Applied Numerical Mathematics 9 (1992) 157–181
North-Holland

An explicit–implicit method for a class of time-dependent partial differential equations

F.W. Wubs
University of Groningen, P.O. Box 800, 9700 AV Groningen, Netherlands

E.D. de Goede
Centre for Mathematics and Computer Science, P.O. Box 4079, 1009 AB Amsterdam, Netherlands

Abstract

Wubs, F.W. and E.D. de Goede, An explicit–implicit method for a class of time-dependent partial differential equations, Applied Numerical Mathematics 9 (1992) 157–181.

For the integration of partial differential equations, we distinguish explicit and implicit methods. In this paper, we consider an explicit–implicit method, which follows from the truncation of the solution process of a fully implicit method. Such a method is of interest because not only better vectorizing properties can be obtained by increasing the explicit part, but the method also fits well in a domain-decomposition approach. In this paper, we focus on the feasibility of such methods by studying their stability and accuracy properties. Nevertheless, we also did some experiments on vectorcomputers to show that for a sufficient degree of explicitness our method is more efficient than fully implicit methods.

Keywords. Partial differential equations, explicit–implicit methods, method of lines, tridiagonal systems, incomplete cyclic reduction, stability, vector and parallel computers, domain decomposition.

1. Introduction

For the integration of partial differential equations, we distinguish explicit and implicit methods. In this paper, we consider an explicit–implicit method, which follows from the truncation of the solution process of a fully implicit method. Such a method is of interest because not only better vectorizing properties can be obtained by increasing the explicit part, but the method also fits well in a domain-decomposition approach. To illustrate the last point, suppose that the computational domain is split into parts and an implicit method is used. Then, during the solution process a lot of communication between the various subdomains is required. At this stage, we use an explicit approximation in order to decrease the communication. Thus, an explicit–implicit method arises. In this paper, we focus on the feasibility of such methods by studying their stability and accuracy properties. We concentrate on one-dimensional problems to facilitate the analysis. To illustrate the theory, we did some experiments on vectorcomputers to show that for a sufficient degree of explicitness our method is more efficient than fully implicit methods. The resulting method can be used in alternating direction

methods for multi-dimensional problems as will be shown for the two-dimensional Burgers equation.

We confine ourselves to partial differential equations of the form

$$u_t(t, x) = f(u, u_x, u_{xx}, x, t) \qquad x \in \Omega \subset \mathbb{R},\ t > 0,\ u(0, x) = \hat{u}(x) \tag{1.1}$$

with appropriate boundary condition and $\hat{u}(x)$ a given function. After replacing the spatial derivatives by discrete approximations, we apply an implicit time integrator to the resulting system of ordinary differential equations. We assume that the space discretization is done in such a way that the time integration leads to the solution of linear tridiagonal systems of the form

$$TV = B, \tag{1.2}$$

where T is a tridiagonal matrix, V denotes the unknowns at the advanced time level and B is a column vector. Our method is based on the approximate solution of this system. This approximation consists of two steps. In the first step, the system is reduced to a smaller tridiagonal system. This is, loosely speaking, the implicit part of the method. In this paper, we use two algorithms for this reduction process, (i) the incomplete cyclic reduction method [7,8] and (ii) a variant of the method of Wang [20]. Both methods are described in the appendix. Let us consider the incomplete cyclic reduction method. Setting $T_0 = T$, $V_0 = V$, $B_0 = B$, we can generate a sequence of systems $T_l V_l = B_l$, which after k reduction steps gives

$$T_k V_k = B_k, \quad k \geqslant 1, \tag{1.3}$$

where system (1.3) is of much smaller order than system (1.2). If T in (1.2) satisfies certain diagonal dominance conditions, then with increasing l the off-diagonal elements of T_l become smaller and smaller with respect to the diagonal elements. Theorems on this behaviour were given by Heller [7] (see also Section 3). In the second step, system (1.3) is solved approximately by replacing the inverse of T_k by a Neumann series. This is the explicit part of the method. The price to be paid for the approximation of the reduced system is a possible drop in accuracy. However, it will be shown that even for small values of k the accuracy is hardly reduced. When the system for V_k has been solved, the other unknowns are computed by back substitution.

The main purpose of this paper is to construct an explicit–implicit method that has an acceptable accuracy and stability behaviour. For model problems, using the incomplete cyclic reduction method, we have been able to derive stability conditions. These conditions show that the maximum allowed time step increases *exponentially* with k (see Section 4.2). Our method is conditionally stable. However, for any time step, k can be chosen such that it becomes stable. For two problem classes we have proved that the approximation of the solution of the reduced system has no influence on the order of accuracy.

As mentioned before, we only consider tridiagonal systems. Such systems can also be solved by direct methods like Gaussian elimination, cyclic reduction or the method of Wang. On a scalar computer (CDC Cyber 750) the computation times for the three methods are comparable. On vector computers the method of Wang and the cyclic reduction method have good vectorizing properties (see [9,12,13,19]). However, the near-explicitness of our approach gives rise to a better performance. For a nonlinear convection–diffusion problem it will be shown that our explicit–implicit method requires less computation time than the fully implicit alternative. In the numerical experiments, the reduction in computation time is about 30% for

a sufficient degree of explicitness. The stability conditions determine the value of k and thus the gain factor. The smaller the value of k, the higher the gain factor.

In Section 2, we show how a system of equations arising from an implicit scheme can be reduced to a smaller tridiagonal system that corresponds to the unknowns V_k. In Section 3, we consider the approximation of the implicit relations by explicit ones. In Section 4, we investigate the consistency and the stability of this explicit–implicit method. The nonlinear case is considered in Section 5. And finally in Section 6, we show, by a number of numerical experiments, the impact of varying the explicitness, and thus the implicitness, on the stability. Moreover, we compare the performance of our method with that of the method of Wang on vector computers (CDC Cyber 205 and Cray X-MP/28).

2. Construction of the reduced system

Consider the partial differential equation (1.1). Using the method of lines, it is space discretized on a uniform grid $\Omega_\Delta := \{j\Delta x\}_j$. This gives a system of ordinary differential equations [11]

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{U} = \boldsymbol{F}(\boldsymbol{U}, t), \quad t > 0, \tag{2.1}$$

where $U_j(t)$ approximates $\boldsymbol{u}(j\Delta x, t)$ and $\boldsymbol{F}(\boldsymbol{U}, t)$ is a vector function approximating the right-hand side function. Thereafter, a time integrator is applied to (2.1). We confine ourselves to difference formulae, which involve only two adjacent time levels. For the time integration of (2.1), explicit or implicit time integrators can be used. If the solution of (2.1) varies only slowly in time, then usually implicit time integrators are used, which in most cases are stable for any time step. For the time discretization of (2.1), we consider the θ-method [14]

$$\boldsymbol{U}^{n+1} = \boldsymbol{U}^n + \Delta t\{\theta \boldsymbol{F}(\boldsymbol{U}^{n+1}, t^{n+1}) + (1-\theta)\boldsymbol{F}(\boldsymbol{U}^n, t^n)\}, \quad \tfrac{1}{2} \leqslant \theta \leqslant 1, \tag{2.2}$$

where we have the second-order trapezoidal rule for $\theta = \frac{1}{2}$ and the backward Euler method for $\theta = 1$ (see [14]).

The equations in system (2.2) may be nonlinear. In the analysis, we confine ourselves to the linear case. In Section 6, we will indicate how the nonlinear case can be treated. In the linear case, $\boldsymbol{F}(\boldsymbol{U}, t)$ is the affine operator

$$\boldsymbol{F}(\boldsymbol{U}, t) = J\boldsymbol{U} + \boldsymbol{g}(t). \tag{2.3}$$

Hence, in each time step the following system has to be solved

$$(I - \theta\Delta t J)\boldsymbol{U}^{n+1} = \boldsymbol{B}, \tag{2.4}$$

where

$$\boldsymbol{B} = \boldsymbol{U}^n + \Delta t\{\theta \boldsymbol{g}(t^{n+1}) + (1-\theta)\boldsymbol{F}(\boldsymbol{U}^n, t^n)\}.$$

In the following, we assume that J is a tridiagonal matrix and of order $N = 2^p - 1$, where p is some positive integer. This choice for N has been made to facilitate the analysis. Applying k steps of the standard incomplete cyclic reduction method to system (2.4) yields

$$\begin{bmatrix} T_k & O \\ E & L \end{bmatrix}\begin{bmatrix} \boldsymbol{V}_k^{n+1} \\ \boldsymbol{W}_k^{n+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{B}_{1,k} \\ \boldsymbol{B}_{2,k} \end{bmatrix}, \tag{2.5}$$

where O is a null matrix, L a lower triangular matrix and E may be a full matrix. Furthermore,

$$\begin{bmatrix} V_k^{n+1} \\ W_k^{n+1} \end{bmatrix} = PU^{n+1},$$

where P is a permutation matrix. The indices of the grid points, corresponding to the so-called reduced system

$$T_k V_k^{n+1} = B_{1,k},$$

are given by the set

$$\{j \mid j = l \cdot 2^k,\ l = 1, \ldots, 2^{p-k} - 1\}. \tag{2.6}$$

Because L in (2.5) is a lower triangular matrix, the elements of W_k^{n+1} can be solved straightforwardly once V_k^{n+1} is known. So far the method is similar to cyclic reduction. The difference occurs in the solution of the reduced system. In incomplete cyclic reduction, this system is approximated by

$$D_k V_k^{n+1} = B_{1,k}, \tag{2.7}$$

where D_k is simply the diagonal part of T_k. It is difficult to analyse the impact of this truncation on the stability and accuracy behaviour of the time stepping method. Such an analysis is possible if we start, instead of (2.4), from its equivalent form

$$(I - \theta \Delta t J)(U^{n+1} - U^n) = \tilde{B}, \tag{2.8}$$

where

$$\tilde{B} = \Delta t\{JU^n + \theta g(t^{n+1}) + (1-\theta) g(t^n)\}.$$

Moreover, experiments (see e.g. Table 3) show that this is a better choice. We now arrive at a reduced system of the form

$$T_k(V_k^{n+1} - V_k^n) = \tilde{B}_{1,k}. \tag{2.9}$$

In the next section, we will approximate this system by an explicit expression.

