Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Chapter 3

Specific Efficient Methods for the Solution of Ordinary and Partial Fractional Differential Equations

In the previous chapter we had introduced a number of numerical methods for (ordinary and partial) fractional differential equations. Our goal now is to look more closely at some particularly important methods from this selection. In particular we want to provide the reader with some basic knowledge that allows to decide for a concrete algorithm for a specific problem at hand. Moreover we want to mention a few techniques that are generally applicable to many classes of algorithms and that allow to obtain slightly modified versions that retain the convergence behavior but that require less computational effort.

The source codes of some very important special algorithms will be listed in Appendix A.

3.1 Methods for Ordinary Differential Equations

We shall begin with the investigation of algorithms for ordinary fractional differential equations; partial fractional differential equations will be treated later.

3.1.1 *Dealing with non-locality: The finite memory principle, nested meshes, and the approaches of Deng and Li*

A key problem in connection with the numerical solution of fractional differential equations is the fact that fractional differential operators are never local. This means that, whenever we want to compute the value of the

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

expression ${}^{\mathrm{C}}D^{\alpha}y(x)$, we need to take into consideration the entire history of the function y, i.e. the function values on the complete interval $[0, x]$. In contrast, for a derivative of integer order it would be sufficient to have information on an arbitrarily small neighborhood of x. This crucial difference implies, as can be seen from a close inspection of the numerical algorithms introduced in Chapter 2, the computation of a numerical solution of an ordinary fractional differential equation on a fixed interval $[0, b]$, say, by one of the standard algorithms using a step size of h has an arithmetic complexity of $O(h^{-2})$ which is much more effort than the $O(h^{-1})$ operation count that we can observe for differential equations of first order. In particular if one is interested in a solution over a rather long interval, this can be a serious problem. Therefore a number of ideas have been proposed in order to overcome this difficulty. We shall look at the three best known and most frequently used concepts and discuss their merits.

The first of these three methods is the *finite memory principle* (sometimes also denoted as *short memory principle*). This very intuitive idea is described, e.g., in Section 7.3 of [453]. Basically, one defines a fixed value $T > 0$, the so-called memory length, and modifies the Caputo derivative

$$ {}^{\mathrm{C}}D^{\alpha}y(x) = \frac{1}{\Gamma(n-\alpha)} \int_0^x (x-t)^{n-\alpha-1} y^{(n)}(t)dt $$

(where $n = \lceil \alpha \rceil$) of the function y if $x > T$. (No changes are necessary for $x \leq T$.) In the case $x > T$ the length of the range of integration, i.e. the length of the actual memory used in the definition of the Caputo operator, is greater than the prescribed memory length T, and it grows even larger as x grows. The idea of the finite memory principle is then not to use the complete memory but only a portion of length T of it. Thus one uses

$$ \frac{1}{\Gamma(n-\alpha)} \int_{x-T}^x (x-t)^{n-\alpha-1} y^{(n)}(t)dt $$

instead of ${}^{C}D^{\alpha}y(x)$ if $x > T$. In mathematical terms, we increase the lower terminal point of the interval of integration, i.e. we choose to take into account the right part of $[0, x]$ and to ignore the contribution of the left part of this interval. This choice has been taken because of the monotonicity property of the kernel $(x-t)^{n-\alpha-1}$: the exponent of this expression is negative, and so the contribution of the left part is likely to be much smaller

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

than the contribution of the right part, thus this choice minimizes the error introduced by the scheme. In this way one always needs to approximate an operator that only takes into account function values from the interval $[x - T, x]$ whose length never exceeds T. Hence, this very simple strategy allows to reduce the arithmetic complexity of standard numerical methods to the $O(h^{-1})$ count that we know from algorithms for first-order equations.

Of course, there is a price that we have to pay for this reduction in computational cost, and this is a loss of accuracy that this due to the fact that we simply ignore a part of the integration interval. Some authors claim that it is possible to keep this loss of accuracy so small that the error introduced by the numerical scheme for the differential equation itself still dominates the newly introduced contribution, thus allowing us to effectively neglect the latter. However, the careful inspection of Ford and Simpson [224] reveals that in order to really achieve this goal, one must choose the memory length T so large that the real improvement of the run time is negligible too. Thus it seems that this method is too simple to be successful in practice.

To overcome these problems, Ford and Simpson [224] have then derived a modified approach that allows to retain the order of convergence of the underlying basic numerical algorithm and at the same time reduce the arithmetic complexity to $O(h^{-1} \ln |h|)$ which is much better than the original $O(h^{-2})$ operation count and only marginally worse than the $O(h^{-1})$ that we know from the theory of first-order equations. Their idea is best explained by looking at a concrete example. For this purpose we shall choose the quadrature-based direct method developed in Subsection 2.2.2 and consider the practically most important case $0 < \alpha < 1$. We recall that it was based on a direct approximation of the differential operator in the given initial value problem

$$ {}_{\mathrm{C}}D^{\alpha} y(x) = f(x, y(x)), \qquad y^{(k)}(0) = y_0^{(k)}, \quad k = 0, 1, \ldots, \lceil \alpha \rceil - 1. \quad (3.1.1) $$

with the help of a quadrature formula with equispaced nodes. Thus, at the grid point x_j we need to take into consideration function values of the given function f at all previously handled grid points $x_0, x_1, \ldots, x_{j-1}$ which gives rise to an $O(j^2)$ operation count after j steps and hence $O(N^2) = O(h^{-2})$ operations at the end point of the interval of interest. Now we recall the observation that Podlubny had already used in his finite memory

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

principle but use it in a different way. To be precise, in the analysis of the finite memory approach we had seen that we cannot afford to ignore the contribution coming from the parts that are a long way away from x completely. But it is possible to approximate the corresponding part of the integral with a more economical formula, i.e. a formula that uses a smaller number of nodes. This will evidently reduce the arithmetic complexity and, if this new quadrature formula is chosen carefully, then we can still retain the order of convergence.

Specifically the idea is to choose two parameters $w > 0$ and $T > 0$, compute (for each x) the smallest integer m that satisfies $x < w^{m+1}T$ (i.e. $m = \lceil (\ln x - \ln T)/\ln w \rceil - 1$) and then to decompose the interval of integration according to

$$[0, x] = [0, x - w^m T] \cup [x - w^m T, x - w^{m-1} T] \cup \cdots \cup [x - wT, x - T] \cup [x - T, x].$$

One can then distribute the computational effort over the history of the function in a logarithmic instead of a uniform way. This means that we choose a basic step size $h > 0$ but use this step size only on the rightmost subinterval of the partition mentioned above, i.e. on $[x - T, x]$. As me move farther and farther away from the point x, we then increase the step size according to a simple strategy: On an interval of the form $[x - w^j T, x - w^{j-1} T]$ we use a step size of $w^{j-1}h$. The leftmost interval may not be amenable to this approach because it is possible that its length is not an integer multiple of the required step size; one can simply use the basic step size h here without sacrificing too much run time because this interval is rather small. An example of the distribution of the quadrature nodes is given in Fig. 3.1 where we have chosen $w = 2$. The scheme requires only 91 nodes for this particular choice of the parameters whereas a uniform grid would use 201 nodes. It is clear that the advantage of the nested mesh approach grows as x becomes larger; to be precise the number of nodes used at the grid point $x = kh$ reduces from $k + 1$ to $O(\ln k)$. This means that the overall complexity of a differential equation solver with step size h that uses this approach is only $O(h^{-1} |\ln h|)$ and not $O(h^{-2})$ as a standard approach would have. Nevertheless it is possible to prove a very pleasant result about the error of the scheme (see Theorem 1 of [224]):

Theorem 3.1. *The nested mesh scheme preserves the order of the underlying quadrature rule on which it is based.*

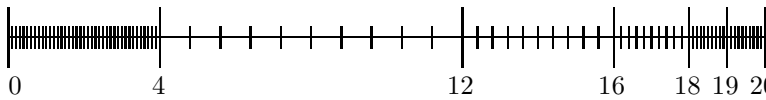

Fig. 3.1 Distribution of quadrature nodes according to the nested mesh principle of Ford and Simpson for $x = 20$, $w = 2$, $h = 1/10$ and $T = 1$. The step sizes are $h = 1/10$ in the leftmost subinterval $[0, 4]$ and in the two rightmost subintervals $[18, 19]$ and $[19, 20]$, $h = 2/10$ in $[16, 18]$, $h = 4/10$ in $[12, 16]$ and $h = 8/10$ in $[4, 12]$.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Yet another promising approach (that can actually be combined with the nested mesh idea to produce an even stronger method) has been suggested by Deng [166]. It is based on an analytic observation that has its origin in a well known property of first-order equations. In order to explain the idea we shall once again start from the usual initial value problem

$$ {}^{\mathrm{C}}D^{\alpha}y(x) = f(x, y(x)), \qquad y^{(k)}(0) = y_0^{(k)}, \quad k = 0, 1, \ldots, \lceil \alpha \rceil - 1, \tag{3.1.2} $$

and recall that it can be rewritten in the equivalent form of a Volterra integral equation

$$ y(x) = \sum_{k=0}^{\lceil \alpha \rceil - 1} \frac{y_0^{(k)}}{k!} x^k + \frac{1}{\Gamma(\alpha)} \int_0^x (x - t)^{\alpha - 1} f(t, y(t)) dt \tag{3.1.3} $$

(see, e.g., Lemma 6.2 of [172]). Now we look at this relation at two consecutive grid points, i.e. for $x = x_j$ and $x = x_{j+1}$, and subtract the equations from each other. This yields

$$ \begin{aligned} & y(x_{j+1}) - y(x_j) \\ &= \sum_{k=0}^{\lceil \alpha \rceil - 1} \frac{y_0^{(k)}}{k!} (x_{j+1}^k - x_j^k) + \frac{1}{\Gamma(\alpha)} \int_{x_j}^{x_{j+1}} (x_{j+1} - t)^{\alpha - 1} f(t, y(t)) dt \\ &\quad + \frac{1}{\Gamma(\alpha)} \int_0^{x_j} \left((x_{j+1} - t)^{\alpha - 1} - (x_j - t)^{\alpha - 1} \right) f(t, y(t)) dt \end{aligned} $$

or, equivalently,

$$ \begin{aligned} y(x_{j+1}) = y(x_j) &+ \sum_{k=0}^{\lceil \alpha \rceil - 1} \frac{y_0^{(k)}}{k!} (x_{j+1}^k - x_j^k) \\ &+ \frac{1}{\Gamma(\alpha)} \int_{x_j}^{x_{j+1}} (x_{j+1} - t)^{\alpha - 1} f(t, y(t)) dt \\ &+ \frac{1}{\Gamma(\alpha)} \int_0^{x_j} \left[(x_{j+1} - t)^{\alpha - 1} - (x_j - t)^{\alpha - 1} \right] f(t, y(t)) dt. \end{aligned} \tag{3.1.4} $$

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Now when we want to approximately compute $y(x_{j+1})$ then we have already computed the expression $y(x_j)$ that appears on the right-hand side so we only need to look this value up which can be achieved in constant time. In addition, the sum on the right-hand side of Eq. (3.1.4) is zero for $0 < \alpha \leq 1$ and it can be easily computed in $O(1)$ time, independent of j, if $\alpha > 1$. Moreover the first integral on the right-hand side extends over an interval of length h, also independent of j, and therefore it can be computed by an appropriate quadrature formula in $O(1)$ time as well. Thus all that remains to be done is to construct an efficient approximation for the second integral on the right-hand side, i.e. the integral that contains the history of the function. It is well known that this integral is zero for the classical (non-fractional) case $\alpha = 1$ (this is clear because the factor in brackets vanishes). Unfortunately no such simplification is available when $\alpha \neq 1$, but at least we may still observe that $(x_{j+1} - t)^{\alpha-1} \approx (x_j - t)^{\alpha-1}$ if $x_j - t \gg x_{j+1} - x_j$. In other words, the factor in brackets is very small at least for those t that are relatively far away from x_j and thus it is unproblematic to use a less accurate but computationally cheap quadrature formula (e.g. a formula with a rather coarse mesh) in this range.

It is sometimes also helpful to use the following alternative representation discussed by Li [342] for the Riemann-Liouville integral operator J_a^α.

Theorem 3.2. *Let $f \in C[0, b]$, $\alpha > 0$ and $x \in [0, b]$. Then,*

$$J^\alpha f(x) = \frac{1}{\Gamma(\alpha)\Gamma(1-\alpha)} \int_0^\infty g(\xi, x)\xi^{-\alpha} d\xi$$

where

$$g(\xi, x) = \int_0^x \exp(-\xi(x-t)) f(t) dt.$$

Remark 3.1. Note that the function g is actually independent of α. Thus, if one needs to compute $J^\alpha f$ for various different values of α then it suffices to determine g once; only the integration of $g(\cdot, x)(\cdot)^{-\alpha}$ needs to be performed a multiple number of times.

The main application of Theorem 3.2 is based on the following property of the function g which is an immediate consequence of its definition.

