Damping of Crank–Nicolson error oscillations

D. Britz a,⁎, O. Østerby b, J. Strutwolf a

a Department of Chemistry, Aarhus University, 8000 Århus C, Denmark
b Department of Computer Science, Aarhus University, 8000 Århus C, Denmark

Received 23 May 2002; received in revised form 27 June 2002; accepted 10 September 2002

Abstract

The Crank–Nicolson (CN) simulation method has an oscillatory response to sharp initial transients. The technique is convenient but the oscillations make it less popular. Several ways of damping the oscillations in two types of electrochemical computations are investigated. For a simple one-dimensional system with an initial singularity, subdivision of the first time interval into a number of equal subintervals (the Pearson method) works rather well, and so does division with exponentially increasing subintervals, where however an optimum expansion parameter must be found. This method can be computationally more expensive with some systems. The simple device of starting with one backward implicit (BI, or Laasonen) step does damp the oscillations, but not always sufficiently. For electrochemical microdisk simulations which are two-dimensional in space and using CN, the use of a first BI step is much more effective and is recommended. Division into subintervals is also effective, and again, both the Pearson method and exponentially increasing subintervals methods are effective here. Exponentially increasing subintervals are often considerably more expensive computationally. Expanding intervals over the whole simulation period, although capable of satisfactory results, for most systems will require more cpu time compared with subdivision of the first interval only.

© 2002 Elsevier Science Ltd. All rights reserved.

Keywords: Crank–Nicolson; Oscillations; Damping

1. Introduction

1.1. Simulated systems

We solve the diffusion equation

\[ \frac{∂c}{∂t} = b \frac{∂^2c}{∂x^2} \]  

(1)

in which \( c \) denotes concentration, \( t \) the time and \( x \) the space axis in one-dimensional systems. All variables are dimensionless. After discretisation of the equation, the factor \( b \) is equal to unity for equal spatial intervals but for unequally spaced points \( b \) is a function of the distance from the electrode, as will be shown below. The equation must be supplemented with boundary conditions. In the present paper, these will be restricted to those of the Cottrell experiment (Cottrell, 1903), which is a potential step at time zero such that the concentration at the electrode \( (x = 0) \) is pulled to zero. The boundary conditions are thus

\[ \begin{align*}
    t < 0: & \quad c(t, x) = 1 \\
    t \geq 0: & \quad c(t, 0) = 0 \\
    x \to \infty, & \quad \text{all } t: c = 1
\]  

(2)

This is mathematically equivalent to the heat equation with an initial temperature defect, with which this shares the oscillation problem when using the Crank–Nicolson (CN) method for the solution. The equation has an analytical solution for both \( c(t, x) \) and the flux at the electrode (see standard texts such as Bard and Faulkner, 2001 or Galus, 1994), that simulations can be compared with. The reason for choosing the Cottrell system is that it is representative of the class of systems with an initial singularity, and these are a major cause of problems with the CN method.

It is recognised that when Eq. (1) is discretised onto an array of evenly spaced points in time and space \( (b = 1) \), more points in space are needed for a given target accuracy than with an unequal distribution of points.
One way of implementing unequal intervals is to transform the space coordinate to another, in which equal intervals correspond to a more suitable unequal spacing in $x$. This was first suggested (for electrochemical simulations) by Joslin and Fletcher (1974) and a transformation that has some convenient properties was described (Britz, 1980). It is the transformation into $y$-space, according to the function

$$y = \ln(1 + ax)$$

where $a$ is the parameter that varies the distribution of points. Large values of this parameter produce a more unequal distribution, while equal intervals in both $x$ and $y$ are approached as $a$ approaches zero. With equal intervals in $y$, this transformation also produces a very similar distribution of points in $x$ to the exponentially expanding intervals series proposed by Pao and Daugherty (1969) in a non-electrochemical context and by Feldberg (1981) in the electrochemical area. In the present paper, only the transformation method will be considered. When it is applied to Eq. (1), that equation becomes

$$\frac{\partial c}{\partial t} = a^2 \exp(-2y) \left( \frac{\partial^2 c}{\partial y^2} - \frac{\partial c}{\partial y} \right)$$

where now $b = a^2 \exp(-2y)$, i.e. variable.