3. Approximation of the solution of the reduced system

Let

$$T_k = D_k + C_k. \tag{3.1}$$

The precise form of C_k (and consequently of D_k) will be given later. For this moment, we assume that D_k^{-1} exists and can be computed at low costs. Premultiplying (2.9) by $(I + D_k^{-1}C_k)^{-1}D_k^{-1}$ gives

$$V_k^{n+1} = V_k^n + (I + D_k^{-1}C_k)^{-1} D_k^{-1} \tilde{B}_{1,k}. \tag{3.2}$$

Using the truncated Neumann series

$$(I + D_k^{-1}C_k)^{-1} \approx I - D_k^{-1}C_k,$$

we obtain

$$V_k^{n+1} - V_k^n \approx \left(I - D_k^{-1}C_k\right)D_k^{-1}\tilde{\boldsymbol{B}}_{1,k}.$$

Now, the formula

$$\tilde{V}_k^{n+1} = V_k^n + \left(I - D_k^{-1}C_k\right)D_k^{-1}\tilde{\boldsymbol{B}}_{1,k} \tag{3.3}$$

will be used to compute an approximation for the solution of the reduced system. Approximating the inverse of a matrix using truncated Neumann series is commonly applied in the construction of iterative algorithms on vector computers (see [1,3,5,18]).

Assume that D_k is simply the diagonal of T_k, then we have the following result due to Heller [7].

Theorem 3.1. *Suppose that* D_i^{-1} *exists,* $i = 0, \ldots, k$, *and* $\| D_0^{-1}C_0 \|_\infty < 1$, *then*
(1) $\| D_i^{-1}C_i \|_\infty \leqslant \| (D_{i-1}^{-1}C_{i-1})^2 \|_\infty < 1$,
(2) $\| V_k^{n+1} - V_k^n \|_\infty = \| \boldsymbol{U}^{n+1} - \boldsymbol{U}^n \|_\infty$.

For the error due to the approximation, we obtain by elimination of $\tilde{\boldsymbol{B}}_{1,k}$ from (3.2) and (3.3)

$$V_k^{n+1} - \tilde{V}_k^{n+1} = \left(D_k^{-1}C_k\right)^2\left(V_k^{n+1} - V_k^n\right).$$

Taking norms and using the second part of Theorem 3.1, it follows that

$$\| V_k^{n+1} - \tilde{V}_k^{n+1} \|_\infty \leqslant \| D_k^{-1}C_k \|_\infty^2 \| \boldsymbol{U}^{n+1} - \boldsymbol{U}^n \|_\infty.$$

Repeated substitution of the first part of the theorem finally yields

$$\| V_k^{n+1} - \tilde{V}_k^{n+1} \|_\infty \leqslant \| D_0^{-1}C_0 \|_\infty^{2^{k+1}} \| \boldsymbol{U}^{n+1} - \boldsymbol{U}^n \|_\infty. \tag{3.4}$$

Hence, if $\| D_0^{-1}C_0 \|_\infty$ is less than unity, the error due to the approximation decreases exponentially with k. This situation occurs for example with parabolic partial differential equations, as will be shown in Section 4.

Now, we will derive a suitable choice for the matrices D_k and C_k. The choice of C_k (and consequently of D_k) is determined by the following considerations:

(i) D_k should be easily invertible, e.g., a diagonal matrix.
(ii) The replacement of (3.2) by (3.3) should not disturb a possible numerical conservation property of (2.8).

For a discussion of conservation properties of numerical schemes, we refer to [14,16]. Note that in general (2.7) does not satisfy the second requirement. We will now derive a condition for the splitting (3.1) such that (ii) is satisfied. Comparing (3.2) and (3.3), we find that instead of (2.9) we have solved

$$\begin{bmatrix} T_k + H_k & O \\ E & L \end{bmatrix}\begin{bmatrix} \tilde{V}_k^{n+1} - V_k^n \\ \tilde{W}_k^{n+1} - W_k^n \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{B}}_{1,k} \\ \tilde{\boldsymbol{B}}_{2,k} \end{bmatrix}, \tag{3.5}$$

where

$$H_k = C_k\left(D_k^{-1}C_k\right)\left(I - D_k^{-1}C_k\right)^{-1}. \tag{3.6}$$

The cyclic reduction process can be represented by a matrix R satisfying

$$RP(I-\theta\Delta tJ)P^T=\begin{bmatrix}T_k & O\\ E & L\end{bmatrix}. \tag{3.7}$$

Define a matrix M by $M=P(I-\theta\Delta tJ)P^{\mathrm{T}}$, which is partioned as before by

$$M=\begin{bmatrix}M_{11} & M_{12}\\ M_{21} & M_{22}\end{bmatrix}.$$

Now, R assumes the form

$$R=\begin{bmatrix}I & -M_{12}M_{22}^{-1}\\ O & R_{22}\end{bmatrix}, \tag{3.8}$$

where R_{22} is nonsingular. Applying R^{-1} to (3.5) yields

$$\left(P(I-\theta\Delta tJ)P^{\mathrm{T}}+\begin{bmatrix}H_k & O\\ O & O\end{bmatrix}\right)\begin{bmatrix}\tilde{V}_k^{n+1}-V_k^n\\ \tilde{W}_k^{n+1}-W_k^n\end{bmatrix}=R^{-1}\begin{bmatrix}\tilde{B}_{1,k}\\ \tilde{B}_{2,k}\end{bmatrix}$$

or in reordered form

$$\left((I-\theta\Delta tJ)+P^{\mathrm{T}}\begin{bmatrix}H_k & O\\ O & O\end{bmatrix}P\right)(\tilde{U}^{n+1}-U^n)=\tilde{B}. \tag{3.9}$$

Comparing (3.9) to (2.8), we find that the perturbation is given by

$$P^{\mathrm{T}}\begin{bmatrix}H_k & O\\ O & O\end{bmatrix}P(\tilde{U}^{n+1}-U^n).$$

We now require that

$$e^{\mathrm{T}}H_k=\mathbf{0}^{\mathrm{T}}, \tag{3.10}$$

where $e^{\mathrm{T}}=[1,1,\ldots,1]$. This assures that in the case where (2.8) represents a conservation law, this remains so after the approximation. From (3.6), we have that (3.10) is satisfied if $e^{\mathrm{T}}C_k=\mathbf{0}^{\mathrm{T}}$. For the reduced system, it can be easily verified that $e^{\mathrm{T}}C_k=\mathbf{0}^{\mathrm{T}}$ if we choose

$$D_k=\mathrm{diag}(e^{\mathrm{T}}T_k), \tag{3.11}$$

where $\mathrm{diag}(v^{\mathrm{T}})$ is a diagonal matrix with $(\mathrm{diag}(v^T))_{ii}=v_i$.

The matrix D_k can also be seen to originate from a lumping process on the columns of T_k. Lumping is often used in finite element methods (see [17,15]) in order to obtain a diagonal matrix. Furthermore, it is used in the context of multigrid methods [4].

Summarizing, the method proceeds as follows:

(a) The system of equation (2.8) is reduced to system (2.9).
(b) D_k and C_k are constructed as denoted by (3.1) and (3.11).
(c) The explicit expression (see (3.3)) is used to approximate the solution for the reduced system.
(d) The other unknowns are solved by back substitution.

In the remainder of the paper, we will denote by D_k the diagonal of T_k and by D our choice (3.11). Also $C_k=T_k-D_k$ and $C=T_k-D$.

Remark 3.2. In Appendix B, it is shown how a reduced system can be obtained using a variant of Wang's method. If the same grid points are chosen for this system as those appearing after k steps of the cyclic reduction method, the results are identical. This stems from the fact that only the submatrix R_{22} in (3.8) is different for Wang's method. But the precise form of this submatrix plays no role in (3.9). In this paper, we do not consider other possibilities for the grid points. Hence, by restricting attention to these reduced systems, the results on consistency and stability derived below are also valid for this variant of Wang's method.

4. Consistency and stability

In the previous section, our method has been defined. Now, we come to two important questions: (i) is the constructed method consistent with (2.1), (ii) is the method stable? If the answers to these questions are positive, we know that the method is convergent. It will be clear that these questions are not trivial for our method. For some problem classes we have been able to prove the consistency and to give stability conditions.

4.1. Consistency

In this section, we shall prove the consistency for two problem classes. For the definition of these classes it is convenient to introduce the following property.

Definition. A matrix A is called an M^*-matrix if A and its transpose are strictly diagonally dominant and the diagonal elements are positive whereas the nondiagonal elements are negative.