Lemma 3.1. *Let $f \in C[0,b]$, $\alpha > 0$, $h > 0$ and $x, x+h \in [0,b]$. Then, the function g introduced in Theorem 3.2 satisfies*

$$g(\xi, x+h) = \exp(-\xi h) g(\xi, x) + \tilde{g}(\xi, x, h)$$

for all $\xi \geq 0$, where

$$\tilde{g}(\xi, x, h) = \int_x^{x+h} \exp(-\xi(x+h-t)) f(t) dt.$$

Lemma 3.1 in conjunction with the representation of Theorem 3.2 allows us to establish a useful way of computing $J^\alpha f(x+h)$ for some $h > 0$ if $J^\alpha f(x)$ is already known. Specifically, while a direct application of the definition of J^α would require us to recompute the entire integral from 0 to x, and thus to re-evaluate the complete history of f, we are now in a position to invoke the auxiliary function g. Assuming that we have already computed $g(\cdot, x)$, the computation of $g(\cdot, x+h)$ requires only the multiplication of $g(\cdot, x)$ by a constant and the addition of a locally defined integral, i.e. an integral over the short interval $[x, x+h]$, followed by the evaluation of the integral $\int_0^\infty g(\xi, x) \xi^{-n} d\xi$ by a suitable method. Efficient numerical techniques for the latter problem have been discussed by Li [342].

Remark 3.2. For the sake of completeness we remark that the function g from Theorem 3.2 solves the first-order initial value problem

$$\frac{\partial}{\partial x} g(\xi, x) = -\xi g(\xi, x) + f(x), \quad g(\xi, 0) = 0$$

which can be easily verified by inserting the definition of g. Note that, since ξ can be considered to be a constant parameter in this differential equation, we effectively deal with a linear inhomogeneous ordinary differential equation with constant coefficients. Thus, this approach for the Riemann-Liouville integral is very similar to the techniques for Caputo-type derivatives mentioned at the end of Section 2.5.

3.1.2 *Parallelization of algorithms*

Another straightforward approach to tackle the computational complexity introduced by the non-locality of the fractional differential operators is based on using parallel computers. We shall explain this approach, first

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

discussed in [173], on the basis of the Adams-Bashforth-Moulton method in its standard form. However, the basic principle can also be applied to modifications of this algorithm like the nested mesh concept described above or even to many completely different classes of numerical methods.

To this end, we shall use the usual initial value problem

$$ {}^{\mathrm{C}}D^{\alpha}y(x) = f(x, y(x)), \qquad y^{(k)}(0) = y_0^{(k)}, \quad k = 0, 1, \ldots, \lceil \alpha \rceil - 1 \tag{3.1.5} $$

and recall the standard Adams-Bashforth-Moulton method from Subsection 2.3.2. Our first observation is that the computation of the weights a_{jk} and b_{jk} of the method as given by Eqs. (2.1.7) and (2.1.9), respectively, is computationally extremely cheap compared to the effort required for the evaluation of the approximate solution according to Eqs. (2.3.5) and (2.3.6), respectively. Thus we shall concentrate on the parallelization of the algorithm for the latter equations.

For this purpose, we assume to have a system with p cores. Our idea is then to divide the set $y_1, y_2, \ldots, y_N$ of values to be computed into blocks of size p in such a way that the ℓth block contains the variables $y_{(\ell-1)p+1}, \ldots, y_{\ell p}$ ($\ell = 1, 2, \ldots, \lceil N/p \rceil - 1$), and the $\lceil N/p \rceil$th block contains the remaining variables $y_{(\lceil N/p \rceil - 1)p+1}, \ldots, y_N$. (This last block contains p elements if and only if p is a divisor of N; otherwise it will contain less than p elements. No problems will result from this fact.) The processing will then happen by handling one block after the other, and during the handling of each block, each processor will be assigned to exactly one of the block's variables. After the computations for each block have been finished, all processors communicate their results to the other processors. This implies that, at the beginning of the ℓth block, all y_j computed in the previous blocks are known to all processors. Based on this structure, the exact procedure used in the computations for the ℓth block can now be given.

The key to an efficient parallelization of the Adams method is that we exploit the structure of the sums in Eqs. (2.3.6) and (2.3.5). Specifically we rewrite these two sums in the forms

$$ y_{j+1,0} = I_{j+1} + h^{\alpha} H_{j,\ell}^{(0)} + h^{\alpha} L_{j,\ell}^{(0)} \tag{3.1.6} $$

and

$$ y_{j+1} = I_{j+1} + h^{\alpha} H_{j,\ell} + h^{\alpha} L_{j,\ell}, \tag{3.1.7} $$

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

respectively. Here,

$$I_{j+1} = \sum_{k=0}^{\lceil\alpha\rceil-1} \frac{x_{j+1}^k}{k!} y_0^{(k)} \tag{3.1.8}$$

is a sum with a fixed and typically very small number of summands that appears in both sums. Moreover, the other quantities appearing in Eqs. (3.1.6) and (3.1.7) are defined by

$$H_{j,\ell}^{(0)} = \sum_{k=0}^{(\ell-1)p} b_{j-k} f(x_k, y_k),$$

$$L_{j,\ell}^{(0)} = \sum_{k=(\ell-1)p+1}^{j} b_{j-k} f(x_k, y_k),$$

$$H_{j,\ell} = c_j f(x_0, y_0) + \sum_{k=1}^{(\ell-1)p} a_{j-k} f(x_k, y_k),$$

and

$$L_{j,\ell} = \sum_{k=(\ell-1)p+1}^{j} a_{j-k} f(x_k, y_k) + \frac{f(x_{j+1}, y_{j+1,0})}{\Gamma(\alpha+2)}. \tag{3.1.9}$$

We need to recall here that, in the block ℓ presently under consideration, these expressions must be calculated for $j = (\ell-1)p, (\ell-1)p+1, \ldots, \ell p-1$. Each of our p processors will then be assigned with the task of computing $y_{j+1,0}$ and y_{j+1} for exactly one value of j. To this end, each processor first computes the sums I_{j+1}, $H_{j,\ell}^{(0)}$ and $H_{j,\ell}$ for its value of j. With respect to these computations, we remark a few observations:

(1) For these computations, one needs to know only the initial values (which are given as part of the initial value problem), the weights of the algorithm (that have been computed in advance) and the y_k for $k \leq (\ell-1)p$, i.e. approximate solution values that have already been computed in previous blocks. Thus, at the beginning of the ℓth block, all required data are available.

(2) The computations to be performed by any of the processors are completely independent of the computations of the others processors. In particular, no communication is required between the processors in this part of the block.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

(3) Each processor has the task of computing I_j, i.e. a sum of $\lceil\alpha\rceil$ terms, and two sums of $(\ell-1)p+1$ terms each, namely $H^{(0)}_{j,\ell}$ and $H_{\ell,j}$. Thus, the work load is distributed across the processors in a uniform way; we hence have a very good load balancing, and it may be expected that all processors require similar amounts of time for this part of the task.
(4) The amount of time required for this part of the block depends on ℓ in a linear way. When ℓ is small, it can be completed very quickly, but as ℓ increases it will take longer and longer.

In the second part of the block we will deal with the remaining sums $L^{(0)}_{j,\ell}$ and $L_{j,\ell}$. In this context we note that the sum $L^{(0)}_{j,\ell}$ is empty for $j=(\ell-1)p$ (i.e. the smallest of the indices under consideration in the current block). Thus, for this value of j the value $y_{j+1,0}$ can be computed by the appropriate processor. Moreover, the expression $L_{j,\ell}=L_{(\ell-1)p,\ell}$ in this case reduces to $f(x_{(\ell-1)p+1},y_{(\ell-1)p+1,0})/\Gamma(\alpha+2)$ which can now also be computed, and so we can complete the calculation of $y_{(\ell-1)p+1}$. The processor associated with this task then additionally computes $f(x_{(\ell-1)p+1},y_{(\ell-1)p+1})$ and sends all its results to all other processors. Its job is finished for this block, and so it becomes idle. (Strictly speaking, it would not have been necessary to compute and store the value $f(x_j,y_j)$, but since this value will be used again and again in future steps, it is more efficient to do so than to call the function f with the same set of arguments for a large number of times.) The other processors have had to wait for this information about $y_{(\ell-1)p+1}$, but once they have received it, they are in a position to compute the first summands of their respective values $L^{(0)}_{j,\ell}$ and $L_{j,\ell}$. This sum only consists of this one summand for the next value of j, viz. $j=(\ell-1)p+1$, so the associated processor can complete the calculation of $y_{j+1,0}=y_{(\ell-1)p+2,0}$ and subsequently also the computation of $y_{(\ell-1)p+2}$ and $f(x_{(\ell-1)p+2},y_{(\ell-1)p+2})$ itself since all the data required for the evaluation of $L_{(\ell-1)p+1,\ell}$ are now present. This processor then also passes the result of the computation to all other processors (including the first one that has already completed its task and that needs this value in the next block). In this fashion, we work our way through all y_j so that, at the end of this process, all processors have computed their y_j and $f(x_j,y_j)$ and passed the results to all other processors. This concludes the ℓth block, and we now have the data of y_j, $j=1,2,\ldots,\ell p$, available at all processors. With respect to this part of the

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

block, none of our four observations above applies. Instead we find:

(1) Only the first value $y_{(\ell-1)p+1}$ could be immediately computed. All other y_j needed to have input from earlier values computed in this block and had to wait for the other processors to provide these values.
(2) As a consequence of the first comment, a certain amount of communication between the processors is required. Specifically, each processor computes its assigned value y_j in the indicated way and then passes the result to all other processors.
(3) The work load is small for the first processor and monotonically increases up to the last processor, so in this part of the block we do not have a good load balance. On average, each processor remains idle for half of the time spent in the second part of the block.
(4) The absolute amount of time required for this part of the block is independent of ℓ. Thus, in view of our previous observation on the first part of the block, the relative amount of time that this part of the block takes becomes smaller as ℓ increases, and hence, as the algorithm proceeds, the inefficiency caused by the load imbalance becomes less and less severe.

The procedure described above for the ℓth block then needs to be repeated in an iterative manner for $\ell = 1, 2, \ldots, \lceil N/p \rceil - 1$. This will give us the numerical solution at the points $x = x_j$, $j = 1, 2, \ldots p(\lceil N/p \rceil - 1)$. For the remaining values x_j with $j = p(\lceil N/p \rceil - 1) + 1, \ldots, N$ we proceed in an almost identical way, except that now the number of values that need to be computed may be smaller than the number p of processors. (As mentioned above, the numbers will be equal if and only if p is a divisor of N.) Thus, for this last block we only employ as many processors as we have approximate function values that need to be computed, and these processors perform their computations in the same way as they did in the previous blocks; the other processors may remain idle.

For a detailed investigation of the performance of this parallelization of the Adams scheme, we refer to [173]. In particular it can be seen from the results reported there that this approach works very well for shared memory multiprocessor systems.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

3.1.3 *When and when not to use fractional linear multistep formulas*

Traditionally, the fractional linear multistep formulas (in particular, the fractional backward differentiation formulas) proposed by Lubich that we had introduced in Subsection 2.1.3 and in Section 2.4 have been considered to be very good methods for the numerical solution of fractional differential equations due to the fact that they combine a conceptual simplicity, a straightforward method to compute their coefficients from the corresponding coefficients of well known methods for first-order differential equations, and a high order of convergence. This opinion has been supported by many numerical examples in the early literature on numerical methods for fractional differential equations where these methods have been used very successfully [256, 355]. However, in those early days of this field of research, almost all fractional differential equations that arose in real-world applications were equations of order $1/2$, and so this was the case that was usually tested.

More recently it has turned out that one needs to be somewhat more careful in judging the quality of fractional linear multistep formulas [175]. Specifically, while it is indeed true that they work very well for certain values of the order α of the differential equation, they can seriously fail for other values of α. To explain the background of this phenomenon, it is useful to recall the construction of these methods. As seen in Definition 2.1, a fractional linear multistep method consists of two components, the convolution quadrature and the starting quadrature. In particular, the latter is responsible for eliminating all lower order terms in the error expansion, and hence for making sure that the convergence order is sufficiently high. Theorem 2.13 gives us a representation for the weights of both parts. The concept for the convolution part is explicit and rather simple to implement; in particular this can easily be done in a numerically stable way. The difficulty lies in the the starting quadrature. The weights w_{nj} of this formula are only given implicitly as the solution of the linear equation system (2.4.1). This system reads

$$\sum_{j=0}^{s} w_{nj} j^{\gamma} = \frac{\Gamma(\gamma+1)}{\Gamma(\gamma+\alpha+1)} n^{\gamma+\alpha} - \sum_{j=0}^{n} \omega_{n-j} j^{\gamma}, \quad \gamma \in A, \tag{3.1.10}$$

where $A = \{\gamma = k + j\alpha : j, k \in \mathbb{N}_0, \gamma \leq p-1\}$, $s+1$ is the number

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

of elements of A, and p is the order of convergence of the formula under consideration. The computation of the expressions on the right-hand side of this system requires the knowledge of the weights ω_{n-j} of the convolution quadrature, but as mentioned above this is not a major problem. Rather, the main obstacle lies in the structure of the coefficient matrix V of the system that is evident from the left-hand side. This matrix depends on α in a somewhat irregular way. If, for example, $p = 2$ and $\alpha = 1/2$ then we have $A = \{0, 1/2, 1\}$, and hence $s = 2$ and the matrix V has the form

$$V = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1^{1/2} & 2^{1/2} \\ 0 & 1^1 & 2^1 \end{pmatrix}.$$

This is essentially a Vandermode matrix which is known to be mildly ill-conditioned. A number of reliable algorithms for the accurate solution of systems with coefficient matrices of this type are known [98, 279]. These methods can, in principle, be used whenever $\alpha = 1/k$ with some integer k but, due to the structure of the equation system, they tend to become more and more unstable as k increases [175]. What is more, they cannot be applied directly at all if α is not the reciprocal of an integer.

