alpha_{N-1}/q}Y(t) + g(t)
\end{aligned} \tag{8}$$

together with the initial conditions,

$${}^iY(0) = \begin{cases} y_0^{(k)} & \text{if } iv = k \in \mathbb{N},\\ 0 & \text{else,} \end{cases} \tag{9}$$

in the following sense.

1. *Whenever $Y := ({}^0Y, \ldots, {}^{v-1}Y)^T$ with ${}^0Y \in C^{\lceil \alpha_N \rceil}[0,c]$ for some $c > 0$ is the solution of the system (8), equipped with the corresponding initial conditions, the function $y := {}^0Y$ solves the multi-term equation (5) and satisfies the initial conditions (4).*

2. *Whenever $y \in C^{\lceil \alpha_N \rceil}[0,c]$ is a solution of the multi-term equation (5) satisfying the initial conditions (4), the vector-valued function*

   $$Y := ({}^0Y, \ldots, {}^vY)^T := (y, D^q y, D^{2q} y, \ldots, D^{(v-1)q} y)^T$$

   satisfies the system (8) and the initial conditions (9).

For this result (and the subsequent ones in this section) the following Lemma is important:

Lemma 4.1 *Let $f \in C^k[a,b]$ for some $a < b$ and some $k \in \mathbb{N}$. Moreover, let $0 < \nu < 1$ and assume that $j \in \mathbb{Q}$ is such that there is no integer number strictly between $j\nu$ and $(j+1)\nu$. Then if $(j+1)\nu < k$ we have*

$$D^\nu D^{j\nu} f = D^{(j+1)\nu} f. \tag{10}$$

It follows that we can rewrite equation (5) as a system of v single-term equations,

$$D^q Y(t) = G(t, Y(t)) \tag{11}$$

with initial conditions

$$Y(0) = Y_0$$

where $Y_0 = ({}^0Y(0), {}^1Y(0), \ldots, {}^{v-1}Y(0))^T$.

In principle, one can now apply any single-term equation solver from [11] to solve the system (11). In practice, we shall confine ourselves to the two methods we already showed to be efficient in [15]. These are the backward differentiation scheme described in [3] and the predictor-corrector scheme from [10]

4.2 Method 2

The method introduced by Edwards, Ford and Simpson aimed to produce a variant that leads to a system of equations of lower dimension. This is achieved by allowing the orders of different equations within the system to vary.

We write each order α_i, as the sum of $[\alpha_i]$ (its whole number part) and $\beta_i = \alpha_i - [\alpha_i]$ (its fractional part.)

For example, we consider a 5-term test equation with $\alpha_N = 2$:

$$[D^2 + b_3 D^{\beta_3+1} + b_2 D + b_1 D^{\alpha_1} + b_0]y(t) = g(t), \text{ where } \alpha_1, \beta_1 \in (0,1), b_i \in \mathbb{R} \tag{12}$$

subject to initial conditions,

$$y^{(k)}(0) = y_0^{(k)}, k = 0, 1$$

would be written as,

$$\begin{array}{l} {}^0Y(t) = y(t) \\ {}^1Y(t) = D^{\alpha_1}\ {}^0Y(t) \\ {}^2Y(t) = D\ {}^0Y(t) \\ {}^3Y(t) = D^{\beta_3}\ {}^2Y(t) \\ {}^4Y(t) = D\ {}^2Y(t) \end{array} \tag{13}$$

together with initial conditions

$${}^kY(0) = \begin{cases} y_0^{(k)} & \text{for } k = 0 \text{ and } k = 2, \\ 0 & \text{otherwise.} \end{cases} \tag{14}$$

In matrix form, the system can be represented

$$\begin{pmatrix} D^{\alpha_1} & 0 & 0 & 0 \\ D & 0 & 0 & 0 \\ 0 & 0 & D^{\beta_3} & 0 \\ 0 & 0 & D & 0 \end{pmatrix} \begin{pmatrix} {}^0Y(t) \\ {}^1Y(t) \\ {}^2Y(t) \\ {}^3Y(t) \end{pmatrix} = \begin{pmatrix} {}^1Y(t) \\ {}^2Y(t) \\ {}^3Y(t) \\ g(t) - \sum_{i=0}^{3} b_i\ {}^iY(t). \end{pmatrix} \tag{15}$$