Herewith, we define the problem classes

Class 1: *T_0 is an M^*-matrix.*

Class 2: *T_0 is, apart from the diagonal, similar to a skew-symmetric matrix and T_1 is an M^*-matrix.*

With respect to consistency, we require that the perturbation introduced by the approximate solution of the implicit relations does not influence the order of accuracy of the applied θ-method. Hence, for a smooth solution of (2.1) substituted into (3.9), we should have that

$$P^{\mathrm{T}}\begin{bmatrix} H & O \\ O & O \end{bmatrix} P(\boldsymbol{U}((n+1)\Delta t) - \boldsymbol{U}(n\Delta t)) \tag{4.1}$$

contributes at most with $O(\Delta t^2)$ or $O(\Delta t^3)$ for $\theta \neq \frac{1}{2}$ or $\theta = \frac{1}{2}$, respectively. We will see that this requirement is met in a much stronger way. In terms of H we may say that the method is consistent if H is $O(\Delta t)$ or $O(\Delta t^2)$ for $\theta \neq \frac{1}{2}$ or $\theta = \frac{1}{2}$, respectively. As H is given by (3.6), its convergence is determined by that of $CD^{-1}C$. Let N_k be the order of the matrix T_k. If the matrix T_k is given by

$$\begin{aligned} (T_k)_{i,i-1} &= \alpha_i^{(k)}, \quad i = 2, \ldots, N_k, \\ (T_k)_{i,i} &= \beta_i^{(k)}, \quad i = 1, \ldots, N_k, \\ (T_k)_{i,i+1} &= \gamma_i^{(k)}, \quad i = 1, \ldots, N_k - 1, \end{aligned} \tag{4.2}$$

with all other entries zero, then the following lemma holds.

Lemma 4.1.

(1) *If T_l is an M^*-matrix, then T_{l+1} is.*

(2) *If* $\| D^{-1}C \|_\infty < 1$, *then*

$$\frac{\| H \|_\infty \leqslant \| D \|_\infty \| D^{-1}C \|_\infty^2}{1 - \| D^{-1}C \|_\infty}.$$

(3) *If the transpose of T_k is diagonal dominant, then*

$$\| D^{-1}C \|_\infty \leqslant \frac{\| D_k^{-1}C_k \|_\infty + \| C_k D_k^{-1} \|_1}{1 - \| C_k D_k^{-1} \|_1}.$$

(4) *If T_1 is an M^*-matrix, then*

$$\| D \|_\infty \leqslant \max_i \left| \beta_i^{(l)} \right| \text{ for } l = 1, \ldots, k.$$

Proof.

(1) We omit the superscripts (l). From the cyclic reduction process, it follows that

$$\begin{aligned} \alpha_i^{(l+1)} &= -\frac{\alpha_{2i-1}\alpha_{2i}}{\beta_{2i-1}}, \qquad \gamma_i^{(l+1)} = -\frac{\gamma_{2i+1}/\gamma_{2i}}{\beta_{2i+1}}, \\ \beta_i^{(l+1)} &= \beta_{2i} - \frac{\alpha_{2i}\gamma_{2i-1}}{\beta_{2i-1}} - \frac{\gamma_{2i}\alpha_{2i+1}}{\beta_{2i+1}}. \end{aligned} \tag{4.3}$$

Note that the sign of the off-diagonal elements is correct. For the dominancy property, $\beta_i^{(l+1)} + \alpha_i^{(l+1)} + \gamma_i^{(l+1)}$ should be positive. This is true, since

$$\begin{aligned} \beta_i^{(l+1)} + \alpha_i^{(l+1)} + \gamma_i^{(l+1)} &= \beta_{2i} + \alpha_{2i}\frac{-(\alpha_{2i-1} + \gamma_{2i-1})}{\beta_{2i-1}} + \gamma_{2i}\frac{-(\alpha_{2i+1} + \gamma_{2i+1})}{\beta_{2i+1}} \\ &> \beta_{2i} + \alpha_{2i} + \gamma_{2i} > 0. \end{aligned}$$

Here, we have used that $0 \leqslant -(\alpha_i + \gamma_i)/\beta_i < 1$, which follows from T_l being an M^*-matrix.

(2) This part follows from basic norm manipulations and the property that if for any induced matrix norm and any matrix A it holds that $\| A \| < 1$, then $(\| I - A \|)^{-1} \leqslant 1/(1 - \| A \|)$ (see e.g. [2]).

(3) It holds that $D + C = D_k + C_k$. From this equality we deduce

$$\begin{aligned} D^{-1}C &= D^{-1}D_k(I + D_k^{-1}C_k) - I \\ &= (D_k^{-1}D)^{-1}(I + D_k^{-1}C_k - D_k^{-1}D). \end{aligned}$$

At the right-hand side of this equality we substitute $D = D_k + \operatorname{diag}(e^T C_k)$, which follows from equation (3.11). Furthermore, we use the identity

$$D_k^{-1} \operatorname{diag}(e^T C_k) = \operatorname{diag}(e^T C_k) D_k^{-1} = \operatorname{diag}(e^T C_k D_k^{-1}),$$

in order to derive that

$$D^{-1}C = \{I + \operatorname{diag}(e^T C_k D_k^{-1})\}^{-1}\{D_k^{-1}C_k - \operatorname{diag}(e^T C_k D_k^{-1})\}.$$

By taking infinity norms at both sides and using the inequality $\| \operatorname{diag}(e^T C_k D_k^{-1}) \|_\infty \leqslant$

$\| C_k D_k^{-1} \|_1$, we arrive at statement (3) if $\| C_k D_k^{-1} \|_1 < 1$ or equivalently the transpose of T_k is diagonal dominant.

(4) From part (1) and (4.3), it follows that $\beta_i^{(l+1)} \leqslant \beta_{2i}^{(l)}$. This leads to the desired bound. □

We will now derive a bound on $\| D_k^{-1} C_k \|_\infty$ and $\| C_k D_k^{-1} \|_1$ in terms of Δt. Once these bounds are found, we can use Lemma 4.1 to bound $\| H \|_\infty$. We start with matrices having constant elements. The results are used later to bound the nonconstant elements case. The quantities $\| D_k^{-1} C_k \|_\infty$ and $\| C_k D_k^{-1} \|_1$ will appear at exactly the same places in the subsequent lemmas. Therefore, we use one symbol ν_k to denote them.

Lemma 4.2. *Let the tridiagonal matrix T_l be given by* (4.2) *with* $\alpha_i^{(l)} = \alpha^{(l)}$, $\beta_i^{(l)} = \beta^{(l)}$, $\gamma_i^{(l)} = \gamma^{(l)}$ *and* $\alpha_i^{(l)} \neq 0$ *or* $\gamma_i^{(l)} \neq 0$. *If T_l is an M^*-matrix and $N_{l+1} \geqslant 3$, then*

$$\frac{\nu_{l+1}}{1-\nu_{l+1}} = \frac{(\alpha^{(l)})^2 + (\gamma^{(l)})^2}{(\alpha^{(l)} + \gamma^{(l)})^2} \frac{\nu_l^2}{1-\nu_l^2}. \tag{4.4}$$

Proof. In the proof, we omit the superscripts (l). From (4.3), it can easily be established that for $N_{l+1} \geqslant 3$, apart from the first and the last, a row from T_{l+1} assumes the form

$$\left[0, \ldots, 0, \frac{-\alpha^2}{\beta}, \beta - \frac{2\alpha\gamma}{\beta}, \frac{-\gamma^2}{\beta}, 0, \ldots, 0\right].$$

And for the first and the last row, we have

$$(T_{l+1})_{1,2} = \frac{-\gamma^2}{\beta}, \qquad (T_{l+1})_{N_{l+1}, N_{l+1}-1} = \frac{-\alpha^2}{\beta};$$

$$(T_{l+1})_{1,1} = \beta - \frac{2\alpha\gamma}{\beta} > 0, \quad (T_{l+1})_{N_{l+1}, N_{l+1}} = \beta - \frac{2\alpha\gamma}{\beta} > 0.$$

(If $N_1 \neq 2^p - 1$ then both " = " signs in the last line become " $\geqslant$ ".) The last inequality follows from the diagonally dominance property. Now,

$$\nu_{l+1} = \left[\frac{\alpha^2+\gamma^2}{\beta}\right] \Big/ \left[\beta - \frac{2\alpha\gamma}{\beta}\right] = \left[\frac{\alpha^2+\gamma^2}{\beta^2}\right] \Big/ \left[1 - \frac{2\alpha\gamma}{\beta^2}\right].$$

Substitution into the left-hand side of (4.4) leads to

$$\frac{\nu_{l+1}}{1-\nu_{l+1}} = \frac{[(\alpha^2+\gamma^2)/\beta^2]}{1 - 2\alpha\gamma/\beta^2 - (\alpha^2+\gamma^2)/\beta^2} = \frac{\alpha^2+\gamma^2}{(\alpha+\gamma)^2} \frac{(\alpha+\gamma)^2/\beta^2}{1-(\alpha+\gamma)^2/\beta^2}.$$

Using that $\nu_l = |\alpha + \gamma|/\beta$, we obtain the desired result. □

Remark 4.3. For the case $\gamma^{(l)} = \alpha^{(l)}$, this result is equal to the result mentioned by Heller in [7, Remark 3].

In the following lemma, we will show how the norm of $D_l^{-1} C_l$ for the nonconstant element case is majorized by that of the constant element case.

