Nonstationary Parallel Relaxed Multisplitting Methods

José Mas*

Departament de Matemàtica Aplicada
Universitat Politècnica de València
E-46071 València, Spain

Violeta Migallón† and José Penadés‡

Departament de Tecnologia Informàtica i Computació
Universitat d’Alacant
E-03080 Alacant, Spain

and

Daniel B. Szyld§

Department of Mathematics
Temple University
Philadelphia, Pennsylvania 19122-2585

Submitted by Ludwig Elsner

ABSTRACT

Relaxed nonstationary multisplitting methods are studied for the parallel solution of nonsingular linear systems. Convergence results of the synchronous and asynchronous versions for systems with H-matrices are presented. Computational results of these methods on a shared memory multiprocessor vector computer are reported. These results show that nonstationary methods (synchronous and asynchronous) are better than the standard ones, especially when the matrix of the linear system has a relatively small bandwidth. Moreover, asynchronous versions always behave better than the synchronous ones.
1. INTRODUCTION

We study the parallel solution of large linear systems of the form
\[ Ax = b \]  \hspace{1cm} (1)
using relaxed multisplitting methods. Here \( A \) is a square nonsingular matrix and \( x, b \in \mathbb{R}^n \). The multisplitting method was introduced by O'Leary and White [22] and further studied by different authors; see, e.g., [8, 10, 11, 21, 27–29]. Multisplitting can be seen as a generalization of the classic block Jacobi method, especially suited for parallel architectures; see, e.g., [6, 12, 13].

A set \( \{F_j, G_j, E_j\}_{j=1}^r \) is called a multisplitting of \( A \) if
\[ A = F_j - G_j, \quad j = 1, 2, \ldots, r, \]  \hspace{1cm} (2)
\( F_j \) are nonsingular, and \( E_j \), called weighting matrices, are nonnegative diagonal matrices such that \( \sum_{j=1}^r E_j = I \). Each splitting (2) induces an iterative method, described by the operator \( P_j: \mathbb{R}^n \rightarrow \mathbb{R}^n \) defined as
\[ P_jx = F_j^{-1}G_jx + F_j^{-1}b, \quad j = 1, 2, \ldots, r. \]  \hspace{1cm} (3)

In an efficient implementation of a multisplitting method on a multiprocessor computer, each operator \( P_j \) represents the task assigned to one of the \( r \) processors. The overall iterate vector is constructed using the \( r \) weighting matrices \( E_j, j = 1, 2, \ldots, r \), as
\[ x^{(l+1)} = \sum_{j=1}^r E_j P_j x^{(l)}, \quad l = 0, 1, 2, \ldots, \]
where the vector \( x^{(0)} \) is an initial approximation to the solution of (1).

In the standard multisplitting method each local approximation is updated exactly once using the same previous iterate \( x^{(l)} \). On the other hand, it is possible to update the local approximations more than once, using different iterates computed earlier. In this case, we call this method a nonstationary multisplitting method, cf. [6, 24]. The main idea of the nonstationary method is that at the \( l \)th iteration each processor \( j \) solves the system defined by its operator \( P_j, q(l, j) \) times, using in each time the new calculated vector to update the right-hand side; i.e., we have the following algorithm:

ALGORITHM 1.1 (Nonstationary Multisplitting). Given the initial vector \( x^{(0)} \)
For $l = 0, 1, \ldots$, until convergence

In processor $j$, $j = 1$ to $r$

\[ y_j^{(0)} = x^{(l)} \]

For $k = 1$ to $q(l,j)$

\[ F_j y_j^{(k)} = G_j y_j^{(k-1)} + b \]

\[ x^{(l+1)} = \sum_{j=1}^{r} E_j y_j^{(q(l,j))}. \]

Bru et al. [2] studied two nonstationary models (synchronous and asynchronous) based on the multisplitting method (they used the term “chaotic” for these nonstationary methods, but elsewhere in the literature, chaotic is synonymous with asynchronous, e.g., as in [7]). They showed the convergence of both models when $A$ is a monotone matrix, i.e., the inverse has nonnegative entries, denoted $A^{-1} \geq 0$, and the splittings (2) are weak regular; i.e., $F_j$ are monotone, and $F_j^{-1} G_j \geq 0$. Numerical experiments (see [4, 5, 16, 20, 23]) have shown that the nonstationary models have better performance than the standard (stationary) method in multiprocessors, provided a good load balancing among the processors is achieved.