Another system used as an example here is the ultramicrodisk electrode (UME), with Cottrell conditions applied, that is a potential jump to an extreme potential. This is a spatially two-dimensional system and has a mathematical singularity at the disk edge due to the abrupt change in boundary conditions there. Simulations of this were first attempted by Evans and Gourlay (1977), Crank and Furzeland (1977) and Heinzl and Shain (1959). It is analogous to the trapezium method in the field of ordinary differential equations (Lapidus and Seinfeld, 1971; Jain, 1984), and in effect expresses both sides of the above diffusion equations as finite approximations centred on a point in time midway between the old time and the new at which new values are to be calculated. For example, Eq. (1), when going from known concentration values at the $j$th time level $j \delta t$ to new values at time $(j+1) \delta t$ (these being indicated by superscripts), gives rise to the discrete system of equations

$$\frac{c_i^{j+1} - c_i^j}{\delta t} = \frac{b}{2} \left( \frac{c_{i-1}^j - 2c_i^j + c_{i+1}^j}{\delta x^2} \right)$$

one for each $i$ in the range $1 < i < N$, that is, we have $N$ internal concentration points in space and the boundary points at $i = 0$ (the electrode) and $i = N+1$ (the bulk solution point) (see for example Britz, 1988 for more details and a description of how this is implemented). In the following, the parameter $\lambda$ is used. For the discretisation of Eq. (1) this is simply

$$\lambda = \frac{\delta t}{\delta x^2}$$

while for Eq. (4) $\delta x$ is replaced by $\delta y$. For the UME, there are two $\lambda$ values, one corresponding to $\theta$ and the other to $\Gamma$. 

\[254\]


\[263\]
The important factor in the following considerations is the product $b\lambda$. For Eq. (1) this is a constant; for Eq. (4) the largest value occurs at the first interval in space. For the UME the value corresponding to $\theta$ is usually larger than for $\Gamma$, the largest value being obtained for small $\theta$ and $\Gamma$.

1.3. The problem

The CN method is attractive from several points of view. It makes use of only two time levels, sharing this property with the explicit method (Courant et al., 1928) and that of Laasonen (1949), see also Lapidus and Pinder (1982), also called backwards or fully implicit or BI. CN is of second-order accuracy with respect to time, whereas the other two are first order. It is formally stable (except under rather exceptional circumstances, rarely encountered in practice (Bieniasz et al., 1995a,b, 1997) and therefore, roundoff errors will not accumulate for any value of $b\lambda$, in contrast with the explicit method, thus allowing the use of rather larger time intervals than that method. There is, however, a problem, already recognised by Crank and Nicolson (1947), who note that for large values of (what is here called) $b\lambda$, and sharp initial transient conditions, the numerical solution shows oscillations not connected with roundoff but with the initial conditions. These may persist over a large number of steps. Standard texts such as those of Richtmyer and Morton (1967), Smith (1985) or Strikwerda (1989) all give this problem attention. Greenwood (1962) concluded that because of these oscillations, the less accurate BI method is preferable, having a non-oscillatory response, and this method is also the basis for the higher-order backward differentiation formulae (BDF), first proposed for electrochemistry by Mocak and Feldberg (1994) for the same reason. These authors note that CN might under some circumstances keep oscillating during the entire simulation period. Similar motives prompted the extrapolation method described by Lawson and Morris (1978) and Strutwolf and Schoeller (1997), also based on BI. On the other hand, Gresho and Lee (1981) suggest not suppressing the oscillations, as they make errors visible, which smooth error responses do not. One would have to agree with Mocak and Feldberg, however, that if the oscillations persist over the whole simulated period, they are unacceptable. It would therefore be convenient if a way of damping the oscillations could be found for the CN method. The other simulation methods of choice at this time, such as BDF (or FIRM, as the Feldberg group calls it) and extrapolation, all make use of either several time levels (BDF) or several steps with different $\delta t$, which makes programming less straightforward. The apparently highly efficient Rosenbrock method, recently suggested by Bieniasz (1999) is not for the occasional non-specialist electrochemist programmer.

2. Theory

In order to study the behaviour of finite difference schemes it is instructive to use the Fourier transform approach (von Neumann and Richtmyer, 1950, O'Brien et al., 1950) as more recently described in Strikwerda (1989) and many other texts. The propagation of a finite difference solution from one time step to the next is governed by the growth factor which for the explicit method is

$$g(\varphi) = 1 - 4b\lambda\sin^2\frac{\varphi}{2} \quad -\pi < \varphi < \pi$$

in which $\varphi$ is the parameter in the frequency domain, in this case, inverse $x$; $\varphi$ close to $0$ corresponds to slowly varying components and $\varphi$ close to $\pi$ corresponds to highly oscillatory components of the solution. The latter are present when there are discontinuities in the initial and boundary conditions.