Lemma 4.4. *Let the diagonally dominant matrix T_l be given by (4.2) and T_l' by T_l from Lemma 4.2. Furthermore, let $\alpha_i\gamma_{i-1} \geqslant 0$ and $\beta_i > 0$. If for $i = 2, \ldots, N_l - 1$*

$$\frac{|\alpha_i^{(l)}|}{\beta_{i-1}^{(l)}}, \frac{|\alpha_i^{(l)}|}{\beta_i^{(l)}} \leqslant \frac{|\alpha^{(l)}|}{\beta^{(l)}} \text{ and } \frac{|\gamma_i^{(l)}|}{\beta_i^{(l)}}, \frac{|\gamma_i^{(l)}|}{\beta_{i+1}^{(l)}} \leqslant \frac{\|\gamma^{(l)}|}{\beta^{(l)}},$$

then

(1) $\nu_l \leqslant \nu_l'$,

(2) $\nu_{l+1} \leqslant \nu_{l+1}'$,

(3)
$$\frac{|\alpha_i^{(l+1)}|}{\beta_{i-1}^{(l+1)}}, \frac{|\alpha_i^{(l+1)}|}{\beta_i^{(l+1)}} \leqslant \frac{|\alpha^{(l+1)}|}{\beta^{(l+1)}},$$
$$\frac{|\gamma_i^{(l+1)}|}{\beta_i^{(l+1)}}, \frac{|\gamma_i^{(l+1)}|}{\beta_{i+1}^{(l+1)}} \leqslant \frac{|\gamma^{(l+1)}|}{\beta^{(l+1)}}.$$

Proof. In this proof we will omit the superscripts (l). Part (1) follows from the fact that, for all specified i, the expressions $|\alpha_i + \gamma_i|/\beta_i$ and $|\alpha_{i+1} + \gamma_{i-1}|/\beta_i$ are both less than $|\alpha + \gamma|/\beta$. Part (2) follows in the same way from part (3) if the latter is true. Part (3) can be proved using (4.3), e.g. it holds that

$$\frac{|\alpha_i^{(l+1)}|}{\beta_i^{(l+1)}} = \frac{|\alpha_{2i-1}\alpha_{2i}|/(\beta_{2i-1}\beta_{2i})}{1-\alpha_{2i}\gamma_{2i-1}/(\beta_{2i-1}\beta_{2i}) - \gamma_{2i}\alpha_{2i+1}/(\beta_{2i}\beta_{2i+1})}$$
$$\leqslant \frac{\alpha^2/\beta^2}{1 - 2\alpha\gamma/\beta^2} = \frac{|\alpha^{(l+1)}|}{\beta^{(l+1)}}$$

and

$$\frac{|\alpha_i^{(l+1)}|}{\beta_{i-1}^{(l+1)}} = \frac{|\alpha_{2i-1}\alpha_{2i}|/(\beta_{2i-1}\beta_{2i-2})}{1 - \alpha_{2i-2}\gamma_{2i-3}/(\beta_{2i-2}\beta_{2i-3}) - \gamma_{2i-2}\alpha_{2i-1}/(\beta_{2i-2}\beta_{2i-1})}$$
$$\leqslant \frac{\alpha^2/\beta^2}{1 - 2\alpha\gamma/\beta^2} = \frac{|\alpha^{(l+1)}|}{\beta^{(l+1)}}.$$

Likewise, we can prove the two remaining conditions in (3). □

As a consequence of this lemma, we can consider the constant element case in order to find an upper bound for ν_k. Defining

$$\mu_l = \frac{\nu_l}{1 - \nu_l}, \tag{4.5}$$

formula (4.4) can be written as

$$\mu_{l+1} = \frac{(\alpha^{(l)})^2 + (\gamma^{(l)})^2}{(\alpha^{(l)} + \gamma^{(l)})^2} \frac{\mu_l^2}{1 + 2\mu_l}. \tag{4.6}$$

This leads to two approximations

$$\mu_{l+1} \leqslant \frac{(\alpha^{(l)})^2 + (\gamma^{(l)})^2}{(\alpha^{(l)} + \gamma^{(l)})^2} \tfrac{1}{2}\mu_l \text{ for large } \mu, \tag{4.7a}$$

$$\mu_{l+1} \leqslant \mu_l^2 \text{ for small } \mu. \tag{4.7b}$$

The latter is a more stringent approximation (and therefore better) than the former if

$$\mu_l \leqslant \tfrac{1}{2} \text{ or } \nu_l \leqslant \tfrac{1}{3}. \tag{4.8}$$

In the following, we will switch from (4.7a) to (4.7b) if (4.8) is satisfied. The next step is to bound μ_0 or μ_1 by an expression in Δt. Apart from the first and the last, the ith row of T_0 is of the form (see (2.8) and (4.2))

$$[0,\ldots,0,\ \Delta t a_i,\ 1+\Delta t b_i,\ \Delta t c_i,\ 0,\ldots,0]. \tag{4.9}$$

Now, we treat the two specified classes (see the beginning of Section 4.1) separately.

Class 1. It follows that

$$\frac{\|D_0^{-1}C_0\|_\infty}{1-\|D_0^{-1}C_0\|_\infty} = \Delta t \frac{|a_j + c_j|}{1+\Delta t(b_j - |a_j + c_j|)}, \tag{4.10}$$

where j is that integer for which the maximum norm is assumed. As we have assumed that T_0 is diagonally dominant (this may restrict Δt), we have, for sufficient small Δt, that (4.10) is $O(\Delta t)$. A similar expression emerges if $\|D_0^{-1}C_0\|_\infty$ is replaced by $\|C_0 D_0^{-1}\|_1$. Hence, $\mu_0 = O(\Delta t)$.

It should be noted that for partial differential equations of parabolic and hyperbolic type, where in the latter one-sided differences are used, $b_j - |a_j + c_j| \approx 0$. Hence, in this case expression (4.10) or equivalently μ_0 is a measure for the spectral radius of T_0. Similarly, μ_l is a measure for the spectral radius of T_l. Using Lemma 4.4, it follows from (4.7) that this measure reduces in each reduction step by a factor 4 in the parabolic case ($a_i \approx c_{i-1}$ yielding $\alpha_i^{(0)} \approx \gamma_i^{(0)}$) and by a factor 2 in the hyperbolic case (a_i or c_i is zero yielding $\alpha_i^{(0)}$ or $\gamma_i^{(0)}$ is zero).

Class 2. A straight-forward computation yields

$$\frac{\|D_1^{-1}C_1\|_\infty}{1-\|D_1^{-1}C_1\|_\infty} = \Delta t^2 \frac{\dfrac{|a_{2j-1}|}{1+\Delta t b_{2j-1}}\dfrac{|a_{2j}|}{1+\Delta t b_{2j}} + \dfrac{|c_{2j+1}|}{1+\Delta t b_{2j+1}}\dfrac{|c_{2j}|}{1+\Delta t b_{2j}}}{1-\Delta t^2\left(\dfrac{|a_{2j}|}{1+\Delta t b_{2j}}\dfrac{|a_{2j-1}|-|c_{2j-1}|}{1+\Delta t b_{2j-1}} - \dfrac{|c_{2j}|}{1+\Delta t b_{2j}}\dfrac{|a_{2j+1}|-|c_{2j+1}|}{1+\Delta t b_{2j+1}}\right)}, \tag{4.11}$$

where j is the integer for which the maximum norm is assumed. By a similar reasoning as in the previous case, (4.11) if of $O(\Delta t^2)$ for sufficient small Δt. For central differences in the hyperbolic case, we have that $a_i \approx -c_i$ and $b_i = 0$ and therefore the denominator of (4.11) is approximately one. So (4.11) is a measure for the spectral radius of T_1.

Combining our results, the following bound for μ_k emerges

$$\mu_k \leqslant (\mu_i)^{2^{k-i}} \leqslant \left\{ \mu_0 \left(\tfrac{1}{2}\right)^i \prod_{l=0}^{i} \frac{(\alpha^{(l)})^2 + (\gamma^{(l)})^2}{(\alpha^{(l)} + \gamma^{(l)})^2} \right\}^{2^{k-i}}, \tag{4.12}$$

where i is the smallest integer for which the expression between braces is less than $\frac{1}{2}$. From (4.10) and (4.11), it follows that

$$\mu_k = \begin{cases} O\big((\Delta t)^{2^{k-i}}\big) & \text{for Class 1,} \\ O\big((\Delta t)^{2^{k-i+1}}\big) & \text{for Class 2.} \end{cases}$$

For sufficient large k, it follows from (4.12) that $\mu_k \ll 1$ and thereby from (4.5) $\nu_k \approx \mu_k$. Hence, using Lemma 4.1 we find that

$$\| H \|_\infty = \begin{cases} O\big((\Delta t)^{2^{k-i+1}}\big) & \text{for Class 1.} \\ O\big((\Delta t)^{2^{k-i+2}}\big) & \text{for Class 2.} \end{cases} \tag{4.13}$$

From (4.13), we observe that the lowest order occurs if $i = k$. But this order is still two and four in the respective cases. Furthermore, we see that the order increases rapidly with $k - i$. Recalling that our integration method is at most of order two, we may expect that there is hardly any effect of the truncation on the the accuracy of the method. The results in Section 6 reflect this conclusion clearly.

Summarizing, we have proved the following theorem;

Theorem 4.5. *If Δt is sufficient small, then, for problems in the Classes 1 and 2, the order accuracy of the θ-method is not decreased by the perturbation introduced by the approximate solution of the implicit relations.*

4.2. Stability

In this section, we study the stability of the perturbed scheme. Hence, (3.9) assumes the form

$$(I - \theta \Delta t J + E)(Z^{n+1} - Z^n) = \Delta t J Z^n, \tag{4.14}$$

where

$$E = P^{\mathrm{T}} \begin{bmatrix} H & O \\ O & O \end{bmatrix} P.$$

In general, the following lemma holds.

Lemma 4.6. *For $\theta \geqslant \frac{1}{2}$, the scheme* (4.14) *is stable if $(I+E)^{-1}J$ exists, is simple and has nonpositive eigenvalues.*

Proof. Premultiplying (4.14) by $(I+E)^{-1}$ yields

$$\big(I - \theta \Delta t (I+E)^{-1} J\big)(Z^{n+1} - Z^n) = \Delta t (I+E)^{-1} J Z^n.$$

Thus we find that the stability analysis of the perturbed scheme is equivalent to the stability analysis of the θ-scheme with Jacobian matrix $(I+E)^{-1}J$. As the θ-scheme, for $\theta \geqslant \frac{1}{2}$, is stable for all problems with a nonpositive simple Jacobian matrix, the lemma follows. □

We now confine ourselves to the class of problems (Class 3) satisfying the following condition: There exists a diagonal matrix $S = \mathrm{diag}(s_1, s_2, \ldots, s_N)$ such that

$$s_i J_{i,i-1} s_{i-1}^{-1} = s_{i-1} J_{i-1,i} s_i^{-1}$$

or

$$s_i J_{i,i-1} s_{i-1}^{-1} = -s_{i-1} J_{i-1,i} s_i^{-1}$$

To this class, the two classes considered in Section 4.1 belong.