When the linear systems (4) are not solved exactly, but instead their solutions approximated by iterative methods, one obtains a two-stage multisplitting method, [6, 18, 24, 25]. In this method, the splittings (2) are called outer splittings, and the splittings $F_j = M_j - N_j$, used to approximate the linear systems (4), are called inner splittings. When one performs a fixed number of inner iterations, one obtains a stationary two-stage multisplitting method; if the number of inner iterations varies for each splitting and for each outer iteration one has a nonstationary two-stage multisplitting algorithm. We point out that Algorithm 1.1 can be viewed as a special case of the nonstationary two-stage multisplitting methods, when the outer splittings are $A = A - O$.

In the context of relaxed (or extrapolated) methods, O'Leary and White [22] considered a multisplitting method using a relaxation parameter, as is done, e.g., in the extrapolated Jacobi method [30]. Frommer and Mayer [10] extended those results and proved the convergence of this extrapolated method when $A = [a_{ij}]$ is an $H$-matrix, i.e., when its comparison matrix

\[ \langle A \rangle = [\alpha_{ij}]: \alpha_{ij} = \begin{cases} |a_{ii}|, & i = j, \\ -|a_{ij}|, & i \neq j, \end{cases} \quad 1 \leq i, j \leq n, \]

is an $M$-matrix; see, e.g., [26].

In this paper we study relaxed nonstationary multisplitting methods when the matrix $A$ is an $H$-matrix. In other words, we extend the two models presented by Bru et al. [2] to the case considered by Frommer and
Mayer [10]. In Section 2, we construct the nonstationary synchronous and asynchronous models and, in Section 3, we study their convergence when $A$ is an $H$-matrix. The last section contains computational results on a shared memory multiprocessor vector computer.

In the rest of this section we present some notation, definitions, and preliminaries needed in the following sections.

For a vector $x \in \mathbb{R}^n$, $x \geq 0$ ($x > 0$) denotes that all components of $x$ are nonnegative (positive). Similarly, if $x, y \in \mathbb{R}^n$, $x \geq y$ ($x > y$) means that $x - y \geq 0$ ($x - y > 0$). For a vector $x \in \mathbb{R}^n$, $|x|$ denotes the vector whose components are the absolute values of the corresponding components of $x$. These definitions carry immediately over to matrices.

Let $x > 0$; we consider the vector norm

$$\|y\|_x = \inf\{\beta > 0 : -\beta x \leq y \leq \beta x\}. \quad (5)$$

This vector norm is monotonic and satisfies $\|Bx\|_x = \|B\|_x$, where $\|B\|_x$ denotes the matrix norm of $B$ corresponding to the norm defined in (5). Let $\rho(B)$ denote the spectral radius of $B$.

Let $A$ be an $H$-matrix. The splitting $A = M - N$ is called an $H$-splitting if the matrix $\langle M \rangle - |N|$ is monotone, and it is called an $H$-compatible splitting if $\langle A \rangle = \langle M \rangle - |N|$; see [14]. These definitions extend to multisplittings in a natural way.

**LEMMA 1.1.** If $A = D - B$ is an $H$-matrix, with $D = \text{diag}(A)$, then $|D|$ is nonsingular and \(\rho(|D|^{-1}|B|) < 1\).

**LEMMA 1.2.** If $A$ is an $H$-matrix, then $|A^{-1}| \leq \langle A \rangle^{-1}$.

**LEMMA 1.3.** Let $A = M - N$ be a splitting of $A$.

(i) If the splitting is an $H$-splitting, then $A$ and $M$ are $H$-matrices and \(\rho(M^{-1}N) \leq \rho(\langle M \rangle^{-1}|N|) < 1\).

