INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 4, Number 1, Pages 63–73

SOLUTION OF A SINGULAR INTEGRAL EQUATION BY A SPLIT-INTERVAL METHOD

TERESA DIOGO, NEVILLE J. FORD, PEDRO M. LIMA, AND SOPHY M. THOMAS

Abstract. In this paper we give details of a new numerical method for the solution of a singular integral equation of Volterra type that has an infinite class of solutions. The split-interval method we have adopted utilises a simple robust numerical method over an initial time interval (which includes the singularity) combined with extrapolation. We describe the method and give details of its order of convergence together with examples that show its effectiveness.

Key Words. Numerical methods, extrapolation, singular integral equation, Volterra equation.

1. Introduction

This paper is concerned with the development of efficient numerical methods for the solution of Volterra integral equations which have a singularity in the kernel that leads to there being more than one solution. Therefore we are not here concerned with equations of Abel type where, despite a singularity in the kernel there is nevertheless a unique solution. More details of numerical methods for Abel type equations may be found, for example, in [1], [2]. As prototype for the type of problems we are interested in we shall investigate the equation

$$u(t) = g(t) + \int_0^t \frac{s^{\mu-1}}{t^\mu} u(s)\, ds \tag{1}$$

which has just the kind of properties that present real challenges to numerical analysts.

The precise analysis of equation (1) depends critically on the value of μ. For the case when $\mu < 1$, it has been shown in [8] that there is an infinite set of solutions, one of which is continuously differentiable whenever g is continuously differentiable, and all the other solutions have infinite gradient at the origin. This infinity of solutions arises because for $\mu < 1$ the singularity in the integrand persists at $s = 0$ for all $t > 0$. Of course when $\mu > 1$ there is still a singularity in the integrand for $t = 0$ but this does not persist when $t > 0$ and it turns out that the equation then has a unique solution.

Thus we shall concentrate on the case $0 < \mu < 1$. We are aided in our work by the fact that an explicit formula for the set of solutions has been derived. The solution formula is

$$u(t) = ct^{1-\mu} + g(t) + \frac{g(0)}{\mu - 1} + t^{1-\mu} \int_0^t s^{\mu-2}(g(t) - g(0))\, ds \tag{2}$$

Received by the editors January 25, 2005 and revised January 10, 2006.
2000 *Mathematics Subject Classification.* 65R20.
Teresa Diogo and Pedro Lima are pleased to acknowledge the support from FCT, project POCTI/MAT/45700/2002.

where $g \in C^1$, and c is an arbitrary constant , which determines the particular trajectory to be followed. When $c = 0$, we have the smooth solution (see [8]).

All this makes our choice of prototype equation ideal. The underlying equation has the complicated properties we wish to investigate in our numerical schemes but we have the luxury of an exact solution with which we can make comparisons of our numerical approximations.

1.1. Previous numerical approaches. The derivation and theoretical background of equation (1) are given in [9] and the references cited therein. In this context, the first use of product integration formulae (see [12]) was applied to the related equation

$$y(t) + \int_0^t p(t,s)y(s)ds = f(t) \tag{3}$$

$$p(t,s) := \frac{1}{\sqrt{\pi}} \frac{1}{\sqrt{\log(t/s)}} \left(\frac{s}{t}\right)^{\mu} \frac{1}{s}\, ds.$$

The product Euler and product trapezoidal methods were applied to equation (3) for the case when $\mu > 1$, with the restriction on the forcing function that $f(t) \in C^m$, where $m = 1$ for the product Euler and $m = 2$ for the product trapezoidal scheme, with convergence orders 1 and 2 respectively.

Equation (3) may be transformed into (1), where the forcing functions are related by the formula

$$g(t) = -\int_0^t p(t,s)f(s)\,ds + f(t).$$

This process is used in [7], and Hermite-type collocation is applied to the simplified equation, again for $\mu > 1$, and the convergence is of order 4, but subject to the more stringent restriction that $g \in C^5$.

Extrapolation procedures were introduced in conjunction with the product Euler scheme in [10] for the solution of (1), still with $\mu > 1$, but now allowing the g to have a singularity at the origin. Richardson's extrapolation is applied to the results, but we find that where the error expansion contains non-integer powers of h, the extrapolation is not always effective.

