An Adaptive Higher Order Method in Time for Partial Integro-Differential Equations

Jörg Frochte

Fachhochschule Südwestfalen, Elektrische Energietechnik Soest, Lübecker Ring 2, 59494 Soest

Abstract. The numerical solution of time-dependent partial integro-differential equations is considered. This includes both variants, the initial value problem and the prehistory problem. One problem concerning partial integro-differential equations is the data storage. To deal with this problem an adaptive method of third order in time is developed to save storage data at smooth parts of the solution. Beyond this a post-processing step adaptively thins out the history data.

Keywords: Finite element method, Integro-Differential Equations, delay term, higher order, adaptivity
PACS: 2.60Nm, 2.70Dh

INTRODUCTION

Let $T > 0$ and $\Omega_T = \Omega \times (0,T]$ where Ω is an open bounded region in $\mathbb{R}^n$. First let us consider time dependent partial integro-differential equations. Such equations appear for example (see [1], [2], [3], [4] chapter 1) in different contexts of heat conduction in materials with memory, viscoelasticity and population models. We will study a special class of time dependent partial integro-differential equations. We call the following the initial value problem

$$\begin{aligned} \frac{d}{dt}u - \varepsilon\nabla^2 u + k\cdot\nabla u &= f + \int_0^t K(s-t)u\,ds \quad \text{in } \Omega_T, \\ u &= \hat{u} \quad \text{on } \partial\Omega\times(0,T], \\ u &= u_0 \quad \text{in } \Omega, \text{for } t = 0 \end{aligned} \tag{1}$$

and we also consider the prehistory problem, where d is the fixed length of the delay:

$$\begin{aligned} \frac{d}{dt}u - \varepsilon\nabla^2 u + k\cdot\nabla u &= f + \int_{t-d}^t K(s-t+d)u\,ds \quad \text{in } \Omega_T, \\ u &= \hat{u} \quad \text{on } \partial\Omega\times(0,T]\times(0,T], \\ u &= u_{history} \quad \text{in } \Omega, \text{for } t\in[-d,0] \end{aligned} \tag{2}$$

$\varepsilon > 0$ is the diffusion coefficient and $k : \Omega_T \to \mathbb{R}^n \times (0,T]$ the convection coefficient. K is a $C^1(\mathbb{R} \supset I_K \to \mathbb{R})$ function called kernel, I_K is $[-d,0]$ in the prehistory case and $[0,T]$ for the initial value problem.

HIGHER ORDER INTEGRATION SCHEME

First we present a higher order scheme with a fixed time step size using a third order discretisation in time, in our case the BDF scheme, which is suitable to deal with stiff problems. The higher order in the integral term is performed using Hermite interpolation. The resulting integration scheme is independent of the discretisation $\tau = \{t^0,\ldots t^n\}$ in time. Hence, to achieve a method with a variable time step size we will be able to apply common mechanics for adaptive time step size control for parabolic PDEs straightforward to the method.

Integration using Hermite interpolation

Let us consider a function $\hat{f}(s) : \mathbb{R} \to \mathbb{R}$ and let $t^j < t^{j+1} \in \mathbb{D}_{\hat{f}}$. So, if we have $f(t^j)$, $\frac{d}{dt}f(t^j)$, $f(t^{j+1})$ and $\frac{d}{dt}f(t^{j+1})$, we can interpolate f on $[t^j,t^{j+1}]$ with a cubic polynomial. Let now the functions in our integral term at a fixed point

in space $\hat{x} \in \Omega$ be denoted as $\hat{f}(s) := K(s-t) \cdot u(\bar{x},s)$. We can now first interpolate $\hat{f}(s)$ using the cubic Hermite interpolation over $[t^j, t^{j+1}]$ with $z = (s-t^j)/(t^{j+1}-t^j)$ and

$$u_{tj} = (1+2z)(1-z)^2, \qquad u_{tj+1} = (3-2z)z^2 \qquad v_{tj} = z(1-z)^2, \qquad v_{tj+1} = -z^2(1-z)\,. \tag{3}$$

Thus we get: $\hat{f}(s) \approx p(s) = u_{tj}(s)\hat{f}(t^j) + u_{tj+1}(s)\hat{f}(t^{j+1}) + (t^{j+1}-t^j)\left(v_{tj}(s)\frac{d\hat{f}(t^j)}{ds} - v_{tj+1}(s)\frac{d\hat{f}(t^{j+1})}{ds}\right)$
This cubic polynomial can be integrated exactly by using Simpson's rule.