Of course, now we have a system which involves integer order derivatives as well as derivatives of possibly several different fractional orders. In addition, the matrix form highlights one of the potential limitations of the approach: one must take particular care to ensure that the discretised version does not have a singular matrix approximating the derivatives. We chose to solve the differential equation using a trapezium rule for the integer-order derivatives and the backward differentiation method of Diethelm (in [3]) for solving the fractional order components. This avoids the possibility of a singularity in the algebraic scheme to be solved and also has the particular advantage that the weights needed are given explicitly in [3] and so there is no additional work involved in setting up the solver for the multiple fractional order derivatives we need.

Remark 4.2 *If one attempted to employ the predictor-corrector formula to (15) then the algebraic problem would become singular.*

This approach leads to the need to solve a linear system of the form,

$$\begin{pmatrix} -^{\alpha_1}w_{0,j} & {}^{\alpha_1}\eta_j & 0 & 0 \\ 1 & 0 & -\frac{h}{2} & 0 \\ 0 & 0 & -^{\beta_1}w_{0,j} & {}^{\beta_1}\eta_j \\ \frac{hb_0}{2} & \frac{hb_1}{2} & 1+\frac{hb_2}{2} & \frac{hb_3}{2} \end{pmatrix} \begin{pmatrix} {}^1Y_j \\ {}^2Y_j \\ {}^3Y_j \\ {}^4Y_j \end{pmatrix} = \begin{pmatrix} {}^1S_j \\ {}^2S_j \\ {}^3S_j \\ {}^4S_j \end{pmatrix}$$

where

$$\begin{aligned} {}^1S_j &= \sum_{k=1}^{j} {}^{\alpha_1}w_{k,j}\,{}^1Y_{j-k} + {}^1Y_0/\alpha_1, \\ {}^2S_j &= {}^1Y_{j-1} + \frac{h}{2}{}^3Y_{j-1}, \\ {}^3S_j &= \sum_{k=1}^{j} {}^{\beta_1}w_{k,j}\,{}^3Y_{j-k} + {}^3\,Y_0/\beta_1, \\ {}^4S_j &= {}^3Y_{j-1} + G_j - \frac{h}{2}\sum_{i=1}^{4} b_{i-1}\,{}^iY_{j-1}, \end{aligned}$$

$G_j = h(g_j + g_{j-1})/2, g_j = g(jh)$,
and ${}^{\alpha_1}\eta_j = (jh)^{\alpha_1}\Gamma(-\alpha_1)$.
and the values ${}^iY_0$ are given by the initial conditions.

4.3 Method 3

Methods 1 and 2 produce different systems of equations. In the case of method 1, the system may be of quite high dimension. Method 2 keeps the dimension of the system reasonably small and independent of the precise orders in the equation. However, this comes at the cost of producing a left-hand side matrix that can be inconvenient in calculations and can restrict the choice of numerical method. However, because the Edwards, Ford and Simpson method uses an explicit BDF-based quadrature, the potential problem was avoided. Our decomposition in Method 3 is designed to avoid this difficulty and permit a wider range of numerical algorithms to be employed.

Method 3 decomposes the multi-term equation into a system of fractional equations of varying orders. This time we do not calculate the integer order derivatives directly, as we did in Method 2, but instead we calculate each derivative in the equation by reference to the next lower order derivative.

We form an ascending list $\{\gamma_j\}$ of all derivative orders that appear in the equation (including any missing natural number orders in the list) and for each $i > 1$, we find the *offset* β_i of γ_i above γ_{i-1}. Thus $\beta_1 = \gamma_1$ and $\beta_i = \gamma_i - \gamma_{i-1}$ for $i > 1$.