From the form of $g$ in Eq. (8) it is apparent that $g(\varphi)$ is always less than 1. But if $b\lambda > 1/2$ then $g(\varphi)$ may become less than $-1$. An absolute value of $g > 1$ implies that the corresponding solution component will be magnified: we have instability. We note that instability appears first (and most strongly) for $\varphi \approx \pm \pi$, i.e. for the highly oscillatory components, and that since $g < -1$ these will be propagated as oscillations in time.

For CN the growth factor is

$$g(\varphi) = \frac{1 - 2b\lambda\sin^2\frac{\varphi}{2}}{1 + 2b\lambda\sin^2\frac{\varphi}{2}} \quad -\pi < \varphi < \pi$$

It is apparent that $|g(\varphi)| \leq 1$ indicating that CN is unconditionally stable irrespective of $b\lambda$ and $\varphi$ (again, with the rare exceptions noted above). But we note that when $\varphi \approx \pm \pi$ then $g(\varphi) \approx -1$, especially when $b\lambda$ is large. This means that the oscillatory components are propagated as weakly damped oscillations in time, a feature well known for CN.

For BI the growth factor is

$$g(\varphi) = \frac{1}{1 + 4b\lambda\sin^2\frac{\varphi}{2}} \quad -\pi < \varphi < \pi$$

It is easily seen that $0 \leq |g(\varphi)| \leq 1$ for all $b\lambda$ and $\varphi$, so the BI method is unconditionally stable and furthermore, it will never produce oscillations in time. An interesting point is that $g(\varphi)$ is very small for $\varphi \approx \pm \pi$ and large $b\lambda$, so the components for which CN displays the most annoying behaviour are the same components that are damped most strongly by BI.

Table 1 shows the growth factor for BI and CN for the highest frequency component for various values of $b\lambda$. The growth factor for BI is always less than 1, but for CN, it may become less than $-1$, especially when $b\lambda$ is large. This means that the oscillatory components are propagated as weakly damped oscillations in time, a feature well known for CN.
Table 1
Growth factors as in Eqs. (9) and (10) for BL and CN at various $b\lambda$

<table>
<thead>
<tr>
<th>$b\lambda$</th>
<th>BI</th>
<th>CN</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.7143</td>
<td>0.6667</td>
</tr>
<tr>
<td>0.5</td>
<td>0.3333</td>
<td>0.0000</td>
</tr>
<tr>
<td>1</td>
<td>0.2000</td>
<td>-0.3333</td>
</tr>
<tr>
<td>10</td>
<td>0.0244</td>
<td>-0.9048</td>
</tr>
<tr>
<td>100</td>
<td>0.0025</td>
<td>-0.9900</td>
</tr>
</tbody>
</table>

$b\lambda$. As $b\lambda$ gets bigger the unwanted oscillations receive less and less damping from CN. At $b\lambda = 100$ the damping is only 1% per time step. But one initial step with BL will immediately reduce the amplitude of such oscillations by a factor 0.0025.

Another obvious suggestion from Table 1 is to choose the initial $\delta t$ such that $b\lambda = 0.5$ in which case CN itself will eliminate the high frequency components, but it is not necessary to use so small a $\delta t$. Even $b\lambda$ values of 2 or 3 used over 10 or 20 steps will provide considerable damping.

3. Some damping methods

Three ways of minimising the oscillations of CN as a response to an initial discontinuity are described and were tested.

The three simulation methods mentioned above: explicit, CN and BL, can be regarded as points on a continuum of implicitness, on a scale from zero to unity lying at 0, 0.5 and 1.0, respectively. CN causes oscillations, and BI does not. In this light, some workers (Patankar and Baliga, 1978; Ganesan and Salamon, 1995) have attacked the problem by using a degree of implicitness between 0.5 and 1. This is difficult to do while retaining the second-order nature of CN. Chawla et al. (2000) described a method for calculating an optimum degree of implicitness. Wood and Lewis (1975) examined the oscillation problem and sought to eliminate it by averaging the initial values with the result of the very first CN step. It can readily be shown that this corresponds to taking a BI step of half a time interval. This would have the effect of shifting the time back by $1/2(\delta t)$ from that point on, as time usually does not appear in the equations to be solved but is counted by the number of steps taken. Since 1982, several workers have used a number of initial BI steps to damp oscillations: Rannacher (1982, 1984); Luskin and Rannacher (1982); Khaliq and Wade (2001); Østerby (2002). They all discuss the optimal number of such steps, and generally tend to use 2 or even 4 of them. It is known from their work and can readily be verified that any fixed number of initial BI steps, followed by CN steps for the remainder of the process, retains the second-order behaviour of the global error; this is so because BI itself has a local second-order error. However, as the number of initial BI steps is increased, the error coefficient in front of $\delta t^2$ becomes larger, and so does the error. Our own tests indicate that a single BI step appears to be about optimum and can in some cases be an effective damper of oscillations caused by CN. BI is known for its lack of oscillations but also for its global first-order error, so here we might have the benefit of the one without the drawback of the other. It turns out that such an initial BI step (hereafter called the BI method) indeed does damp the oscillations caused by CN, and is especially effective when $b\lambda$ is large as is the case with our microdisk simulations. Finally, we mention the work of Bank et al. (1985), who used a combination of CN and a second-order BDF at every step, which does not appear to hold any advantages in the present context.