The case where $0 < \mu \leq 1$, and the implication of multiple solutions, is first considered in [9]. The product Euler method is now shown to converge to the single smooth solution, where the input function $g(t) \in C^1[0,t]$. If $\mu = 1$, there is the additional constraint that $g(0) = 0$. For $g \in C^2[0,T]$ and $\mu < 1$, the error expansion is now shown to be of the form

$$e_k^h = Ah^{\mu} + O(h).$$

If g is not sufficiently smooth, the techniques of [10] may be employed, and if $g(t) = t^{\tilde{\gamma}}\overline{g}(t)$, $\overline{g} \in C^2$, $\tilde{\gamma} \in R$, such that $\mu + \tilde{\gamma} < 1$, the order will depend on $\mu + \tilde{\gamma}$, not μ alone. We further note that if $\mu = 1$ or $\mu = 2$, there is a logarithmic term in the error expansion. The numerical results reflect this weak order of convergence, so extrapolation is again used to improve the accuracy. The scheme used in the paper is the E-algorithm of Brezinski (see [3] or [9]), which is a very general scheme, applicable to any situation for which the error expansion is known.

Up to this point, attention has been focused on approximating the smooth solution, and numerical methods for its trajectory which commence at the origin. In [4], approximation of the non-smooth solutions is considered. A specific member of the non-smooth family may be uniquely identified at a point $t = \alpha$ ($\alpha = kh, k \in Z$), and its subsequent approximation can be by any of the classical methods (the trapezoidal method is used as an example). This theme is continued in [6], where the product Euler and product trapezoidal rules are applied to a non-smooth solution away from the origin.

1.2. The aim of this investigation. As we have seen already, it has been possible to develop low order numerical methods for solving equations of the form (1). One can further improve the order through the use of extrapolation. However these methods are highly expensive to use in practice because of the very small step sizes needed for the computations and therefore we seek an alternative higher order method. The approach we shall use is to exploit the precise properties of our equation to enable the application of an established higher order method for the regularised problem.

2. A split-interval scheme

A developing theme over recent years in numerical analysis is the need to use a numerical scheme that reflects the underlying structure of an equation (or its solution) if one is to be successful in deriving a really efficient and effective method. Accordingly, here we present a method that reflects the underlying structure of equation (1), namely the fact that (for $\mu < 1$) the singularity in the integrand persists for all values of t but arises only at the initial point $s = 0$. Thus it is natural to present a method that utilises two distinct schemes, one close to $s = 0$ where the singularity has to be taken into consideration, and a second (higher order) formula for the remainder of the interval (where the solution is smoother and therefore a higher order scheme can be used with impunity).

Following the approach in [6], we rewrite (1) in the form

$$u(t) = g(t) + \int_0^\alpha \frac{s^{\mu-1}}{t^\mu} u(s)ds + \int_\alpha^t \frac{s^{\mu-1}}{t^\mu} u(s)ds. \tag{4}$$

Once the solution has been evaluated over the initial interval $[0, \alpha]$, this equation is equivalent to the equation

$$u(t) = \tilde{g}(t) + \int_\alpha^t \frac{s^{\mu-1}}{t^\mu} u(s)ds, \quad t > \alpha. \tag{5}$$

Now we can see that (5) is not singular. Indeed over any finite interval $t \in [\alpha, T]$ by standard analytical theory (see, for example, [11]) the equation has a unique solution. Moreover, standard numerical methods for Volterra equations can be applied to solve (5) with known order of convergence.

With this approach in mind, three decisions have to be made:

(1) we need to choose the formula that will be used over the initial sub-interval $[0, \alpha]$
(2) we need to choose the formula that will be used over the sub-interval $[\alpha, T]$ where we want to solve the equation over the interval $[0, T]$
(3) we need to choose an appropriate value of α at which we shall change over from one formula to the other.