As a particular example, let $p = 2$ and $\alpha = 0.499$ (i.e. we only perturb α by a rather small amount in comparison to the case described above). Then we have $A = \{0, 0.499, 0.998, 1\}$, hence $s = 3$ and the matrix V has the form

$$V = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1^{0.499} & 2^{0.499} & 3^{0.499} \\ 0 & 1^{0.998} & 2^{0.998} & 3^{0.998} \\ 0 & 1^1 & 2^1 & 3^1 \end{pmatrix} \approx \begin{pmatrix} 1 & 1 & 1.00000 & 1.00000 \\ 0 & 1 & 1.41323 & 1.73015 \\ 0 & 1 & 1.99723 & 2.99342 \\ 0 & 1 & 2.00000 & 3.00000 \end{pmatrix}.$$

Thus we observe that the small perturbation of α has resulted in a change of the dimension of the matrix and, what is much worse, the last two rows of the matrix are almost identical. In other words, the matrix is very close to a singular matrix; therefore its condition is extremely poor. Diethelm *et al.* [175] have reported results of a detailed comparison of various numerical approaches for handling this ill-conditioned system. The result was that the standard GMRES method [493, 494] and its modification using the Householder transform [573] usually gave the best results, but it was

rather slow. In contrast, a classical LU decomposition was much faster and produced only slightly worse results. The accuracy of the Björck-Pereyra method was comparable only for $\alpha = 1/2$, somewhat worse for $\alpha = 1/3$ and extremely poor for $\alpha = 1/k$ with $k > 3$; as mentioned above, this algorithm is not applicable at all for other values of α.

Thus the conclusion is that high order fractional linear multistep methods are a very useful concept in theory but as long as the problem of finding an accurate scheme for the numerical computation of the starting weights is unsolved, they cannot be reliably computed in practice except for a small set of values for α and therefore one should avoid using them unless the order of the differential equation under consideration happens to be in this set.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

3.1.4 *The use of series expansions*

An essential feature of fractional multistep formulas is that they consist of two components, the convolution quadrature and the starting quadrature. The main problem of these formulas that we had found in the previous subsection was that it is extremely difficult to compute the starting quadrature with a sufficient accuracy. No such problems are associated with the convolution quadrature. Thus one might be tempted to find some approach that avoids the use of the starting quadrature. In order to construct such an approach it is useful to recall the background that has lead us to introducing the starting quadrature in the first place. In fact the reason was that we wanted to have a high order of convergence for a most general class of equations, and it is known that the solutions of such equations are typically not smooth at the origin; rather they have an asymptotic expansion in powers of the form $x^{j+k\alpha}$ with $j, k \in \mathbb{N}$ (see [401] or Section 6.4 of [172]). The introduction of the starting quadratures was an attempt to make the complete formula exact for linear combinations of such powers with small j and k, thus eliminating all low order terms in the asymptotic expansion of the error with respect to the step size.

From this derivation we can see that there is an alternative to the introduction of the notoriously difficult starting quadrature. Specifically we can try to modify the initial value problem

$$ {}^{\mathrm{C}}D^{\alpha} y(x) = f(x, y(x)), \qquad y^{(k)}(0) = y_0^{(k)}, \quad k = 0, 1, \ldots, \lceil \alpha \rceil - 1, \tag{3.1.11} $$

so that the initial value problem newly obtained in this way has two important properties:

(a) Near the starting point 0, the solution $\tilde{y}$, say, of the modified problem has an asymptotic expansion of the form

$$\tilde{y}(x) = \sum_{j=0}^{\mu} \sum_{k=0}^{\nu} \tilde{c}_{jk} x^{j+k\alpha} + o(x^{\rho})$$

with $\rho > \mu + \nu\alpha$ and

$$\tilde{c}_{jk} = 0 \quad \text{for } k \neq 0. \tag{3.1.12}$$

(b) The solution of the original problem can be computed from the solution of the modified problem in an efficient and numerically stable manner.

Condition (3.1.12) then asserts that those terms of the asymptotic expansion of the original solution that required the use of starting quadratures to obtain the desired order of convergence are absent. Thus, as a consequence of this condition, we can dispense with the starting quadrature completely and use only the convolution quadrature part of the fractional multistep method without losing the order of convergence.

The key point in this approach is, of course, that we need to make sure that (3.1.12) is satisfied. This is by no means trivial but may be possible in certain situations. In particular, if we know the coefficients c_{jk} in the asymptotic expansion

$$y(x) = \sum_{j=0}^{\mu} \sum_{k=0}^{\nu} c_{jk} x^{j+k\alpha} + o(x^{\rho}) \tag{3.1.13}$$

of the exact solution of the initial value problem (3.1.11) then we can define

$$\tilde{y}(x) = y(x) - \sum_{j=0}^{\mu} \sum_{k=1}^{\nu} c_{jk} x^{j+k\alpha}$$

which immediately implies our required condition (3.1.12). Moreover, as a consequence of this definition and the original initial value problem (3.1.11)

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

we can see that $\tilde{y}$ solves the modified differential equation

$$\begin{aligned} {}^{\mathrm{C}}D^{\alpha}\tilde{y}(x) &= {}^{\mathrm{C}}D^{\alpha}y(x) - \sum_{j=0}^{\mu}\sum_{k=1}^{\nu}\frac{c_{jk}\Gamma(j+1+k\alpha)}{\Gamma(j+1+(k-1)\alpha)}x^{j+(k-1)\alpha} \\ &= f(x,\tilde{y}(x)+\sum_{j=0}^{\mu}\sum_{k=1}^{\nu}c_{jk}x^{j+k\alpha}) \qquad (3.1.14a) \\ &\quad -\sum_{j=0}^{\mu}\sum_{k=1}^{\nu}\frac{c_{jk}\Gamma(j+1+k\alpha)}{\Gamma(j+1+(k-1)\alpha)}x^{j+(k-1)\alpha} \end{aligned}$$

whereas the initial conditions can remain unchanged, i.e.

$$\tilde{y}^{(k)}(0) = y_0^{(k)}, \quad k = 0, 1, \ldots, \lceil\alpha\rceil - 1. \qquad (3.1.14b)$$

Thus the remaining problem is that we need to find the coefficients of the first terms of the asymptotic expansion (3.1.13). In a number of special cases it may be possible to achieve this via a differential transform method [431] or an Adomian decomposition [9]. Note that we do not need to find more than the first few coefficients of the expansion, so the poor convergence properties usually associated with Adomian's method (see our comments in Section 2.5) may be irrelevant in this context.

3.1.5 *The generalized Adams methods as an efficient tool for multi-order fractional differential equations*

In Section 2.7 we had dealt with the basics of multi-term equations, i.e. equations involving more than one fractional derivative. In particular we had constructed a number of possible methods to convert such equations into systems of single-order equations. In some of these approaches, all differential equations of the resulting system were of fractional order (see, e.g., Theorem 2.14); in other approaches (Theorem 2.15) we had a mixture of fractional-order and integer-order equations. The detailed investigations of Ford and Connolly [222] indicate that in most examples the former approach will lead to a computationally more efficient algorithm than the latter.

Of course, the reformulation of the given multi-term equation in the form of a system of single-term equations alone is not sufficient to obtain a numerical solution. Rather we need to solve the system that we have

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

created by a suitable numerical method. According to [198] and [221] one can say that the Adams method described in Subsection 2.3.2 will usually lead to an algorithm with satisfactory convergence and stability properties.

As an example we consider the initial value problem

$$ {}^{\mathrm{C}}D^{1.8}y(x) + 3x^2\ {}^{\mathrm{C}}D^{0.4}y(x) + 5(\sin x)y(x) = \exp x, \tag{3.1.15a} $$

$$ y(0) = 1, \quad y'(0) = -7. \tag{3.1.15b} $$

Using the approach of Theorem 2.14, we may rewrite this problem in the form

$$ \begin{aligned} {}^{\mathrm{C}}D^{0.4}y_1(x) &= y_2(x), \\ {}^{\mathrm{C}}D^{0.6}y_2(x) &= y_3(x), \\ {}^{\mathrm{C}}D^{0.8}y_3(x) &= -3x^2y_2(x) - 5(\sin x)y_1(x) + \exp x, \\ y_1(0) = 1, &\quad y_2(0) = 0, \quad y_3(0) = -7. \end{aligned} \tag{3.1.16} $$

The first component y_1 of the solution vector $(y_1, y_2, y_3)^{\mathrm{T}}$ of this system is then identical to the required solution y of our multi-term equation (3.1.15). A numerical computation using the Adams method of the corresponding order for each of the three constituent equations of the system (3.1.16) with step sizes $h = 1/10$, $h = 1/20$ and $h = 1/40$, respectively, produced the results depicted in Fig. 3.2.

Of course, an explicit expression for the exact solution of this system is not known, but the plots of the approximate solutions (and the underlying numerical data) clearly indicate that the method converges. The error analysis of Edwards *et al.* [198] confirms this observation; indeed we know the convergence order of each single equation from the theory of Subsection 2.3.2, and using the methods from [198] one can see that the convergence order of the overall method is just the minimum of the convergence orders of each individual equation.

Now let us look at a slight modification of the problem (3.1.15), namely the equation

$$ 0.01\ {}^{\mathrm{C}}D^{1.8}y(x) + 3x^2\ {}^{\mathrm{C}}D^{0.4}y(x) + 5(\sin x)y(x) = \exp x, \tag{3.1.17a} $$

$$ y(0) = 1, \quad y'(0) = -700 \tag{3.1.17b} $$

that is obtained by multiplying the highest order derivative in the original equation by a small coefficient and by multiplying the highest order initial

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

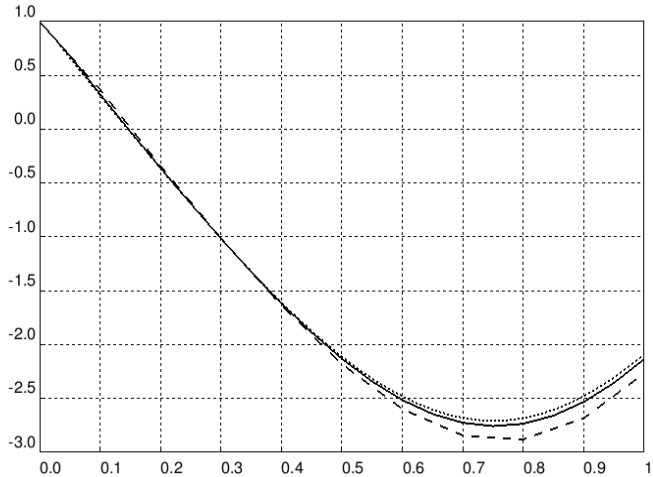

Fig. 3.2 Numerical solutions of Eq. (3.1.15) for $h = 1/10$ (dashed line), $h = 1/20$ (continuous line) and $h = 1/40$ (dotted line).

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

value by the reciprocal of this coefficient. Using the same approach as above, we may rewrite this problem in the form

$$\begin{aligned} {}^{\mathrm{C}}D^{0.4}y_1(x) &= y_2(x), \\ {}^{\mathrm{C}}D^{0.6}y_2(x) &= y_3(x), \\ {}^{\mathrm{C}}D^{0.8}y_3(x) &= -300x^2y_2(x) - 500(\sin x)y_1(x) + 100\exp x, \\ y_1(0) = 1, &\quad y_2(0) = 0, \quad y_3(0) = -700. \end{aligned} \tag{3.1.18}$$

In order to obtain a reliable approximation of the exact solution we have first computed the solution numerically with a number of very small step sizes until the differences between consecutive approximate solutions were negligible. It was sufficient to use a step size of $h = 1/2000$; the resulting graph is shown in Fig. 3.3 as a continuous line. Clearly, the total variation of the solution is much larger than in the previous example. This is not surprising in view of the modification of the initial values.

In particular, we find that $y(1) \approx -1.502667$. If we use step sizes in the approximation that are similar to those of the previous example then we arrive at completely meaningless results as indicated in Table 3.1. Only significantly smaller step sizes yield a useful numerical result.

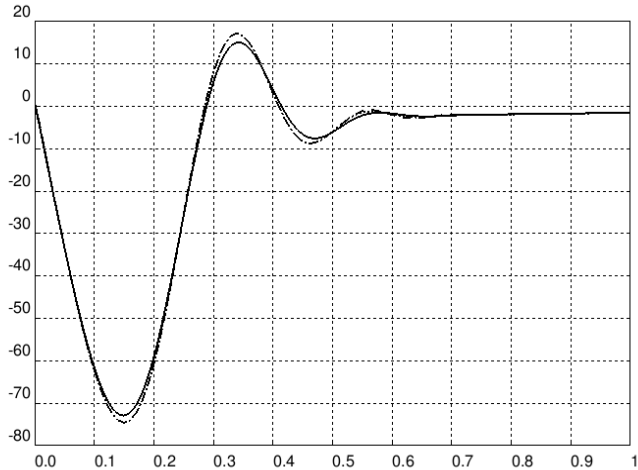

Fig. 3.3 Numerical solutions of Eq. (3.1.15) for uniform mesh with $h = 1/2000$ (continuous line; "exact" solution), uniform mesh with $h = 1/100$ (dotted line) and non-uniform mesh with $h = 1/100$ in the left half and $h = 1/60$ in the right half (dashed line).