Focusing on the first time interval, we also have the option of subdividing it into a number of smaller subintervals. This might work, because, as was shown above, it is the value of $b\lambda$ that determines whether or not there will be oscillations, and subdivision can yield sufficiently small values. The question then is, might the oscillations resume when subsequent time intervals use whole $\delta t$ or the whole $b\lambda$ again. If so, increasing intervals might be preferred, as they might provide a smoother transition to the whole time intervals starting at the second whole step. Equal intervals subdivision was first tried by Pearson (1965) and this work was followed by that of Fang and Chen (1997). This will be called the Pearson method below. Expanding subintervals were suggested in private correspondence between one of us (DB) and Feldberg (1981), and later used by Britz and Østerby (1994), Britz (1996), Georganopoloulou et al. (2000) and mentioned by Mocak and Feldberg (1994). This will be termed exponentially increasing subintervals.

Both Pearson and exponentially increasing subintervals can be combined into one description, using two parameters. Let the first time interval $\delta t$ be divided into $M$ subintervals $\tau_k$, $k = 1, \ldots, M$ and let the following hold:

$$\tau_k = \beta \tau_{k-1}$$

(11)

$$\sum_{k=1}^{M} \tau_k = \delta t$$

(12)

If we set $\beta = 1$ then this corresponds to $M$ Pearson steps, but if $\beta > 1$, we have a sequence of $M$ subintervals of increasing length.

Another option is to use exponentially increasing time intervals throughout a simulation. One might thus begin with such small time intervals that no oscillations appear and as the time intervals grow, the initial transient that would cause oscillations, has been ‘forgotten’. This was
first suggested by Peaceman and Rachford (1955), in their paper describing the alternating implicit (ADI) method, and introduced to electrochemical simulations by Lavagnini et al. (1991) and Feldberg and Goldstein (1995). It is now routinely used by the Svir group in their simulations of electrochemically induced chemiluminescence, for example Svir and Golovenko (2001). This is a special case of the above exponentially increasing subintervals method, that is, the case in which the whole simulation is done in a single large ‘first’ step of length \( \delta t \).

Gavaghan (1998) used a mixture of the Pearson and exponentially increasing intervals techniques. He started the (microdisk) simulations with a base \( \delta t \) for \( M \) steps, then a further \( M \) steps with length 10\( \delta t \), etc. We find this unnecessary, as either method works on its own.

A further option is to use adaptive time interval control, as implemented by Bieniasz (1994) and in an improved version (Bieniasz, 2002). This will not be dealt with in this paper.

### 3.1. Theory of the Pearson method and exponentially increasing subintervals

In the Pearson strategy the first interval of length \( \delta t \) is divided into \( M \) subintervals of equal length \( \tau = \delta t / M \).

With CN, the growth factor for each of these subintervals and for the highest frequency component is therefore according to Eq. (9)

\[
g = \frac{1 - 2b\lambda / M}{1 + 2b\lambda / M}
\]

and the cumulative damping is given by \( g^* = g^M \). If we wish a total damping of say \( 10^{-p} \) then

\[
\left| \frac{1 - 2b\lambda / M}{1 + 2b\lambda / M} \right| = 10^{-p/M}
\]

or

\[
\left| \frac{2b\lambda}{M} \right| = \left( \frac{1 + 2b\lambda / M}{M} \right) 10^{-p/M}
\]  

(14)

Only large \( b\lambda \) are of interest for these considerations so we can assume \( b\lambda > M/2 \) and we have

\[
\frac{2b\lambda}{M} - 1 = \left( \frac{2b\lambda}{M} + 1 \right) 10^{-p/M}
\]  

(15)

or

\[
b\lambda = \frac{M}{2} \left( 1 + 10^{-p/M} \right)
\]  

(16)

We thus have a relationship between \( b\lambda \) and \( M \), enabling us to achieve a given damping of \( 10^{-p} \). This relationship is shown graphically in Fig. 1 for \( p = 1, 2, 3, 4 \). We note that if \( b\lambda = 100 \) we can achieve a damping of 0.0001 by choosing \( M = 30 \) or higher.