Over the sub-interval $[0, \alpha]$ our priority needs to be the effective computation of the integral over an initial interval including the singularity. In previous work we have seen that there is no point looking for a high order scheme (this is absolutely clear in the case of the approximation of solutions that are not even smooth near $t = 0$). Because of the singularity in the integrand, it is quite hard to find high order numerical methods, so we shall use the product Euler rule as presented and analysed in [11] and (in the context of the present equation) [10], [9] over $[0, \alpha]$.

We take a uniform grid with fixed $h > 0$ where $T = Nh$ and put $t_i = ih, i = 0, \dots, N$. Now let $u_i \approx u(t_i)$ and $g_i = g(t_i)$. We define $\alpha = mh$ as the end point of the primary interval for some $m < N$.

The discrete form of equation (1) for $n = 0, \dots, m$ is

$$u_n = g_n + \sum_{i=0}^{n-1} w_i^{(n)} u_i \tag{6}$$

where the weights $w_i^{(n)}$ are given by the product integration formula

$$w_i^{(n)} = t_n^{-\mu} \int_{t_i}^{t_{i+1}} s^{\mu-1} ds = \frac{(t_{i+1}^{\mu} - t_i^{\mu})}{\mu t_n^{\mu}} \tag{7}$$

The construction of this scheme, together with results on convergence to the smooth solution, and an error analysis, are provided in [10] and [9].

For the second sub-interval, the discretised formula is extended for $t_n \in [\alpha, T]$ through the use of a higher order scheme over the second sub-interval:

$$u_n = g_n + \sum_{i=0}^{m} w_i^{(n)} u_i + \sum_{i=m}^{n} v_i^{(n)} u_i \tag{8}$$

where the $v_i^{(n)}$ are the weights of some suitable quadrature rule for the interval $[\alpha, nh]$. We shall return to the choice of weights $v_i^{(n)}$ later.

As we have seen in [6], the choice of α depends on T and the method we are planning to use over $[\alpha, T]$. If α was chosen to be too small, then there would be excessive errors in the approximation over the second sub-interval.

2.1. Convergence theory. It has been shown in [9] that the convergence order of the product Euler method with fixed step length $h > 0$ when $\mu < 1$ is $O(h^\mu)$, and it is the way in which this error is propagated into the second stage which now concerns us. We proceed in quite a general way and assume that the error in the value $u(\alpha)$ provided by the scheme in the first interval has an asymptotic error expansion $\mathcal{A}_1(h)$.

One can see from the formula (2) (or alternatively by the argument given in [5]) that every member of the family of exact solutions passes through the point $(0, \frac{\mu g(0)}{\mu-1})$ and thereafter different solutions do not intersect. Indeed, for every $\gamma \in \mathbb{R}$ and for every $\alpha \in \mathbb{R}^+$ there exists precisely one solution that passes through (α, γ). One can also show quite easily ([5]) that the rate of separation of two trajectories as time t elapses is proportional to $t^{1-\mu}$.

In principle, a particular solution trajectory may be selected by defining the (exact) value $u(\alpha)$ and this can now be used to provide insight into the error in the approximation for the split-interval scheme in the second sub-interval.

For the second sub-interval, the analysis of error propagation in the numerical scheme is straightforward. This is because we will be using a well-behaved numerical scheme with known order on an integral equation which satisfies all the usual analytical requirements (continuous forcing function, non-singular Lipschitz kernel and so on) and therefore the scheme will display its known order over any finite interval $[\alpha, T]$.

Now suppose that, at the end of the first sub-interval, the approximation of the exact solution $u(\alpha)$ is subject to an error (induced by the scheme) $\mathcal{E}$. Thus we are going to approximate in the second sub-interval using the wrong starting value

$u(\alpha) = \mathcal{U} + \mathcal{E}$ instead of $u(\alpha) = \mathcal{U}$ and let us suppose that we are using a method of order p.

The error in the solution induced by the second sub-interval method is $\mathcal{O}(h^p)$ and let us assume that for some fixed $t \in (\alpha, T)$ the error has an asymptotic expansion $\mathcal{A}_2(h)$.

We can now combine the errors $\mathcal{A}_1$ and $\mathcal{A}_2$ in the following way.