$$\int_{t^j}^{t^{j+1}} \hat{f}\,ds \quad \approx \quad \int_{t^j}^{t^{j+1}} p\,ds = \frac{t^{j+1}-t^j}{2}\left(\hat{f}(t^j)+\hat{f}(t^{j+1})\right) + \frac{(t^{j+1}-t^j)^2}{12}\left(\frac{d\hat{f}(t^j)}{ds} - \frac{d\hat{f}(t^{j+1})}{ds}\right)$$

If we now combine this approach with a time discretisation of third order, we obtain a method of third order in time. Now, the question is left how to evaluate $\frac{d\hat{f}(s)}{ds} = u(\bar{x},s)\cdot\frac{dK(s-t)}{ds} + K(s-t)\cdot\frac{du(\bar{x},s)}{ds}$ of the same order as the time discretisation. While we generally assume that $\frac{dK(s-t)}{ds}$ is given as an analytic expression, if not we use the same technique as described for u below, we will have to approximate $\frac{du(\bar{x},s)}{ds}$ anyway. To do this we again choose a BDF approach of the same order as the one used for the time discretisation. Generally it is supposed to compute $\frac{du(\bar{x},s)}{ds}$ once and save it, so that the CPU costs are only spent once. Of course, alternatively it is possible to trade CPU time vs. memory and compute $\frac{du(\bar{x},s)}{ds}$ at every time step.
This approach can be applied to the inner part of the integral. It will not work with the recently added part of the delay integral $[t^n, t^{n+1}]$. Because u^{n+1} is unknown, we have to treat it differently from the ones before. Many approaches are possible like an extrapolation, a kind of predictor-corrector scheme etc.. In numerical tests it turns out that the influence of this part is generally small enough just to use the left rectangular rule for the subsequent examples.
We also have to consider the lower limit of the integral. In this case we have to distinguish between the initial value and the prehistory problem. In both cases it is not clear how to compute an approximation for $\frac{du(\bar{x},0)}{ds}$, respectively $\frac{du(\bar{x},-d)}{ds}$. For in the prehistory case it is quite easy, we can approximate $\frac{du(\bar{x},-d)}{ds}$ using forward instead of backward differences. In the initial value case we can use the same ansatz, but not right from the start. We have to wait a few time steps until enough data has been accumulated. So, in the case of the initial value problem until the third time step we will only have an approximation of first or second order for the integral term.

Adaptive step size control

One major advantage of the presented Hermite integration scheme is the fact that we are able to choose the time step size in every step independent of the ones chosen before. So, we are only limited by the used time stepping technique. With a variable step size the coefficients of the BDF scheme have to be recomputed in every time step.
To construct an adaptive scheme we need an approximation $\bar{u}_h(t^{j+1})$ to compare the computed solution $u_h(t^j)$ with. The computation of this approximation should require low CPU costs, however it should not force the algorithm to unnecessary changes in the time step size. To achieve this we use an extrapolation of third order based on the polynomial of the BDF scheme: $\gamma_i = \prod_{j=0, j\neq i}^{k} \frac{t^{n+1}-t^{n+1-j}}{t^{n+1-i}-t^{n+1-j}}$, $\gamma_0 = 1 - \sum_{i=1}^{k}\gamma_i$ $\bar{u}^{n+1} = \sum_{i=0}^{k}\gamma_i u^{n-i}$. Now we set up the error function as follows: $\eta = \|\bar{u}^{n+1} - u_{\Delta t^n}^{n+1}\| \leq \mathrm{Rtol}\cdot\max\{\|\bar{u}^{n+1}\|, \|u^{n+1}\|\} + \mathrm{Atol} = E$, with two parameters Rtol and Atol to be chosen, and the solution $u_{\Delta t^n}^{n+1}$ computed with the time step size from the last time step. Based on this the next time step size is chosen. For the used BDF scheme with variable time step size the choice of Δt^{n+1} is restricted by the conditions published by Grigorieff, see [7] and [8]. For the most practical problems the boundaries published by Grigorieff are quite pessimistic, so that for our solver we choose $\Delta t^{n+1} = \alpha\Delta t^n$ with $0.75 \leq \alpha \leq 1.25$ instead of $0.836 \leq \alpha \leq 1.127$ from [7] .