Table 3.1. Numerical approximation for $y(1)$ from Eq. (3.1.17) using the Adams method and various step sizes.

step size h	approx. value of $y(1)$
1/10	984034.14
1/20	209443.96
1/40	−6.914879
1/100	−1.515596

The reason for this apparently strange behavior becomes evident by looking at the graph of the "exact" solution in Fig. 3.3. Specifically, the gradient of the solution is negative and very large in modulus near the starting point $x = 0$ (notice the scale on the y-axis of Fig. 3.3 in comparison to the scale of the y-axis in Fig. 3.2), then it becomes large and positive very quickly near $x = 0.15$, becomes negative again near $x = 0.35$ and only then decreases in modulus. The Adams method (or, indeed, most other methods) cannot follow these abrupt changes of the slope of y if the step size is too large, and so the numerical solution strongly deviates from the

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

exact solution. However, it is apparent from Fig. 3.3 that this problem does not persist throughout the entire interval $[0, 1]$ where the solution is sought. For $x > 0.5$ we observe only very mild variations in the slope of y, and therefore one could use a coarser mesh in this range while retaining the fine mesh in the left half of the interval. It is one of the advantages of the Adams method that such a non-uniform mesh can be implemented very easily. To be precise, denoting the grid points by t_j, the resulting formula for differential equations of order α with initial point 0 becomes (see, e.g., pp. 198–200 of [172])

$$y_{k+1} = \sum_{j=0}^{\lceil\alpha\rceil-1} \frac{t_{k+1}^j}{j!} y_0^{(j)} + \frac{1}{\Gamma(\alpha)} \left(\sum_{j=0}^{k} a_{j,k+1} f(t_j, y_j) + a_{k+1,k+1} f(t_{k+1}, y_{k+1}^{\mathrm{P}}) \right) \tag{3.1.19}$$

(this is the Adams-Moulton formula) with weights

$$a_{0,k+1} = \frac{(t_{k+1} - t_1)^{\alpha+1} + t_{k+1}^{\alpha}(\alpha t_1 + t_1 - t_{k+1})}{t_1 \alpha(\alpha+1)}, \tag{3.1.20a}$$

$$\begin{aligned} a_{j,k+1} &= \frac{(t_{k+1} - t_{j-1})^{\alpha+1} + (t_{k+1} - t_j)^{\alpha}(\alpha(t_{j-1} - t_j) + t_{j-1} - t_{k+1})}{(t_j - t_{j-1})\alpha(\alpha+1)} \\ &\quad + \frac{(t_{k+1} - t_{j+1})^{\alpha+1} - (t_{k+1} - t_j)^{\alpha}(\alpha(t_j - t_{j+1}) - t_{j+1} + t_{k+1})}{(t_{j+1} - t_j)\alpha(\alpha+1)} \end{aligned} \tag{3.1.20b}$$

if $1 \le j \le k$, and

$$a_{k+1,k+1} = \frac{(t_{k+1} - t_k)^{\alpha}}{\alpha(\alpha+1)}. \tag{3.1.20c}$$

The Adams-Bashforth predictor here is given by

$$y_{k+1}^{\mathrm{P}} = \sum_{j=0}^{\lceil\alpha\rceil-1} \frac{t_{k+1}^j}{j!} y_0^{(j)} + \frac{1}{\Gamma(\alpha)} \sum_{j=0}^{k} b_{j,k+1} f(t_j, y_j). \tag{3.1.21}$$

with

$$b_{j,k+1} = \frac{(t_{k+1} - t_j)^{\alpha} - (t_{k+1} - t_{j+1})^{\alpha}}{\alpha}. \tag{3.1.22}$$

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

We have applied this concept to the problem given in Eq. (3.1.17) with a mesh size of $1/100$ on $[0, 0.5]$ and $1/60$ on $[0.5, 1]$. The result is plotted in Fig. 3.3 as a dashed line. Specifically we obtained a value $y(1) \approx -1.514199$, so the accuracy was roughly comparable to the accuracy obtained by using a uniform grid with a mesh size $h = 1/100$ (the dotted line in Fig. 3.3; the dashed and dotted lines can hardly be distinguished from each other). However, the uniform grid has $N = 100$ mesh points whereas the non-uniform grid has only $N = 80$ mesh points. Since the computational cost is proportional to N^2, the use of the uniform grid is more than 50% more expensive than the non-uniform grid.

Another point worth mentioning in connection with the application of the Adams method to multi- or single-order systems arising from the reformulation of multi-term equations is connected to the structure of the initial value vector of the system. Specifically, a look at the equivalence results (Theorems 2.14 and 2.15) reveals that the initial conditions, given in Eqs. (2.7.7) and (2.7.12), respectively, may contain a large number of zeros. To be precise, only the coefficients that correspond to integer-order derivatives of the solution of the given multi-term equation may be non-zero values; all other components must vanish. Let us briefly look at an example to demonstrate why this may be a problem and what can be done to solve it. The simple linear problem

$$\sum_{k=0}^{14} (k+1)\, {}^{\mathrm{C}}D^{k/10} y(x) = \sin x, \qquad y(0) = 1, \quad y'(0) = 2, \tag{3.1.23}$$

will be sufficient to demonstrate the relevant effects. According to Theorem 2.14, the equivalent system of equations reads

$$\begin{aligned} {}^{\mathrm{C}}D^{1/10} y_j &= y_{j+1}, \qquad j = 1, 2, \ldots, 13, \qquad &\text{(3.1.24a)} \\ {}^{\mathrm{C}}D^{1/10} y_{14} &= \frac{1}{15} \left(\sin x - \sum_{k=1}^{14} k y_k(x) \right) \end{aligned}$$

subject to the initial conditions

$$y_k(0) = \begin{cases} 1 & \text{for } k = 1, \\ 2 & \text{for } k = 11, \\ 0 & \text{else.} \end{cases} \tag{3.1.24b}$$

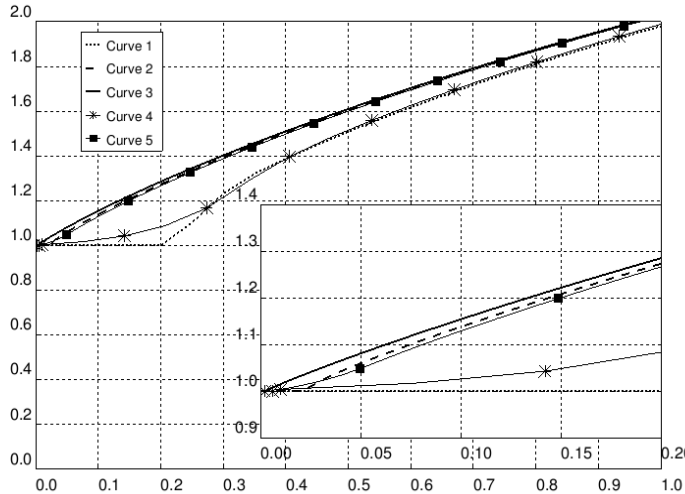

Fig. 3.4 Numerical solutions of Eq. (3.1.24) on $[0, 1]$ for uniform mesh with $h = 1/20$ (dotted line, Curve 1), uniform mesh with $h = 1/200$ (dashed line, Curve 2), uniform mesh with $h = 1/2000$ (continuous line, Curve 3; "exact" solution), non-uniform mesh with 5 steps of length $h = 1/500$ followed by 15 steps of length $h = 66/1000$ (continuous line marked with stars, Curve 4) and non-uniform mesh with 5 steps of length $h = 1/500$ followed by 100 steps of length $h = 99/10000$ (continuous line marked with squares, Curve 5). The inset in the bottom right corner is an enlarged view of the range $[0, 0.2]$ of the complete graph.

Figure 3.4 shows the approximate solution of this system on the interval $[0, 1]$ that we have obtained using the Adams-Bashforth-Moulton method with a uniform step size of $h = 1/20$ (dotted line). It is evident (in particular from the inset picture in the bottom left corner that shows an enlarged version of the graphs in the range $[0, 0.2]$) that this numerical solution is constant on the subinterval $[0, 0.2]$, i.e. on the first four steps of the algorithm. A comparison with a numerical solution with a much smaller step size ($h = 1/2000$; the continuous line in Fig. 3.4) that we once again consider to be very close to the exact solution reveals that this means that the exact solution moves away from the numerical solution for $h = 1/20$ (which, as stated above, remains constant) on a relatively large initial subinterval. Only when we have left this subinterval, the numerical solution for $h = 1/20$ can start to move towards the exact solution and diminish the approximation error. Of course, this task is much more difficult than a similar problem where no such initial interval problem exists.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

This behavior of the numerical solution does not seem to be justified by the analytical properties of the initial value problem. However, a close inspection of numerical values for other choices of the step size reveals that we can *always* observe this phenomenon: The approximate values for $y(x_1), \ldots, y(x_4)$ coincide with the initial value $y(0)$ in all these cases. The reason behind this observation becomes apparent when we look at how the Adams-Bashforth-Moulton method interacts with the system of equations (3.1.24) and, in particular, with the initial conditions (3.1.24b). To be precise, we note that the vector containing the initial conditions consists of the entry $y_0^{(0)}$ in the first component, followed by nine zeros. Only then the next non-zero entry may appear; its value comes from the given initial values of the original problem (3.1.23). The Predict-Evaluate-Correct-Evaluate (PECE) form of the Adams algorithm means that the non-zero elements are propagated by two rows in each step, and hence the algorithm needs to take five steps before it can produce an approximation that differs from the initial value.

Two possible strategies are immediately obvious in order to overcome these difficulties. The first approach uses the fact that we always need the same number of steps to get away from the initial value, no matter what step size we have chosen. Thus we can decide to choose a very small step size for the first few (in this example: four) steps, so that the exact solution will not move away too far from the initial value (i.e. from the numerical solution) on the interval covered by these first steps. Hence it will be much easier for the algorithm to bridge the gap and return to the neighborhood of the graph of the exact solution. Once the next (here: the fifth) step is reached, it is possible to return to a much coarser mesh in order to avoid excessive computational costs. The numerical results depicted in Fig. 3.4 (Curves 4 and, in particular, 5) demonstrate that this concept indeed allows us to obtain much better approximate solutions without increasing the complexity too much.

Alternatively we can also follow a completely different path originally proposed in [169]. As we had noted above, the problem that we are facing is due to the fact that there are very many (in this case: nine) zeros in the initial value vector, and that only two components are propagated at each step, so that it here takes five steps to eliminate all the zeros. The propagation by two components takes place in two parts of the algo-

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

rithm, one in the predictor (the Adams-Bashforth method) and one in the corrector (the Adams-Moulton method). Thus we can also solve the problem by introducing additional corrector iterations, hence moving from the plain PECE structure to a $\mathrm{P(EC)}^m\mathrm{E}$ form of the algorithm as indicated in Eq. (2.3.7). In our case, $m = 8$ would be an appropriate choice. According to Theorem 2.10 this approach has the pleasant side effect of increasing the convergence order. Moreover, a close inspection of the formula (2.3.7b) reveals that the corrector may be written in the form

$$y_{k,\mu} = \gamma_k + h^\alpha a_{kk} f(x_k, y_{k,\mu-1})$$

where

$$\gamma_k = \sum_{j=0}^{\lceil\alpha\rceil-1} \frac{x_k^j}{j!} y_0^{(j)} + h^\alpha \sum_{j=0}^{k-1} a_{jk} f(x_j, y_j)$$

is independent of μ (the index of the corrector iteration). Thus the total arithmetic complexity of the corrector part of the kth step (taking us from x_{k-1} to x_k) is $O(k)$ for the calculation of γ_k plus $O(m)$ for the m corrector steps, the sum of which (since m is constant) is asymptotically the same as the complexity in the case $m = 1$. We hence conclude that the computational cost of an Adams-Bashforth-Moulton predictor-corrector method with a large but fixed number of corrector iterations is asymptotically identical to the computational cost of the same method with only one application of the corrector. In summary, the use of a $\mathrm{P(EC)}^m\mathrm{E}$ approach simultaneously solves our problem that was induced by the large number of zeros in the initial value vector (3.1.24b) and increases the convergence order of the algorithm and yet it only introduces a negligible amount of additional computational complexity. We have plotted some graphs of approximate solutions obtained in this way in Fig. 3.5. The good quality of the numerical solutions is evident. In particular, the zoom of the subinterval $[0, 0.2]$ clearly shows that we have completely succeeded in removing the unpleasant constant behavior of the approximate solution near the initial point (except for the dotted curve which corresponds to only three corrector iterations where we can confirm the theoretical expectation that the subinterval with a constant approximate solution decreases in size by a half).

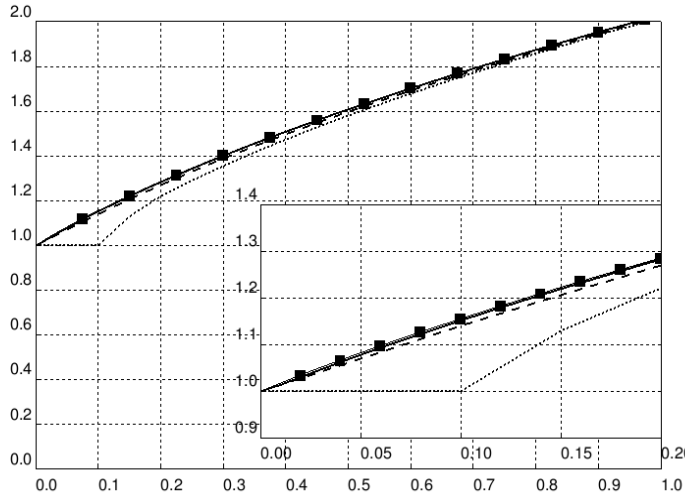

Fig. 3.5 Numerical solutions of Eq. (3.1.24) on $[0, 1]$ for uniform mesh with $h = 1/20$ and three corrector iterations (dotted line), $h = 1/20$ and nine corrector iterations (dashed line), $h = 1/100$ and nine corrector iterations (continuous line), and $h = 1/2000$ and nine corrector iterations (continuous line marked with star symbols; "exact" solution). The inset in the bottom right corner is an enlarged view of the range $[0, 0.2]$ of the complete graph.

