Improved Accuracy for Locally One-Dimensional Methods for Parabolic Equations

Jim Douglas, Jr.* and Seongjai Kim†

Abstract

Classical alternating direction (AD) and fractional step (FS) methods for parabolic equations, based on some standard implicit time stepping procedure such as Crank-Nicolson, can have errors associated with the AD or FS perturbations that are much larger than the errors associated with the underlying time stepping procedure. We show that minor modifications in the AD and FS procedures can virtually eliminate the perturbation errors at an additional computational cost that is less than ten per cent of the cost of the original AD or FS method. Moreover, after these modifications, the AD and FS procedures produce identical approximations of the solution of the differential problem. It is also shown that the same perturbation of the Crank-Nicolson procedure can be obtained with AD and FS methods associated with the backward Euler time stepping scheme. An application of the same concept is presented for second-order wave equations.

Key words. Alternating direction method, fractional step method, locally one-dimensional (LOD) method, Crank-Nicolson procedure, splitting error.

AMS subject classifications. 65M06, 65M12, 80M20.

1 Introduction

Over the last five decades, a number of time-stepping procedures have been introduced to allow multidimensional parabolic problems to be approximated numerically

*Department of Mathematics, Purdue University, W. Lafayette, Indiana 47097 USA Email: douglas@math.purdue.edu
†Department of Mathematics, University of Kentucky, Lexington, Kentucky 40506-0027 USA Email: skim@ms.uky.edu
through finite difference or finite element methods that treat the spatial variables individually in a cyclic fashion and we shall call any such procedure a *locally one-dimensional* (LOD) method. We shall be concerned with two families of these methods, namely the alternating direction (AD) methods first introduced in three papers [1, 3, 7] by Douglas, Peaceman, and Rachford and the fractional step (FS) procedures introduced by the Russian mathematicians D’yakonov, Marchuk, and Yanenko [5, 6, 9, 10]. These methods can be interpreted as perturbations of some underlying implicit multidimensional numerical method, such as the Crank-Nicolson or backward Euler method. In most of these LOD methods, the splitting error terms related to the perturbations are of the same or higher order in the time step $k$ as the truncation error terms associated with the underlying numerical methods. Thus, the asymptotic rates of convergence for an LOD method should be of the same order in the spatial and temporal discretization parameters as those for its associated underlying method. However, at practical levels of discretization, the actual errors associated with an LOD method can be much larger than that for the underlying method.

The first objective of this paper is to propose modifications to the right hand sides in the AD and FS algorithms based on Crank-Nicolson time differencing that reduce the splitting error from $O(k^2)$ to $O(k^3)$ for the AD methods and from $O(k)$ to $O(k^3)$ for the FS methods. In fact, after modification of the right hand sides, the AD and FS methods become identical, assuming that their solutions are initiated with the same values at the first two time levels, $t^0 = 0$ and $t^1 = k$.

The second objective will be to show that exactly the same perturbed the Crank-Nicolson procedure can be obtained from AD and FS methods based on a backward Euler time differencing by modifications of the right hand sides in the computational algorithms. This somewhat surprising observation can lead to coding simplifications in the implementation of the perturbed Crank-Nicolson method.

A rather large literature has grown around these LOD methods over the past decades; we shall not try to indicate references to papers not essential to our development.

An outline of the paper is as follows:

- In §2, we shall consider two examples approximated by the same AD procedure. In one example, the splitting error is essentially negligible, while in the other it totally dominates the error for the Crank-Nicolson procedure and, thereby, motivates trying to modify the AD procedure so as to reduce the splitting error. The difference in the results is due to the nature of the solutions of the differential problems.
- In §3, we reformulate the Douglas-Gunn [2] version of AD methods related
to Crank-Nicolson time discretization and identify the splitting perturbation terms.

- In §4, we derive a general form of Crank-Nicolson-based FS methods and again identify the splitting perturbation terms.

- In §5, we introduce correction terms to the right hand sides of AD and FS methods which reduce the splitting error of these methods to $O(t^3)$. For the improved AD method, a convergence proof is established, along with the numerical verification of improved accuracy and the preservation of the efficiency of the AD method.

- In §6, improved AD and FS methods based on backward Euler methods are presented; for appropriately interpreted time steps and correction terms, these first-order LOD methods turn out to be equivalent to the improved AD method based on Crank-Nicolson time stepping.

- In §7, some extensions to parabolic equations with more general coefficients and wave equations are treated.

- The last section includes conclusions.

2 Motivation

Let $\Omega = (a_x, b_x) \times (a_y, b_y) \subset \mathbb{R}^2$ with boundary $\Gamma = \partial \Omega$, and let $J = (0, T]$ be the time interval. Consider the parabolic problem

\begin{align}
  u_t - \nabla \cdot (a \nabla u) + cu &= f, \quad (x, t) \in \Omega \times J, \quad (2.1a) \\
  u_{\nu} + bu &= g, \quad (x, t) \in \Gamma \times J, \quad (2.1b) \\
  u &= u_0, \quad x \in \Omega, \quad t = 0, \quad (2.1c)
\end{align}

where the coefficients $a$, $c$, and $b$ are given functions, the subscript $\nu$ denotes the outer unit normal on $\Gamma$, $u_0$ is the prescribed initial value of the solution at $t = 0$, and $f$ and $g$ represent external sources and sinks.

Let

$$A_1 u = -(au_x)_x + \frac{1}{2} cu, \quad A_2 u = -(au_y)_y + \frac{1}{2} cu.$$ 

Then, (2.1a) can be rewritten as

$$u_t + A_1 u + A_2 u = f. \quad (2.2)$$
Let $J = (0, T]$ and $t^n = nk$, $k = T/n$. Also, let
\[ x_i = a_x + i h_x, \quad i = 0, 1, \ldots, n_x, \quad h_x = (b_x - a_x)/n_x, \]
\[ y_j = a_y + j h_y, \quad j = 0, 1, \ldots, n_y, \quad h_y = (b_y - a_y)/n_y. \]
(2.3)

Employ Crank-Nicolson time discretization and replace the operators $A_1$ and $A_2$ by central finite difference operators $A_{1h}$ and $A_{2h}$ to obtain the difference relation
\[ \frac{u^n - u^{n-1}}{k} + \frac{1}{2}(A_{1h} + A_{2h})(u^n + u^{n-1}) = f^{n-\frac{1}{2}} + O(h^2 + k^2), \]
(2.4)

where
\[ f^{n-\frac{1}{2}} = \frac{1}{2}(f^n + f^{n-1}). \]

Therefore, the basic truncation error associated with the Crank-Nicolson procedure is $O(h^2 + k^2)$.