For the initial value problem the adaptivity can be applied straightforward. We just have to add the new value to the history. Considering the prehistory problem there is some additional work to do. We have to watch carefully that for any new time step size Δt the delay integral is of the given length d. To do this we have to consider three different cases illustrated by figure 1. Let $\Delta t^0, \Delta t^1, ..., \Delta t^{n-1}$ be the time step size (= interval length) of the last n time steps and $u(t^0), u(t^1), ..., u(t^n)$ the corresponding values of the solution. If $\Delta t^n = \Delta t^0$, case a] in fig.1, we obtain the same case as for the non-adaptive technique. We have just to delete Δt^0 and $u(t^0)$ from the history. In the case that $\Delta t^n < \Delta t^0$ (b])

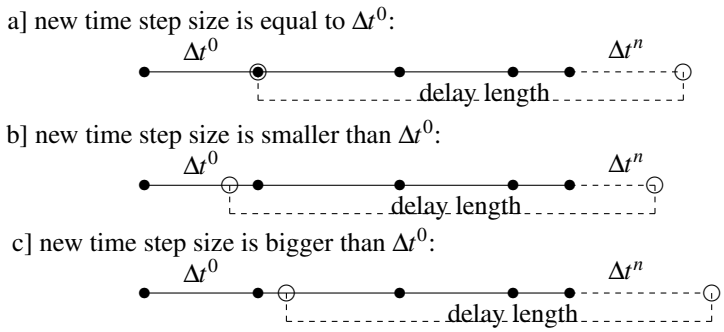

FIGURE 1. An outline of the different cases concerning prehistory problems and adaptivity

we interpolate u at the required position $t^{n+1}-d$ and set $\Delta t^0 := t^1 - (t^{n+1}-d)$. Finally the case $\Delta t^n > \Delta t^0$ (c]) is left. Here like in case a] we start deleting the first value and afterwards continue as in case b].

Adaptive post-processing of the history

With this technique we have the ability to deal with a non-equidistant discretisation of the memory term. We can now use this ability to adapt this discretisation in a post-processing step to a given accuracy. This approach is limited to two different types of kernels, first a fading kernel with the "fading condition" :$K(t_1) < K(t_2)$ for all $t_1 < t_2$ or at least second a kernel, which fulfils the fading condition for all $t_1, t_2 < \hat{t}$. A common example for the second type of kernel is a kind of bell curve with the maximum at $\hat{t}$. In this case the condition is not true for all t in the kernel space, but for all t before the peak. These two classes are very important when modelling memory problems. In population dynamics a bell curve is popular and in many other applications a fading kernel would be used.
The post-processing is now performed at the part of the history where fading condition is true. First we compare the approximation of the integral $\int_{t^n}^{t^{n+2}} K(s-t+d)u$ using the information at the discretisation point t^{n+1} with an approximation that just depends on t^n and t^{n+2}. If the difference between these two approximations is small enough, t^{n+1} is deleted from the memory term. This thinning obviously saves memory and CPU-time as well because the computed integral in the next steps is smaller. On the other hand the post-processing itself requires some CPU resources, so it should be only applied every m time steps, where m is depending on the computed problem. In numerical tests $5 < m < 20$ has been a good choice.

NUMERICAL RESULTS

For the discretisation in space we use linear finite elements, if necessary with streamline diffusion stabilisation, see e.g. [9] for details. The mesh is fixed during the simulation. The presented numerical results concentrated on the interaction of the adaptive techniques for the chosen time step size and the post-processing of the memory term. Results for higher convection dominated problems and for those violating the C^1 assumption of the kernel K, but without the described post-processing techniques, can be found in [10]. Let us now consider the following prehistory testproblem:

$$\frac{d}{dt}u - \nabla^2 u + (1,1)^t \cdot \nabla u = f + \int_{t-2}^{t} K(s)u\,ds \quad t \in [0,4]\,, \tag{4}$$

with $K = \exp(s-t)$. f is chosen in a way that $u = \frac{(\sin(2\pi t)+1)/2}{2\cosh(x-m)\cosh(y-n)}$ is the solution of the testproblem over $\Omega = [0,1]\times[0,1]$. To verify the order in time the results for the Hermite approach with BDF(3) are displayed in table 1 (left). Until the error in space becomes dominant we can see reduction rates of third order. The error is measured as follows: $\|u-u_h\|_{L_2}^{\max} = \max_{t^j\in\tau} \|u(t^j)-u_h(t^j)\|_{L_2}$. In table 1 we can see the effects of the presented adaptive techniques. Together with figure 2 this shows that common adaptive techniques for parabolic PDEs in time can be applied to integro-differential equations. The additional adaptive post-processing for the saved history is able to save a notable amount of memory. Used with care the loss in the accuracy is minimal as table 1 shows.