We now rewrite the (possibly non-linear) multi-term equation using the sequence of orders $\{\beta_i\}$. Once again, the approach is most easily understood through an example:

4.4 Example: a five-term Equation

For the equation,

$$D^2 y = f(t, y(t), D^{\alpha_1} y(t), Dy(t), D^{\alpha_3} y(t)), \text{ where } \alpha_1, \in (0,1), \alpha_3 \in (1,2) \tag{16}$$

subject to initial conditions,

$$y^{(k)}(0) = y_0^{(k)}, k = 0, 1$$

it is easy to see that $\{\beta_i\} = \{\alpha_1, 1-\alpha_1, \alpha_3 - 1, 2 - \alpha_3\}$.

We can establish the following equivalence theorem by repeated application of Lemma 4.1.

Theorem 4.2 *The equation (16), with the relevant initial conditions, is equivalent to the system of equations*

$$\begin{array}{l} {}^0Y(t) = y(t) \\ {}^1Y(t) = D^{\alpha_1} y(t) = D^{\beta_1}\ {}^0Y(t) \\ {}^2Y(t) = Dy(t) = D^{\beta_2}\ {}^1Y(t) \\ {}^3Y(t) = D^{\alpha_3} y(t) = D^{\beta_3}\ {}^2Y(t) \\ {}^4Y(t) = D^2 y(t) = D^{\beta_4}\ {}^3Y(t) \end{array} \tag{17}$$

together with initial conditions

$${}^kY(0) = \begin{cases} y_0^{(k)} & \text{for } k = 0 \text{ and } k = 2, \\ 0 & \text{else.} \end{cases} \tag{18}$$

Expressed in matrix form, this leads to

$$\begin{pmatrix} D^{\beta_1} & 0 & 0 & 0 \\ 0 & D^{\beta_2} & 0 & 0 \\ 0 & 0 & D^{\beta_3} & 0 \\ 0 & 0 & 0 & D^{\beta_4} \end{pmatrix} \begin{pmatrix} {}^0Y(t) \\ {}^1Y(t) \\ {}^2Y(t) \\ {}^3Y(t) \end{pmatrix} = \begin{pmatrix} {}^1Y(t) \\ {}^2Y(t) \\ {}^3Y(t) \\ f(t,\ {}^0Y(t),\ {}^1Y(t),\ {}^2Y(t),\ {}^3Y(t)) \end{pmatrix}.$$

With the decomposition proposed in method 3, it is possible to apply the full range of numerical methods discussed in [11]. (Indeed, one could apply this decomposition to simple non-linear problems too) However, one must bear in mind the comparatively high set-up costs for the method because of the multiple fractional orders one might need. Once again, the backward differentiation approach and the predictor-corrector algorithm turn out to be attractive choices that combine efficiency with fairly low start-up costs.

5 Counting the cost

The three approaches to reformulating the underlying multi-term equation considered in Section 4 each lead to one or more numerical algorithm for solving the multi-term equation. Here we discuss the components in the overall computational cost of the algorithms.

5.1 Component 1: the dimension of the system

The first, and probably the most significant, factor in the computational cost of these methods is the dimension of the resulting matrix system of equations. The variation in these dimensions depends both on the dimension of the original multi-term equation, and on the specific orders included in the original problem.

First we consider two specific examples before we give the general conclusion.

Example 5.1 *Let the orders of fractional derivative in the multi-term equation be, respectively* $0.5, 1, 1.5, 2, 2.5$. *Using method 1, the highest common factor of these orders is* 0.5 *and the dimension of the resulting system is* $5 = \frac{2.5}{0.5}$. *Using method 2, the orders used will be* $0.5, 1, 0.5, 2, 0.5$ *so the dimension will again be* 5. *Using method 3, the orders used will be* $0.5, 0.5, 0.5, 0.5, 0.5$ *and the dimension will be* 5. *For this example, the dimension of the system is independent of the method chosen.*

Remark 5.1 *For example 5.1, methods 1 and 3 actually yield exactly the same decomposition of the original equation. Of course, this is not a general pattern as we shall see in our next example.*