### 2.1 The AD algorithm

The original AD method [1, 3, 7] for (2.1) is a perturbation of the Crank-Nicolson difference equation that has a splitting error of $O(k^2)$, so that it is second-order correct in both space and time. Let us formulate it in an equivalent way that will coincide with the general formulation [2] of AD methods employed below. Given an approximation $w^0$ to $u_0$, for $n \geq 1$, let
\[ \frac{w^{n,1} - w^{n-1}}{k} + \frac{1}{2}A_{1h}(w^{n,1} + w^{n-1}) + A_{2h}w^{n-1} = f^{n-\frac{1}{2}}, \quad (2.5a) \]
\[ \frac{w^n - w^{n-1}}{k} + \frac{1}{2}A_{1h}(w^{n,1} + w^{n-1}) + A_{2h}(w^n + w^{n-1}) = f^{n-\frac{1}{2}}. \quad (2.5b) \]

Equivalently,
\[ (1 + \frac{k}{2}A_{2h})w^{n,1} = (1 - \frac{k}{2}A_{1h} - kA_{2h})w^{n-1} + kf^{n-\frac{1}{2}}, \quad (2.6a) \]
\[ (1 + \frac{k}{2}A_{2h})w^n = w^{n,1} + \frac{k}{2}A_{2h}w^{n-1}. \quad (2.6b) \]

The intermediate value $w^{n,1}$ can be found (implicitly) as
\[ w^{n,1} = w^n + \frac{k}{2}A_{2h}(w^n - w^{n-1}), \quad (2.7) \]
so that
\[ (1 + \frac{k}{2}A_{1h})(1 + \frac{k}{2}A_{2h})w^n = (1 - \frac{k}{2}A_{1h})w^{n-1} + kf^{n-\frac{1}{2}}, \quad (2.8) \]
where $A_h = A_{1h} + A_{2h}$. Multiply out the left hand side and rewrite the result as
\[
\frac{w^n - w^{n-1}}{k} + \frac{1}{2} A_h(w^n + w^{n-1}) + \frac{k}{4} A_{1h} A_{2h} (w^n - w^{n-1}) = f^{n-\frac{1}{2}}.
\tag{2.9}
\]
Thus, the splitting term is given by
\[
\frac{k}{4} A_{1h} A_{2h} (w^n - w^{n-1});
\tag{2.10}
\]
evaluated on the solution $u$ of (2.1), this term is $O(k^2)$ for a smooth solution, as claimed above.

Some theoretical aspects of the method were treated in detail in [1], while some practical aspects of the method were considered in the companion paper [7]. In each half of the calculation, the matrix to be inverted is tridiagonal, so that the algorithm requires $O(N = n_t n_x n_y)$ flops.

### 2.2 Accuracy of the AD algorithm: Two examples

Let us consider two examples. Let $\Omega \times J = (0,1)^2 \times (0,1)$, $a \equiv 1$, and $c = b \equiv 0$ in (2.1). Consider two different solutions:

\[
\begin{align*}
  u_+ &= \sin(2\pi \nu t) + \sin(2\pi \nu_x x) + \sin(2\pi \nu_y y), \quad (2.11a) \\
  u_\infty &= \sin(2\pi \nu t) \cdot \sin(2\pi \nu_x x) \cdot \sin(2\pi \nu_y y). \quad (2.11b)
\end{align*}
\]

For the moment, take $\nu_t = \nu_x = \nu_y = 1$. The sources $f$ and $g$ are evaluated so that (2.1) is satisfied. Also, let $n := n_t = n_x = n_y$. To compare computation cost and accuracy, we implemented three algorithms: an LU-based algorithm and a PCG-ILU0 procedure for the Crank-Nicolson equation derivable from (2.4) and the AD procedure of (2.9). Here, PCG-ILU0 denotes the conjugate gradient method preconditioned by the zero-level (not allowing fill-in) incomplete LU-factorization. The PCG-ILU0 procedure was initialized at each time level by the extrapolation
\[
u^{n,0} = 2u^{n-1} - u^{n-2}, \quad n \geq 2,
\]
and the iteration stopped when the residual was reduced by a factor of $10^{-5}$. The main/driver routines are written in C++ and C, and the core routines in F77. Each computation was carried out on a 400 MHz laptop having 196M memory and a Linux operating system.

Table 1 presents the elapsed times and numerical errors for $u_+$ for various grid sizes. One can see from the table that the three different algorithms show the same errors and their second-order convergence. The elapsed time (CPU) is measured
<table>
<thead>
<tr>
<th></th>
<th>$n = 40$</th>
<th>$n = 80$</th>
<th>$n = 160$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU $L^2$-error</td>
<td>CPU $L^2$-error</td>
<td>CPU $L^2$-error</td>
</tr>
<tr>
<td><strong>$LU$-based</strong></td>
<td>0.74 4.10e-3</td>
<td>9.07 1.00e-3</td>
<td>126 2.47e-4</td>
</tr>
<tr>
<td><strong>$PCG$-$ILU_0$</strong></td>
<td>0.46 4.11e-3</td>
<td>5.67 1.00e-3</td>
<td>53.4 2.47e-4</td>
</tr>
<tr>
<td><strong>$AD$</strong></td>
<td>0.26 4.10e-3</td>
<td>2.16 1.00e-3</td>
<td>17.9 2.47e-4</td>
</tr>
</tbody>
</table>

Table 1: The performances of the $LU$-based, $PCG$-$ILU_0$, and $AD$ methods for $u = u_+$. For the computation, $a = 1$, $c = b = 0$, and $\nu_t = \nu_x = \nu_y = 1$; $n := n_t = n_x = n_y$.