The error in the *exact* solution at time $t \in (\alpha, T)$ induced by the error $\mathcal{E}$ at time α is of the form $\mathcal{E}\dfrac{t^{1-\mu}}{\alpha^{1-\mu}}$. The additional error caused by the numerical method on the second sub-interval is then represented by $\mathcal{A}_2(h)$. This leads to the following conclusion:

THEOREM 2.1. *Under the assumptions given, the asymptotic error expansion of the total error at time* $t \in (\alpha, T)$ *is given by*

$$\mathcal{A}_1(h)\frac{t^{1-\mu}}{\alpha^{1-\mu}} + \mathcal{A}_2(h) \tag{9}$$

We have now established the details of the way that errors propagate from the first sub-interval to the second sub-interval. We can see, in particular, that errors introduced in the first sub-interval do not go away and indeed the asymptotic expansion of the errors from the first sub-interval are retained in the full asymptotic error expansion in a very natural way. This means that one cannot construct a high order scheme directly through the use of a low order scheme on the initial interval since the low order terms will be preserved in the full error expansion. However, the general approach we have taken to developing the asymptotic error formula permits us to develop two approaches to extrapolation for the improvement of the order of the split-interval method.

2.2. An example. The basic idea behind extrapolation methods is the use of approximations obtained by a method with known asymptotic error expansion and by using several different step lengths so that the lowest order terms in the error expansion may be eliminated, thereby providing a higher order of accuracy in the extrapolated scheme.

Obviously, since we have an asymptotic error expansion for our split-interval scheme, it would be possible to employ an extrapolation method on the combined scheme. However, as we shall see later, it may well be more efficient to use extrapolation at the value α to reduce the error in the initial sub-interval before the second (higher order) scheme takes over.

In order to show how extrapolation will work out in practice, we introduce here a simple prototype split interval scheme. For the time being, we seek to approximate the continuously differentiable solution. As we remarked earlier, there seems to be little benefit in using any scheme other than the product Euler scheme over the initial interval. Thereafter we could choose any well-behaved quadrature scheme. We shall choose to use the classical trapezium rule since the approach is clearly illustrated using this pair of schemes.

In the notation of the previous section, one knows that the asymptotic error expansions are, respectively, of the forms

$$\mathcal{A}_1(h) = a_1 h^{\mu} + a_2 h + a_3 h^{\mu+1} + a_4 h^2 + \ldots \tag{10}$$

and

$$\mathcal{A}_2(h) = b_2 h^2 + b_3 h^3 + b_4 h^4 + \ldots . \tag{11}$$

This means that the total error of the combined scheme takes the form

$$\mathcal{B}(h) = c_1 h^\mu + c_2 h + c_3 h^{\mu+1} + c_4 h^2 + \dots. \tag{12}$$

In other words, the combined scheme has the same error expansion as the low order product Euler rule and this means that there will be little benefit in employing the combined scheme directly to solve the equation.

On the other hand, if we are able to use extrapolation at the value α to eliminate the first three terms in the expansion for $\mathcal{A}_1$ then this will change the error expansion for the total error in the following way. $\mathcal{B}$ will now take the form

$$\mathcal{B}(h) = d_1 h^2 + d_2 h^{2+\mu} + d_3 h^3 + \dots. \tag{13}$$

This means that we will then have an overall method of the same order as the trapezium rule. However one needs to take note that the initial interval still has influenced the asymptotic error expansion of the combined scheme. The form of the error term contains non-integer powers of h and any extrapolation method to improve further the order of the split-interval method will need to take this into account. For example, two extrapolation stages will be needed (to eliminate the terms in h^2 and $h^{2+\mu}$) to improve the order from 2 to 3 instead of the classical one stage extrapolation.

3. Extrapolation

We are planning to use a modified version of the classical Richardson extrapolation. The modification to the technique arises through the presence of non-integer powers of h in the asymptotic error expansions. This means that the calculations are more complicated than in the classical case and that the improvement in convergence order requires more extrapolation steps.