Example 5.2 *Let the orders of fractional derivative in the multi-term equation be, respectively* $0.49, 1, 1.5, 2, 2.5$. *Using method 1, the highest common factor of these orders is* 0.01 *and the dimension of the resulting system is* $2500 = \frac{2.5}{0.01}$. *Using method 2, the orders used will be* $0.49, 1, 0.5, 2, 0.5$ *so the dimension will be* 5. *Using method 3, the orders used will be* $0.49, 0.51, 0.5, 0.5, 0.5$ *and the dimension will be* 5. *Thus we can see that the dimension of the system using method 1 is highly unfavourable compared with the other two methods.*

Remark 5.2 *One can observe an unexpected phenomenon here: under method 1, the dimension of the system would increase if the order given here as* 0.49 *was allowed to vary and become closer to* 0.5. *It is known (see for example the discussion in [7]) that the exact solution to a multi-term equation is stable with respect to small perturbations in the order(s). This feature will be reflected in algorithms developed using methods 2 and 3, but will be hidden in algorithms based on method 1.*

In the general case, let the orders present in a certain multi-term equation be $\{\alpha_1, \alpha_2, \ldots, \alpha_N\}$ then the dimension of the system generated by method 1 will be $\frac{\alpha_N}{q}$ where q is the largest rational number for which $\{1, \alpha_1, \alpha_2, \ldots, \alpha_N\}$ are all integer multiples of q. The dimensions of the systems generated by methods 2 and 3 will be bounded by $N + [\alpha_N]$. The precise dimension in this case will depend on how many of the natural number orders $1, 2, \ldots, \alpha_N - 1$ are already present in the list $\{\alpha_1, \alpha_2, \ldots, \alpha_N\}$ since the list must be augmented as necessary to include all the intervening integer orders.

5.2 Component 2: the cost of *setting up* the fractional solvers

The computational cost of setting up the fractional solvers will depend on the number of different fractional derivatives that appear in the system of equations, and on the choice of numerical method. For the methods under consideration here, for each fractional order,

the cost of calculating the parameters of the numerical method will be quite low. In the case of the backward differentiation formula, the weights are tabulated in [3]. For the predictor-corrector scheme, the parameters are given in [9]. The total cost of setting up will be a small multiple of the number of fractional orders.

Remark 5.3 *The cost of setting up the solvers can become significant if a higher order scheme is preferred. We considered in [15] the competitiveness of fractional multistep methods (see [16]) and found them to be usually (surprisingly) uncompetitive in practice. Further discussion of the practical implications of implementing the methods is contained in [5]. The main issue here is the need to calculate special starting weights for the method in order to preserve the order of convergence. The calculation of these weights is a poorly-conditioned problem and inaccuracies in the weights can lead to loss of accuracy in the method. The calculation of starting weights for a system of equations involving several different fractional order derivatives seems likely to be a major obstacle to the use of fractional multistep methods here.*

5.3 Component 3: executing the algorithm

The main factor (apart from the dimension of the system) that will affect the time taken to execute the algorithms we are discussing is the number of corrector steps when we are using PECE predictor-corrector algorithm. Multiple corrector iterations will increase the execution time (and number of calculation steps) proportionately. In the next section we recall how to choose the number of corrector iterations to achieve an appropriate order for the numerical scheme.

6 The orders of the methods and the number of corrector iterations

The conventional error analysis of numerical schemes for the solution of differential equations depends on the use of a series expansion for the exact solution of the problem. Thus, one assumes that the solution satisfies certain smoothness properties and then establishes a bound on the error. Typically, one can prove that the solution will satisfy the smoothness assumptions if the underlying differential equation satisfies natural conditions.

The analysis of errors for fractional differential equations presents additional challenges because the fractional order derivative of a smooth function is typically not itself smooth. Therefore it is natural to present an error analysis for a method either based on assumptions about the smoothness of the underlying equation or of its solution. Typically we shall see a loss of order of the method if the solution (and its fractional derivatives) cannot be shown to be smooth.