<table>
<thead>
<tr>
<th></th>
<th>$n = 40$</th>
<th>$n = 80$</th>
<th>$n = 160$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU $L^2$-error</td>
<td>CPU $L^2$-error</td>
<td>CPU $L^2$-error</td>
</tr>
<tr>
<td><strong>$LU$-based</strong></td>
<td>0.91 2.46e-4</td>
<td>10.5 5.98e-5</td>
<td>136 1.47e-5</td>
</tr>
<tr>
<td><strong>$PCG$-$ILU_0$</strong></td>
<td>0.83 2.46e-4</td>
<td>12.5 5.97e-5</td>
<td>121 1.42e-5</td>
</tr>
<tr>
<td><strong>$AD$</strong></td>
<td>0.45 8.44e-3</td>
<td>3.62 2.02e-3</td>
<td>29.0 4.90e-4</td>
</tr>
</tbody>
</table>

Table 2: The performances of the $LU$-based, $PCG$-$ILU_0$, and $AD$ methods for $u = u_-$. The coefficients and algorithm parameters are the same as in Table 1.

in seconds and the $L^2$-norm of the error is evaluated at $t = 1$. An average of 7-9 iterations per time step were required for the $PCG$-$ILU_0$ iteration to converge.

In Table 2, we present the results for $u = u_-$. The computation cost for the $AD$ method increases linearly as the number of grid points grows, while the $PCG$-$ILU_0$ calculation shows a slight superlinearly in its computation cost, as one can expect. Here, 12 to 22 iterations were required for the $PCG$-$ILU_0$ iteration to converge, with the iteration counts becoming larger for more grid points. In the table, the algorithms show second-order accuracy. However, the $AD$ method produces an error approximately 34 times larger than that for the $LU$-based or $PCG$-$ILU_0$ methods for the same grid size. Note the errors for $n = 40$ for the $LU$-based/$PCG$-$ILU_0$ calculations and $n = 160$ for the $AD$ calculation. The number of grid points for $AD$ is 64 times larger than that for $PCG$-$ILU_0$, but the error is still twice as large. In summary, the $AD$ method can produce a large perturbation error that degrades the effectiveness of the algorithm.

The local truncation error for the Crank-Nicolson difference equation is of the form
\[
\mathcal{O}\left( h_x^2 \frac{\partial^1 u}{\partial x^1} \right) + \mathcal{O}\left( h_y^2 \frac{\partial^1 u}{\partial y^1} \right) + \mathcal{O}\left( k^2 \frac{\partial^3 u}{\partial t^3} \right),
\]
while the splitting error of the $AD$ method is
\[
\mathcal{O}\left( k^2 \frac{\partial^2 u}{\partial x^2 \partial y^2 \partial t} \right).
\]
That is, roughly speaking, why the AD method introduces no splitting error for $u_+$ and a large splitting error for $u_-$.

## 3 The General AD Procedure

Consider a parabolic problem of the form

\[
\frac{\partial u}{\partial t} + \sum_{i=1}^{m} A_i u = f, \quad x \in \Omega, \ 0 < t \leq T, \quad (3.1a)
\]

\[
u = 0, \quad x \in \partial \Omega, \ 0 < t \leq T, \quad (3.1b)
\]

\[
u = u_0, \quad x \in \Omega, \ t = 0. \quad (3.1c)
\]

If $A = A_1 + \cdots + A_m$, then the basic Crank-Nicolson approximation to (3.1a) is given by

\[
\frac{w^n - w^{n-1}}{k} + \frac{1}{2} A(w^n + w^{n-1}) = f^{n-\frac{1}{2}}, \quad n \geq 1. \quad (3.2)
\]

(Here, we are interested in the time discretization of (3.1); consequently, we shall ignore spatial discretization for the moment.) The Douglas-Gunn algorithm [2] for AD time discretization of (3.1) is as follows:

For $\kappa = 1, \ldots, m$, find $w^{n,\kappa}$ such that

\[
\frac{w^{n,\kappa} - w^{n-1}}{k} + \frac{1}{2} \sum_{i=1}^{\kappa} A_i(w^{n,\kappa} + w^{n-1}) + \sum_{i=\kappa+1}^{m} A_i w^{n-1} = f^{n-\frac{1}{2}}, \quad (3.3)
\]

and then to set

\[
w^n = w^{n,m}. \quad (3.4)
\]

In the above,

\[
\sum_{m+1}^{m} A_i w^{n-1} := 0.
\]

This algorithm is equivalent to

\[
\left(1 + \frac{k}{2} A_1\right) w^{n,1} = \left(1 - \frac{k}{2} A_1 - k \sum_{i=2}^{m} A_i \right) w^{n-1} + k f^{n-\frac{1}{2}}, \quad (3.5a)
\]

\[
\left(1 + \frac{k}{2} A_\kappa\right) w^{n,\kappa} = w^{n,\kappa-1} + \frac{k}{2} A_\kappa w^{n-1}, \quad \kappa = 2, \ldots, m, \quad (3.5b)
\]

\[
w^n = w^{n,m}. \quad (3.5c)
\]
The intermediate values $w^{n,1}, \ldots, w^{n,m-1}$ can be eliminated by recursively operating on (3.5b) by $1 + \frac{k}{2}A_\kappa$ for $\kappa = m - 1, \ldots, 1$. The resulting equation can be written in the form

$$\frac{w^n - w^{n-1}}{k} + \frac{1}{2}A(w^n + w^{n-1}) + B_k(w^n - w^{n-1}) = f^{n-\frac{1}{2}}, \quad (3.6)$$

where

$$B_k = \frac{k}{4} \sum_{1 \leq i_1 < i_2 \leq m} A_{i_1}A_{i_2} + \frac{k^2}{8} \sum_{1 \leq i_1 < i_2 < i_3 \leq m} A_{i_1}A_{i_2}A_{i_3} + \cdots + \frac{k^{m-1}}{2^m} A_1 A_2 \cdots A_m. \quad (3.7)$$