Lemma 4.7.

(1) $\bar{S}_{11} T_k (\bar{S}_{11})^{-1}$ *has similar symmetry properties to* SJS^{-1}, *where* $\bar{S}_{11}$ *is the* (1, 1) *block of* $\bar{S} \equiv PSP^{\mathrm{T}}$ (*partioned as in* (2.5)).

(2) *If* T_k *is similar to a symmetric matrix then* E *is.*

(3) *The eigenvalues of* $I+E$ *are positive if those of* $D^{-1}C$ *are less than one.*

Proof.

(1) Observe that $T_k = M_{11} - M_{12} M_{22}^{-1} M_{21}$ (see (3.7) and below). Then, it holds that

$$\bar{S}_{11} T_k (\bar{S}_{11})^{-1} = \bar{S}_{11} M_{11} (\bar{S}_{11})^{-1} - \bar{S}_{11} M_{12} (\bar{S}_{22})^{-1} \bar{S}_{22} M_{22} (\bar{S}_{22})^{-1} \bar{S}_{22} M_{21} (\bar{S}_{11})^{-1}.$$

Hence, if T_k is obtained after k steps of cyclic reduction on $P^{\mathrm{T}}MP$, then $\bar{S}_{11} T_k (\bar{S}_{11})^{-1}$ can be viewed as obtained after k steps of cyclic reduction on $P^{\mathrm{T}}\bar{S}M(\bar{S})^{-1}P (\equiv SP^{\mathrm{T}}MPS^{-1})$. Moreover, starting cyclic reduction with a matrix which is, up to signs, symmetric, yields in every step of the process such a type of matrix. Consequently, $\bar{S}_{11} T_k (\bar{S}_{11})^{-1}$ has similar symmetry properties to $SP^{\mathrm{T}}MPS^{-1}$ or SJS^{-1}.

(2) Starting with T_k, we find in succession that C, $D^{-1/2}CD^{-1/2}$ and H are similar to a symmetric matrix by the same matrix as T_k is. Here, it is used that diagonal matrices commute and that H can be written as (see (3.6))

$$H = D^{1/2}(D^{-1/2}CD^{-1/2})^2 (I - D^{-1/2}CD^{-1/2})^{-1} D^{1/2}. \tag{4.15}$$

(3) This part holds if the eigenvalues of H are positive, which is according to (4.15) the case if those of $D^{-1/2}CD^{-1/2}$ are less than one. The latter matrix is similar to $D^{-1}C$ which proves the statement. □

Now, $(I+E)^{-1}J$ is similar to $(S(I+E)S^{-1})^{-1}SJS^{-1}$, where $(S(I+E)S^{-1})$ is a symmetric matrix. If the eigenvalues of $D^{-1}C$ are less than one, then the last matrix is positive definite and $(I+E)^{-1}J$ is similar to

$$(S(I+E)S^{-1})^{-1/2} SJS^{-1} (S(I+E)S^{-1})^{-1/2}. \tag{4.16}$$

It is straightforward to show, that if J has nonpositive eigenvalues, then (4.16) does and

consequently $(I+E)^{-1}J$ does. From Lemma 4.6 the stability follows. Summarizing, we have the following theorem:

Theorem 4.8. *Scheme* (4.15) *is stable for problems in Class 3 if* T_k *is similar to a symmetric matrix and* Δt *and* k *are such that the eigenvalues of* $D^{-1}C$ *are less than one.*

Remark 4.9. One should be aware of the fact that stability in terms of eigenvalues does not imply that there may not be some growth in for example the 2-norm. A detailed analysis reveals that this growth is bounded by the square root of the condition number of the symmetric matrix $S^2(I+E)$. This number should not be too large.

Using Theorem 4.8, we can find stability conditions for model problems. This will be done for a parabolic and a hyperbolic problem. In terms of the coefficients in (4.9), we specify the parabolic problem by $a_i = c_i = -\theta/\Delta x^2$ and $b_i = -2a_i$, whereas the hyperbolic problem is determined by $a_i = -c_i = -\frac{1}{2}\theta/\Delta x$ and $b_i = 0$. From (4.10) and (4.11), it follows that

$$\mu_0 = \frac{2\theta\Delta t}{\Delta x^2}$$

in the parabolic case and that

$$\mu_1 = 2\left(\frac{\theta\Delta t}{2\Delta x}\right)^2$$

in the hyperbolic case. As a result we have that the matrix T_k is symmetric and has constant coefficients. Then, the eigenvalues of $D^{-1}C$ are less than one if $\beta^{(k)}/\alpha^{(k)} > 6$ (see (4.2) for the definition of $\alpha^{(k)}$ and $\beta^{(k)}$). This corresponds to $\nu_k < \frac{1}{3}$ (due to the symmetry the infinity norm and the 1-norm are equal) or equivalently $\mu_k < \frac{1}{2}$ (see (4.5)). From Theorem 3.1, it follows that this requirement is met if

$$\nu_0 \leqslant \left(\tfrac{1}{3}\right)^{1/2^k} \approx 1 - \ln(3)\left(\tfrac{1}{2}\right)^k.$$

However, a much stronger result is possible using relations derived in Section 4.1. From (4.7a), it follows that for this symmetric case

$$\mu_{l+1} \leqslant \tfrac{1}{4}\mu_l.$$

Now, $\mu_k < \frac{1}{2}$ if in the parabolic case $\mu_0 < \frac{1}{2}\cdot 4^k$ and in the hyperbolic case $\mu_1 < \frac{1}{2}\cdot 4^{k-1}$. For $\theta = \frac{1}{2}$, we obtain in the parabolic case the stability condition

$$\frac{\Delta t}{\Delta x^2} \leqslant 2(4^{k-1}) \text{ or } \frac{\Delta t}{(2^k\Delta x)^2} \leqslant \tfrac{1}{2}. \tag{4.17}$$

A more precise computation using (4.6) yields a stability condition of the form

$$\frac{\Delta t}{(2^k\Delta x)^2} \leqslant \delta_k.$$

In Table 1, we list some values of δ_k. Here, we have

$$\lim_{k\to\infty} \delta_k \uparrow 0.643651.$$

Table 1
Stability coefficients for the parabolic model problem

k	0	1	2	3	4
δ_k	0.5	0.60355	0.63334	0.64105	0.64299

It will be clear that our estimate (4.17) is sufficient sharp for practical purposes. The stability condition given here is clearly reflected in the results of Section 6.1. For the hyperbolic case, we obtain for $\theta = \frac{1}{2}$ the condition

$$\frac{\Delta t}{\Delta x} \leqslant 2^k \quad \text{or} \quad \frac{\Delta t}{2^k \Delta x} \leqslant 1, \qquad k \geqslant 1. \tag{4.18}$$

Using (4.6), a condition of the form

$$\frac{\Delta t}{2^k \Delta x} \leqslant \eta_k$$

is found, where $\eta_k = \sqrt{2\delta_{k-1}}$. The values of δ_k are the same as those in Table 1.

Remark 4.10. In these model problems, the method reduces to the modified Euler method [11] for $k = 0$. The stability region of the modified Euler method has only the trivial intersection with the imaginary axis. Hence, the method is not stable for problems in which the Jacobian matrix has purely imaginary eigenvalues, as in the hyperbolic case. Therefore, it is not surprising that in the hyperbolic case no stability condition emerges for $k = 0$.

5. The nonlinear case

The technique derived in the previous sections can also be applied to nonlinear equations. Starting from (2.2), we arrive at a linear system of equations by introducing a so-called splitting function $G(Z, \tilde{Z}, t)$ [10]. We choose G in such a way that it is linear in its second variable, i.e.

$$G(Z, \tilde{Z}, t) = J(Z, t)\tilde{Z} + g(Z, t), \tag{5.1}$$

where J and g are chosen such that the splitting condition

$$G(Z, Z, t) = F(Z, t)$$

is satisfied. For example, J can be the Jacobian matrix of $F(U)$. Equation (2.2) is now approximately solved by the iteration process

$$\begin{aligned} &Z^{(0)} = U^n, \\ &Z^{(q)} = U^n + \Delta t\{\theta G(Z^{(q-1)}, Z^{(q)}, t^{n+1}) + (1-\theta)G(U^n, U^n, t^n)\}, \quad q = 1, \ldots, Q, \\ &U^{n+1} = Z^{(Q)}. \end{aligned} \tag{5.2}$$

In this equation, the iterate $Z^{(q)}$ has to be solved from a linear system of equations. In order to approximate (2.2) accurately by (5.2), Q has to be chosen large. for $Q \geqslant 1$, (5.2) constitutes a first-order accurate scheme. It becomes even second-order accurate if $\theta = 0.5$ and $Q \geqslant 2$ ($Q \geqslant 1$ in the linear case). Two specific instances of (5.2) are (i) Newton's method for (2.2) if J is the Jacobian matrix of $F(U)$ and (ii) the modified Euler method if $J = 0$, $\theta = 0.5$ and $Q = 2$.

