A third order method for Convection-Diffusion Equations with a Delay term

Jörg Frochte

Abstract The numerical solution of a parabolic convection diffusion equation with delay term is considered. This includes both variants, the initial value problem and the prehistory problem. Equations with a delay or memory term, often called integrodifferential problems, appear in different contexts of heat conduction in materials with memory, viscoelasticity and population models. This work concentrates on the linear convection diffusion case of the prehistory and the initial value problem. One problem concerning delay or memory problems is the data storage. To deal with this problem an adaptivity method of third order in time is developed to save storage data at smooth parts of the solution. Numerical results for higher Péclet numbers are presented.

1 Introduction

Let $T > 0$ and $Q_T = \Omega \times (0,T]$ where Ω is an open bounded region in $\mathbb{R}^n$. First let us consider a parabolic partial integrodifferential equation of the kind

$$\begin{aligned} \frac{d}{dt}u - Au = f + \int_0^t B(t,s)u\,ds \quad &\text{in } \Omega, \\ u = \hat{u} \quad &\text{on } \partial\Omega, \\ u = u_0 \quad &\text{in } \Omega, \text{for } t = 0\,, \end{aligned} \tag{1}$$

with a linear elliptic operator A and a problem depending operator B. Such equations appear for example (see [12], [9], [3], [2] chapter 1) in different contexts of heat conduction in materials with memory, viscoelasticity and population models. These models tend to be nonlinear, but of course first the numerical behaviour of linear

Jörg Frochte
Fachhochschule Sdwestfalen Elektrische Energietechnik Soest Lbecker Ring 2 59494 Soest , e-mail: joerg.frochte@uni-due.de

problems has to be studied. Up to now the research in this area has been concentrated on linear and semi-linear cases, see e.g. [11], [5] [7], [6]. We will study a special case of (1).

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

We call this the initial value problem and we also consider the prehistory problem:

$$\frac{d}{dt}u - \varepsilon\nabla^2 u + k\cdot\nabla u = f + \int_{t-d}^t K(s-t+d)u\,ds \quad \text{in } \Omega, \tag{3}$$
$$u = \hat{u} \quad \text{on } \partial\Omega,$$
$$u = u_{history} \quad \text{in } \Omega, \text{for } t \in [-d,0]$$

K is a $C^1(\mathbb{R} \to \mathbb{R})$ function called kernel, $\varepsilon > 0$ is the diffusion coefficient and $k \in C^0(\mathbb{R}^n \to \mathbb{R}^n)$ the convection coefficient. In the prehistory case, d is the fixed length of the delay.
So additionally to the common problems of partial integrodifferential equations like e.g. memory storage we have to deal with the problems arising in the context of convection diffusion equations. With a rising global Péclet number defined as

$$Pe = \frac{h_\Omega \|k\|_{\max}}{\varepsilon} \tag{4}$$

h_Ω is the characteristic length of the domain Ω. To solve this problem we will use linear finite elements with streamline diffusion stabilisation, see e.g. [8] for details. One of the first approached solver strategies for the initial value problem was published in [10] and uses a left trapez rule to deal with the integral term and a backward Euler scheme for the time discretisation. The result is a scheme of first order for the initial value problem. If we apply this to our initial value problem (2) we achieve:

$$\frac{u^{n+1}-u^n}{\Delta t} - \varepsilon\nabla^2 u^{n+1} + k\cdot\nabla u^{n+1} = f + \Delta t\sum_{j=0}^{n} K(j\Delta t)u^{n-j} \tag{5}$$

To deal with the memory storage problem of the integral term e.g. Thomée advocates integration techniques of higher order so that not every time step has to be stored.

2 Higher order integration scheme

In this paper we propose another approach. First we present a higher order scheme for a fixed time step size using a third order discretisation in time, in our case the BDF schemes, 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 $\tau = \{t_0, ...t^n\}$ in time so that we can straightforward apply common mechanics for adaptive time step size control to the scheme.

2.1 Integration using hermite interpolation

Let us consider a function $\hat{f}(t) : \mathbb{R} \to \mathbb{R}$ 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 $\hat{x} \in \Omega$ be denoted as

$$\hat{f}(t) := K(t - t_1) \cdot u(\overline{x}, t) \; .$$

We can now first interpolate $\hat{f}(t)$ using the cubic hermite interpolation over $[t^j, t^{j+1}]$ with $s = (t - t^j)/(t^{j+1} - t^j)$ and

$$u_{t^j} = (1+2s)(1-s)^2, \qquad u_{t^{j+1}} = (3-2s)s^2 \tag{6}$$

$$v_{t^j} = s(1-s)^2, \qquad v_{t_{j+1}} = -s^2(1-s) \; . \tag{7}$$

Thus we get:

$\hat{f}(t) \approx p(t) = u_{t^j}(t)\hat{f}(t^j) + u_{t^{j+1}}(t)\hat{f}(t^{j+1}) + (t^{j+1} - t^j)\left(v_{t_j}(t)\frac{d\hat{f}(t_j)}{dt} + v_{t_{j+1}}(t)\frac{d\hat{f}(t_{j+1})}{dt}\right)$

This cubic polynomial can be integrated exactly by using Simpson's rule.

$$\int_{t_j}^{t^{j+1}} \hat{f}\, dt \approx$$

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

If we now combine this approach with a time discretisation of third order we obtain a method of third order in time. Now there is the question left how to evaluate

$$\frac{\hat{f}(t)}{dt} = u(\overline{x}, t) \cdot \frac{dK(t - t_1)}{dt} + K(t - t_1) \cdot \frac{du(\overline{x}, t)}{dt}$$

of the same order as the time discretisation. While we generally assume that $\frac{dK(t-t_1)}{dt}$ is given as an analytical expersion, if not we use the same technique as described for u below, we will have to approximate $\frac{du(\overline{x},t)}{dt}$ 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 $\frac{du(\bar{x},t)}{dt}$ 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 $\frac{du(\bar{x},t)}{dt}$ in every time step.

This approach can be applied in 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 different from the ones before. Many approaches are possible like an extrapolation, a kind of predictor-corrector scheme etc.. In numerical test it turns out that the influence of this part is generally small enough just to use the left trapezium rule for the subsequent exampels.
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 $\frac{du(\bar{x},0)}{dt}$, respectively $\frac{du(\bar{x},-d)}{dt}$. For in the prehistory case it is quite easy, we can approximate $\frac{du(\bar{x},-d)}{dt}$ 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.

2.1.1 Numerical Results

Let us now consider the following prehistory testproblem:

$$\frac{d}{dt}u - \varepsilon\nabla^2 u + k\cdot\nabla u = f + \int_{t-4}^{t} K(s)u\,ds \quad t\in[0,4]\ , \tag{8}$$

with $K = \exp(s-t)$. f is chosen in a way that

$$u = \frac{g(t)}{2\cosh(a(x-m))\cosh(a(y-n))} \tag{9}$$

is the solution. With $g(t) = (\sin(2\pi t)+1)/2$ we call this testproblem I.

To verify the order in time we choose the parameters ($a = 1$, $\varepsilon = 1$ and $k = (1,1)^t$) for the testproblem which causes only minor difficulties in space. In table 1 the results for hermite approach with BDF(3) are displayed. Until the error in space becomes dominant we can see reduction rates of third order. For higher Péclet numbers the stabilisation of the galerkin method is performed by streamline diffusion.

Table 1 Results for the hermite integration and a BDF(3) scheme on testproblem I.1 with $a = 5$. The left table shows the order in time and the right one the behaviour for different Péclet numbers.

Δt	Pe	$\|u-u_h\|_{L_2}$	Quotient
1/8	1	1.322e-2	-
1/16	1	1.794e-3	7.369
1/32	1	2.274e-4	7.889
1/64	1	2.847e-5	7.987
1/128	1	4.112e-6	6.923

Δt	Pe	$\|u-u_h\|_{L_2}$
1/32	10^0	2.274e-4
1/32	10^1	9.301e-4
1/32	10^2	1.286e-3
1/32	10^3	1.329e-3
1/32	10^4	1.335e-3

2.2 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. For BDF(3) we achieve:

$$\beta_3 = \frac{(t^{j+1}-t^j)(t^{j+1}-t^{j-1})(t^{j+1}-t^j)}{(t^{j-2}-t^{j-1})(t^{j-2}-t^j)(t^{j-2}-t^{j+1})}\ ; \quad \beta_2 = \frac{(t^{j+1}-t^j)(t^{j+1}-t^{j-2})(t^{j+1}-t^j)}{(t^{j-1}-t^{j-2})(t^{j-1}-t^j)(t^{j-1}-t^{j+1})}$$

$$\beta_1 = (-1)\frac{(t^{j+1}-t^{j-2})(t^{j+1}-t^{j-1})}{(t^j-t^{j-1})(t^j-t^{j-2})}\ ; \quad \beta_0 = 1-\beta_3-\beta_2-\beta_1$$

To construct an adaptive scheme we need an approximation $\overline{u}_h(t^{j+1})$ 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:

$$\gamma_i = \prod_{j=0, j\neq i}^{k} \frac{t^{n+1}-t^{n+1-j}}{t^{n+1-i}-t^{n+1-j}}\ , \quad \gamma_0 = 1-\sum_{i=1}^{k}\gamma_i \quad \overline{u}^{n+1} = \sum_{i=0}^{k}\gamma_i u^{n-i}$$