TABLE 1. Results for the Hermite integration and a BDF(3) scheme (33025 degrees of freedom in space). The left table shows the order in time and the right one compares the behaviour for different adaptive techniques.

Δt	$\|u-u_h\|_{L_2}^{\max}$	quotient
1/8	1.324e-2	-
1/16	1.789e-3	7.40
1/32	2.296e-4	7.79
1/64	3.029e-5	7.58
1/128	4.441e-6	6.82

adaptivity in time	adaptive post-proc.	averaged Δt	$\|u-u_h\|_{L_2}^{\max}$	averaged history size
no	no	1/64	3.029e-5	128
no	yes	1/64	3.22e-5	95
yes	no	0.198	1.59e-4	109
yes	yes	0.198	1.58e-4	87

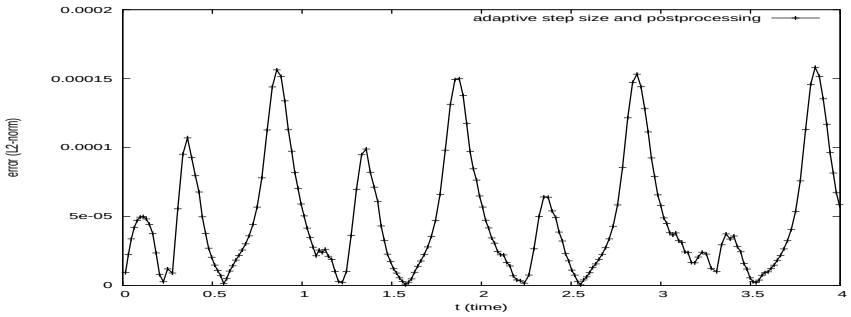

FIGURE 2. L_2-Error of the full adaptive approach with Rtol=1e-5 and Atol=1e-7

CONCLUSION AND FUTURE PROSPECTS

A third order algorithm for convection-diffusion problems with a delay term was presented. The design of the algorithm makes it possible to reach adaptivity in time using common techniques for parabolic PDEs. The adaptivity in time is one approach to deal with the memory storage problem. Beyond this the presented adaptive post-processing for the saved history is able to economise memory usage with only a minimal loss of accuracy. Together with the adaptivity in time this leads to a very effective technique for the presented classes of integro-differential equations.
Up to now the research in this area has been concentrated on linear and semi-linear cases, see e.g. [11], [12], [13], [14]. Future prospects will be the extension on non-linear problems as the population model in [2], p.868 and on problems with so called weak kernels $|K(t)| < Ct^{-\alpha}$ with $0 < \alpha < 1$ for $t > 0$, see e.g. [15].

ACKNOWLEDGMENTS

The author would like to thank Wolfgang Ruess for bringing the field of integro-differential equations to his attention.

REFERENCES

1. H.-M. Yin, *SIAM J. Math. Anal.* **22**, 1723–1737 (1991).
2. W. Ruess, *Advances in Differential Equations* **4**, 843–876 (1999).
3. M. Dehghan, *International Journal of Computer Mathematics* **84**, 123–129 (2006).
4. C. Chen, and T. Shih, *Finite Element Methods for Integrodifferential Equations*, World Scientific Publishing, Singapore, 1998.
5. R. K. Sinha, and A. K. Pani, *J. Integral Equations and Applications* pp. 219–249 (1998).
6. H. Sloan, and V. Thomée, *SIAM J. Numer. Anal.* **23** (1986).
7. R. D. Grigorieff, *Numerische Mathematik* **42**, 359–377 (1983).
8. M. Calvo, T. Grande, and R. D. Grigorieff, *Numerische Mathematik* **57**, 39–50 (1990).
9. H.-G. Roos, S. M., and T. L., *Numerical Methods for Singulary Perturbed Differential Equations*, Springer, Berlin, 1996.
10. J. Frochte, *Proceedings of the ENUMATH 2007 [to be published]* (2008).
11. V. Thomée, and N.-V. Zhang, *Mathematics of Computation* **53**, 121–139 (1989).
12. Y. Lin, V. Thomée, and L. B. Wahlbin, *SIAM J. Num. Ana.* **28**, 1047–1070 (1991).
13. A. K. Pani, V. Thomée, and L. B. Wahlbin, *Journal of Integral Equations and Applications* **4**, 231–252 (1992).
14. A. K. Pani, and T. E. Peterson, *SIAM Journal on Numerical Analysis* **33**, 1084–1105 (1996).
15. C. Chen, V. Thomée, and L. B. Wahlbin, *Mathematics of Computation* **58**, 587–602 (1992).