With exponentially increasing subintervals, we choose an initial substep \( \tau_1 \) and a factor \( \beta > 1 \) such that the following substeps are \( \tau_2 = \beta \tau_1, \tau_3 = \beta^2 \tau_1, \) etc. and such that \( M \) substeps make up one whole subsequent time step:

\[
\delta t = \tau_1 + \beta \tau_1 + \beta^2 \tau_1 + \ldots + \beta^{M-1} \tau_1 = \tau_1 \frac{\beta^M - 1}{\beta - 1}
\]

(18)

or

\[
\tau_1 = \delta t \left( \frac{\beta - 1}{\beta^M - 1} \right)
\]

(19)

Defining \( \lambda_i = \tau_i / \delta x^2 \), \( i = 1, \ldots, M \) we have

\[
\lambda_i = \frac{\beta - 1}{\beta^M - 1}, \quad \lambda_i = \beta^{i-1} \lambda_1, \quad i = 2, \ldots, M
\]

(20)

and a total damping of

\[
g^* = \prod_{i=1}^{M} \left| \frac{1 - 2b\lambda_i}{1 + 2b\lambda_i} \right| = \prod_{i=1}^{M} \left| \frac{1 - 2\beta^{-i+1}b\lambda_1}{1 + 2\beta^{-i+1}b\lambda_1} \right|
\]

(21)

For a given \( b\lambda \) we can calculate \( g^* = g^*(M, \beta) \). The surface \( g^* \) as a function of \( M \) and \( \beta \) contains a large number of isolated zeros which occur if one of the involved \( b\lambda \) becomes equal to 0.5. This means that there are several possibilities of achieving complete damping of the highest frequency component. But there are other components (corresponding to \( |\varphi| < \pi \)) which may give rise to undesirable oscillations. These components effectively correspond to (slightly) smaller \( b\lambda \)-values and therefore different \( b\lambda \)-values. It is therefore better to focus attention on a ‘worst case’ where the \( b\lambda \) values are as far from 0.5 as possible. The resulting \( g^{**} \) indicates the ‘assured’ damping in a neighbourhood of \( \beta \)- and \( b\lambda \)-values, such that the observed damping will always be better than or at least as good as \( g^{**} \).
For a given \( b \lambda \) and a given \( M \) we choose \( \beta \) such that
\[
\beta b \lambda_1 < 0.5, \quad b \lambda_2 = \beta b \lambda_1 > 0.5
\]
and such that
\[
\beta b \lambda_1 - 0.5 = 0.5 - b \lambda_1
\]  
(22)

or
\[
b \lambda_1 (\beta + 1) = 1
\]
(23)

Using Eq. (20) we get
\[
b \lambda (\beta^2 - 1) - (\beta M - 1) = 0
\]
(24)

This equation can be solved numerically for \( \beta \). With, e.g. \( b \lambda = 100, \ M = 8 \), we get \( \beta \approx 2.061 \). In the context of the value of \( g^* \), it is not necessary to require excessive accuracy in \( \beta \) since the value of \( g^* \) exhibits a rather small variation when we are away from the zeros.

Alternatively one could determine \( \beta \) such that
\[
1 - 2b \lambda_1 = g_1 = -g_2 = -\frac{1 - 2\beta b \lambda_1}{1 + 2\beta b \lambda_1}
\]
(25)

or
\[
4\beta (b \lambda_1)^2 = 1
\]
(26)

or
\[
b \lambda (\beta - 1) - \frac{1}{2} \beta^{-1/2} (\beta M - 1) = 0
\]
(27)

The solution is a slightly different value of \( \beta \) and with a slightly different value of \( g^* \), but the difference is small. In the above example \( (b \lambda = 100, \ M = 8) \) \( \beta \) comes to 2.037.

The assured damping is achieved when using this particular value of \( M \). Larger values of \( M \) are allowed, but the gain in damping is limited, especially for larger values of \( b \lambda \).

The effect of a change in \( \lambda \) can be explained as follows. When \( b \lambda \) and \( M \) are not too small then the last 1 in Eq. (24) contributes very little and can be left out. If \( b \lambda \) is now replaced by \( b \lambda / \beta \) then we get a solution to the modified equation when \( M \) is replaced by \( M - 1 \). So the assured damping is almost the same but it is obtained for a smaller value of \( M \), i.e. with less work. So the assured damping is mostly dependent on \( \beta \), assuming that \( M \) is chosen large enough.