For the purpose of comparison we have also built up Table 3.2 that displays the absolute errors of the various methods that we have tried at the point $x = 1$. The $\mathrm{P(EC)}^m\mathrm{E}$ method is seen to give the best relation between computational effort (essentially proportional to N^2 where N is the total number of grid points) and accuracy.

3.1.6 *Two classes of singular equations as application examples*

We conclude this section with a subsection dealing with two particularly difficult types of applications. The purpose of the first of these examples, taken from [188], is to demonstrate that certain algorithms can be successfully applied even in singular situations where the standard assumptions are not satisfied. Specifically we shall look at the equation

$$ {}^{\mathrm{C}}D^{1/2}y(x) = \ln y(x) + E\frac{q(x)}{y(x)}, \qquad y(0) = 0. \tag{3.1.25}$$

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Table 3.2. Absolute errors of numerical approximations for $y(1)$ from Eq. (3.1.23) using the Adams method with various step sizes and various numbers of corrector iterations.

grid	total no. of points	corrector iterations	abs. error at $x = 1$	visualization
uniform, $h = 1/20$	20	1	$5 \cdot 10^{-2}$	Fig. 3.4, Curve 1
uniform, $h = 1/200$	200	1	$4 \cdot 10^{-3}$	Fig. 3.4, Curve 2
uniform, $h = 1/2000$	2000	1	$2 \cdot 10^{-4}$	Fig. 3.4, Curve 3
$h = 1/500$ (5 steps), $h = 66/1000$ (15 steps)	20	1	$4 \cdot 10^{-2}$	Fig. 3.4, Curve 4
$h = 1/500$ (5 steps), $h = 99/10000$ (100 steps)	105	1	$5 \cdot 10^{-3}$	Fig. 3.4, Curve 5
uniform, $h = 1/20$	20	3	$1 \cdot 10^{-2}$	Fig. 3.5, dotted
uniform, $h = 1/20$	20	9	$9 \cdot 10^{-3}$	Fig. 3.5, dashed
uniform, $h = 1/100$	100	9	$2 \cdot 10^{-3}$	Fig. 3.5, contin.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

This equation has been used [297] to model the propagation of a flame in the context of a thermo-diffusive model with high activation energies using a gaseous mixture with simple chemistry $A \to B$. Specifically, $y(x)$ denotes the radius of the flame at the time instant x; the given function q models the energy input into the system via a point source.

The model described by Eq. (3.1.25) — which can be justified in a mathematically rigorous way [335] — has some rather natural important questions associated with it. Apart from the most obvious one for an (exact or approximate) solution for a specific choice of the parameters E and q, one is often strongly interested in the bifurcation behavior of the equation. Analytically, we know from [45] for a large class of functions q with $q(x) = 0$ for all $x > x_0$ with some x_0 that there exists a value $E_{\text{crit}}(q)$ such that

- if $E > E_{\text{crit}}(q)$ then y is defined on $[0, \infty)$ and $\lim_{x \to \infty} y(x) = \infty$,
- if $E = E_{\text{crit}}(q)$ then y is defined on $[0, \infty)$ and $\lim_{y \to \infty} y(x) = 1$,
- if $E < E_{\text{crit}}(q)$ then there exists some finite $x_{\max} > x_0$ such that y is defined on $[0, x_{\max}]$ and $\lim_{x \to x_{\max}} y(x) = 0$.

Stated explicitly, it says that the flame will quench in finite time if the energy added to the system is smaller than the critical level E_{crit}, and it

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

will burn persistently if the energy is above E_{crit}. For safety considerations it is therefore very important to find out the value of E_{crit} (or at least lower bounds for it) if one is interested in keeping the fire under control. On the other hand, sometimes one is interested in constructing a permanently burning flame, and then one needs to know E_{crit} (or at least upper bounds for it) in order to find an efficient process that uses as little energy as possible.

For the numerical solution of Eq. (3.1.25) one needs to take into account that the right-hand side of the differential equation is singular at the initial point. Thus, since the right-hand side is undefined at $(0, y(0))$, we cannot use a numerical method that needs to evaluate the right-hand side at this point. It is possible, however, to use the backward differentiation method of Subsection 2.2.2 or the fractional generalization of the backward Euler method according to Lubich's approach (see Subsection 2.1.3 and Section 2.4). Both approaches prove to be successful [188]; we concentrate on the former here.

A classical test case frequently considered in the literature (see the references cited in [188]) is

$$q(x) = \begin{cases} x^{0.3}(1-x) & \text{if } 0 \le x \le 1, \\ 0 & \text{else.} \end{cases}$$

For this function q we have attempted to numerically calculate the critical value $E_{\text{crit}}(q)$ with the backward differentiation method of Subsection 2.2.2. To this end we have solved the initial value problem (3.1.25) with various values of E and a very small step size $h = 1/3200$. The results are depicted in Fig. 3.6, and they indicate that the critical value is $E_{\text{crit}}(q) \approx 7.6655$.

The second type of examples that we shall discuss in this subsection arises in a completely different context. Specifically, these equations originate from problems in the context of the fractional calculus of variations as discussed in detail in Chapter 5. To be precise, we first deal with the system of equations

$$ {}^{\mathrm{C}}D^{\alpha}_{a+} q_2(t) = \lambda, \tag{3.1.26a}$$

$$ {}^{\mathrm{C}}D^{\alpha}_{a+} q_1(t) = p_1(t) - \lambda, \tag{3.1.26b}$$

$$ {}^{\mathrm{RL}}D^{\alpha}_{b-} p_1(t) = 0, \tag{3.1.26c}$$

$$ {}^{\mathrm{RL}}D^{\alpha}_{b-} p_2(t) = 0, \tag{3.1.26d}$$

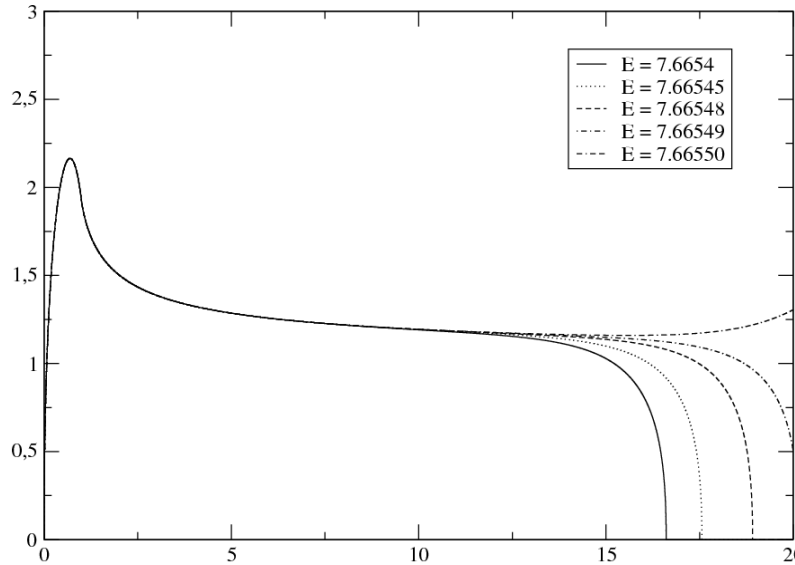

Fig. 3.6 Numerical solutions of Eq. (3.1.25) for $h = 1/3200$ and various values of E.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

that (in slightly more complicated notation whose use is not necessary here) forms the canonical equations of a fractional Euler-Lagrange problem, see Eq. (5.2.45) and its derivation. The interesting features of this system of equations are (a) that it contains both Caputo and Riemann-Liouville differential operators, (b) that it contains both left-sided and right-sided operators, and (c) that it is usually connected with initial conditions that lead to unbounded solutions.

Specifically, we shall consider the system (3.1.26) on the interval $[a, b] = [0, 1]$ with $\alpha = 0.85$, $\lambda = 3$ and the initial conditions

$$q_2(0) = 0.5, \tag{3.1.27a}$$

$$q_1(0) = 1.0, \tag{3.1.27b}$$

$$J^{\alpha}_{b-} p_1(b) = \Gamma(\alpha), \tag{3.1.27c}$$

$$J^{\alpha}_{b-} p_2(b) = \Gamma(\alpha). \tag{3.1.27d}$$

We first introduce the auxiliary functions $\tilde{p}_j(t) = p_j(t) - (b-t)^{\alpha-1}$ $(j = 1, 2)$, and notice that we can then replace equations (3.1.26c) and (3.1.26d) and

the corresponding initial conditions (3.1.27c) and (3.1.27d), respectively, by

$$
\begin{aligned}
{}^{\mathtt{C}}D^{\alpha}_{b-}\tilde{p}_1(t) &= 0, && \text{(3.1.28a)}\\
{}^{\mathtt{C}}D^{\alpha}_{b-}\tilde{p}_2(t) &= 0, && \text{(3.1.28b)}\\
\tilde{p}_1(b) &= 0, && \text{(3.1.28c)}\\
\tilde{p}_2(b) &= 0. && \text{(3.1.28d)}
\end{aligned}
$$

It is then clear that the system (3.1.28) has the solutions $\tilde{p}_1(t) = \tilde{p}_2(t) = 0$. This can be seen by an immediate analytical discussion, and the application of an arbitrary numerical method will lead to the same result. Thus, returning to our original unknowns, we derive that

$$p_1(t) = p_2(t) = (b-t)^{\alpha-1}.$$

We now have to solve the differential equations (3.1.26a) and (3.1.26b) together with the corresponding initial conditions (3.1.27a) and (3.1.27b) in order to determine the remaining unknowns q_1 and q_2. To this end we must take into consideration that the function p_1 that appears on the right-hand side of Eq. (3.1.26b) is singular at $t = b$, i.e. at the right end point of the interval of interest. Thus, following the concept known in the theory of numerical integration as "avoiding the singularity" [109, 472], we use a numerical method that does not evaluate the given function at the singular point $t = b$. An obvious choice here is the Adams-Bashforth (or forward Euler) method introduced in Subsection 2.3.2 above, i.e. the method obtained by only computing the predictor and using its value as the final approximation without employing the corrector step.

For the example with the parameters mentioned above, we have computed the numerical solution via this method with a step size of $h = 1/200$. Some numerical values for the solutions are given in Table 3.3, and plots of the approximate solutions are provided in Fig. 3.7.

The equation system

$$
\begin{aligned}
{}^{\mathtt{C}}D^{\alpha}_{a+}q_2(t) &= p_2(t), && \text{(3.1.29a)}\\
{}^{\mathtt{C}}D^{\alpha}_{a+}q_1(t) &= p_1(t), && \text{(3.1.29b)}\\
{}^{\mathtt{RL}}D^{\alpha}_{b-}p_1(t) &= 0, && \text{(3.1.29c)}\\
{}^{\mathtt{RL}}D^{\alpha}_{b-}p_2(t) &= 0 && \text{(3.1.29d)}
\end{aligned}
$$

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Table 3.3. Numerical values for the components q_1, q_2, p_1 and p_2 of the solution of the system (3.1.26) in combination with the initial conditions (3.1.27), with parameters $a = 0$, $b = 1$, $\lambda = 3$ and $\alpha = 0.85$, computed with the Adams-Bashforth method with step size $h = 1/200$.

t	$q_1(t)$	$q_2(t)$	$p_1(t)$	$p_2(t)$
0.00	1.000000	0.500000	1.000000	1.000000
0.10	0.702446	0.948135	1.015930	1.015930
0.25	0.356250	1.476467	1.044097	1.044097
0.50	−0.143113	2.260083	1.109569	1.109569
0.75	−0.577773	2.984338	1.231144	1.231144
0.95	−0.861913	3.537202	1.567309	1.567309
0.99	−0.896892	3.645564	1.995262	1.995262
1.00	−0.898149	3.672551	∞	∞

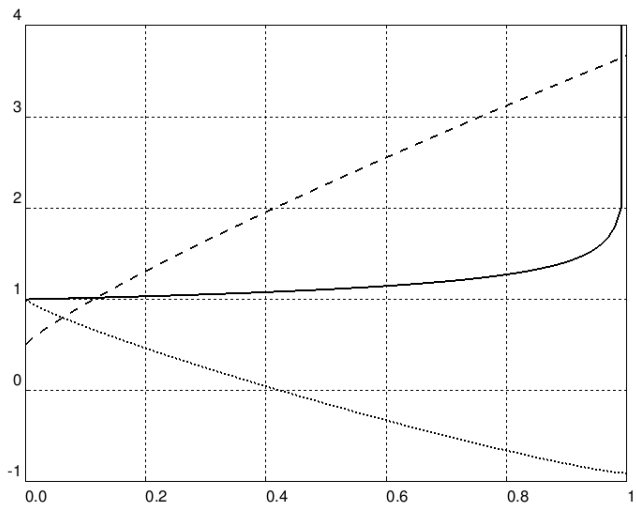

Fig. 3.7 Numerical values for the components q_1 (dotted), q_2 (dashed) and p_1 (continuous line) of the solution of the system (3.1.26) in combination with the initial conditions (3.1.27), with parameters $a = 0$, $b = 1$, $\lambda = 3$ and $\alpha = 0.85$, computed with the Adams-Bashforth method with step size $h = 1/200$. The solution component p_2 coincides with p_1.

with which we shall conclude this section has a similar background; it arises in the context of fractional canonical equations in the extended phase space, see Eq. (5.2.54). For our numerical experiments we use the parameter $\alpha = 0.8$ and look for a solution on the interval $[a, b] = [0, 10]$ with the

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Table 3.4. Numerical values for the components q_1, q_2, p_1 and p_2 of the solution of the system (3.1.29) in combination with the initial conditions (3.1.30), with parameters $a = 0$, $b = 10$ and $\alpha = 0.8$, computed with the Adams-Bashforth method with step size $h = 1/20$.

t	$q_1(t)$	$q_2(t)$	$p_1(t)$	$p_2(t)$
0.00	1.000000	−1.000000	1.000000	1.000000
0.50	1.391091	−0.608909	0.637463	0.637463
1.00	1.684942	−0.315058	0.644394	0.644394
2.50	2.453250	0.453250	0.668325	0.668325
5.00	3.631618	1.631618	0.724780	0.724780
7.00	4.595544	2.595544	0.802742	0.802742
9.00	5.712011	3.712011	1.000000	1.000000
9.50	6.069920	4.069920	1.148698	1.148698
9.95	6.522876	4.522876	1.820564	1.820564
10.00	6.610243	4.610243	∞	∞

initial conditions

$$q_2(0) = -1.0, \tag{3.1.30a}$$

$$q_1(0) = 1.0, \tag{3.1.30b}$$

$$J_{b-}^{\alpha} p_1(b) = \Gamma(\alpha), \tag{3.1.30c}$$

$$J_{b-}^{\alpha} p_2(b) = \Gamma(\alpha). \tag{3.1.30d}$$

Our numerical results have been obtained in a similar way as above. They are reported in Table 3.4 and Fig. 3.8.