The two numerical schemes that we are most concerned with here are the backward differentiation scheme from [3] which is known (under appropriate smoothness assumptions) to have an error of $\mathcal{O}(h^{2-\alpha})$ when used to solve a single-term equation of fractional order $0 < \alpha < 1$. The predictor-corrector (PECE) scheme from [10] is known to have an error of $\mathcal{O}(h^{1+\alpha})$ which rises to $\mathcal{O}(h^{\delta})$ where $\delta = \min(2, 1+q\alpha)$ when the corrector step is

applied $q-$times (see [4]). These orders are attained, for example, under the assumption that the $\alpha-$order derivative of the solution is twice continuously differentiable.

These remarks on order and smoothness will help us to understand the numerical results that follow.

7 Choice of numerical examples

For our numerical experiments we have chosen to investigate the behaviour of the methods for linear equations where the exact solution is known explicitly and is smooth. As we noted in the previous section, results on the order of convergence of the method are dependent on the smoothness not only of the solution but of its fractional derivative. By choosing equations whose exact solution is $u(t) = t^3$, the condition $D^\alpha u \in \mathcal{C}^2$ is satisfied for every $\alpha \in (0, 1)$. For those equations whose exact solution is $u(t) = t^2$, the condition $D^\alpha u \in \mathcal{C}^1$ is satisfied for every $\alpha \in (0, 1)$ but $D^\alpha u \in \mathcal{C}^2$ is not satisfied so the order of convergence may be compromised.

We let the exact analytical solution, for a fixed interval T be $y(T)$ and the approximation at T using n step lengths be $y_n(T)$. EOC, represents the experimental estimated order of convergence evaluated using the formula

$$EOC = \log_2 \left(\frac{\mid y(T) - y_n(T) \mid}{\mid y(T) - y_{2n}(T) \mid} \right).$$

In all the examples that follow we consider approximations over the interval $[0, 1]$, and give the errors at $t = 1$.

The captions in the tables have the following meanings:

- Method 1a refers to the use of the decomposition described as Method 1 with the $PECE$ (Adams-type) numerical method from [10].
- Method 1a(m) is the same approach as in 1a but this time using the $P(EC)^m E$ scheme with the corrector step applied $m-$times
- Method 1b uses the decomposition from Method 1 but with the simpler numerical scheme from [3]
- Method 2 is implemented using the backward differentiation scheme as described in [14]
- Method 3 uses the decomposition described in this paper with the $PECE$ (Adams-type) numerical method from [10]
- Method 3(m) is the same approach as in 3 but this time using the $P(EC)^m E$ scheme with the corrector step applied $m-$times.

8 Numerical results

In the tables that follow we give the errors in the approximate solutions of each of three equations. We also tabulate the number of floating point operations and the average time to execute the algorithm (based on an average time over 10 'runs'). The aim is to give a fair basis for comparison of the actual errors against the actual *cost* of the computation. Of course, with modern computer software, there is a complex interplay between the number of calculations undertaken and the time taken to perform them, and therefore one can reach subtly different conclusions depending on which basis is used to measure the computational cost.

8.1 Example 1

For the first example we investigate the equation

$$D^2y + D^{1/2}y + y = t^3 + 6t + \frac{3.2t^{2.5}}{\Gamma(0.5)} \tag{19}$$

which, under the appropriate initial conditions, has the exact solution

$$y(t) = t^3. \tag{20}$$

In each case, this results in a system containing only the single fractional order $\frac{1}{2}$. We would expect that Methods 1a, 1b, 2 and 3 would all have order 1.5 and that methods 1a(2) and 3(2) would have order 2. These expectations are confirmed by the results in Table 1.

8.2 Example 2

The second example shows how the loss of smoothness in the fractional derivative of the solution can lead to a reduction in order of the methods. We consider

$$D^2y + D^{1/2}y + y = t^2 + 2 + \frac{2.6666666667t^{1.5}}{\Gamma(0.5)}. \tag{21}$$

The equation is superficially very similar to Example 1 but this time (subject to suitable initial conditions) the solution is $y(t) = t^2$ and has the property that $D^{1/2}y$ is not twice continuously differentiable. The orders of convergence observed for Methods 1a(2) and 3(2) confirm that the order 2 is no longer attained.