Large values of \( \beta \) (around 2 or higher) are not recommended because successive values of \( b \lambda_1 \) and \( b \lambda_{i+1} = \beta b \lambda_i \) can be very far (on either side) from the optimal value of 0.5, and previous and successive \( \lambda_i \) quickly get so far away that they contribute little to the cumulative damping. Smaller values of \( \beta \) allow the \( b \lambda_i \) to pack closer around 0.5, and successive values are close enough that they can contribute appreciably to \( g^* \). But smaller values of \( \beta \) require larger values of \( M \) and thus more work.

In the contour plot (Fig. 2) we supply values of the assured cumulative damping, \( g^{**} \) as a function of \( M \) and \( \beta \) for \( b \lambda = 100 \). The damping as a function of \( \beta \) for other values of \( b \lambda \) can also be estimated from this figure. Only the value of \( M \) will be different. We note that a damping of 0.001 is achieved for \( 1.5 < \beta < 1.8 \) and a value of \( M \) of about 12, whereas a damping of 0.0001 requires \( \beta \) between 1.4 and 1.5 and \( M = 14 \). This value can be compared with the value of \( M = 30 \) required with the Pearson strategy for \( b \lambda = 100 \) and the same damping. For some simulation problems, the advantage of a smaller \( M \) may be counteracted by a greater computing time with increasing subintervals, see below in the section on microdisk simulations.

4. Computational remarks

Simulations were run at Aarhus University on an 800 MHz PC running under Linux, using FORTRAN 90 with roughly 16-decimal precision throughout. Contour calculations were done using MATLAB on a Sun microsystems Enterprise 2.

5. Results

5.1. Cottrell system

In all the simulations reported in this section, 100 time intervals of 0.01 each were used, except where indicated otherwise. Different values of \( b \lambda \) are obtained by varying the spatial interval \( \Delta x \). Both current and the relative error are shown, with the relative error defined as
\[
e = \frac{I_{\text{sim}} - I_{\text{anal}}}{I_{\text{anal}}}
\]  
(28)
where $I_{\text{sim}}$ is the simulated dimensionless current, and $I_{\text{anal}}$ the known analytical value.

5.1.1. Using the BI method

Fig. 3 shows the computed current $I_{\text{sim}}$ for $\lambda = 30$. It is clear that the oscillations persist throughout the simulation period. For larger $b\lambda$ the results are worse still. If our aim is to have no appreciable oscillations past, say, $t = 0.1$, then we are restricted to a $b\lambda$ value of about unity, as also shown in the Figure. The goal then is to lift the restriction on $b\lambda$ by damping the oscillations. Fig. 4 shows the relative errors for $b\lambda = 3$, 10, and 30, respectively, for pure CN and superimposed results of using a single BI first step. Note the different scales required to contain the error envelopes as $b\lambda$ changes. We observe that an initial BI step works better for larger $b\lambda$, where the CN oscillations are worse. The large scale in Fig. 4(c) obscures the fact that, even though the first BI step reduces the errors markedly, the error is still unacceptably large at $t = 1$ (about \pm 1), so we conclude that in this case, the first BI step does not help sufficiently. For $b\lambda$ values of 100 or greater, a first BI step gives good results.

5.1.2. Pearson

Section 2 suggests that if $M$ is chosen such that $b\lambda/M$ falls below about 0.5, there should be no oscillations. Results indicate furthermore that this is somewhat conservative. So a suitable $M$ value might damp the oscillations, at least during the $M$ subintervals. It is not clear, a priori, what happens after that; in principle, the oscillations might resume. Fig. 5 shows the result of applying increasing numbers $M$ of Pearson subintervals for the rather large $b\lambda$ value, 100. As $M$ increases, the scale needed to contain the error decreases, until at $M = 30$ the error is down to an acceptable level, keeping within about \pm 0.003 over most of the simulation duration except for an initial settling-in period. One might have expected to have needed an $M$ value of 200 to achieve this ($b\lambda/M = 0.5$) but as was noted above, this is too conservative. We find that ratios $b\lambda/M$ equal to 1 or 2 give sufficient damping when $M$ is large.

By reference to Fig. 1, a suitable $M$-value for a given $b\lambda$ can be found, for a desired degree of damping.

The Pearson method, then, works rather well. An advantage of the method is, as we shall see below, that it is often computationally cheap (although more expensive than the BI method), requiring only relatively little

![Fig. 3. Simulated current $I_{\text{sim}}$ for a CN simulation of the Cottrell system, using parameters $\delta t = 0.01$ (i.e. 100 time steps in the range) and $\lambda = 30$ and (bold line) $\lambda = 1$. Current was calculated using a six-point approximation.](image1)

![Fig. 4. Relative error $e$ of the simulated current as in Fig. 3, using (a) $\lambda = 3$, (b) $\lambda = 10$ and (c) $\lambda = 30$. The larger amplitude, normal thickness lines are for a plain CN simulation, and the heavier lines (smaller amplitude) are for a BI first step followed by CN.](image2)
extra computing compared with exponentially increasing subintervals.