3.2 Methods for Partial Differential Equations

In our discussion of numerical methods for partial fractional differential equations, we shall first concentrate on generalizations of classical problems like diffusion or wave equations obtained by replacing the integer-order time derivatives with a fractional differential operator. In particular, the first three subsections of this section will be devoted to numerical algorithms for problems of this type. Mainly we shall describe how classical approaches can be extended to the fractional setting. In Subsection 3.2.4 we shall then briefly deal with numerical methods for equations that are of fractional order with respect to the space variables.

Roughly speaking, we can classify the numerical approaches that we shall discuss into two categories, the fully discrete schemes and the semi-

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

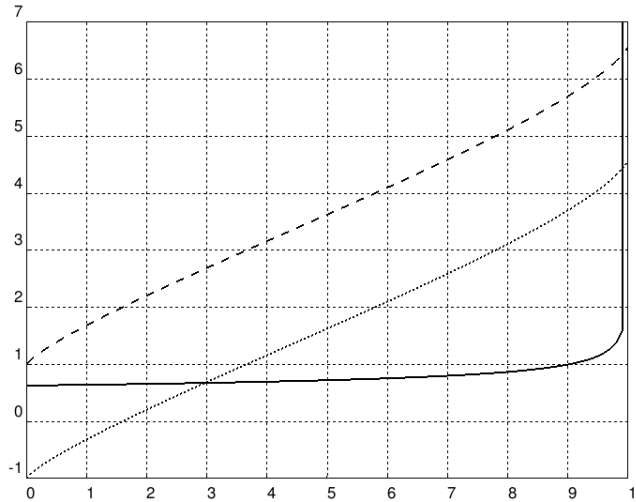

Fig. 3.8 Numerical values for the components q_1 (dashed), q_2 (dotted) and p_1 (continuous line) of the solution of the system (3.1.29) in combination with the initial conditions (3.1.30), with parameters $a = 0$, $b = 10$ and $\alpha = 0.8$, computed with the Adams-Bashforth method with step size $h = 1/20$. The solution component p_2 coincides with p_1.

discrete schemes. In the former approach, we immediately discretize with respect to all variables and directly obtain a procedure to compute the required approximate solutions. In the latter, we discretize with respect to some of the variables (usually, either with respect to the space variables only or with respect to the time variable only) and thus obtain a system of problems of a simpler structure as an intermediate result. These simpler problems can then sometimes be solved exactly to obtain the required approximate solutions; more frequently they will also be solved by numerical schemes tailored for this simpler class of problems.

3.2.1 *The method of lines*

The first method that we shall look at explicitly, the *vertical method of lines*, belongs to the class of semi-discrete methods. This is a well known concept in the theory of classical partial differential equations of integer order where it is frequently employed for parabolic problems, see, e.g., [514] or Section 7.2 of [319]. It is best illustrated by using a simple example. To this end, we shall look at the time-fractional diffusion equation in one space

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

dimension,

$$ {}^{\mathrm{C}}D_t^{\alpha} y(x,t) = K(x,t)\frac{\partial^2}{\partial x^2} y(x,t) + f(x,t) \tag{3.2.1a} $$

for $t \in [0,T]$ and $x \in [a,b]$, say, with $0 < \alpha < 1$ and a strictly positive function K, subject to the initial condition

$$ y(x,0) = y_0(x) \quad \text{for } x \in [a,b] \tag{3.2.1b} $$

and the boundary conditions

$$ y(a,t) = r_1(t) \quad \text{and} \quad y(b,t) = r_2(t) \qquad \text{for } t \in [0,T] \tag{3.2.1c} $$

with certain given functions K, f, y_0, r_1 and r_2. From the description that we shall now give it will be evident that the method can easily be modified to handle other related problems like, for example, equations in more than one space dimension, equations with other types of boundary conditions or time-fractional wave equations, i.e. equations like (3.2.1) but with $1 < \alpha < 2$ and then, of course, with a second initial condition.

The idea is, as already indicated, to discretize the differential equation with respect to only one variable at first. In the case of interest here, the vertical method of lines, this is the space variable x. There are a number of possibilities to implement such a discretization. One might choose, for example, a finite element approach or a finite difference method. For our example, we will go for the latter choice, but it is no problem for the vertical method of lines to use a different scheme (see, e.g., [225] for such an example). Thus, we select a uniform grid in the space variable that is given by the grid points $x_j = a + j(b-a)/N$, $j = 0, 1, \ldots, N$, with a certain value $N \in \mathbb{N}$. For these grid points we then discretize the differential operator with respect to the space variables via a standard central difference formula,

$$ \frac{\partial^2}{\partial x^2} y(x_j,t) \approx \frac{1}{h^2}\left(y(x_{j+1},t) - 2y(x_j,t) + y(x_{j-1},t)\right) $$

for $j = 1, 2, \ldots, N-1$ where we have introduced the parameter $h = (b-a)/N$. Notice that we do not need to consider $j = 0$ and $j = N$ as the solution $y(x_j,t)$ for these two values of j is already known from the

boundary condition (3.2.1c). Combining this discretization with the originally given differential equation (3.2.1a) leads to the system of equations

$$ {}^{\mathrm{C}}D_t^{\alpha} y(x_j,t) = \frac{K(x_j,t)}{h^2}\left(y(x_{j+1},t) - 2y(x_j,t) + y(x_{j-1},t)\right) + f(x_j,t) \tag{3.2.2} $$

for $j = 1, 2, \ldots, N-1$. Clearly, this is a system of $N-1$ ordinary fractional differential equations of order α that can be combined with the initial conditions

$$ y(x_j, 0) = y_0(x_j), \qquad j = 1, 2, \ldots, N-1, $$

and the boundary conditions

$$ y(x_0, t) = r_1(t) \quad \text{and} \quad y(x_N, t) = r_2(t) \qquad \text{for } t \in [0, T] $$

to obtain a system that can be uniquely solved. Depending on the nature of the functions f, K, r_1 and r_2, it may be possible to solve these equations analytically in closed form, thus producing the required approximation of the solution $y(x_j, t)$ to the fractional diffusion problem for the points of our grid with respect to the space variable and all t. In the more common case where an exact solution of the system of ordinary fractional differential equations cannot be found, one needs to apply a numerical scheme. In Chapter 2 we have presented a number of possible choices from which one may select here.

Remark 3.3. Using vector notation, we may reformulate the $N-1$ equations (3.2.2) in the form

$$ {}^{\mathrm{C}}D^{\alpha} Y(t) = \frac{1}{h^2} F(t, Y(t)) + \begin{pmatrix} f(x_1,t) \\ \vdots \\ f(x_{N-1},t) \end{pmatrix} $$

with a suitably chosen function F. It is then clear that the Lipschitz constant of the right-hand side with respect to the variable Y may be very large if h is small, i.e. if the space variable has been discretized on a fine grid. This feature may lead to a problematic behavior of some explicit solvers for ordinary fractional differential equations and may force us either to use very small step sizes in the time variable or to revert to implicit algorithms. A corresponding stability analysis has been provided in [592].

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

As the name *vertical* method of lines indicates, there is also a *horizontal method of lines*, also known as *Rothe's method* (see Section 7.2 of [319]). In this approach, one starts by discretizing with respect to the time variable. We can also do this here. Let us use a fractional linear multistep method as described in Section 2.4 for this purpose. Thus, using the time variable discretization $t_j = j\tau$ with $\tau = T/M$ for some $M \in \mathbb{N}$ and invoking Theorem 2.13 this means that our fractional initial-boundary value problem (3.2.1) is approximated by the system of elliptic differential equations

$$y(x,t_n) = y_0(x) + \tau^\alpha \sum_{j=0}^{n} \omega_{n-j} \left[K(x,t_j) \frac{\partial^2}{\partial x^2} y(x,t_j) + f(x,t_j) \right] + \tau^\alpha \sum_{j=0}^{s} w_{n,j} \left[K(x,t_j) \frac{\partial^2}{\partial x^2} y(x,t_j) + f(x,t_j) \right] \tag{3.2.3a}$$

for $n = 1, 2, \ldots, M$ that does not contain any differential operators of non-integer order. This system needs to be augmented by the boundary conditions

$$y(a,t_n) = r_1(t_n) \quad \text{and} \quad y(b,t_n) = r_2(t_n) \tag{3.2.3b}$$

for all $n = 1, 2, \ldots, M$ in order to be uniquely solvable, and its solution can once again be computed exactly if the given data permit or numerically otherwise.

3.2.2 *Backward difference formulas for time-fractional parabolic and hyperbolic equations*

For non-fractional partial differential equations of parabolic type such as, e.g., equations of the form (3.2.1) with $\alpha = 1$, the finite difference methods are a standard choice (see, e.g., Section 9.2 of [330]). It is therefore not surprising that the fractional generalizations of these methods are very popular tools when it comes to the numerical solution of fractional differential equations of the form (3.2.1), at least with $0 < \alpha < 1$. In this context it is particularly helpful to know that many well known properties of these algorithms can be carried over from the integer-order to the fractional case. Thus we shall now look at this class of method in more detail.

Specifically, a finite difference method is characterized by the fact that it discretizes the given differential equation by using finite differences to

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

replace the derivative operators. For derivatives of second order like those appearing on the right-hand side of Eq. (3.2.1a), it is natural to choose a central difference of second order for this purpose. If a step size of h is used, then this yields an $O(h^2)$ approximation with respect to the space variable. There is no problem in transferring this idea from the case $\alpha = 1$ to the case $0 < \alpha < 1$. The first order derivative with respect to time that appears in a classical parabolic equation can be discretized in a number of ways. Denoting the step size by k, the most important versions are the forward difference

$$z'(t) \approx \frac{z(t+k) - z(t)}{k}$$

and the backward difference

$$z'(t) \approx \frac{z(t) - z(t-k)}{k}$$

that lead to numerical methods of the forward Euler form

$$\frac{z_\ell - z_{\ell-1}}{k} = f(t_{\ell-1}, z_{\ell-1})$$

and the backward Euler form

$$\frac{z_\ell - z_{\ell-1}}{k} = f(t_\ell, z_\ell),$$

respectively, for the first-order equation $z'(t) = f(t, z(t))$ with respect to the time variable. It is well known that both these formulas have a convergence order of $O(k)$. But of course there is not only this similarity; there are also major differences between these approaches. Evidently, the forward Euler method is explicit, i.e. we can directly compute z_ℓ from $z_{\ell-1}$. The backward Euler method, on the other hand, is implicit because the unknown value z_ℓ also appears inside the given function f on the right-hand side. This means that typically the computational effort required for a backward Euler method is significantly larger than the effort for a forward Euler method. However, this advantage for the forward scheme comes with the price of stability problems. Specifically the forward method will become unstable if the step sizes are too large, whereas the backward method is unconditionally stable and therefore does not exhibit such problems. This is a very important aspect when one uses such methods for the approximation

of the time derivative in our parabolic partial differential equation. As a matter of fact it implies that a finite difference method with a forward Euler method for the time integration can only converge if the step sizes in space and time are related by the inequality

$$k \leq ch^2$$

where c is a constant that depends on the coefficient function K of the differential equation (3.2.1a). Thus, if a small step size h is used for the space discretization then a very much smaller step size must be used for the time discretization. No such condition is required for the backward Euler scheme.

A sort of a compromise between these two methods is the Crank-Nicolson approximation

$$\frac{z_\ell - z_{\ell-1}}{k} = \frac{1}{2}(f(t_{\ell-1}, z_{\ell-1}) + f(t_\ell, z_\ell))$$

that is formally obtained by taking the arithmetic mean of the two equations defining the forward Euler and the backward Euler method, respectively. Like the backward Euler method it is implicit, and it shares some, but not all, of its favorable stability properties (see Chapter 7 of [319]). The advantage of this method is that it yields a better convergence order, namely $O(k^2)$, than the two methods described above.