8.3 Example 3

In the third example, we explore the effect of dimension increase when the original decomposition as a system of a single order is used. For the equation

$$D^2y + D^{3/4}y + y = t^3 + 6t + \frac{8.533333333t^{2.25}}{\Gamma(0.25)} \tag{22}$$

h	Method 1a				Method 1b			
	error y(1)	KFlops	Time	EOC	error y(1)	KFlops	Time	EOC
1/32	1.86e-02	14.8	0.00		1.60e-02	16.0	0.00	
1/64	6.62e-03	46.0	0.06	1.49	5.85e-03	41.2	0.00	1.45
1/128	2.35e-03	157.4	0.11	1.49	2.12e-03	119.2	0.05	1.47
1/256	8.36e-04	576.8	0.22	1.49	7.59e-04	385.7	0.17	1.48
1/512	2.96e-04	2202.2	0.50	1.50	2.71e-04	1361.2	0.32	1.49
	Method 2				Method 3			
1/32	1.27e-03	9.9	0.06		1.86e-02	27.0	0.05	
1/64	4.32e-04	22.8	0.05	1.56	6.62e-03	70.1	0.11	1.49
1/128	1.49e-04	57.7	0.11	1.54	2.35e-03	205.4	0.17	1.49
1/256	5.14e-05	164.6	0.22	1.54	8.36e-04	672.6	0.49	1.49
1/512	1.79e-05	525.6	0.55	1.52	2.96e-04	2393.4	1.21	1.50
	Method 1a(2)				Method 3(2)			
1/32	4.61e-04	22.1	0.00		4.61e-04	39.0	0.11	
1/64	8.96e-05	68.7	0.06	2.36	8.96e-05	102.4	0.16	2.36
1/128	1.80e-05	235.6	0.16	2.32	1.80e-05	302.8	0.33	2.32
1/256	3.76e-06	864.3	0.28	2.26	3.76e-06	998.4	0.82	2.26
1/512	8.14e-07	3301.4	0.77	2.21	8.14e-07	3569.4	1.87	2.21

Table 1 $D^2y + D^{1/2}y + y = t^3 + 6t + \frac{3.2t^{2.5}}{\Gamma(0.5)}$. Exact solution $y(t) = t^3$

which, with appropriate initial conditions, has solution $y(t) = t^3$. The dimension of the system to be solved using methods 1a and 1b is 8, while methods 2 and 3 use a system of dimension 3. The orders of convergence are expected to be 1.25 for methods 1a, 2 and 3, 1.75 for method 1b and 2.0 for methods 1a(4) and 3(4). These are reflected in Table 3.

However, the results in Table 3 are very interesting because they show the importance of considering execution time as well as the number of operations in calculating the cost of applying an algorithm. Method 1a(4) uses considerably more computations than method 3(4). However, the execution speed is faster for method 1a(4) because the system being solved is all of the same order and therefore the software can take advantage of this and use various short-cuts. This means that method 1a(n) may be preferred above method 3(n) even when the dimension of the system to be solved using method 1a is somewhat larger than the dimension of the system using method 3. However, our detailed experimentation has shown that any advantage of method 1a over method 3 vanishes before the dimension of the method 1a decomposition reaches three times the dimension of the method 3 decomposition. Notice also that method 2 is by far the fastest method for producing a reasonably accurate solution, but its lower order of accuracy makes it uncompetitive for problems where higher degrees of accuracy are demanded.