The splitting perturbation is given by $B_k(w^n - w^{n-1})$, and for sufficiently smooth solutions $u$ of the differential equation (3.1),

$$B_k(u^n - u^{n-1}) = O(k^2), \quad (3.8)$$

so that the splitting error is of the same order in $k$ as the basic Crank-Nicolson local truncation error. Now, since the operators $A_i$ usually represent second-order differential operators in an $x_i$ direction, it should not be surprising that the higher-order derivatives in $B_k$ contribute bigger local errors than were present in the Crank-Nicolson error. We shall see in §5 that it is not only possible but also quite feasible to modify the algorithm (3.3) in a rather simple fashion to raise the splitting error to $O(k^3)$.

## 4 The General FS Procedure

We shall consider the same parabolic problem (3.1) for a FS time discretization. For reasons that will appear below, it is not the usual case to look for an FS procedure based on the Crank-Nicolson equation (3.2); however, it is useful for us to do so. The appropriate FS algorithm is given by

$$\frac{w^{n,1} - w^{n-1}}{k} + \frac{1}{2}A_1(w^{n,1} + w^{n-1}) = f^{n-\frac{1}{2}}, \quad (4.1a)$$

$$\frac{w^{n,\kappa} - w^{n,\kappa-1}}{k} + \frac{1}{2}A_\kappa(w^{n,\kappa} + w^{n-1}) = 0, \quad \kappa = 2, \ldots, m - 1, \quad (4.1b)$$

$$\frac{w^n - w^{n,m-1}}{k} + \frac{1}{2}A_m(w^n + w^{n-1}) = 0. \quad (4.1c)$$
Equivalently,

\[
\left(1 + \frac{k}{2} A_1 \right) w^{n,1} = \left(1 - \frac{k}{2} A_1 \right) w^{n-1} + k f^{n-\frac{1}{2}}, \quad (4.2a)
\]

\[
\left(1 + \frac{k}{2} A_\kappa \right) w^{n,\kappa} = w^{n,\kappa-1} - \frac{k}{2} A_\kappa w^{n-1}, \quad \kappa = 2, \ldots, m - 1, \quad (4.2b)
\]

\[
\left(1 + \frac{k}{2} A_m \right) w^n = w^{n,m-1} - \frac{k}{2} A_m w^{n-1}. \quad (4.2c)
\]

Again, the intermediate values can be eliminated, with the result being the equation

\[
\frac{w^n - w^{n-1}}{k} + \frac{1}{2} A(w^n + w^{n-1}) + B_k(w^n + w^{n-1}) = f^{n-\frac{1}{2}}, \quad (4.3)
\]

with \(B_k\) being the same as for the \(AD\) equation (2.10).

Thus, for the Crank-Nicolson version of the \(FS\) method, the splitting perturbation term becomes \(B_k(w^n + w^{n-1})\). Now,

\[
B_k(w^n + w^{n-1}) = O(k); \quad (4.4)
\]

i.e., the splitting error term is worse than the inherent local error in the Crank-Nicolson equation. (This is the reason that (4.1) is not usually employed.) However, we shall be able to modify the procedure in an equally simple fashion to reduce the splitting error to \(O(k^3)\) below.

## 5 Improved Accuracy for \(LOD\) Procedures

We present a strategy to reduce the perturbation error of \(AD\) and \(FS\) procedures and essentially to recover the accuracy of the Crank-Nicolson difference equation for an additional computational cost that is a small fraction of the standard \(AD\) or \(FS\) cost.

### 5.1 Correction term for the \(AD\) method

Recall from (3.6) and (3.7) that, if the right hand side term in the \(AD\) algorithm (3.3) is \(f^{n-\frac{1}{2}}\), the right hand side of (3.6) is also \(f^{n-\frac{1}{2}}\) and that the splitting perturbation is given by \(B_k(w^n - w^{n-1})\). If we could add \(B_k(w^n - w^{n-1})\) to the right hand side, then we could cancel the perturbation term completely; but since we do not know \(w^n\), we cannot make this modification in the algorithm. Our best estimate of \(w^n - w^{n-1}\) is \(w^{n-1} - w^{n-2}\) and let us modify the \(AD\) algorithm to the following:
For $n \geq 2$,
\begin{align}
F^n_{AD} &= f^{n-\frac{1}{2}} + B_k(z^{n-1} - z^{n-2}), \\
(1 + \frac{k}{2}A_1)z^{n,1} &= \left(1 - \frac{k}{2}A_1 - k\sum_{i=2}^{m} A_i\right)z^{n-1} + kF^n_{AD}, \\
(1 + \frac{k}{2}A_\kappa)z^{n,\kappa} &= z^{n,\kappa-1} + \frac{k}{2}A_\kappa z^{n-1}, \quad \kappa = 2, \ldots, m, \\
z^n &= z^{n,m}.
\end{align}
(5.1.a, 5.1.b, 5.1.c, 5.1.d)

The evaluation of $z^1$ will be discussed below by interpreting the modified method as an iterative procedure; for practical purposes, assume that $z^1$ is obtained by solving the Crank-Nicolson equation for this single time step.

By eliminating the intermediate values (or referring to (3.6)), we see that $z^n$ satisfies the equation
\begin{equation}
\frac{z^n - z^{n-1}}{k} + \frac{1}{2}A(z^n + z^{n-1}) + B_k(z^n - 2z^{n-1} + z^{n-2}) = f^{n-\frac{1}{2}}
\end{equation}
for $n \geq 2$. Now, for a smooth solution $u$ of (2.1),
\begin{equation}
B_k(u^n - 2u^{n-1} + u^{n-2}) = O(k^3),
\end{equation}
and the local splitting error is now higher order in $k$ than the local error for the Crank-Nicolson equation and than the local splitting error in the $AD$ procedure (3.3). We shall both prove the convergence of the solution of (5.1) to that of (2.1) under certain circumstances and demonstrate that the error in the solution of (5.1) is reduced essentially to that of the Crank-Nicolson procedure for the example $u_x$ considered above, for which the splitting error was many times as large as the Crank-Nicolson error.