When it comes to extending these concepts to time-fractional differential equations, it is rather simple to generalize the forward Euler method and the backward Euler method. All that needs to be done is that, as indicated in Section 2.8, one must replace the first-order (forward or backward) difference by an appropriately chosen fractional counterpart. We had seen in Chapter 2 that there are a number of possible fractional versions of the finite difference; any of these can be used. It is known that the stability properties of the resulting fractional methods are essentially identical to those of the underlying classical (integer-order) methods. Specifically, a fractional forward difference scheme will only be stable under the restriction

$$k^\alpha \leq ch^2$$

where once again the constant c depends on the given function K. Since $0 < \alpha < 1$, this restriction is even more severe than its integer-order counterpart

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

($\alpha = 1$), and so we refrain from looking at fractional forward difference methods for time-fractional diffusion equations.

For the Crank-Nicolson method we also run into certain difficulties. To be precise, the background behind the Crank-Nicolson method is that, under standard smoothness assumptions, the forward and backward Euler methods have an error expansion at the point t_ℓ that has the form

$$z(t_\ell) - z_\ell = c_1 k + c_2 k^2 + \text{higher order terms}$$

where the constants c_1 and c_2 depend on the given differential equation and on the chosen method. However, for the constant c_1 one can show that this constant has the same absolute value for both algorithms, but different signs. Thus, the construction of the Crank-Nicolson scheme via an arithmetic mean of these two formulas implies that the coefficient of k in the error expansion of the Crank-Nicolson method vanishes, and the dominating term is of the order k^2 as mentioned above. For the fractional analog of the Euler schemes we have two problems in this connection. Firstly, no information about the relation of the constants c_1 for the two methods is available. Thus, even if it may be possible to eliminate the leading term of the error by a suitably weighted mean of forward and backward method, it is not known which values one should choose for the weights. And moreover, even if one would be able to find the correct values for these weights then one would typically still have to deal with the situation that the second term in the asymptotic expansion, i.e. the first term that would not be eliminated, is only going to be of the order $k^{1+\alpha}$ and not k^2, so the gain in accuracy would be less substantial than in the non-fractional case. An alternative path to a fractional version of the Crank-Nicolson scheme with the same order of convergence as in the classical case has been provided by McLean and Mustapha [386] whose approach however requires the use of a special non-uniform discretization for the time variable that may not always be desired. Thus we shall not pursue the Crank-Nicolson idea in the fractional setting any further either and concentrate only on the backward Euler scheme.

For the backward Euler method we may proceed in a similar way. Of course we need to use a backward difference instead of the forward difference with respect to time now. We may choose the formula from Eqs. (2.1.10) and (2.1.11) which then yields, after a suitable rearrangement of terms, the

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

overall algorithm

$$\begin{aligned} y_{0m} &= r_1(t_m), \\ y_{jm} &= \Gamma(2-\alpha)\frac{k^\alpha}{h^2}K(x_j,t_m)(y_{j+1,m} - 2y_{jm} + y_{j-1.m}) \\ &\quad + (1-\alpha)m^{-\alpha}y_0(x_j) + \Gamma(2-\alpha)k^\alpha f(x_j,t_m) \qquad (3.2.4) \\ &\quad - \sum_{\mu=0}^{m-1} B_{\mu m}y_{j\mu} \qquad\qquad (1 \le j < N), \\ y_{Nm} &= r_2(t_m) \end{aligned}$$

for $m = 0, 1, 2, \ldots, M$. In this algorithm, N is the discretization parameter for the mesh with respect to the space variables, $h = (b-a)/N$ is the corresponding step size, M is the discretization parameter for the time mesh and $k = T/M$ is the time step where T is the upper bound of the time interval on which we are looking for a solution. The grid points are then $t_\mu = \mu k$ and $x_j = a + jh$, and the weights are given by

$$B_{\mu m} = \begin{cases} (m-1)^{1-\alpha} - (m-1+\alpha)m^{-\alpha} & \text{for } \mu = 0, \\ (m-\mu+1)^{1-\alpha} - 2(m-\mu)^{1-\alpha} + (m-\mu-1)^{1-\alpha} & \text{else} \end{cases}$$

(which follows from Eqs. (2.1.10) and (2.1.11)). Finally the functions y_0, r_1 and r_2 are those given in the initial condition (3.2.1b) and the boundary conditions (3.2.1c), respectively. The values y_{jm} that we compute in this way can then be interpreted as approximations to the exact solution values $y(x_j, t_m)$. Clearly we have an implicit method because the values y_{jm}, $j = 0, 1, \ldots, N$, i.e. the unknowns of the system at the mth time step, appear not only on the left-hand side of the equation system (3.2.4) but also on the right-hand side. It is convenient to rewrite the system in matrix-vector form, thus obtaining

$$\begin{pmatrix} y_{0m} \\ y_{1m} \\ y_{2m} \\ \vdots \\ y_{N-1,m} \\ y_{Nm} \end{pmatrix} = \Gamma(2-\alpha)\frac{k^\alpha}{h^2}G_m \begin{pmatrix} y_{0m} \\ y_{1m} \\ y_{2m} \\ \vdots \\ y_{N-1,m} \\ y_{Nm} \end{pmatrix} + H_m \qquad (3.2.5)$$

with the vector

$$H_m = \begin{pmatrix} r_1(t_m) \\ (1-\alpha)m^{-\alpha}y_0(x_1) + \Gamma(2-\alpha)k^\alpha f(x_1,t_m) - \sum_{\mu=0}^{m-1} B_{\mu m} y_{1\mu} \\ (1-\alpha)m^{-\alpha}y_0(x_2) + \Gamma(2-\alpha)k^\alpha f(x_2,t_m) - \sum_{\mu=0}^{m-1} B_{\mu m} y_{2\mu} \\ \vdots \\ (1-\alpha)m^{-\alpha}y_0(x_{N-1}) + \Gamma(2-\alpha)k^\alpha f(x_{N-1},t_m) - \sum_{\mu=0}^{m-1} B_{\mu m} y_{N-1,\mu} \\ r_2(t_m) \end{pmatrix}$$

and the matrix $G_m = (g_{\rho\sigma}^{(m)})_{\rho,\sigma=0}^{N}$ with

$$g_{\rho\sigma}^{(m)} = \begin{cases} -2K(x_\rho, t_m) & \text{for } 1 \le \rho = \sigma \le N-1, \\ K(x_\rho, t_m) & \text{for } 1 \le \rho \le N-1 \text{ and } |\rho - \sigma| = 1, \\ 0 & \text{else} \end{cases}$$

(note that, for the sake of notational convenience, we have started the row and column indices at 0, not at 1). Hence our solution vector $Y_m = (y_{0m}, y_{1m}, \dots y_{Nm})^{\mathrm{T}}$ at the mth time step can be obtained as the solution of the linear system

$$\left(I - \Gamma(2-\alpha)\frac{k^\alpha}{h^2} G_m\right) Y_m = H_m$$

where I is the unit matrix.

We now demonstrate the performance of this algorithm by means of an example problem. To this end we look at the initial-boundary value problem

$${}^{\mathrm{C}}D_t^{0.7} y(x,t) = \left(2 + \frac{1 + x(2\pi - x)}{5 + 5t}\right) \frac{\partial^2}{\partial x^2} y(x,t) + \cos x \cos t \qquad (3.2.6a)$$

for $0 \le t \le 10$ and $0 \le x \le 2\pi$ with initial condition

$$y(x,0) = \cos\frac{x}{2} \qquad (3.2.6b)$$

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

and boundary conditions

$$y(0,t) = \exp(-t) \quad \text{and} \quad y(2\pi,t) = -\exp(-t). \tag{3.2.6c}$$

The resulting approximate solutions for various choices of the discretization parameters are shown in Figs. 3.9 and 3.10. It is evident from these figures that the algorithm indeed behaves in a stable manner in the sense that an arbitrary combination of step sizes with respect to time and with respect to space yields useful values.

For classical hyperbolic equations, i.e. equations of the form (3.2.1) with $\alpha = 2$ (and two initial conditions in Eq. (3.2.1b) instead of just one), two principal versions of finite difference approaches are frequently used. The first of these approaches is formally identical to the approach presented above for parabolic problems, where we only have to set $\alpha = 2$ now. Of course one here needs to use second order differences with respect to time;

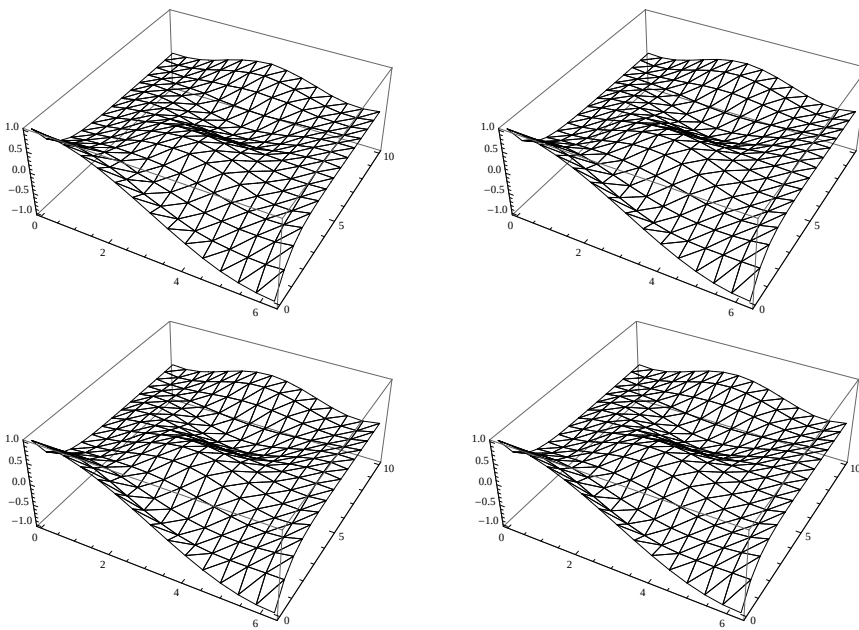

Fig. 3.9 Numerical solutions of Eq. (3.2.6) on $[0, 2\pi] \times [0, 10]$ for backward Euler method with space mesh size h and time mesh size k, where $h = \pi/5$, $k = 1$ (top left), $h = \pi/10$, $k = 0.5$ (top right), $h = \pi/50$, $k = 0.5$ (bottom left), $h = \pi/500$, $k = 0.5$ (bottom right).

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

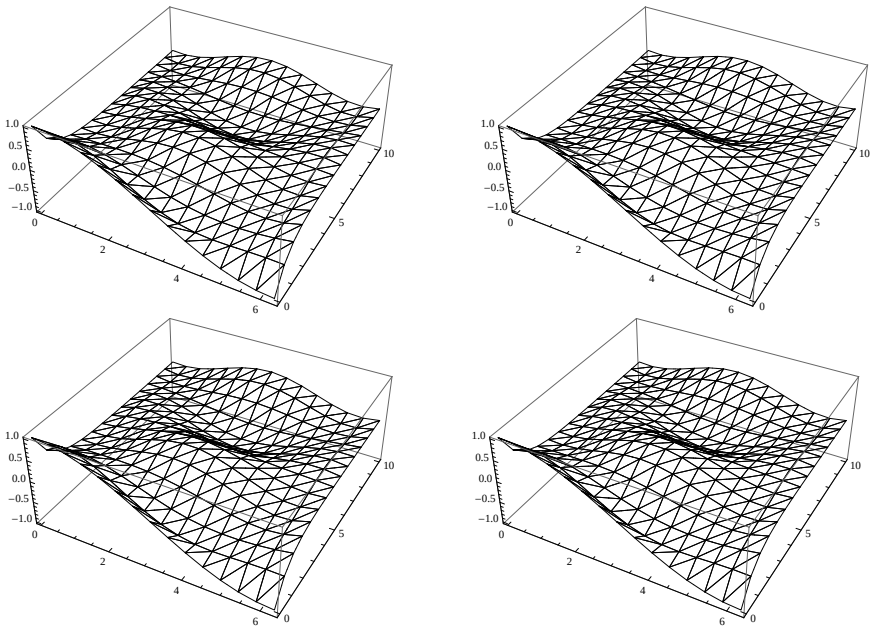

Fig. 3.10 Numerical solutions of Eq. (3.2.6) on $[0, 2\pi] \times [0, 10]$ for backward Euler method with space mesh size h and time mesh size k, where $h = \pi/50$, $k = 0.1$ (top left), $h = \pi/500$, $k = 0.1$ (top right), $h = \pi/10$, $k = 0.01$ (bottom left), $h = \pi/50$, $k = 0.01$ (bottom right).

centered differences are the usual method of choice. It is then possible to extend the previous theory and to prove convergence and stability under similar conditions as above. In particular, for appropriately chosen differences with respect to the space variable the stability condition then reads $k^2 \leq c'h^2$ or, more simply,

$$k \leq ch$$

where once again the constant c', and hence also c, depends on the coefficient function K, and once again no such restriction is required in the backward Euler method. For details we refer to [35] or Chapter 14 of [286] where one may also find a description of the very rich class of phenomena that may arise in connection with hyperbolic equations (in particular in the nonlinear case). This concept can be directly generalized to the fractional setting $1 < \alpha < 2$, thus immediately extending the results for the case $0 < \alpha < 1$.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

The second common possibility to approach the classical hyperbolic equation is to introduce a new two-dimensional function Y via the definition

$$Y = \begin{pmatrix} \partial y/\partial t \\ \partial y/\partial x \end{pmatrix}$$

Using this new variable, the given equation (3.2.1a) with $\alpha = 2$ can be rewritten in the form of a two-dimensional system of partial differential equations of first order,

$$\begin{aligned} \frac{\partial}{\partial t} Y_1(x,t) \qquad\quad - \frac{\partial}{\partial x} Y_2(x,t) &= 0, \\ \frac{\partial}{\partial t} Y_2(x,t) - K(x,t) \frac{\partial}{\partial x} Y_1(x,t) &= f(x,t), \end{aligned}$$

that we can write in matrix-vector form as

$$\frac{\partial}{\partial t} Y(x,t) + A(x,t) \frac{\partial}{\partial x} Y(x,t) = \begin{pmatrix} 0 \\ f(x,t) \end{pmatrix}$$

with

$$A = \begin{pmatrix} 0 & -1 \\ -K(x,t) & 0 \end{pmatrix}.$$

This approach can be generalized to the fractional situation too. Indeed, using the same new variable Y as above, we come to the system

$$\begin{aligned} \frac{\partial}{\partial t} Y_1(x,t) \qquad\quad - \frac{\partial}{\partial x} Y_2(x,t) &= 0, \\ {}^{\mathrm{C}}D_t^{\alpha-1} Y_2(x,t) - K(x,t) \frac{\partial}{\partial x} Y_1(x,t) &= f(x,t), \end{aligned}$$

that evidently is a system of two differential equations of different orders. As we had seen above, it is in principle possible to deal with such a system numerically. However, as demonstrated in Section 14.4 of [286] (see also Chapter 12 of [330]), the special case $\alpha = 2$ already needs to be handled very carefully and tediously in order to obtain practically useful results, and the implications of the additional complications introduced by the fractional generalization seem to be unclear at the moment. We therefore do not follow this path any further here.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

3.2.3 *Other methods*

It seems indeed that methods based on finite differences with respect to both the space and the time variables are the most popular choices in concrete applications. However there are also some other possibilities. A brief survey of finite difference and other methods for fractional partial differential equations has been given in Section 1 of [391]; see also the references cited therein. We shall now briefly describe a few potential alternatives to the finite difference concept.