Assume that $y^{(h)}$ is an approximation to y using a method with stepsize h and that the asymptotic error expansion is $\mathcal{E}^{(h)}$. In other words

$$y = y^{(h)} + \mathcal{E}^{(h)}. \tag{14}$$

Now the corresponding equation for steplength $2h$ is

$$y = y^{(2h)} + \mathcal{E}^{(2h)}. \tag{15}$$

One can use the expressions for $\mathcal{E}$ to obtain an improved approximation for y in the following way.

Assume that

$$\mathcal{E}^{(h)} = e_1 h^\mu + e_2 h + e_3 h^{\mu+1} + e_4 h^2 + \dots \tag{16}$$

and

$$\mathcal{E}^{(2h)} = e_1 (2h)^\mu + e_2 (2h) + e_3 (2h)^{\mu+1} + e_4 (2h)^2 + \dots \tag{17}$$

then by multiplying (14) by 2^μ and subtracting (15) we obtain an expression of the form

$$y = y_1^{(h)} + \mathcal{E}_1^{(h)}, \tag{18}$$

where

$$\mathcal{E}_1^{(h)} = e_2^{(1)} h + e_3^{(1)} h^{\mu+1} + e_4^{(1)} h^2 + \dots \tag{19}$$

which gives an improvement from a single extrapolation in the order of the method from $\mathcal{O}(h^\mu)$ to $\mathcal{O}(h)$.

Method	Method 1	Method 2	Method 3
Native order	μ	μ	same as at α
Work for order 1	9600	9600	6720
Work for order 2	12000	12000	6960
Work for order 3	11320	11320	11260

Table 1. Number of steps in calculation of solution at $T = 10$ with $h = \frac{1}{640}$ and $\alpha = 1$ for different extrapolation schemes

The same approach may be repeated successively (by calculating $\mathcal{E}_1^{(2h)}$, $\mathcal{E}_2^{(h)}$ etc) to eliminate the higher order terms in the error and, in principle at least, extrapolation may yield a method of arbitrarily high order. However, it is clear that extrapolation involves extra computational cost and so one must take this into account. In the following discussion we shall see how the computational cost of several competing extrapolation procedures compares for the problem under consideration.

First we should identify the possibilities we shall consider:

Method 1: use of the product Euler rule over the interval $[0, T]$ with extrapolation at $t = T$

Method 2: use of the product Euler rule over the interval $[0, \alpha]$ and the classical trapezium rule over $[\alpha, T]$ with extrapolation at $t = T$

Method 3: use of the product Euler rule over the interval $[0, \alpha]$ and the classical trapezium rule over $[\alpha, T]$ with extrapolation at $t = \alpha$ to raise the order (for each step size) of the initial approximation to 2 and further extrapolation at $t = T$ if needed to raise (further) the order of the overall method

Table 1 shows the number of steps of calculation required to obtain the solution at $T = 10$ by each of these methods. One needs to compare the amount of work in Table 1 with the errors we report in the subsequent Tables. Supposing we want to obtain an error of order h by Method 1 or Method 2 then we need to use two meshes of 6400 and 3200 nodes respectively. With Method 3 we would need to use meshes with 640 and 320 nodes over $[0, 1]$ but then only a further 5760 nodes over $[1, 10]$.

Remark 1. *Note that the non-integer value of μ imposes a practical constraint on implementation of the extrapolation scheme: in practice the computer cannot in general calculate 2^μ exactly and therefore the leading term in the asymptotic error expansion will be zero* only approximately according to the machine precision. *Asymptotically, as $h \to 0$, the non-zero coefficient of the dominant error terms will become more significant and the order gained through extrapolation could be lost.*

4. Numerical Experiments

For our numerical experiments we consider two examples.

Example 1. We take the equation:

$$u(t) = 1 + t + \int_0^t \frac{s^{\mu-1}}{t^\mu} u(s)\, ds \tag{20}$$

with the two values of $\mu = 0.4$ and $\mu = 0.8$.

We shall calculate the value of $u(10)$ by each of our three extrapolation schemes using $\alpha = 1$ for the split interval methods. Tables 2-5 give the absolute errors in the approximate solution (compared to the exact analytical solution) in each case.