9 Concluding remarks

All the methods we have considered in this paper can be used effectively for the solution of multi-term equations and the choice of a suitable method for any particular applica-

h	Method 1a				Method 1b			
	error y(1)	KFlops	Time	EOC	error y(1)	KFlops	Time	EOC
1/32	4.84e-03	14.8	0.00		1.30e-02	16.0	0.00	
1/64	1.66e-03	45.8	0.06	1.54	7.40e-03	41.1	0.00	0.81
1/128	5.74e-04	157.1	0.11	1.53	4.02e-03	118.9	0.06	0.88
1/256	2.00e-04	576.3	0.22	1.52	2.12e-03	385.2	0.16	0.92
1/512	7.00e-05	2201.2	0.50	1.51	1.10e-03	1360.2	0.38	0.95
	Method 2				Method 3			
1/32	9.08e-04	9.8	0.06		4.84e-03	26.9	0.05	
1/64	3.32e-04	22.6	0.05	1.45	1.66e-03	69.9	0.11	1.54
1/128	1.20e-04	57.5	0.11	1.47	5.74e-04	205.1	0.21	1.53
1/256	4.30e-05	164.0	0.22	1.48	2.00e-04	672.0	0.55	1.52
1/512	1.54e-05	524.6	0.60	1.48	7.00e-05	2392.4	1.27	1.51
	Method 1a(2)				Method 3(2)			
1/32	6.06e-04	22.0	0.06		6.06e-04	39.0	0.05	
1/64	2.73e-04	68.5	0.06	1.15	2.73e-04	102.2	0.16	1.15
1/128	1.13e-04	235.3	0.17	1.27	1.13e-04	302.5	0.33	1.27
1/256	4.46e-05	863.8	0.33	1.34	4.46e-05	997.9	0.77	1.34
1/512	1.70e-05	3300.4	0.77	1.39	1.70e-05	3568.4	1.97	1.39

Table 2: $D^2y + D^{1/2}y + y = t^2 + 2 + \frac{2.6666666667t^{1.5}}{\Gamma(0.5)}$ Exact solution $y(t) = t^2$

tion must take account of the precise fractional orders encountered in the problem and the necessary level of accuracy required in the solution. For highly accurate solutions, one should consider the decompositions given by method 1 and method 3 (according to the dimension of the resulting system of equations). If the dimensions are reasonably similar then method 1 combined with the predictor-corrector algorithm is likely to be most efficient. If the dimension of the method 1 decomposition is much larger than the dimension of the method 3 decomposition, then we would prefer the method 3 decomposition combined with the predictor-corrector algorithm. Note that the optimal number of corrector iterations can be determined formulaically using the approach introduced in [4] and outlined in Section 6.

References

[1] Bagley, R. L. & Torvik, P. J. (1984), *On the appearance of the fractional derivative in the behaviour of real materials*, J. Appl. Mech., 51, 294–298.

[2] Chern, J. T. (1993), *Finite element modelling of viscoelastic materials on the theory of fractional calculus*, PhD Thesis, Pennsylvania State University.

[3] Diethelm, K. (1997), *An algorithm for the numerical solution of differential equations of fractional order.* Elec. Transact. Numer. Anal. 5, 1–6.

[4] Diethelm, K. (2003), *Efficient solution of multi-term fractional differential equations using $P(EC)^mE$ methods*, Computing 71, 305–319.

h	Method 1a				Method 1b			
	error y(1)	KFlops	Time	EOC	error y(1)	KFlops	Time	EOC
1/32	1.12e-01	25.3	0.00		7.37e-03	44.7	0.05	
1/64	4.72e-02	83.4	0.06	1.24	2.34e-03	106.6	0.05	1.66
1/128	1.99e-02	297.7	0.11	1.24	7.30e-04	282.8	0.11	1.68
1/256	8.41e-03	1119.6	0.22	1.24	2.26e-04	844.0	0.22	1.69
1/512	3.54e-03	4336.2	0.72	1.25	6.93e-05	2802.0	0.49	1.71
	Method 2				Method 3			
1/32	3.86e-03	9.9	0.00		2.07e-02	27.0	0.06	
1/64	1.61e-03	22.8	0.06	1.26	8.20e-03	70.1	0.11	1.34
1/128	6.71e-04	57.7	0.05	1.26	3.29e-03	205.4	0.22	1.32
1/256	2.81e-04	164.6	0.22	1.26	1.33e-03	672.6	0.55	1.31
1/512	1.18e-04	525.6	0.55	1.25	5.43e-04	2393.4	1.21	1.29
	Method 1a(4)				Method 3(4)			
1/32	1.42e-03	64.0	0.05		1.48e-03	51.1	0.11	
1/64	3.07e-04	209.9	0.11	2.21	3.62e-04	134.7	0.22	2.03
1/128	6.62e-05	747.3	0.28	2.21	8.75e-05	400.2	0.49	2.05
1/256	1.43e-05	2805.3	0.60	2.21	2.11e-05	1324.3	1.10	2.05
1/512	3.10e-06	10853.4	1.81	2.21	5.07e-06	4745.5	2.53	2.06