Now, let us interpret (5.2) as the iterative procedure related to the matrix splitting [8]
\[ 1 + \frac{k}{2}A = (1 + \frac{k}{2}A + B_k) - B_k. \]
Consider the algorithm: Find $\zeta^\ell$, $\ell \geq 1$, by recursively solving
\begin{equation}
(1 + \frac{k}{2}A + B_k)\zeta^\ell = B_k\zeta^{\ell-1} + (1 - \frac{k}{2}A)\gamma + f^{n-\frac{1}{2}}.
\end{equation}
(5.4)
The solution $w^n$ of the original $AD$ method (3.3) is the first iterate $\zeta^1$ of (5.4) for $
\gamma = w^{n-1}$ starting with the initial value
\begin{equation}
\zeta^0 = w^{n-1}.
\end{equation}
(5.5)
On the other hand, the solution \( z^n \) of (5.1) is the first iterate of (5.4) with \( \gamma = z^{n-1} \) and the initial value

\[
\zeta^0 = 2z^{n-1} - z^{n-2}.
\] (5.6)

Consequently, the algorithm (5.1) will be called the alternating direction method with improved initialization (AD-II).

If the general time step code for (5.1) is written to perform the iteration (5.4), then, for \( n \geq 2 \), (5.6) would be used to initialize the “iteration” and one step of iteration calculated, while for \( n = 1 \), (5.5) would be used to initialize the iteration and two or more iterations would give \( z^1 \) to the desired accuracy.

It might seem reasonable to use a higher-order extrapolation than (5.6), but experiments have shown that instability can result unless the time step is small enough that the higher-order extrapolation is irrelevant. It has also been observed that (5.6) can over-correct for large time steps, and it is possible that the use of

\[
\zeta^0 = z^{n-1} + \eta(z^{n-1} - z^{n-2}), \quad 0 \leq \eta \leq 1,
\] (5.7)

could lead to better computational results for large time steps. However, experiments have shown that, when the time step is reasonably chosen (e.g., \( k \leq h \) when the coefficient of \( u_t \) is one), AD-II methods have worked better than AD methods for various heterogeneous media; see Tables 3 and 4 in §5.4. So, (5.7) does not seem necessary for solving heat equations in practice.

### 5.2 Correction term for the FS method

The FS difference equation (4.3) preserves the right hand side of the FS algorithm (4.2) and exhibits the splitting perturbation \( B_k(w^n + w^{n-1}) \). Modify (4.2) as follows. For \( n \geq 2 \), let

\[
F^n_{FS} = f^{n-\frac{1}{2}} + B_k(3z^{n-1} - z^{n-2}),
\] (5.8a)

\[
\left(1 + \frac{k}{2}A_1\right)z^{n,1} = \left(1 - \frac{k}{2}A_1\right)z^{n-1} + kF^n_{FS},
\] (5.8b)

\[
\left(1 + \frac{k}{2}A_{\kappa}\right)z^{n,\kappa} = z^{n,\kappa-1} - \frac{k}{2}A_{\kappa}z^{n-1}, \quad \kappa = 2, \ldots, m - 1,
\] (5.8c)

\[
\left(1 + \frac{k}{2}A_m\right)z^n = z^{n,m-1} - \frac{k}{2}A_mz^{n-1}.
\] (5.8d)

After the intermediate values are eliminated, we see that \( z^n \) satisfies the equation

\[
\frac{z^n - z^{n-1}}{k} + \frac{1}{2}A(z^n + z^{n-1}) + B_k(z^n - 2z^{n-1} + z^{n-2}) = f^{n-\frac{1}{2}},
\] (5.9)
which is identical to the equation (5.2) satisfied by the solution of the AD-III algorithm (5.1). Thus, we have not only shown how to reduce the splitting perturbations for the AD and FS methods but also to discover that their improved procedures lead to identical results (after several decades of being considered to be different techniques). Again, it is advisable to obtain $z^1$ as discussed in the previous subsection.

If the values of $A_i z^{n-1}$ are saved, then there is essentially no difference in the implementation of algorithms (5.1) and (5.8). That being the case, we shall address both algorithms as pertaining to the AD-III method.

### 5.3 A convergence proof for the AD-III method

Let $\| \cdot \|$ denote the $L^2(\Omega)$ or $L^p(\Omega)$ norm and $\| \cdot \|$ the norm on either $H^1(\Omega)$ or $h(\Omega)$, as appropriate. (That is, depending on spatial discretization by finite elements or finite differences.) Assume that the operators $\{A_i\}$ commute:

$$A_i A_j = A_j A_i, \quad i, j = 1, \ldots, m, \quad (5.10)$$

and that

$$(A_i z, z) \geq \alpha \| z \|^2, \quad \alpha > 0. \quad (5.11)$$

By (5.10) and (5.11), it follows that

$$(B_k z, z) \geq 0.$$

Let $\partial_t v^n = (v^n - v^{n-1})/k$ and $e^n = u^n - z^n$. Then, the error equation associated with (5.2) is

$$\partial_t e^n + \frac{1}{2} A(e^n + e^{n-1}) + B_k(e^n - 2e^{n-1} + e^{n-2}) = \delta^n, \quad (5.12)$$

where

$$\delta^n = O(k^2 + h^p), \quad p \geq 2, \quad (5.13)$$

for any reasonable spatial discretization. Choose $\partial_t e^n$ as a test function. Then, for $n \geq 2$,

$$(\partial_t e^n, \partial_t e^n) + \frac{1}{2} (A(e^n + e^{n-1}), \partial_t e^n) + k^2 (B_k \partial_t^2 e^n, \partial_t e^n) = (\delta^n, \partial_t e^n). \quad (5.14)$$