Product Euler; $\mu = 0.4$; T=10; True sol.34.3333						
h	$e_0^{(n)} = e_n$	$e_1^{(n)}$	$e_2^{(n)}$	$e_3^{(n)}$	$e_4^{(n)}$	$e_5^{(n)}$
1/20	4.9018					
1/40	3.7496	$1.4342e-1$				
1/80	2.8594	$7.3154e-2$	$2.8869e-3$			
1/160	2.1760	$3.7123e-2$	$1.0915e-3$	$3.9725e-6$		
1/320	1.6536	$1.8768e-2$	$4.1290e-4$	$1.1141e-6$	$1.6135e-7$	
1/640	1.2555	$9.4621e-3$	$1.5627e-4$	$3.0133e-7$	$3.0391e-8$	$2.2097e-10$
Product Euler; $\mu = 0.8$; T=10; True sol.18.5						
h	$e_0^{(n)} = e_n$	$e_1^{(n)}$	$e_2^{(n)}$	$e_3^{(n)}$	$e_4^{(n)}$	$e_5^{(n)}$
1/20	$6.2890e-1$					
1/40	$3.8199e-1$	$4.8814e-2$				
1/80	$2.2981e-1$	$2.4474e-2$	$1.3334e-4$			
1/160	$1.3721e-1$	$1.2256e-2$	$3.8219e-5$	$1.0408e-7$		
1/320	$8.1416e-2$	$6.1334e-3$	$1.0953e-5$	$3.1492e-8$	$7.2956e-9$	
1/640	$4.8067e-2$	$3.0683e-3$	$3.1392e-6$	$8.6550e-9$	$1.0427e-9$	$5.6168e-12$

TABLE 2. Method 1 for Example 1, with 5 levels of extrapolation at T=10 ($\mu = 0.4$ and $\mu = 0.8$)

Split Interval; $\mu = 0.4$; T=10; True sol.34.3333						
h	$e_0^{(n)} = e_n$	$e_1^{(n)}$	$e_2^{(n)}$	$e_3^{(n)}$	$e_4^{(n)}$	$e_5^{(n)}$
1/20	-5.2139					
1/40	-3.9333	$7.4645e-2$				
1/80	-2.9616	$7.9624e-2$	$8.4604e-2$			
1/160	-2.2309	$5.6088e-2$	$3.2551e-2$	$7.9201e-4$		
1/320	-1.6825	$3.4021e-2$	$1.1955e-2$	$6.1135e-4$	$1.0791e-3$	
1/640	-1.2704	$1.9175e-2$	$4.3293e-3$	$3.2305e-4$	$2.2695e-4$	$2.7749e-5$
Split Interval; $\mu = 0.8$; T=10; True sol.18.5						
h	$e_0^{(n)} = e_n$	$e_1^{(n)}$	$e_2^{(n)}$	$e_3^{(n)}$	$e_4^{(n)}$	$e_5^{(n)}$
1/20	$6.2360e-1$					
1/40	$3.8147e-1$	$5.4747e-2$				
1/80	$2.3041e-1$	$2.6593e-2$	$1.5618e-3$			
1/160	$1.3782e-1$	$1.2887e-2$	$8.1772e-4$	$5.1797e-4$		
1/320	$8.1830e-2$	$6.2760e-3$	$3.3542e-4$	$1.4112e-4$	$1.5504e-5$	
1/640	$4.8309e-2$	$3.0772e-3$	$1.2168e-4$	$3.5574e-5$	$3.9201e-7$	$2.1417e-6$

TABLE 3. Method 2 for Example 1, with 5 levels of extrapolation at T=10 ($\mu = 0.4$) and $\mu = 0.8$)