Table 3: $D^2y + D^{3/4}y + y = t^3 + 6t + \frac{8.533333333t^{2.25}}{\Gamma(0.25)}$ Exact solution $y(t) = t^3$

[5] Diethelm, K., Ford, J. M., Ford, N. J., & Weilbeer, M. (2006), *Pitfalls in fast numerical solution of fractional differential equations*, J. Comput. Appl. Math. 186, 482–503.

[6] Diethelm, K. & Ford, N. J. (2001), *Numerical solution methods for distributed order differential equations.* Fract. Calc. Appl. Anal. 4, 531–542.

[7] Diethelm, K. & Ford, N. J. (2002), *Analysis of fractional differential equations.* J. Math. Anal. Appl. 265, 229–248.

[8] Diethelm, K. & Ford, N. J. (2002), *Numerical solution of the Bagley-Torvik equation.* BIT 42, 490–507.

[9] Diethelm, K. & Ford, N. J. (2004), *Multi-order fractional differential equations and their numerical solution.* Appl. Math. Comp. 154, 621–640.

[10] Diethelm, K., Ford, N. J. & Freed, A. D. (2004), *Detailed error analysis for a fractional Adams method.* Numerical Algorithms 36, 31–52.

[11] Diethelm, K., Ford, N. J., Freed A. D. & and Luchko, Y. (2005), *Algorithms for the Fractional Calculus: A Selection of Numerical Methods*, Computer Methods in Applied Mechanics and Engineering, 194, 743–773.

[12] Diethelm, K. & Ford, N. J. (2007), *Numerical analysis for distributed order differential equations*, submitted.

[13] Diethelm, K. & Luchko, Y. (2004), *Numerical solution of linear multi-term initial value problems of fractional order*, J. Comput. Anal. Appl. 6, 243–263.

[14] Edwards, J. T., Ford, N. J., & Simpson, A. C. (2002), *The numerical solution of multi-term fractional differential equations:systems of equations*, J. Comput. Appl. Math. 148, 401–418.

[15] Ford, N. J., & Connolly, J. A. (2006), *Comparison of numerical methods for fractional differential equations*, Communications in Pure and Applied Analysis, 5, 289–307.

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

[17] Lubich, C. (1988), *Convolution quadrature and discretized operational calculus. I*, Num. Math. 52, 129–145.

[18] Lubich, C. (1988), *Convolution quadrature and discretized operational calculus. II*, Num. Math. 52, 413–425.

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

[20] Lubich, C. (2004), *Convolution quadrature revisited*, BIT 44, 503–514.

[21] Luchko, Y., & Gorenfl R. (1999), *An operational method for solving fractional differential equations with the Caputo derivative*, Acta Mathematica Vietnamica, 24, 207–233.

[22] Miller, K. S. & Ross, B. (1993), *An Introduction to the Fractional Calculus and Fractional Differential Equations* John Wiley and Sons.

[23] Nkamnang, A. R. (1998), *Diskretisierung von mehrgliedrigen Abelschen Integralgleichungen und gewöhnlichen Differentialgleichungen gebrochener Ordnung*, PhD Dissertation, Freie Universiät Berlin.

[24] Podlubny, I. (1999), *Fractional Differential Equations*, Academic Press, San Diego.

[25] Samko S. G., Kilbas, A. A. & Marichev, O. I. (1997), *Fractional Integrals and Derivatives: Theory and applications,* Gordon and Breach Science Publishers.