Now we set up the error function as follows:

$$\eta = \|\overline{u}^{n+1} - u^{n+1}_{\Delta t^n}\| \leq \mathrm{Rtol}\cdot\max\{\|\overline{u}^{n+1}\|, \|u^{n+1}\|\} + \mathrm{Atol} = E\ , \tag{10}$$

with two parameters Rtol and Atol to be chosen and the solution $u^{n+1}_{\Delta t^n}$ computed with the time step size from the last time step. Based on this the next time step size is chosen as follows:

$$h^{n+1} = \alpha h^n \qquad \alpha = \begin{cases} 0.75 & \text{, if } \left(\frac{E}{\eta}\right)^{1/4} < 0.75 \\ \left(\frac{E}{\eta}\right)^{1/4} & \text{, if } 0.75 \leq \left(\frac{E}{\eta}\right)^{1/4} \leq 1.25 \\ 1.25 & \text{, if } < 1.25 \left(\frac{E}{\eta}\right)^{1/4} \end{cases}$$

For the used variable step size BDF scheme the choice of Δt^{n+1} is restricted by the conditions published by Grigorieff, see [4] and [1]. For the most practical problems the boundaries published by Grigorieff are quite pessimistic so that for our solver we choose $0.75 \leq \alpha \leq 1.25$ instead of $0.836 \leq \alpha \leq 1.127$ from [4].

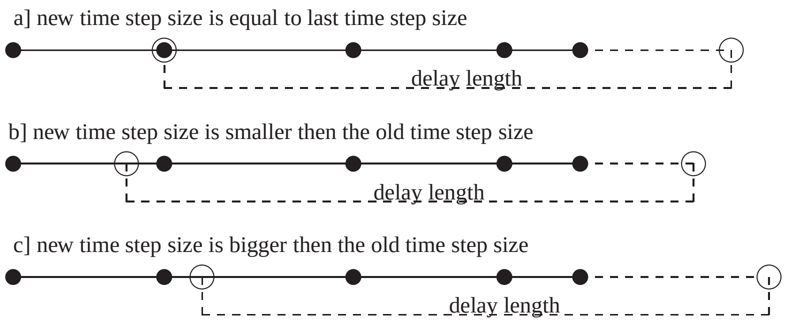

a] new time step size is equal to last time step size
delay length
b] new time step size is smaller then the old time step size
delay length
c] new time step size is bigger then the old time step size
delay length

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

For the initial value problem the adaptivity can be applied straightforward. We have just to add the new value to the history. Considering the prehistory problem there is some additonal 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 this we have to consider three different cases illustrated by figure 1. Let $\Delta t^0, \Delta t^1, \dots, \Delta t^{n-1}$ be the time stepsize (=interval length) of the last n time steps and $u(t^0), u(t^1), \dots, u(t^n)$ the corresponding values of the solution. If $\Delta t^n = \Delta t^0$, case a] in fig.1, we are in 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]) 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 first delete the first value and afterwards continue as in case b].

2.2.1 Numerical Results

If we choose $Atol = Rtol$ =5e-4 in the error indicator (10) we would expect the error control to achieve an accuracy of 1e-3 on the unit square. If we now consider the result for the testproblem I in figure 2, we can see that this accuarcy has been achieved. We can see from our test that if we use the presented adaptive approach the error indicators used for parabolic partial differential equations without delay can also be used for problems with a delay. But one should keep in mind that problems with a delay are more sensible than problems without a delay and so smaller tolerance parameters have to be chosen for a required accuracy compared to a problem without delay.

Let us consider now the testproblem II:

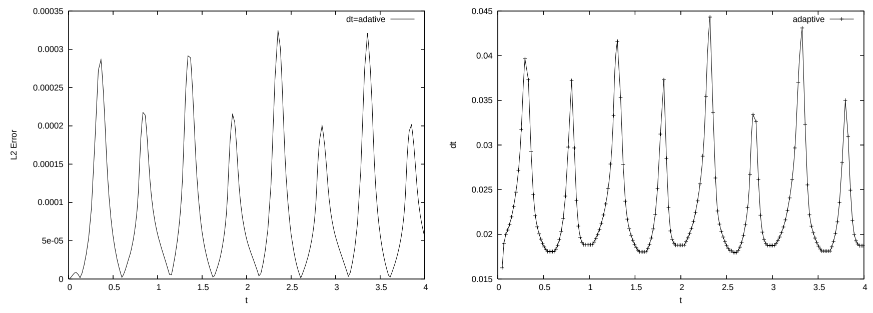


Fig. 2 For the adaptive approach with $Atol = Rtol = 5e-4$ we see the L_2-Error on the left and on the chosen dt the right.