Instead of (2.8), we solve for $q = 1, \ldots, Q$ a linear system of equations of the form

$$(I - \theta \Delta t J(Z^{(q-1)}, t^{n+1}))(Z^{(q)} - Z^{(0)}) = \tilde{B}, \tag{5.3}$$

where

$$\tilde{B} = \Delta t\{\theta J(Z^{(q-1)}, t^{n+1})Z^{(0)} + \theta g(Z^{(q-1)}, t^{n+1}) + (1-\theta)G(U^n, U^n, t^n)\}.$$

This system can be solved similarly to (2.8).

6. Numerical illustration

To illustrate the performance of the method described in Section 2 and 3, we present some experiments for both linear and nonlinear problems. By varying the set of points which are solved explicitly, we vary the stability property of the method. The aim of our experiments is to show the relation between the number of reduction steps, which is a measure for the implicitness, and the stability behaviour of the method. We are also interested in the accuracy behaviour relative to the number of reduction steps. For linear test problems only, we also compare the accuracy of our approach with the incomplete cyclic reduction method (2.7) and with the associated fully implicit method (i.e., the complete cyclic reduction method). To measure the obtained accuracy we define

$$\text{cd} = -{}^{10}\log(\,|\text{maximal global error at the endpoint } t = T\,|),$$

denoting the number of correct digits in the numerical approximation at the endpoint. The calculations in Sections 6.1 and 6.2 were performed on the CDC Cyber 170-750 which has a 48-bit mantissa, i.e., a machine precision of about 14 decimal digits.

6.1. A linear parabolic problem

As a first example, consider the linear parabolic problem

$$\begin{aligned} &u_t = u_{xx}, \qquad 0 < t < T, \quad 0 < x < L, \\ &u_x(0, t) = 2\frac{\pi}{L} e^{-(2\pi/L)^2 t}, \\ &u(L, t) = 0. \end{aligned} \tag{6.1}$$

The exact solution is given by

$$u(x, t) = e^{-(2\pi/L)^2 t} \sin(2\pi x/L). \tag{6.2}$$

We choose $L = 32$. For the space-discretization of (6.1), central differences are used. This yields for J and g (see (2.3))

$$\begin{aligned} (JU)_1 &= \frac{(U_2 - U_1)}{(\Delta x)^2}, & g_1(t) &= -\frac{\left(2(\pi/L)\, e^{-(2\pi/L)^2 t}\right)}{\Delta x}, \\ (JU)_j &= \frac{(U_{j-1} - 2U_j + U_{j+1})}{(\Delta x)^2}, & g_j(t) &= 0 \qquad \text{for } j = 2, \ldots, N-1, \\ (JU)_N &= \frac{(U_{N-1} - 2U_N)}{(\Delta x)^2}, & g_N(t) &= 0. \end{aligned} \tag{6.3}$$

Table 2
Number of correct digits for the linear parabolic problem (6.1) with $T = 320$

Δt	scheme (3.3)				scheme (2.7)
	$p = 4$ $\Delta x = 2$	$p = 5$ $\Delta x = 1$	$p = 6$ $\Delta x = 0.5$	$p = 7$ $\Delta x = 0.25$	$p = 7$ $\Delta x = 0.25$
2	2.00(0)	***(0)			
	2.13(1)	2.64(1)	***(1)		
		2.69(2)	3.00(2)	***(2)	
			3.41(3)	3.40(3)	
			3.43(4)	4.07(4)	***(4)
					3.97(5)
	2.13(TR)	2.70(TR)	3.43(TR)	4.08(TR)	
4	2.12(1)	***(1)			
	2.16(2)	2.75(2)	***(2)		
		2.85(3)	3.49(3)	***(3)	
			3.37(4)	3.31(4)	***(4)
					3.12(5)
	2.16(TR)	2.85(TR)	3.37(TR)	3.32(TR)	
8	2.30(2)	2.28(2)	***(2)		
		2.78(3)	2.38(3)	***(3)	
			2.56(4)	2.40(4)	***(4)
					2.50(5)
	2.30(TR)	2.77(TR)	2.57(TR)	2.49(TR)	
16	2.24(2)	***(2)			
		2.01(3)	***(3)		
			1.94(4)	***(4)	***(4)
				1.91(5)	1.90(5)
	2.16(TR)	1.97(TR)	1.94(TR)	1.93(TR)	

The grid points are chosen $x_j = x_0 + j\Delta x$ with $x_0 = -\frac{1}{2}\Delta x$ and $x_{N+1} = L$. Here, the Jacobian matrix J has real eigenvalues. For the time integration, we use the trapezoidal rule, $\theta = 0.5$ ($Q = 1$ due to linearity). In Table 2, we give the cd-values of the method, obtained at the endpoint $T = 320$. The number of grid points N is equal to 2^p. The number of cyclic reduction steps is given in parenthesis. The values obtained for the complete cyclic reduction method, i.e., the trapezoidal rule (TR), are given in the last row for every time step. For the complete cyclic reduction method, the number of reduction steps equals $p - 1$. In the last column, we list the values for the standard incomplete cyclic reduction in which approximation (2.7) is used. An unstable behaviour of the integration process is denoted by ***.The results clearly show the effect of varying the number of reduction steps:

(a) The error hardly depends on the number of reduction steps as long as the method is stable. Moreover, the accuracy of our method is comparable with that of the (complete) cyclic reduction method, provided that the method is stable. This is in agreement with our conclusions at the end of Section 4.1.
(b) If the mesh size is decreased by a factor two, then one extra reduction step is needed to maintain the same stability boundary on Δt, which agrees with stability condition (4.17).

Scheme (2.7) behaves similar to scheme (3.3) with increasing k. We will discuss its behaviour at the end of the next section.

6.2. A linear hyperbolic problem

As a second example, consider the linear hyperbolic problem

$$u_t = u_x, \qquad 0 < t < T, \quad 0 < x < L, \tag{6.4}$$

with initial condition

$$u(x, 0) = \sin(2\pi x/L),$$

and boundary condition

$$u(L, t) = \sin(2\pi(L + t)/L).$$

The exact solution is given by

$$u(x, t) = \sin(2\pi(x + t)/L).$$

We choose $L = 64$. Central differences are used at all points except for the first point where a commonly used one-sided difference is applied. The discretization is given by ($x_1 = 0$ and $x_{N+1} = L$)

$$\begin{aligned}
(JU)_1 &= \frac{(U_2 - U_1)}{\Delta x} && g_1 = 0, \\
(JU)_j &= \frac{(U_{j+1} - U_{j-1})}{2\Delta x}, && g_j = 0 \qquad \text{for } j = 2, \ldots, N-1, \\
(JU)_N &= -\frac{U_{N-1}}{2\Delta x}, && g_N(t) = \frac{\sin(2\pi(L + t)/L)}{2\Delta x}.
\end{aligned} \tag{6.5}$$

Here, we have $N = 2^p - 1$. Note that for linear hyperbolic systems, the Jacobian matrix J has almost purely imaginary eigenvalues. For the time integration, we use the trapezoidal rule, i.e., $\theta = 0.5$ ($Q = 1$ due to linearity). In Table 3, the results are given in the same form as in Table 2.

Globally, we observe the same effect for this hyperbolic problem as for the parabolic problem (6.1). If the mesh size is decreased by a factor two, then one extra reduction step is needed to maintain the same stability boundary on Δt, which is in agreement with (4.18).

In contrast with the parabolic problem, scheme (2.7) needs two or three extra reduction steps in order to obtain an accuracy that is comparable with scheme (3.3). We think that this difference stems from the fact that for the θ-method with $\theta = 0.5$ all eigenvalues of the amplification matrix in the hyperbolic case have magnitude 1, whereas in the parabolic case these are less than one with some eigenvalues even considerably less than one. This makes the hyperbolic case much more susceptible to perturbations in the scheme than the parabolic case. Scheme (2.7) needs some extra reduction steps to make these perturbations negligible, whereas our method compensates this by the explicit approximation of the reduced system (see (3.3)).

6.3. A nonlinear test problem

In this section, we apply our method to a nonlinear two-dimensional problem. This problem is discretized by means of an ADI method. In the successive steps of this method, there arise

Table 3
Number of correct digits for the linear hyperbolic problem (6.4) with $T = 320$

Δt	scheme (3.3)					scheme (2.7)
	$p=5$ $\Delta x=2$	$p=6$ $\Delta x=1$	$p=7$ $\Delta x=0.5$	$p=8$ $\Delta x=0.25$	$p=9$ $\Delta x=0.125$	$p=8$ $\Delta x=0.25$
1	1.30(1)	1.80(1)	1.87(1)	***(1)		
	1.30(2)	1.80(2)	2.11(2)	2.24(2)	***(2)	***(2)
				2.25(3)	2.27(3)	0.34(3)
						2.03(4)
	1.30(TR)	1.80(TR)	2.12(TR)	2.29(TR)	2.31(TR)	
2	1.19(1)	1.24(1)	***(1)			
	1.19(2)	1.51(2)	1.63(2)	***(2)		
		1.51(3)	1.64(3)	1.67(3)	***(3)	***(3)
					1.68(4)	0.64(4)
						1.68(5)
	1.19(TR)	1.51(TR)	1.65(TR)	1.69(TR)	1.70(TR)	
4	0.78(1)	***(1)				
	0.91(2)	1.01(2)	***(2)			
	0.91(3)	1.04(3)	1.05(3)	***(3)		
		1.04(4)	1.08(4)	1.06(4)		***(4)
						0.88(5)
						1.08(6)
	0.91(TR)	1.04(TR)	1.08(TR)	1.07(TR)		
8	***(1)					
	0.36(2)	***(2)				
	0.43(3)	0.40(3)	***(3)			
		0.46(4)	0.41(4)			
	0.43(TR)	0.46(TR)	0.41(TR)			

systems with tridiagonal matrices to which our method can be applied. Consider the two-dimensional Burgers equations