5.1.3. Exponentially increasing subintervals

One might nevertheless wish fewer subintervals within the first time interval, and exponentially increasing subintervals might be considered, and have been used, as mentioned above. Here, there are three variables, $M$ and $\beta$ as well as $b\lambda$. In previous work, interval doubling has been used ($\beta = 2$), Britz and Østerby (1994), Britz (1996) and Georganopoulou et al. (2000), but this turns out not always to be optimal. Fig. 2 can be used to find a suitable $\beta$ and $M$. The satisfactory regions for a desired damping are now concentrated in the lower right-hand region of the plots, especially for larger $b\lambda$ values.

5.1.4. Unequal spatial intervals

For the Cottrell experiment simulations using unequal spatial intervals, simulation results are shown in Fig. 6. The value of $\delta t$ is now 0.001. The expansion parameter $a$ has been chosen rather large at 30, and the number $N$ of space intervals at 40. Then $\delta y = 0.130$, making $b\lambda$, there equal to about 41. Fig. 6 shows the relative error for BI and for Pearson ($M = 20$). The oscillations do not tend to persist as long as with equal intervals. A first BI step has little effect on the response. The large gap in time before the apparent onset of error oscillations is due to the fact that these were too large at smaller times to fit within the scale of the plot, set so as to make the Pearson result more visible. This is for the ratio $b\lambda_1/M$ of about 2. In this Figure, with its small vertical scale, we note an
apparently persistent relative error at larger times. In fact, the absolute error does converge towards zero, but somewhat slowly. This phenomenon has been discussed with Rudolph (private communication, 2002). It appears that the point method, that is, simulation as implemented here, is limited to a smallest number \( N \) of spatial intervals of about 40 because of this slow error convergence, whereas the method used by the Feldberg group (Feldberg, 1969; Rudolph, 1995) is able to go down to \( N = 13 \) with much smaller errors. This needs to be clarified.

In a run of simulations with \( M = 10 \) with varying \( \beta \) values, it was found that \( \beta = 1.6 \) was about the optimal value, at which the oscillations were reduced but not eliminated. The \( \beta \) value is in reasonable agreement with that determined from Eq. (24), using \( b\lambda = 41 \), since it can be assumed that \( b\lambda \) at \( \delta y \) has the greatest effect (\( b \) being greatest there). It can be concluded that exponentially increasing subintervals are not worthwhile here, Pearson being more simply implemented and possibly even more effective.

5.2. The microdisk

With the microdisk simulations to be reported below, a fixed grid in \( (\Gamma, \theta) \) of \( 30 \times 30 \) was used, and a fixed time interval of 0.001 except where otherwise indicated. Britz (1996) previously reported a smooth response for this value of time interval, but this was achieved (but not reported) by starting with 10 increasing (doubling) subintervals.

Firstly, in order to have some comparison values, cpu-intensive simulations using an initial BI step were run with time intervals as short as \( 10^{-5} \). This yielded a convergent dimensionless current value of 2.246 at \( t = 0.1 \) and one of 1.364 at \( t = 1.0 \). These serve as standards of comparison for what follows. The value at \( t = 1 \) is close to that (1.367) provided by Aoki and Osteryoung (1984) and cited by Gavaghan (1998), and the difference may be due to the somewhat limited \( 30 \times 30 \) grid used in this study.

5.2.1. The BI method

Fig. 7 shows both the plain CN response and that of using BI as the first step. The BI step is seen to be very effective. The time scale in Fig. 7(a) shows only the first 0.1 units of time. Plain CN is grossly in error (an \( I_{\text{sim}} \) of 7.092) at \( t = 0.1 \) and still slightly inaccurate \( (I_{\text{sim}} = 1.366) \) and still oscillating at \( t = 1.0 \), while an initial BI step gives the (steady) values 2.236 and 1.364, respectively. Fig. 7(b), in which only the last 0.1 of the time scale is shown, also makes clear that plain CN has not stopped oscillating at \( t = 1 \), whereas simulation with the initial BI step has. This is encouraging. It seems that the BI method is more successful with the microdisk than for a one-dimensional system. On a \( 30 \times 30 \) grid, and considering Eq. (5), and \( \delta t = 0.001 \), \( \lambda_o \) is \( \delta t/\delta \theta^2 \), and the largest \( b \) factor is that for \( \theta = \delta \theta \) and \( \Gamma = \delta \Gamma \). Not surprisingly, this is the point closest to the disk edge, where the singularity is and from which oscillations are liable to spread. The value for \( b\lambda \) is equal to about 93.