Multiply (5.14) by $k$ and sum beginning at $n = 2$ to have

$$\sum_{j=2}^{n} \| \partial_t e^j \|^2 k + \frac{1}{2} (A e^n, e^n) + k^2 \sum_{j=2}^{n} (B_k \partial_t^2 e^j, \partial_t e^j) k$$

$$= \frac{1}{2} (A e^1, e^1) + \sum_{j=2}^{n} (\delta^j, \partial_t e^j) k. \quad (5.15)$$
Now, since
\[
(B_k \partial_t e^j, \partial_t e^j) + (B_k \partial_t e^{j-1}, \partial_t e^{j-1}) \geq 2(B_k \partial_t e^{j-1}, \partial_t e^j), \quad j = 2, \ldots, n,
\]
we have
\[
\sum_{j=2}^{n} (B_k \partial_t^2 e^j, \partial_t e^j) k = \sum_{j=2}^{n} (B_k [\partial_t e^j - \partial_t e^{j-1}], \partial_t e^j) \geq \frac{1}{2} (B_k \partial_t e^n, \partial_t e^n) - \frac{1}{2} (B_k \partial_t e^1, \partial_t e^1).
\] (5.16)

Apply the Cauchy-Schwarz inequality to the sum involving $\delta^j$ in (5.15). Then utilizing (5.16), one can obtain the following inequality:
\[
\sum_{j=2}^{n} \| \partial_t e^j \|^2 k + (Ae^n, e^n) + k^2 (B_k \partial_t e^n, \partial_t e^n) \leq \sum_{j=2}^{n} \| \delta^j \|^2 k + (Ae^1, e^1) + k^2 (B_k \partial_t e^1, \partial_t e^1), \quad n \geq 2.
\] (5.17)

Thus, the estimation of the error generated by the AD-II method is, in the commutative case, reduced to bounding the errors in $z^0$ and $z^1$, thereby emphasizing the remarks above on the evaluation of $z^1$.

5.4 Accuracy and efficiency of the AD-II method

To check the accuracy and efficiency of the AD-II algorithm, let us choose the domain $\Omega = (0, 1)^2$ and the time interval $J = (0, 1]$, along with the four diffusion coefficients
\[
a_1(x, y) = 1, \quad (5.18a)
\]
\[
a_2(x, y) = 1/(2 + \cos(3\pi x) \cdot \cos(2\pi y)), \quad (5.18b)
\]
\[
a_3(x, y) = \begin{cases} 1 + 0.5 \cdot \sin(5\pi x) + 2y^3, & \text{if } x \leq 0.5, \\ 1.5/(1 + (x - 0.5)^2) + 2y^3, & \text{else}, \end{cases} \quad (5.18c)
\]
\[
a_4(x, y) = \begin{bmatrix} a_2(x, y) & 0 \\ 0 & a_3(x, y) \end{bmatrix}. \quad (5.18d)
\]

The first time step to obtain $z^1$ for the AD-II calculation was made by following the $w^1$-AD calculation by SOR iterations to get the Crank-Nicolson value. We compare the results of four different algorithms, namely the LU-based, PCG-ILU0, AD and AD-II methods. We did not implement the FS method, since its splitting error is much larger than those for the AD and AD-II methods.
Table 3: The performances of the $LU$-based, $PCG-ILU0$, $AD$ and $AD-II$ methods with $c = b \equiv 0$, $\nu_t = 1$, $\nu_x = 4$, $\nu_y = 3$, $n_x = n_y = n_t = 100$ for $u = u_\infty$.

<table>
<thead>
<tr>
<th>Method</th>
<th>$a = a_1$</th>
<th>$a = a_2$</th>
<th>$a = a_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU</td>
<td>$L^2$-error</td>
<td>CPU</td>
</tr>
<tr>
<td>$LU$-based</td>
<td>23.6</td>
<td>1.10e-3</td>
<td>27.2</td>
</tr>
<tr>
<td>$PCG-ILU0$</td>
<td>21.6</td>
<td>1.09e-3</td>
<td>24.0</td>
</tr>
<tr>
<td>$AD$</td>
<td>7.14</td>
<td>1.70e-2</td>
<td>10.9</td>
</tr>
<tr>
<td>$AD-II$</td>
<td>7.77</td>
<td>1.10e-3</td>
<td>11.3</td>
</tr>
</tbody>
</table>

Table 4: The performances of the $LU$-based, $PCG-ILU0$, $AD$ and $AD-II$ methods with $a = a_4$, $c = b \equiv 0$, $\nu_t = 2.0$, $\nu_x = 6.25$, $\nu_y = 7$, $h = h_x = h_y = 1/120$, and $u = u_\infty$.

<table>
<thead>
<tr>
<th></th>
<th>$k = 2h$</th>
<th>$k = h$</th>
<th>$k = h/2$</th>
<th>$k = h/4$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU</td>
<td>$L^2$-error</td>
<td>CPU</td>
<td>$L^2$-error</td>
</tr>
<tr>
<td>$LU$-based</td>
<td>28.4</td>
<td>2.12e-3</td>
<td>49.6</td>
<td>2.13e-3</td>
</tr>
<tr>
<td>$PCG-ILU0$</td>
<td>24.9</td>
<td>2.14e-3</td>
<td>36.5</td>
<td>2.15e-3</td>
</tr>
<tr>
<td>$AD$</td>
<td>8.19</td>
<td>2.01e-3</td>
<td>16.3</td>
<td>6.76e-2</td>
</tr>
<tr>
<td>$AD-II$</td>
<td>8.80</td>
<td>1.10e-2</td>
<td>16.9</td>
<td>2.17e-3</td>
</tr>
</tbody>
</table>

Table 3 presents the performances of the four algorithms for the first three diffusion coefficients in (5.18) for $u = u_\infty$ with $\nu_t = 1$, $\nu_x = 4$, and $\nu_y = 3$. The error for the $AD$ method is 16, 3, and 5 times larger than the Crank-Nicolson error for $a = a_1$, $a_2$, and $a_3$, respectively. The $AD-II$ method requires only about 5-7% extra cost over the $AD$ method and its accuracy hardly differs from that of the direct, $LU$-based solver.