$$u_t = -uu_x - vu_y + (u_{xx} + u_{yy})/\mathrm{Re},$$
$$v_t = -uv_x - vv_y + (v_{xx} + v_{yy})/\mathrm{Re},$$

where u and v denote the velocities and Re the Reynolds number. An exact solution of the Burgers equations can be generated by using the Cole–Hopf transformation [6],

$$u = -\frac{2}{\mathrm{Re}}\frac{\phi_x}{\phi}, \qquad v = -\frac{2}{\mathrm{Re}}\frac{\phi_y}{\phi}, \tag{6.6a}$$

where ϕ is the solution of

$$\phi_t = \frac{1}{\mathrm{Re}}(\phi_{xx} + \phi_{yy}). \tag{6.6b}$$

In our test problem, we choose $\phi = f_1 + f_2$ with

$$f_1(x, y, t) = \exp((-12(x+y) + 9t)\mathrm{Re}/32)$$
$$f_2(x, y, t) = \exp((-4(x+2y) + 5t)\mathrm{Re}/16), \tag{6.7}$$

which yields the exact solution

$$u = \frac{1}{4}\frac{3f_1 + 2f_2}{f_1 + f_2} = \frac{3}{4} - \frac{1}{4}\frac{1}{1 + \exp((-4x + 4y - t)\mathrm{Re}/32)}, \tag{6.8a}$$

$$v = \frac{1}{4}\frac{3f_1 + 4f_2}{f_1 + f_2} = \frac{3}{4} + \frac{1}{4}\frac{1}{1 + \exp((-4x + 4y - t)\mathrm{Re}/32)}. \tag{6.8b}$$

The solution represents a wave front at $y = x + 0.25t$. The speed of propagation is $0.125\sqrt{2}$ and is perpendicular to the wave front. For increasing values of Re, the wave front becomes sharper.

For the space discretization, we use the same second-order central differences as in the linear test examples. For the time integration, we use the ADI scheme of Peaceman and Rachford [14] with $Q = 2$. The Peaceman–Rachford formula reads (cf. (2.2))

$$\boldsymbol{U}^{n+1/2} = \boldsymbol{U}^n + \tfrac{1}{2}\Delta t\left[\boldsymbol{F}_x(\boldsymbol{U}^{n+1/2}) + \boldsymbol{F}_y(\boldsymbol{U}^n)\right], \tag{6.9a}$$

$$\boldsymbol{U}^{n+1} = \boldsymbol{U}^{n+1/2} + \tfrac{1}{2}\Delta t\left[\boldsymbol{F}_x(\boldsymbol{U}^{n+1/2}) + \boldsymbol{F}_y(\boldsymbol{U}^{n+1})\right]. \tag{6.9b}$$

where $\boldsymbol{F}_x$ and $\boldsymbol{F}_y$ represent the space discretizations of the terms containing the x- and y-derivatives, respectively. The Peaceman–Rachford method is a popular analogue of the trapezoidal rule for two-dimensional problems. Note that (6.9a) is explicit in the y-direction and implicit in the x-direction, and vice versa in (6.9b).

With the purpose of testing the (order of) accuracy of the schemes, we first compare the exact solution of the Burgers equations with the numerical solution obtained for grid sizes $\Delta x = \Delta y = 1/17, 1/33, 1/65, 1/129$ and for time steps $\Delta t = 1/10, 1/20, 1/40, 1/80, 1/160, 1/320$. The computational domain is $\Omega = [0, 1] \times [0, 1]$ and the time integration interval is $[0, 2.5]$. We prescribe time-dependent Dirichlet boundary conditions which are taken from the exact solution and we choose $\mathrm{Re} = 100$.

The ADI scheme for two-dimensional problems requires the solution of tridiagonal sets along horizontal and vertical grid lines respectively. We combined the tridiagonal sets to one large tridiagonal system. For the solution of the tridiagonal systems, we use a variant of the method of Wang (ADIW) (see Appendix B) and our explicit–implicit method (ADEI). Here, the reduced system in ADEI is obtained as in Wang's method (see Remark 3.2). In the successive stages of (6.9), reordering of the data structure is performed in order to obtain contiguous data vectors. This is beneficial to both ADIW and ADEI. In Table 4, the cd-values

Table 4
cd_∞-values for the ADIW and ADEI scheme

scheme	$(\Delta x)^{-1}$	correct digits for u-field					
		$\Delta t = \frac{1}{10}$	$\Delta t = \frac{1}{20}$	$\Delta t = \frac{1}{40}$	$\Delta t = \frac{1}{80}$	$\Delta t = \frac{1}{160}$	$\Delta t = \frac{1}{320}$
ADIW	17[17]	1.92	2.29	2.48	2.51	2.52	2.52
	33[121]	2.10	2.56	2.89	3.07	3.15	3.18
	65[325]	2.24	2.75	3.19	3.51	3.68	3.77
	129[1849]	2.25	2.77	3.26	3.67	3.99	4.19
ADEI	17[17]	1.92	2.29	2.48	2.51	2.52	2.52
	33[121]	2.12	2.57	2.89	3.07	3.15	3.18
	65[325]	2.24	2.75	3.19	3.51	3.68	3.77
	129[1849]	2.25	2.78	3.26	3.67	3.98	4.17

Table 5
Execution times in seconds for a 129×129 grid with $t = 2.5$, $\Delta t = \frac{1}{80}$, $\mathrm{Re} = 100$, and $k = 3$

scheme	Execution times (in seconds)	
	Cyber 205	Cray X-MP/28
ADIW	18.6	8.7
ADEI	12.6	6.1

for both approaches are presented. We only list the cd-values for the u-field; for the v-field we obtain nearly the same results. In the brackets [ ], we list the order of the reduced system. An optimal size of the blocks in ADIW (see Appendix B) is chosen. On the Cyber 205, the optimal order of the reduced system is about $5\sqrt{N}$, where N denotes the order of the original system [19].

Remark 6.1. For the problem considered here, a more efficient special purpose method can be designed by exploiting the fact that essentially independent systems with tridiagonal matrices of the same size are solved. However, in many practical problems the sizes of these systems are not equal. We are most interested in methods for such cases and therefore we combined these independent systems to one large system.

For both ADI-type schemes one can observe second-order behaviour in space and time. Note that the accuracy results for the ADIW scheme and the ADEI scheme are comparable, which is in agreement with Theorem 4.5.

Table 5 presents the execution times for both ADI-type schemes obtained for a single example, namely for a 129×129 grid with $t = 2.5$, $\Delta t = \frac{1}{80}$ and $\mathrm{Re} = 100$. For the number of reduction steps we chose $k = 3$. This experiment has been carried out on a (2-pipe) CDC Cyber 205 and on a Cray X-MP/28. On the Cray X-MP only one processor has been used.

With respect to other choices of k, we found that the computation time decreases slightly for $k = 1, 2$, but for $k \geqslant 4$ there is no advantage over ADIW. From the experiments, we conclude that for $k = 1, \ldots, 3$ the explicit–implicit method requires less computation time than the ADI method in which the tridiagonal system is solved by the variant of the method of Wang.

Remark 6.2. The proposed method and Wang's method are well suited for highly parallel computers in which each processor has its own memory. The computational domain can be split in appropriate parts which are distributed over the processors. Only for solving the reduced set of equations interaction between processors is needed. For Wang's method this interaction is global, but with our method this can be kept local (with neighbours only) due to the explicit approximation of the reduced system. Applying the methods in such a way is in fact a domain-decomposition approach.

7. Conclusions

In this paper, we have constructed an explicit–implicit method starting from a one-step implicit method. The method was constructed for time-dependent partial differential equations. It makes use of the fact that the interdependence of the solution at two different points

at the new time level decreases with their physical distance. The constructed method has the following properties:

(a) The accuracy is hardly influenced if we replace the one-step implicit method by an approximating explicit–implicit method as long as the integration is stable.
(b) If the one-step implicit method satisfies a conservation property, then this property is preserved by the approximating explicit–implicit method.
(c) The maximum allowed time step increases exponentially with the number of reduction steps.
(d) The explicit–implicit methods are, for a sufficient degree of explicitness, more efficient than the conventional methods on the considered vector computers.
(e) The methods can be used within ADI-type methods for multi-dimensional problems.

The methods considered in this paper assume that the difference equations are such that tridiagonal matrices arise. We think that this approach can also be used for difference equations which do not lead to a tridiagonal matrices (e.g. higher-order differences), because the decrease of interdependence with the distance between successive point of the reduced system stems from the partial differential equation. And this can in one or the other way be exploited.

Appendix A. Incomplete cyclic reduction

The cyclic reduction algorithm was originally developed by Hockney [8] for the discrete version of Poisson's equation. The cyclic reduction algorithm is well-suited for use on a parallel or vector computer, as many of the quantities involved may be computed independently of the others. This case has been studied by Lambiotte and Voight [12] with attention to a vector computer.