(ii) If the splitting is an $H$-compatible splitting and $A$ is an $H$-matrix, then it is an $H$-splitting and hence, a convergent splitting.

The proofs of Lemmas 1.1 and 1.2 can be found, e.g., in [10], and the proof of Lemma 1.3 in [14].

2. NONSTATIONARY METHODS

Consider the solution of the nonsingular linear system (1) in a multiprocessor computer. Let $\{F_j, G_j, E_j\}_{j=1}^r$ be a multisplitting of $A$. Assume that
the task assigned to each processor \( j \) is to apply the operator \( P_j, 1 \leq j \leq r \), defined in (3), \( q(l,j) \) times in the \( l \)th iteration; we call the numbers \( q(l,j) \) the nonstationary parameters of the scheme. Then, given an initial vector \( x^{(0)} \), the relaxed nonstationary multisplitting method produces the sequence of vectors

\[
x^{(l+1)} = \omega \sum_{j=1}^{r} E_j P_j^{q(l,j)} x^{(l)} + (1 - \omega) x^{(l)}, \quad l = 0, 1, 2, \ldots
\]

The iterative scheme (6) is a rendition of the following algorithm:

**Algorithm 2.1 (Relaxed Nonstationary Multisplitting).** *Given the initial vector \( x^{(0)} \)

\[
\text{For } l = 0, 1, \ldots, \text{ until convergence}
\]

\[
\text{In processor } j, \ j = 1 \text{ to } r
\]

\[
y_j^{(0)} = x^{(l)}
\]

\[
\text{For } k = 1 \text{ to } q(l,j)
\]

\[
F_j y_j^{(k)} = G_j y_j^{(k-1)} + b
\]

\[
x^{(l+1)} = \omega \sum_{j=1}^{r} E_j y_j^{(q(l,j))} + (1 - \omega) x^{(l)}.
\]

It follows that when the nonstationary parameters \( q(l,j) = 1 \), for \( 1 \leq j \leq r, l = 0, 1, 2, \ldots \), Algorithm 2.1 reduces to the one studied by Frommer and Mayer [10].

Note that in this algorithm, all processors complete their local iterations before updating the global approximation \( x^{(l)} \). Thus, this algorithm is synchronous. To construct an asynchronous version of Algorithm 2.1 we consider a different scheme where all processors are always working without waiting for information from other processors. Moreover, it is not necessary that the processors update the global approximation at the same time.

Let \( \{j_l\}_{l=0}^{\infty}, 1 \leq j_l \leq r \) be a sequence of integers that indicates the number of the processor that updates the approximation to the solution at the \( l \)th iteration. Let \( r_l - 1 \) be the number of times that processors other than the \( j_l \)th processor update the approximation of the solution during the time interval in which the \( j_l \)th processor performs its calculations. This implies that \( r_l \) is the smallest positive integer such that \( j_l = j_{l+r_l} \). As is customary in the description of asynchronous algorithms (see, e.g., [1, 2]) we assume that the sequence of integers \( \{j_l\}_{l=0}^{\infty}, 1 \leq j_l \leq r \), is a regulated sequence. This means that there exists a positive integer \( K \) such that each of the integers \( 1, 2, \ldots, r \) appears at least once in every \( K \) consecutive elements of the sequence. It is easy to see that this condition is equivalent
to the assumptions considered by Frommer [9] and other authors, e.g., Frommer and Szyld [15] and the authors cited therein, in their description and analysis of asynchronous algorithms. With this notation we write the following asynchronous algorithm:

**Algorithm 2.2** (Relaxed Asynchronous Nonstationary Multisplitting). Given the initial vector \( x^{(0)} \)

\[
\begin{align*}
\text{In processor } j_l, l = 0, 1, \ldots, \text{ until convergence} \\
y^{(0)}_{j_l} &= x^{(l)}_{j_l} \\
\text{For } k = 1 \text{ to } q(l, j_l) \\
F_{j_l} y^{(k)}_{j_l} &= G_{j_l} y^{(k-1)}_{j_l} + b \\
y^{(q(l, j_l))}_{j_l} &= \omega y^{(q(l, j_l))}_{j_l} + (1 - \omega) x^{(l)}_{j_l} \\
x^{(l+1)}_{j_l} &= (I - E_{j_l})x^{(l+r_{l-1})} + E_{j_l} y^{(q(l, j_l))}_{j_l}.
\end{align*}
\]

Algorithm 2.2 can be rewritten as the following asynchronous iteration: for \( l = 0, 1, 2, \ldots, \)

\[
x^{(l+r_{l})} = (I - E_{j_l})x^{(l+r_{l-1})} + E_{j_l} \left( \omega P^{q(l,j_l)}_{j_l} + (1 - \omega)I \right) x^{(l)}, \tag{7}
\]

and it is a special case of the following asynchronous iteration: for \( l = 0, 1, 2, \ldots, \)

\[
x^{(l+r_{l})} = (I - E_{j_l})x^{(l+r_{l-1})} + E_{j_l} \left( \omega \sum_{j=1}^{r} E_j P^{q(l,j)}_{j} + (1 - \omega)I \right) x^{(l)}, \tag{8}
\]

when the weighting matrices satisfy for \( i, j = 1, \ldots, r, \)

\[
E_i E_j = E_i, \quad \text{if } i = j, \quad E_i E_j = 0, \quad \text{if } i \neq j. \tag{9}
\]

When the relations (9) do not hold, the formulations (7) and (8) represent different algorithms. In this latter case, the iteration (8) implies a certain degree of synchronization between groups of processors. In the next section, we prove the convergence of both asynchronous iterations (7) and (8), while the computational experiments reported in Section 4 correspond to Algorithm 2.2, i.e., the iteration (7).

3. CONVERGENCE

In this section we prove the convergence of Algorithms 2.1 and 2.2. The iteration (6) can be written as

\[
x^{(l+1)} = H^{(l)} x^{(l)} + c^{(l)}, \quad l = 0, 1, \ldots, \tag{10}
\]
where \( H^{(l)} \) are the iteration matrices

\[
H^{(l)} = \omega \sum_{j=1}^{r} E_j \left( F_j^{-1} G_j \right)^{q(l,j)} + (1 - \omega) I, \quad l = 0, 1, \ldots, (11)
\]

and

\[
c^{(l)} = \omega \sum_{j=1}^{r} E_j \left( \sum_{i=0}^{q(l,j)-1} \left( F_j^{-1} G_j \right)^{i} \right) F_j^{-1} b, \quad l = 0, 1, \ldots, (12)
\]

The exact solution \( \xi \) of the linear system (1) is a fixed point of the operators \( P_j, j = 1, 2, \ldots, r \). Thus, the sequence of error vectors \( e^{(l)} \) generated by the iteration (6) satisfies

\[
e^{(l+1)} = x^{(l+1)} - \xi = H^{(l)} e^{(l)} = H^{(l)} H^{(l-1)} \ldots H^{(0)} e^{(0)}, \quad l = 0, 1, \ldots (13)
\]

The following theorem shows the convergence of Algorithm 2.1 when \( A \) is an H-matrix and the relaxation parameter lies, as in [10], in the interval \((0, 2/(1 + \rho(|D|^{-1}|B|)))\), where \( D = \text{diag}(A) \) and \( A = D - B \). Note that from Lemma 1.1, the matrix \(|D|\) is nonsingular and \( \rho(|D|^{-1}|B|) < 1 \).