We first deal with *collocation methods*. The essence of these methods is that one looks for an approximate solution in a finite-dimensional linear space. Given a basis of this space, the coefficients are then computed in such a way that the given differential equation is exactly fulfilled at a certain set of preassigned points. Of course the number of elements of this set must coincide with the dimension of the space of functions, and frequently additional conditions on the location of the points are required in order to obtain unique solutions with a good approximation quality. For the sake of illustration, we consider the problem (3.2.1) with $K \equiv 1$, $f \equiv 0$ and $r_1 \equiv r_2 \equiv 0$. Note, in particular, that the latter condition is not a restriction of generality as it can always be achieved by a suitable transformation.

In the construction of an exemplary special case of the method we follow Section 14.1 of [330] and discretize with respect to the space variable, i.e. we work according to the spirit of a vertical method of lines. Introducing a discretization parameter $N \in \mathbb{N}$, we set $h = (b-a)/N$ and use a piecewise polynomial space

$$S_N = \{v \in C^1[a,b] : v(0) = v(1) = 0, v_{|_{[a+jh,a+(j+1)h]}} \in \mathcal{P}_{r-1} \text{ for all } j\}$$

with some $r \geq 4$, where $\mathcal{P}_{r-1}$ is the set of all polynomials of degree at most $r-1$. Now in each subinterval $[a+jh, a+(j+1)h]$ we define the collocations points $x_{j,1}, x_{j,2}, \ldots, x_{j,r-2}$ by $x_{j,\nu} = a+jh+h(\eta_\nu+1)/2$ where η_ν $(\nu = 1, 2, \ldots, r-2)$ is the νth zero of the Legendre polynomial P_{r-2}. Our problem then is to find a solution of the system of differential equations

$${}^{\mathrm{C}}D_t^\alpha y_N(x_{j,\nu}, t) = \frac{\partial^2}{\partial x^2} y_N(x_{j,\nu}, t), \qquad 1 \leq j \leq M, \quad 1 \leq \nu \leq r-2, \tag{3.2.7}$$

subject to the initial condition $y_N(\cdot, 0) = y_{0,N}$ where $y_{0,N} \in S_N$ is a suitable approximation of the initial datum y_0 given in Eq. (3.2.1b).

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

In comparison to other methods, collocation methods frequently require more restrictive conditions on the regularity of the functions involved in order to converge sufficiently fast.

Our next object of study in this subsection are the *spectral methods* (see, e.g., Section 14.2 of [330] or [343]). Like collocation methods, spectral approximation is often employed with respect to the space variables in a partial differential equation. One possible way of describing the basic idea is once again to use a vertical method of lines approach. For the description it is again convenient to assume homogeneous boundary conditions $r_1 \equiv r_2 \equiv 0$ to be present in the original partial differential equation (3.2.1c) and hence also in the boundary conditions (3.2.3b). As mentioned above, this is not a loss of generality. Moreover, we once again assume $K \equiv 1$ as in our description of the collocation method. However, we do not impose any restriction on the inhomogeneity f of the given partial differential equation (3.2.1). Then we let $\{\phi_j : j = 1, 2, \ldots\}$ be a set of linearly independent and sufficiently smooth functions that span $L^2(a, b)$ and define S_N to be the linear span of $\{\phi_j : 1 \le j \le N\}$.

As in the collocation approach, we attempt to find an approximate solution in the set S_N (that now has a different meaning than above). To this end we invoke a Galerkin idea and require, instead of Eq. (3.2.7), that

$$({}^{\mathrm{C}}D_t^{\alpha} y_N, z) = \left(\frac{\partial^2}{\partial x^2} y_N, z\right) + (f, z) \qquad \text{for all } z \in S_N \text{ and } t > 0 \quad (3.2.8)$$

where $(\cdot, \cdot)$ denotes the standard L^2 inner product on (a, b). Once again the resulting differential equations with respect to t are equipped with the initial condition $y_N(\cdot, 0) = y_{0,N}$ where $y_{0,N} \in S_N$ is a suitable approximation of the initial datum y_0 given in Eq. (3.2.1b).

If Π_N is the orthogonal projection from $L^2(a, b)$ to S_N then we may rewrite Eq. (3.2.7) as

$${}^{\mathrm{C}}D_t^{\alpha} y_N = A_N y_N + \Pi_N f$$

where $A_N = \Pi_N L \Pi_N$ and $L = \partial^2/\partial x^2$. With the basis representation $y_N(x, t) = \sum_{j=1}^{N} a_j(t)\phi_j(x)$ this amounts to a matrix-vector equation of the form

$$B\ {}^{\mathrm{C}}D_t^{\alpha} a(t) + Aa(t) = b(t), \qquad t > 0,$$

with the elements of the $(N \times N)$ matrices A and B being given by $(L\phi_i, \phi_j)$ and (ϕ_i, ϕ_j), respectively. In particular, the matrix B becomes the unit matrix if the basis function ϕ_j are orthonormal. A number of methods have been proposed in this book for solving this system of fractional differential equations with respect to t; any of these methods can be used to compute the function a and hence the final solution of our given partial fractional differential equation.

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

3.2.4 *Methods for equations with space-fractional operators*

It is mathematically no problem to replace the derivatives with respect to the space variables in a partial differential equation by their fractional-order generalizations. However, as indicated in the previous chapter, the resulting models have some features that are inconsistent with certain fundamental physical principles (see Section 2.3.2 of [281]). For certain other physical models, the corresponding authors argue in favor of space-fractional equations also based on ideas taken from physical principles [86]. Therefore the application of such space-fractional equations (or, even more generally, partial differential equations with fractional derivatives with respect to both space and time) needs to be done very carefully in order to obtain valid models. Nevertheless there may be situations where these models are appropriate, and therefore we shall devote a short section to their numerical solution too.

In this description, we discuss the fully general case of a partial differential equation that is fractional with respect to space and time. The case of an equation that has fractional derivatives only with respect to the space variable but not with respect to the time variable is contained *a fortiori* as a special case in this general setting.

The model problem that we shall numerically solve here has the convection-diffusion form

$$ {}^{\mathrm{C}}D_t^{\alpha} y(x,t) = -b(x)\frac{\partial}{\partial x} y(x,t) + a(x)\, {}^{\mathrm{RL}}D_x^{\beta} y(x,t) + q(x,t) \tag{3.2.9a} $$

for $x \in [0, L]$ and $t \in [0, T]$ with $0 < \alpha \leq 1$, $1 < \beta \leq 2$, continuous functions $a > 0$, $b > 0$ and q, the initial condition

$$ y(x,0) = f(x) \tag{3.2.9b} $$

for all $x \in [0, L]$ and the homogeneous boundary conditions

$$y(0,t) = y(L,t) = 0 \tag{3.2.9c}$$

for all $t \in [0, T]$. Obviously, the case of inhomogeneous boundary conditions can be reduced to this problem via the classical techniques.

The numerical method for the solution of the problem (3.2.9) that we shall propose here is due to Zhang [598]. It is based on approximating the three differential operators appearing in the differential equation (3.2.9a) by appropriate finite differences. Specifically, we introduce equispaced grids with respect to both variables, i.e. we define a time step $\tau > 0$ such that $T = N\tau$ with some $n \in \mathbb{N}$ and a space step $h = L/M$ with some $M \in \mathbb{N}$ and denote the grid points by $x_i = ih$ $(i = 0, 1, \ldots, M)$ and $t_j = j\tau$ $(j = 0, 1, \ldots, N)$. Then, the Caputo derivative with respect to the time variable is approximated by

$$\begin{aligned} {}^{\mathtt{C}}D_t^{\alpha} y(x_i, t_{k+1}) & \\ \approx \frac{\tau^{-\alpha}}{\Gamma(2-\alpha)} \Big(& y(x_i, t_{k+1}) \\ & + \sum_{j=0}^{k} y(x_i, t_{k-j}) \left[(j+2)^{1-\alpha} - 2(j+1)^{1-\alpha} + j^{1-\alpha} \right] \\ & - y(x_i, t_0) \left[(k+1)^{1-\alpha} - k^{1-\alpha} \right] \Big). \end{aligned} \tag{3.2.10}$$

This discretization can essentially be obtained by representing the Caputo differential operators in terms of their Riemann-Liouville counterparts using the well known relation (2.1.5) in combination with an approximation of the Riemann-Liouville derivatives via the formula developed in Eqs. (2.1.10) and (2.1.11). For the first-order space derivative we use the classical first difference

$$\frac{\partial}{\partial x} y(x_i, t_{k+1}) \approx h^{-1} \left(y(x_i, t_{k+1}) - y(x_{i-1}, t_{k+1}) \right). \tag{3.2.11}$$

And finally the fractional derivative with respect to the space variable x is approximated by the shifted Grünwald formula

$${}^{\mathtt{RL}}D_x^{\beta} y(x_i, t_{k+1}) \approx h^{-\beta} \sum_{j=0}^{i+1} g_j y(x_{i-j+1}, t_{k+1}) \tag{3.2.12a}$$

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.

with

$$g_j = \frac{(-1)^j}{j!} \prod_{\mu=1}^{j} (\beta + 1 - \mu). \tag{3.2.12b}$$

Approximations for $y(x_0, t)$ and $y(x_M, t)$ are not required because these values are already known exactly from the boundary conditions (3.2.9c). Thus, applying the discretizations (3.2.10), (3.2.11) and (3.2.12) to our problem (3.2.9), we construct a fully discretized scheme for the remaining unknowns $Y_k = (y_{1,k}, y_{2,k}, \dots y_{M-1,k})^{\mathrm{T}}$ $(k = 1, 2, \dots, N)$, i.e. the vectors of approximations for the exact solutions $(y(x_1, t_k), \dots, y(x_{M-1}, t_k))^{\mathrm{T}}$. As described in detail in [598], this system can be written in the form

$$Y_0 = (f(x_1), f(x_2), \dots, f(x_{M-1}))^{\mathrm{T}}, \tag{3.2.13a}$$

$$ZY_{k+1} = c_k Y_k - \sum_{\mu=1}^{k} d_\mu Y_{k+1-\mu} + \tau^\alpha Q_{k+1} \quad (0 \le k < N), \tag{3.2.13b}$$

where

$$Q_\mu = \Gamma(2-\alpha)(q(x_1, t_\mu), \dots, q(x_{M-1}, t_\mu))^{\mathrm{T}},$$
$$d_\mu = (\mu+1)^{1-\alpha} - 2\mu^{1-\alpha} + (\mu-1)^{1-\alpha}$$

and

$$c_\mu = (\mu+1)^{1-\alpha} - \mu^{1-\alpha}.$$

Moreover, the matrix $Z = (z_{ij})_{i,j=1}^{M-1}$ in Eq. (3.2.13b) has entries of the form

$$z_{ij} = \begin{cases} 0 & \text{for } i < j-1, \\ -a(x_i)\tau^\alpha h^{-\beta}\Gamma(2-\alpha) & \text{for } i = j-1, \\ 1 + (b(x_i)h^{-1} - a(x_i)h^{-\beta}g_1)\tau^\alpha\Gamma(2-\alpha) & \text{for } i = j, \\ -(b(x_i)h^{-1} + a(x_i)h^{-\beta}g_2)\tau^\alpha\Gamma(2-\alpha) & \text{for } i = j+1, \\ -a(x_i)\tau^\alpha h^{-\beta}g_{i-j+1}\Gamma(2-\alpha) & \text{for } i > j+1 \end{cases}$$

which completes the description of the method.

The fact that this algorithm has useful stability and convergence properties has been demonstrated in [598].

Fractional Calculus Downloaded from www.worldscientific.com
by UFA STATE AVIATION TECHNICAL UNIVERSITY on 09/23/13. For personal use only.