We assume that the system of linear algebraic equations arising from implicit difference formula (2.2), which must be solved at each time step is a special case of the tridiagonal system

$$\alpha_j x_{j-1} + \beta_j x_j + \gamma_j x_{j+1} = b_j,$$

for $1 \leqslant j \leqslant m$, where $\alpha_1 = 0$ and $\gamma_m = 0$. Also, we assume that $m = 2^p - 1$, although this is not essential, where p is some positive integer. In matrix form, we obtain

$$\begin{bmatrix} \beta_1 & \gamma_1 & & & 0 \\ \alpha_2 & \beta_2 & \gamma_2 & & \\ & \vdots & \vdots & \vdots & \\ & & \alpha_{m-1} & \beta_{m-1} & \gamma_{m-1} \\ 0 & & & \alpha_m & \beta_m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{m-1} \\ x_m \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{m-1} \\ b_m \end{bmatrix}. \tag{A.1}$$

The cyclic reduction algorithm separates the system in two subsystems, which involve respectively the rows with even indices and the rows with odd indices. Let us rewrite (A.1) as follows:

$$\alpha_{j-1} x_{j-2} + \beta_{j-1} x_{j-1} + \gamma_{j-1} x_j = b_{j-1},$$
$$\alpha_j x_{j-1} + \beta_j x_j + \gamma_j x_{j+1} = b_j,$$
$$\alpha_{j+1} x_j + \beta_{j+1} x_{j+1} + \gamma_{j+1} x_{j+2} = b_{j+1}.$$

Multiplying the first equation by $-\alpha_j/\beta_{j-1}$, the third by $-\gamma_j/\beta_{j+1}$ and adding to the second equation, we obtain

$$\kappa_j x_{j-2} + \lambda_j x_j + \mu_j x_{j+2} = B_j, \tag{A.2}$$

where

$$\kappa_j = -\alpha_{j-1}\frac{\alpha_j}{\beta_{j-1}},\ \lambda_j = \beta_j - \gamma_{j-1}\frac{\alpha_j}{\beta_{j-1}} - \alpha_{j+1}\frac{\gamma_j}{\beta_{j+1}},$$

$$\mu_j = -\gamma_{j+1}\frac{\gamma_j}{\beta_{j+1}}$$

and

$$B_j = -\frac{\alpha_j}{\beta_{j-1}}b_{j-1} + b_j - \frac{\gamma_j}{\beta_{j+1}}b_{j+1}.$$

Thus, if j is even, the new system of equations involves x_j's with even indices. Similar equations hold for x_2 and x_{m-1}. The process of reducing the equations in this fashion is known as cyclic reduction. Hereby, (A.1) may be written as the following equivalent system:

$$\begin{bmatrix} \lambda_2 & \mu_2 & & & 0 \\ \kappa_4 & \lambda_4 & \mu_4 & & \\ & \vdots & \vdots & \vdots & \\ & & \kappa_{m-3} & \lambda_{m-3} & \mu_{m-3} \\ 0 & & & \kappa_{m-1} & \lambda_{m-1} \end{bmatrix} \begin{bmatrix} x_2 \\ x_4 \\ \vdots \\ x_{m-3} \\ x_{m-1} \end{bmatrix} = \begin{bmatrix} B_2 \\ B_4 \\ \vdots \\ B_{m-3} \\ B_{m-1} \end{bmatrix}, \tag{A.3}$$

and

$$\begin{bmatrix} \beta_1 & 0 & & & 0 \\ 0 & \beta_3 & 0 & & \\ & \vdots & \vdots & \vdots & \vdots \\ & & 0 & \beta_{m-2} & 0 \\ 0 & & & 0 & \beta_m \end{bmatrix} \begin{bmatrix} x_1 \\ x_3 \\ \vdots \\ x_{m-2} \\ x_m \end{bmatrix} = \begin{bmatrix} b_1 \\ b_3 \\ \vdots \\ b_{m-2} \\ b_m \end{bmatrix} - \begin{bmatrix} \gamma_1 & & & & 0 \\ \alpha_3 & \gamma_3 & & & \\ & \vdots & \vdots & & \\ & & & \alpha_{m-2} & \gamma_{m-2} \\ 0 & & & & \alpha_m \end{bmatrix} \begin{bmatrix} x_2 \\ x_4 \\ \vdots \\ x_{m-3} \\ x_{m-1} \end{bmatrix}. \tag{A.4}$$

Since $m = 2^p - 1$ and the new system (A.3) involves only x_j's with even indices, the dimension of the new system is $2^{p-1} - 1$. Note that once (A.3) is solved, it is easy to solve for the x_j's with odd indices, as evidenced by (A.4). The system (A.4) is known as the eliminated equations.

Since system (A.3) is tridiagonal and in the form of (A.1), we can apply the reduction algorithm repeatedly until we have one equation. However, we can stop the process after any step and use another method to solve the reduced system of equations. This is called the incomplete cyclic reduction method. After renumbering, we obtain a system of equations as denoted by (2.5).

Appendix B. A parallel method

Here, we use a variant on Wang's algorithm [20]. Let us assume that the system of linear equations given in (A.1) is of the form

$$\begin{bmatrix} A_1 & d_1 & & & & 0 \\ a_k & \beta_k & c_k & & & \\ & e_2 & A_2 & d_2 & & \\ & & a_l & \beta_l & c_l & \\ & & & e_3 & A_3 & d_3 \\ 0 & & & & a_m & \beta_m \end{bmatrix} x = b,$$

where A_1, A_2, and A_3 are tridiagonal matrices and

$$a_i = [0,\ldots,0,\alpha_i], \qquad c_i = [\gamma_i, 0,\ldots,0],$$
$$d_1 = [0,\ldots,0,\gamma_{k-1}]^T, \quad d_2 = [0,\ldots,0,\gamma_{l-1}]^T, \quad d_3 = [0,\ldots,0,\gamma_{m-1}]^T,$$
$$e_2 = [\alpha_{k+1}, 0,\ldots,0]^T, \quad e_3 = [\alpha_{l+1}, 0,\ldots,0]^T.$$

In this example, we use three block matrices, but this reduction technique can be applied for an arbitrary number of block matrices. For this subdivision x_k, x_l and x_m will be the unknowns of the reduced system of equations. Eliminating the off-diagonal elements of A_i, followed by a scaling of the diagonal elements gives

$$\begin{bmatrix} I & v_1 & & & & 0 \\ a'_k & \beta_k & c'_k & & & \\ & w_2 & I & v_2 & & \\ & & a'_l & \beta_l & c'_l & \\ & & & w_3 & I & v_3 \\ 0 & & & & a'_m & \beta_m \end{bmatrix} x = b',$$

where the v and w are column vectors. Now, we eleminate a'_k, c'_k, a'_l, c'_l, and a'_m, which yields

$$\begin{bmatrix} I & v_1 & & & & 0 \\ & \beta'_k & & \gamma_k & & \\ & w_2 & I & v_2 & & \\ & \alpha_l & & \beta'_l & & \gamma_l \\ & & & w_3 & I & v_3 \\ 0 & & & \alpha_m & & \beta'_m \end{bmatrix} x = b''. \tag{B.1}$$

So far, this method corresponds with the first steps of Wang's algorithm. The elimination of the off-diagonal elements of the matrices A_i can be done independently. Therefore, this approach is well-suited for vector and parallel computers (see [13]). By a simple reordering system (B.1) can be brought to a system of the form (2.5). The kth, lth and the mth row, which do not contain elements of the block matrices, form the reduced system of equations.

Acknowledgement

The authors wish to express their gratitude to the referees, whose comments contributed significantly to the presentation of this paper.

References

[1] L. Adams, m-step preconditioned conjugate gradient methods, *SIAM J. Sci. Stat. Comput.* 6 (1985) 452–463.
[2] K.E. Atkinson, *An Introduction to Numerical Analysis* (Wiley, New York, 1989).
[3] O. Axelsson, A survey of preconditioned iterative methods for linear systems of equations, *BIT* 25 (1985) 166–187.
[4] J.K. Dendy Jr, Black box multigrid for nonsymmetric problems, *Appl. Math. Comput.* 13 (1983) 261–283.
[5] P.P. Dubois, A. Greenbaum and G.H. Rodrigue, Approximating the inverse matrix for use in iterative algorithms on vector computers, *Computing* 22 (1979) 257–268.
[6] C.A.J. Fletcher, A comparison of finite element and finite difference solutions of the one- and two-dimensional Burgers equation, *J. Comput. Phys.* 51 (1983) 159–188.
[7] D. Heller, Some aspects of the cyclic reduction algorithm for block tridiagonal linear systems, *SIAM J. Numer. Anal.* 13 (1976) 484–496.
[8] R.W. Hockney, A fast direct solution of Poisson's equation using Fourier analysis, *J. Assoc. Comput. Mach.* 12 (1965) 95–113.
[9] R.W. Hockney and C.R. Jesshope, *Parallel Computers: Architecture, Programming and Algorithms* (Adam Hilger, Bristol, 1981).
[10] P.J. van der Houwen and J.G. Verwer, One-step splitting methods for semi-discrete parabolic equations, *Computing* 22 (1979) 291–309.
[11] J.D. Lambert, *Computational Methods in Ordinary Differential Equations* (Wiley, London, New York, 1973).
[12] J.J. Lambiotte and R.G. Voight, The solution of tridiagonal linear systems on the CDC Star-100 computer, *ACM Trans. Math. Software* 1 (1975) 308–329.
[13] P.H. Michielse and H.A. van der Vorst, Data transport in Wang's partition method, *Parallel Comput.* 7 (1988) 87–95.
[14] A.R. Mitchell and D.F. Griffiths, *The Finite Difference Method in Partial Differential Equations* (Wiley, Chichester, 1980).
[15] A.R. Mitchell and R.W. Wait, *The Finite Element Analysis and Applications* (Wiley, New York, 1985).
[16] R.D. Richtmyer and K.W. Morton, *Difference Methods for Initial Value Problems* (Wiley, New York, 1967).
[17] G. Strang and J. Fix, *An Analysis of the Finite Element Method* (Prentice-Hall, Englewood Cliffs, NJ, 1973).
[18] H.A. van der Vorst, A vectorizable variant of some ICCG methods, *Siam J. Sci. Stat. Comput.* 3 (1982) 350–356.
[19] H.A. van der Vorst, *Parallel Rekenen en Supercomputers* (Academic Service, Schoonhoven, 1988).
[20] H.H. Wang, A parallel method for tridiagonal system equations, *ACM Trans. Math. Software* 7 (1981) 170–183.