**Theorem 3.1.** Let \( A \) be an H-matrix. Assume that the multisplitting \( \{F_j, G_j, E_j\} \) is H-compatible and that \( \text{diag}(|F_j|) \leq |D| \). If \( 0 < \omega < 2/(1 + \rho) \), where \( \rho = \rho(|D|^{-1}|B|) \), and \( q(l,j) \geq 1 \) for all nonnegative integers \( l \) and \( 1 \leq j \leq r \), then the iteration (6) converges for all \( x^{(0)} \in \mathbb{R}^n \) to \( \xi \), the solution of the linear system \( Ax = b \).

**Proof.** By (13), the iteration (6) converges for all \( x^{(0)} \in \mathbb{R}^n \) if and only if \( \lim_{l \to \infty} H^{(l)} H^{(l-1)} \ldots H^{(0)} = O \). Hence it suffices to prove that

\[
\|H^{(l)}\| \leq \alpha, \quad l = 0, 1, \ldots, \text{for some } \alpha < 1, \text{and some matrix norm } \| \cdot \| \ [3, Lemma 2].
\]

By Lemma 1.3, the matrices \( F_j, j = 1, 2, \ldots, r \) are H-matrices. Then, using Lemma 1.2, some manipulation yields

\[
|F_j^{-1} G_j| \leq I - (F_j)^{-1} |D| (I - J), \quad j = 1, 2, \ldots, r, (14)
\]

where \( J = |D|^{-1}|B| \).

Consider the vector \( u = (1, 1, \ldots, 1)^T \in \mathbb{R}^n \). Since \( J = |D|^{-1}|B| \) is nonnegative, the matrix \( J + \epsilon uu^T \) is irreducible for all \( \epsilon > 0 \), and then (see e.g., [26]) there exists a positive Perron vector \( x_\epsilon \) such that

\[
(J + \epsilon uu^T)x_\epsilon = \rho_\epsilon x_\epsilon, (15)
\]
where \( \rho_\epsilon = \rho(J + \epsilon uu^T) \). The continuity of the spectral radius and Lemma 1.1 ensure that there exists \( \epsilon_0 \) such that \( \rho_\epsilon < 1 \), for all \( 0 < \epsilon \leq \epsilon_0 \). Since \( \langle F_j \rangle \leq |D|, \ j = 1, 2, \ldots, r \), from (14) and (15) we have that

\[
|F_j^{-1}G_j| x_\epsilon \leq \left[ I - \langle F_j \rangle^{-1}|D| (I - (J + \epsilon uu^T)) \right] x_\epsilon = x_\epsilon - \langle F_j \rangle^{-1}|D| (1 - \rho_\epsilon) x_\epsilon \leq \rho_\epsilon x_\epsilon.
\]  
(16)

We then obtain the following inequalities

\[
|H^{(l)}| x_\epsilon \leq \omega \sum_{j=1}^r E_j |F_j^{-1}G_j|^{q(l,j)} x_\epsilon + |1 - \omega| x_\epsilon \\
\leq \omega \sum_{j=1}^r E_j \rho_\epsilon^{q(l,j)} x_\epsilon + |1 - \omega| x_\epsilon \\
\leq (\omega \rho_\epsilon + |1 - \omega|) x_\epsilon, \quad l = 0, 1, \ldots.
\]

Moreover, as \( 0 < \omega < 2/(1 + \rho) \), then \( \omega \rho + |1 - \omega| < 1 \); and the continuity of the spectral radius guarantees that there exists \( \epsilon_1 \) such that \( \alpha_\epsilon = \omega \rho_\epsilon + |1 - \omega| < 1 \) for all \( 0 < \epsilon \leq \epsilon_1 \). Therefore, for all \( 0 < \epsilon \leq \min \{ \epsilon_0, \epsilon_1 \} \),

\[
|H^{(l)}| x_\epsilon < \alpha_\epsilon x_\epsilon,
\]  
(17)

where \( x_\epsilon > 0 \). Thus, using the matrix norm induced by the vector norm (5), one obtains \( \|H^{(l)}\| x_\epsilon \leq \alpha_\epsilon < 1, \quad l = 0, 1, \ldots. \)

We proceed now with the study of the convergence of the asynchronous iteration (8) under similar hypotheses as those for the convergence of the synchronous Algorithm 2.1.