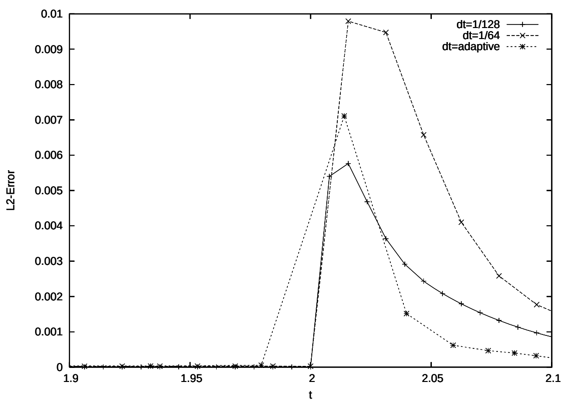
$$\frac{d}{dt}u - \varepsilon\nabla^2 u + k\cdot\nabla u = f + \int_{t-2}^{t} K(s)u\,ds \quad t\in[0,4]\,, \tag{11}$$

with $K = \sin(10s)$. f is chosen in a way that

$$u = \frac{g(t)}{2\cosh(a(x-m))\cosh(a(y-n))} \tag{12}$$

with $a = 1$ and $g(t) = \exp(-|t-2|)$ is the solution. The solution u is non-differentiable

Fig. 3 This figure shows the behaviour of the solution computed with a fixed time step size and the one with an apdaptive chosen time step size in the region around the non-differentiable point $t = 2$.



at $t = 2$ and so it contains a special challenge, especially for higher order multistep methods like BDF schemes, compared to onestep methods. Beyond this, we assumed that $u \in C^1[0,4]$ when we used the hermite interpolation. Table 2 displays the results for testproblem II for two fixed time step sizes and the adaptive chosen one. Figure 3 and table 2 show that this is an intrinsic problem for this technique but it does not tend to diverge in the case the assumptions are violated.
But comparing the result for $dt = 1/64$ and $dt = 1/128$ we see that the reduction rate

Table 2 Results for the hermite integration and a BDF(3) scheme on testproblem II

Δt choosen	mean Δt	$\|u-u_h\|_{L_2}$
fixed	= 0.015625	9.794e-3
adaptive	$\approx$ 0.047014	7.108e-3
fixed	= 0.007812	5.764e-3

is far a away from the third order in such a case. If we compare the mean timestep size we see that with the adaptive scheme we achieved a higher accuracy with less time steps.

3 Conclusion

A third order algorithm for convection-diffusion problems with a delay was presented. The presented technique is robust for higher Péclet numbers and a violation of the C^1 assumption. The design of the algorithm leads straightforward to adaptivity in time with the potentiality to use techniques common from parabolic PDEs. The adaptivity in time is also one aproach to deal with the memory storage problem. A future prospect will be to develop different storange strategies for kernels with different properbilties. A major goal considering the last results is an adaptive choice of the order of the BDF scheme in order to improve the handling of non-differentiable points.

References

1. Calvo, M., Grande, T., Grigorieff, R.D.: On the zero stability of the variable order variable stepsize bdf-formulas. Numerische Mathematik **57**, 39–50 (1990)
2. Chuanmiao, C., Tsimin, S.: Finite Element Methods for Integrodifferential Equations. World Scientific Publishing, Singapore (1998)
3. Dehghan, M.: Solution of a partial integro-differential equation arising from viscoelasticity. International Journal of Computer Mathematics **84**(1), 123–129 (2006)
4. Grigorieff, R.D.: Stability of multistep-methods on variable grids. Numerische Mathematik **42**, 359–377 (1983)
5. Lin, Y., Thomée, V., Wahlbin, L.B.: Ritz-volterra projections to finite-element spaces and applications to integrodifferential and related equations. SIAM J. Num. Ana. **28**, 1047–1070 (1991)
6. Pani, A.K., Peterson, T.E.: Finite element methods with numerical quadrature for parabolic integrodifferential euqations. SIAM Journal on Numerical Analysis **33**(3), 1084–1105 (1996)
7. Pani, A.K., Thomée, V., Wahlbin, L.B.: Numerical methods for hyperbolic and parabolic integro-differential equations. Journal of Integral Equations and Applications **4**(4), 231–252 (1992)
8. Roos, H.G., M., S., L., T.: Numerical Methods for Singulary Perturbed Differential Equations. Springer, Berlin (1996)
9. Ruess, W.: Existence and stability of solutions to partial functional differential equations with delay. Advances in Differential Equations **4**(6), 843–876 (1999)

10. Sloan, H., Thomée, V.: Time discretization of an integro-differential equation of parabolic type. SIAM J. Numer. Anal. **23**(5) (1986)
11. Thomée, V., Zhang, N.V.: Error estimates for semidiscrete finite element methods for parabolic integro-differential equations. Mathematics of Computation **53**(187), 121–139 (1989)
12. Yin, H.M.: On parapolic volterra equations in several space dimensions. SIAM J. Math. Anal. **22**(6), 1723–1737 (1991)