Split Interval; $\mu = 0.4$*; T=1; True sol.2.8333*				
h	$e_0^{(n)} = e_n$	$e_1^{(n)}$	$e_2^{(n)}$	$e_3^{(n)}$
1/20	1.1393			
1/40	$8.9377e-1$	$1.2527e-1$		
1/80	$6.9341e-1$	$6.6295e-2$	$7.3172e-3$	
1/160	$5.3386e-1$	$3.4531e-2$	$2.7673e-3$	$8.7321e-6$
1/320	$4.0890e-1$	$1.7788e-2$	$1.0453e-3$	$5.3380e-6$
1/640	$3.1209e-1$	$9.0916e-3$	$3.9489e-4$	$1.9241e-6$
1/1280	$2.3764e-1$	$4.6204e-3$	$1.4927e-4$	$5.9140e-7$
1/2560	$1.8066e-1$	$2.3384e-3$	$5.6458e-5$	$1.6857e-7$
Split Interval; $\mu = 0.8$*; T=1; True sol.-1.75*				
h	$e_0^{(n)} = e_n$	$e_1^{(n)}$	$e_2^{(n)}$	$e_3^{(n)}$
1/20	$2.9565e-1$			
1/40	$1.8988e-1$	$4.7153e-2$		
1/80	$1.1927e-1$	$2.3996e-2$	$8.3773e-4$	
1/160	$7.3661e-2$	$1.2119e-2$	$2.4180e-4$	$1.7218e-6$
1/320	$4.4901e-2$	$6.0941e-3$	$6.9497e-5$	$8.0114e-8$
1/640	$2.7090e-2$	$3.0570e-3$	$1.9936e-5$	$3.0098e-8$

Table 4. Method 3 for Example 1, showing 3 levels of extrapolation at T=1

Split Interval; $\mu = 0.4$*; T=10; True sol.34.3333*			
h	$e_0^{(n)} = e_n$	$e_1^{(n)}$	$e_2^{(n)}$
1/160	$2.4840e-5$		
1/320	$1.8764e-5$	$1.6746e-5$	
1/640	$7.0404e-6$	$3.1295e-6$	$5.3649e-8$
1/1280	$2.2010e-6$	$5.8593e-7$	$8.6286e-9$
1/2560	$6.3276e-7$	$1.1000e-8$	$1.2495e-9$
Split Interval; $\mu = 0.8$*; T=10; True sol.18.5*			
h	$e_0^{(n)} = e_n$	$e_1^{(n)}$	$e_2^{(n)}$
1/160	$6.4075e-7$		
1/320	$1.0486e-7$	$7.3774e-8$	
1/640	$1.8278e-8$	$1.0581e-8$	$1.3911e-11$

Table 5. Method 3 for Example 1, showing 2 further levels of extrapolation at T=10

The tables illustrate the success of the extrapolation for the split-interval scheme. In Table 5, the right hand column provides evidence in support of the $\mathcal{O}(h^3)$ that we would expect to achieve with the extrapolation scheme we have followed. Thus the extrapolation scheme with three levels of extrapolation at $t = \alpha$ and a further two levels of extrapolation at $t = T$ is efficient in terms of gaining convergence of order 3 at least computational cost. We have not considered in detail in this paper the conditions required for extrapolation to be applied successfully. However

Split Interval; $\mu = 0.4$; T=1; True sol.3.5153				
h	$e_0^{(n)} = e_n$	$e_1^{(n)}$	$e_2^{(n)}$	$e_3^{(n)}$
1/20	1.1402			
1/40	$8.9424e-1$	$1.2450e-1$		
1/80	$6.9365e-1$	$6.5846e-2$	$7.1910e-3$	
1/160	$5.3399e-1$	$3.4285e-2$	$2.7240e-3$	$1.3643e-6$
1/320	$4.0897e-1$	$1.7658e-2$	$1.0312e-3$	$1.6019e-6$
1/640	$3.1212e-1$	$9.0243e-3$	$3.9049e-4$	$4.3763e-7$
1/1280	$2.3766e-1$	$4.5861e-3$	$1.4793e-4$	$6.4933e-8$
1/2560	$1.8067e-1$	$2.3211e-3$	$5.6057e-5$	$5.5540e-9$

TABLE 6. Split interval for Example 2, scheme showing 3 levels of extrapolation at T=1

Split Interval; $\mu = 0.4$; T=10; True sol.8.3266			
h	$e_0^{(n)} = e_n$	$e_1^{(n)}$	$e_2^{(n)}$
1/160	$5.0578e-7$		
1/320	$4.8929e-6$	$6.6925e-6$	
1/640	$1.3712e-6$	$1.9726e-7$	$1.3210e-6$
1/1280	$1.6574e-7$	$2.3607e-7$	$3.3737e-7$
1/2560	$4.5302e-8$	$1.1565e-7$	$8.7499e-8$

TABLE 7. Split interval scheme for Example 2, showing 2 further levels of extrapolation at T=10

we know from other contexts that the usual requirement is that the solution be sufficiently differentiable for the calculation of the asymptotic error expansion for the method to be possible.