Table 4 shows numerical results for various time steps, when $a = a_4$ (an anisotropic diffusivity), $c = b \equiv 0$, $\nu_t = 2$, $\nu_x = 6.25$, and $\nu_y = 7$, and $h = h_x = h_y = 1/120$. The $AD$ calculations show large splitting errors, even for small time steps. Here again the improved initialization (5.6) greatly improves the accuracy of the alternating direction procedure, for a few percent of extra cost. However, as one can see from the table, the $AD-II$ algorithm generates a splitting error that is about four times the Crank-Nicolson error for $k = 2h$. To improve its performance, one may introduce a few SOR iterations following the $AD-II$ sweeps. For $k = 2h$, the combination of the $AD-II$ sweeps and four symmetric SOR iterations recovers the Crank-Nicolson accuracy; this calculation took 11.0 seconds, which is about one quarter more than the pure $AD-II$ calculation.
6 Deriving the AD-II Method from Backward Euler AD and FS Methods

The generalized Douglas-Rachford AD method [4] can be formulated for a time step \( \tau \) as follows:

\[
\frac{w^{n,k} - w^{n-1}}{\tau} + \sum_{i=1}^{k} A_i w^{n,i} + \sum_{i=k+1}^{m} A_i w^{n-1} = \phi^n, \quad k = 1, \ldots, m, \tag{6.1}
\]

or, equivalently,

\[
(1 + \tau A_1) w^{n,1} = \left(1 - \tau \sum_{i=2}^{m} A_i w^{n-1}\right) + \tau \phi^n, \quad \tag{6.2a}
\]

\[
(1 + \tau A_k) w^{n,k} = w^{n,k-1} + \tau A_k w^{n-1}, \quad k = 2, \ldots, m, \tag{6.2b}
\]

\[
w^n = w^{n,m}. \tag{6.2c}
\]

By elimination of the intermediates, we arrive at the equation

\[
\frac{w^n - w^{n-1}}{\tau} + A w^n + C_\tau (w^n - w^{n-1}) = \phi^n, \tag{6.3}
\]

where

\[
C_\tau = \tau \sum_{1 \leq i_1 < i_2 \leq m} A_{i_1} A_{i_2} + \tau^2 \sum_{1 \leq i_1 < i_2 < i_3 \leq m} A_{i_1} A_{i_2} A_{i_3} + \cdots + \tau^{m-1} A_1 A_2 \cdots A_m. \tag{6.4}
\]

Now, take \( \tau = k/2 \) and note that \( C_{\frac{k}{2}} = 2B_k \). Hence, (6.3) would become

\[
\frac{w^n - w^{n-1}}{k} + \frac{1}{2} A w^n + B_k (w^n - w^{n-1}) = \frac{1}{2} \phi^n.
\]

Let

\[
\phi^n = 2 f^{n-\frac{1}{2}} - A w^{n-1} + 2 B_k (w^{n-1} - w^{n-2}); \tag{6.5}
\]

then, the algorithm (6.1) leads to

\[
\frac{w^n - w^{n-1}}{k} + \frac{1}{2} A (w^n + w^{n-1}) + B_k (w^n - 2w^{n-1} + w^{n-2}) = f^{n-\frac{1}{2}}, \tag{6.6}
\]

and the AD-II method is reproduced.

Next, consider the standard FS method, which is based on a backward Euler approach. The algorithm is given by

\[
\frac{w^{n,1} - w^{n-1}}{\tau} + A_1 w^{n,1} = \phi, \quad \tag{6.7a}
\]

\[
\frac{w^{n,k} - w^{n,k-1}}{\tau} + A_k w^{n,k} = 0, \quad k = 1, \ldots, m, \tag{6.7b}
\]

\[
w^n = w^{n,m}. \tag{6.7c}
\]
Thus,
\[ (1 + \tau A_1)(1 + \tau A_2) \cdots (1 + \tau A_m)w^n = w^{n-1} + \tau \phi, \]  
(6.8)

or
\[ \frac{w^n - w^{n-1}}{\tau} + A w^n + C \tau w^n = \phi. \]  
(6.9)

Thus, taking
\[ \tau = \frac{k}{2} \quad \text{and} \quad \phi = 2 f^{n-\frac{1}{2}} - A w^{n-1} + 2 B_k (2 w^{n-1} - w^{n-2}) \]  
(6.10)

leads to exactly the same modified Crank-Nicolson equation (6.6) as we have found three times before. From the point of view of implementation, the algorithm (6.7) with \( \tau \) and \( \phi \) given by (6.10) is perhaps the simplest to code and slightly more efficient in the number of arithmetic operations to carry out.

7 Extensions

In this section, we extend arguments in Sections 5 and 6 to parabolic equations with more general coefficients and to second-order wave equations.

7.1 Parabolic equations with general coefficients

Consider the differential problem associated with the equation
\[ c \frac{\partial u}{\partial t} + \sum_{i=1}^{m} A_i u = f, \]  
(7.1)

where the function \( c \) is specified, as well as the operators \( A_i \). All four of the \( AD \) and \( FS \) methods can be extended to treat (7.1) in such a fashion as to arrive at an \( \mathcal{O}(k^3) \) perturbation of the Crank-Nicolson method. Let us discuss only the backward Euler based \( FS \) procedure, for which the \( FS \) algorithm can be written (with nominal time step and right hand side given by \( \tau \) and \( \phi \), respectively) as

\[ \frac{c w^{n,1} - w^{n-1}}{\tau} + A_1 w^{n,1} = \phi, \]  
(7.2a)

\[ \frac{c w^{n,\kappa} - w^{n,\kappa-1}}{\tau} + A_\kappa w^{n,\kappa} = 0, \quad \kappa = 2, \ldots, m, \]  
(7.2b)