**Theorem 3.2.** Let \( A \) be an \( H \)-matrix. Assume that the multisplitting \( \{F_j, G_j, E_j\}_{j=1}^r \) is \( H \)-compatible, that \( \text{diag}(|F_j|) \leq |D| \), and that \( \{j_l\}_{l=0}^\infty \), \( 1 \leq j_l \leq r \) is a regulated sequence. If \( 0 < \omega < 2/(1 + \rho) \), where \( \rho = \rho(|D|^{-1}|B|) \) and \( q(l,j) \geq 1 \) for all nonnegative integers \( l \) and \( 1 \leq j \leq r \), then the iteration (8) converges for all \( x^{(0)} \in \mathbb{R}^n \) to \( \xi \), the solution of the linear system \( Ax = b \).

**Proof.** We proceed in a way similar to that in the proof of Theorem 2.2 of [2]. To analyze the convergence of the asynchronous iteration (8), we embed this procedure in a procedure in \( \mathbb{R}^{nK} \), with \( K \) being the positive integer that regulates the sequence \( \{j_l\}_{l=1}^\infty \). For that purpose we use the
MULTISPLITTING METHODS 741

following notation. Let $e^{(l)} = x^{(l)} - x$ be the error in the $l$th iteration. Then, let

$$
\bar{e}_l = ((e^{(l)})^T, (e^{(l-1)})^T, \ldots, (e^{(l-K+1)})^T)^T \in \mathbb{R}^{nK},
$$

and $\bar{x} = (x_1^T, x_2^T, \ldots, x_K^T)^T \in \mathbb{R}^{nK}$, where $x_0 > 0$ is the Perron vector of the matrix $J + \epsilon uu^T$, with $J = |D|^{-1}|B|$, $\epsilon > 0$, and $u = (1, 1, \ldots, 1)^T$.

Then, the relation

$$
e^{(l)} = S_l \bar{e}_{l+r_l-1} \tag{18}
$$

holds, where $S_l$, $l = 0, 1, \ldots$, is an $n \times nK$ matrix with an $n \times n$ identity block in the $r_l$ position and the remainder $K - 1$ blocks are null. It is easy to see, using (8) and (18), that $\bar{e}_{l+r_l} = B_{l+r_l} \bar{e}_{l+r_l-1}$, where $B_{l+r_l}$ is the $(nK) \times (nK)$ matrix

$$
B_{l+r_l} = \begin{bmatrix}
I - E_{j_l} & 0 & \cdots & 0 & 0 \\
I & 0 & \cdots & 0 & O \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
O & O & \cdots & I & 0
\end{bmatrix} + \begin{bmatrix}
E_{j_l}H^{(l)}S_l \\
0 \\
\vdots \\
0
\end{bmatrix}, \tag{19}
$$

and $H^{(l)}$ is as defined in (11). It follows then that

$$
\bar{e}_{l+2K-1} = B_{l+2K-1}B_{l+2K-2} \cdots B_{l+1}\bar{e}_l.
$$

In Theorem 3.1 we showed that (17) holds with $\alpha_\epsilon < 1$, for all $0 < \epsilon \leq \epsilon'$, for some $\epsilon'$. Using (17) and reasoning in the same way as in the proof of Theorem 2.2 of [2], there exists $\bar{x} > 0$ and $0 \leq \gamma < 1$, such that

$$
|B_{l+2K-1}B_{l+2K-2} \cdots B_{l+1}|\bar{x} \leq \gamma \bar{x}.
$$

Therefore $\|B_{l+2K-1}B_{l+2K-2} \cdots B_{l+1}\|\bar{x} \leq \gamma$, where $\| \cdot \|\bar{x}$ is the matrix norm induced by the monotonic vector norm (5), and $\lim_{\nu \to \infty} \bar{e}_\nu = 0$. ◼

We conclude this section with several remarks on Theorems 3.1 and 3.2.