With this in mind, we consider a second example.

Example 2. We take the equation

$$u(t) = t + \frac{t^{2.5}}{100} + \int_0^t \frac{s^{\mu-1}}{t^\mu} u(s)\, ds \tag{21}$$

with $\mu = 0.4$.

Here the solution is $\mathcal{C}^2$ but not $\mathcal{C}^3$ and this imposes a constraint on the order improvement through extrapolation. We present (in Tables 6 and 7) the corresponding tables to Tables 4 and 5 for Example 2. Note how the order of convergence obtained this time seems to be $\mathcal{O}(h^2)$.

5. Conclusions and further work

As we have seen, Method 2 provides little, if any, advantage over Method 1. There is no saving in computational effort and the errors are not made smaller. This is as expected since, despite using the trapezium rule over the second sub-interval, Method 2 is of the same order as Method 1. However, Method 3 is considerably more efficient than Methods 1 or 2 when used without extrapolation at T. One sees a loss of efficiency when further extrapolation at T is needed. For the development of higher order methods we would propose an alternative strategy: use a different split-interval scheme with more extrapolation steps at α and a higher order numerical method on $[\alpha, T]$. We shall return to a discusssion of this in a sequel.

Acknowledgements

The authors are pleased to acknowledge the helpful comments of the referees.

References

[1] H. Brunner. *Collocation Methods for Volterra Integral and Related Functional Differential Equations*, Cambridge University Press, 2004.

[2] H. Brunner and P. J. van der Houwen. *The Numerical Solution of Volterra Equations*, North Holland, 1986.

[3] C. Brezinski and M. Redivo-Zaglia. *Extrapolation Methods: Theory and Practice.* North-Holland, Amsterdam, 1991.

[4] T. Diogo, J.T. Edwards, N.J. Ford and S.M. Thomas. *Numerical analysis of a singular integral equation*, Appl. Math. Comp. **167** 2005. p.372–382.

[5] T. Diogo, J.M. Ford, N.J. Ford and P. Lima. *Non-integrable resolvent kernels and qualitative behaviour for exact and approximate solutions to some convolution integral equations*, submitted for publication.

[6] T. Diogo, N. J. Ford, P. Lima and S. Valtchev. *Numerical methods for a Volterra integral equation with non-smooth solutions*, to appear in J. Comp. Appl. Math.

[7] T. Diogo, S. McKee and T. Tang. *A Hermite-type collocation method for the solution of an integral equation with certain weakly singular kernel*, IMA J. Numer. Anal. **11** 1991. p.595–605.

[8] W. Han. *Existence, uniqueness and smoothness results for second-kind Volterra equations with weakly singular kernels*, J. Int. Eq. Appl. **6** 1994. p.365–384.

[9] P. Lima and T. Diogo. *Numerical solution of a nonuniquely solvable Volterra integral equation using extrapolation methods*, J. Comp. Appl. Math. **140** 2002. p.537–557.

[10] P. Lima and T. Diogo. *An extrapolation method for a Volterra integral equation with weakly singular kernel*, Appl. Numer. Math. **24** 1997. p.131–148.

[11] P. Linz. *Analytical and Numerical Methods for Volterra Equations*, SIAM. Philadephia. 1985.

[12] T. Tang, S. McKee and T. Diogo, *Product integration methods for an integral equation with a logarithmic singular kernel*, Appl. Numer. Math. **9** 1992. p.259–266.

Department of Mathematics, Instituto Superior Técnico, Lisbon, Portugal

Department of Mathematics, University of Chester, Chester, CH1 4BJ, UK

Department of Mathematics, Instituto Superior Técnico, Lisbon, Portugal

Department of Mathematics, University of Chester, Chester, CH1 4BJ, UK