Inspection of Table 1 indicates that BI gives a damping of about 0.003, consistent with our results.

5.2.2. Pearson

Firstly, in order to establish what to aim for, a run using plain CN was done with a smaller time interval \( \delta t = 10^{-4} \), and this showed small oscillations only, completely damped at \( t = 0.01 \). The current value at \( t = 0.1 \) is a satisfactory 2.246. This might lead us to expect that with the larger time step of 0.001, a Pearson subdivision of the first step into ten subintervals would damp the oscillations. This is not the case and in fact, \( M = 30 \) is required. These results are consistent with calculations based on Eq. (9).

5.2.3. Exponentially increasing subintervals

Feldberg and Goldstein (1995) and Svir and Golevenko (2001) have used exponentially increasing intervals over the whole simulation period. This was tried here also. It is possible to effect this by letting the first
time interval be the only interval, and to subdivide it suitably. We thus set $\delta t = 1$. The aim was to reduce $M$ to just 100, in order to possibly save cpu time. This leaves the setting of the expansion parameter $\beta$, which then also sets the smallest, first, interval. Now $h\lambda = 93000$, and Eq. (24) suggests a $\beta$ of about 1.1. Fig. 8 shows the sensitivity to $\beta$; a value of 1.07 fails to damp the oscillations, whereas the calculated value, 1.1, produces a satisfactory curve, ending, at $t = 1$ with 1.364 as it should for this grid.

There is a drawback here, however, also pointed out by Gavaghan (1998). A disk simulation using plain CN or one BI step followed by CN (as in Fig. 7) and 1000 steps each of 0.001, used 36 s of computer time. The above runs with various $\beta$ values and only 100 steps, used 52 s. The reason for this is that for a constant $\delta t$, the simulation requires one initial LU-decomposition, and only a back-substitution at every subsequent step. The LU-decomposition is that step requiring the most cpu time, so much of this is saved. With changing time intervals, however, a new LU decomposition must be computed at every step, and this increases computer time drastically. This would not matter in those simulation cases where the discretisation coefficients are themselves time-dependent, where equal time intervals also would necessitate recomputation of the LU-decomposed matrix at every step. Here, this was not so, and it seems that the use of exponentially increasing time intervals over the whole period is not a good idea. Note that the same argument applies to the more effective computation using sparse matrix techniques.

We turn now to exponentially increasing subintervals within the first time interval, followed by further steps. We revert to $\delta t = 0.001$, and simulations were carried forward by 100 whole steps, the first one subdivided. Choosing $M = 10$ and a range of $\beta$ gets better results with increasing $\beta$, but there is not sufficient damping. This is consistent with Fig. 2. With $M = 20$ good damping was seen for $1.2 < \beta < 1.6$, also in good accordance with Fig. 2. However, since the BI method is so effective here and uses less computing time, it is preferred.

6. Conclusions

For the Cottrell system, we conclude that both the Pearson method, and that of exponentially increasing subintervals, can effectively damp CN oscillations. The Pearson method usually requires a greater number $M$ of subintervals, but can on occasion be more cpu-efficient. It is also easier to implement. As a rule of thumb, an $M$ value such that $h\lambda/M$ is about unity, is sufficient (in fact, more than sufficient, especially at high $h\lambda$ values). The use of a single BI step is most effective in damping oscillations for this system at high $h\lambda$ values ($h\lambda \geq 100$).

With unequal spatial intervals for the one-dimensional system, it is noted that the oscillations are inherently damped better than for equal intervals, and that Pearson first-interval division is effective in damping them. Exponentially increasing intervals within the first step are also useful here, and a smaller $M$ value can be used than for Pearson, with a suitable $\beta$ factor found by using Eq. (24).

For microdisk simulations, a single BI step is an option, as it appears to be very effective in damping the oscillations of CN. Using subdivision of the first interval, equal intervals (Pearson) might be the safest option, but a rather large number of subintervals are required. An advantage here is that since the subintervals are all equal, a single LU-decomposition at the beginning of the series of subintervals is enough, followed by just one more when the full steps follow. Exponentially increasing subintervals are not attractive, partly because the results are not very good and partly because of the need for repeated LU-decomposition at every substep. Using exponentially increasing intervals over the whole simulation is possible and works, but with longer computing times in some cases.

Acknowledgements

The authors wish to thank the anonymous referee who pointed out several previous works to us, particularly those of Rannacher and coworkers.

References