\[ w^n = w^{n,m}. \]  
(7.2c)
Equivalently,
\[
\left(1 + \frac{\tau}{c} A_1\right) w^{n,1} = w^{n-1} + \frac{\tau}{c} \phi, \tag{7.3a}
\]
\[
\left(1 + \frac{\tau}{c} A_k\right) w^{n,k} = w^{n,k-1}, \tag{7.3b}
\]
and
\[
\left(1 + \frac{\tau}{c} A_1\right) \left(1 + \frac{\tau}{c} A_2\right) \cdots \left(1 + \frac{\tau}{c} A_m\right) w^n = w^{n-1} + \frac{\tau}{c} \phi. \tag{7.4}
\]
Thus,
\[
c \frac{w^n - w^{n-1}}{\tau} + A w^n + C_r w^n = \phi, \tag{7.5}
\]
where
\[
C_r = \frac{\tau}{c} \sum_{1 \leq i_1 < i_2 \leq m} A_{i_1} \frac{1}{c} A_{i_2} + \left(\frac{\tau}{c}\right)^2 \sum_{1 \leq i_1 < i_2 < i_3 \leq m} A_{i_1} \frac{1}{c} A_{i_2} \frac{1}{c} A_{i_3}
+ \cdots + \left(\frac{\tau}{c}\right)^{m-1} A_1 \frac{1}{c} A_2 \cdots \frac{1}{c} A_m. \tag{7.6}
\]
The choices
\[
\tau = \frac{k}{2} \quad \text{and} \quad \phi = 2f^{n-\frac{1}{2}} - Aw^{n-1} + C_{k/2}(2w^{n-1} - w^{n-2}) \tag{7.7}
\]
again gives us the perturbed Crank-Nicolson equation
\[
\frac{w^n - w^{n-1}}{k} + \frac{1}{2} A(w^n + w^{n-1}) + \frac{1}{2} C_{k/2}(w^n - 2w^{n-1} + w^{n-2}) = f^{n-\frac{1}{2}}, \tag{7.8}
\]
as we wished to derive.

### 7.2 Wave equations

While it is more common to apply explicit methods to hyperbolic equations instead of implicit methods, it is worth noting that the concepts introduced above for parabolic equations can also be applied to second-order wave equations. Consider the initial-boundary problem given by
\[
\frac{\partial^2 u}{\partial t^2} + \sum_{i=1}^m A_i u = f, \quad x \in \Omega, \ t > 0, \tag{7.9a}
\]
\[
u = 0, \quad x \in \partial \Omega, \ t > 0, \tag{7.9b}
\]
\[
u(x, 0) = g_0(x), \quad x \in \Omega, \tag{7.9c}
\]
\[
u_t(x, 0) = g_1(x), \quad x \in \Omega. \tag{7.9d}
\]
Let
\[ \partial_t^2 w^n = (w^{n+1} - 2w^n + w^{n-1})/k^2. \]
Then, the standard explicit approximation of (7.9a) is defined by the equation
\[ \partial_t^2 w^n + Aw^n = f^n, \quad n \geq 1, \quad \text{where } A = \sum_{i=1}^m A_i, \tag{7.10} \]
and it is necessary that a Courant-Friedrichs-Lewy constraint of the form
\[ \frac{k}{h} \leq \text{Const.} \]
be satisfied for stability. The standard implicit procedure, which is not subject to a CFL constraint, is given by the equation
\[ \partial_t^2 w^n + A(\alpha w^{n+1} + (1 - 2\alpha)w^n + \alpha w^{n-1}) = f^n, \quad n \geq 1. \tag{7.11} \]
Algorithm (7.11) is unconditionally stable if \( \alpha \in [0.25, 5]. \)

An LOD time-stepping procedure can be constructed as follows. Given \( z^0, \ldots, z^n, \)
first approximate the solution at \( t^{n+1} \) by the explicit procedure:
\[ \frac{z^{n+1,0} - 2z^n + z^{n-1}}{k^2} + Az^n = f^n, \tag{7.12} \]
and then apply the implicit stepping
\[ \frac{z^{n+1,\kappa} - z^{n+1,\kappa-1}}{k^2} + A_\kappa (\alpha z^{n+1,\kappa} - 2z^n + z^{n-1}) = 0, \quad \kappa = 1, \ldots, m, \tag{7.13a} \]
\[ z^{n+1} = z^{n+1,m}. \tag{7.13b} \]
The intermediate values can be eliminated to see that
\[ \partial_t^2 z^n + A(\alpha z^{n+1} + (1 - 2\alpha)z^n + \alpha z^{n-1}) + B_{a, k}(z^{n+1} - 2z^n + z^{n-1}) = f^n, \tag{7.14} \]
where
\[ B_{a, k} = \alpha^2 k^2 \sum_{1 \leq i_1 < i_2 \leq m} A_{i_1} A_{i_2} + \cdots + \alpha^m k^{2m-2} A_1 A_2 \cdots A_m. \tag{7.15} \]
Equation (7.14) can be written as a Sobolev (difference) equation
\[ (1 + k^2 B_{a, k}) \partial_t^2 z^n + A(\alpha z^{n+1} + (1 - 2\alpha)z^n + \alpha z^{n-1}) = f^n. \]
Note that the perturbation term is \( O(k^4) \) for smooth solutions of (7.9); no CFL condition is required for (7.14). It is not clear to the authors when the additional work associated with (7.14) to allow a larger time step is advantageous.
8 Conclusions

We have shown that a small modification in the right hand sides and sometimes in the nominal time step allow us to transform standard $AD$ and $FS$ methods for parabolic equations into a modified Crank-Nicolson procedure perturbed by a term that is third order ($O(t^3)$) in the time step. The extra expense of the modification requires less than ten per cent additional arithmetic per time step and leads in most cases to a very significant reduction in the splitting perturbation error associated with the $AD$ or $FS$ method.

The codes utilized in this article have been implemented for the model-code library at the University of Kentucky: Graduatel Research and Applications for Differential Equations (GRADE), and are available through internet access to http://www.ms.uky.edu/~skim/GRADE/ or by asking S. Kim (skim@ms.uky.edu).

References