1. The convergence proof of Algorithm 2.2, or equivalently, iteration (7), can be carried out in the same way as that in the proof of Theorem 3.2 by using in (19) the matrix $H^{(l)}$ defined as $H^{(l)} = \omega(F_{j_l}^{-1}G_{j_l})^{q(l,j_l)} + (1 - \omega)I$.

2. Convergence proofs of iterations (6) and (8) can be obtained without the hypothesis $\operatorname{diag}(|F_j|) \leq |D|$, but in that case the upper bound of the range of the relaxation parameter can be characterized but not
easily computed. To see this, observe that if $A$ is an $H$-matrix and the splittings (2) are $H$-compatible, then $\langle A \rangle = \langle F_j \rangle - |G_j|$ are regular splittings. Reasoning as in Theorem 2.1 of [2], there exists a positive vector $x$ and a constant $0 \leq \beta < 1$ such that $(F_j)^{-1}|G_j|x \leq \beta x$. Then it is easy to show that $|H^{(l)}|x \leq (\omega \beta + |1 - \omega|)x$. Thus, in the range $0 < \omega < 2/(1 + \beta)$, the convergence of the iterations (6) and (8) follows as in the proofs of Theorems 3.1 and 3.2.

3. The conditions $\text{diag}(\langle F \rangle) \leq |D|$ and $\langle A \rangle = \langle F \rangle - |G|$ are satisfied if and only if $|f_{ij}| \in [0, |a_{ij}|]$, $i \neq j$ and $f_{ii} = a_{ii}$, $1 \leq i, j \leq n$, where $a_{ij}$ and $f_{ij}$ denote the entries of $A$ and $F$, respectively. These conditions are satisfied, for example, by the classic splittings of Jacobi or Gauss-Seidel.

4. One can use $r$ different relaxation parameters $\omega_1, \omega_2, \ldots, \omega_r$, associated with each of the operators $P_j$, $j = 1, 2, \ldots, r$, as is done, e.g., in the MSOR method; see, e.g., [17]. In this case, our convergence theorems remain valid, and the proofs are almost identical.

5. In the unrelaxed case $\omega = 1$, Algorithms 2.1 and 2.2 can be seen as special cases of Algorithms 4 and 6 of [6], respectively, when the outer splittings are all $A = A - O$ and the inner splittings are $A = F_j - G_j$, $j = 1, 2, \ldots, r$.

6. For $\omega = 1$, i.e., for the unrelaxed multisplitting methods, Algorithm 2.1 coincides with Model A of Bru et al. [2]. Similarly, for $\omega = 1$, and $q(l, j) = 1$ for $j = 1, \ldots, r$, $l = 1, 2, \ldots$, Algorithm 2.2 coincides with Model B of [2]. They proved the convergence when the matrix $A$ is monotone and all splittings are weak regular. Theorems 3.1 and 3.2 give other convergence conditions in the context of $H$-matrices. $H$-matrices are not necessarily monotone. Different interesting matrices, such as $M$-matrices, strictly or irreducibly diagonally dominant matrices, and matrices of which the comparison matrix is symmetric and positive definite, are $H$-matrices; see, e.g., [10, 14, 19].

4. COMPUTATIONAL EXPERIMENTS

To implement Algorithms 2.1 and 2.2, we have considered the problem of Laplace's equation satisfying boundary conditions in the rectangle $\Omega = [0, a] \times [0, b]$. We discretized the domain $\Omega$ with $M \times N$ points equally spaced by $h$. This discretization yields the linear system $Ax = b$, where $A$ is a block tridiagonal matrix, $A = \text{tridiag}[-I, C, -I]$, where $I$ and $C$ are $N \times N$, and $C = \text{tridiag}[-1, 4, -1]$. Therefore $A$ has $M \times M$ square blocks of order $N$. We have ran our experiments on a shared memory multiprocessor computer Alliant FX/80 with eight vector processors, using Fortran FX and parallelization directives. To define the splittings $A =$
\[ F_j - G_j, \quad j = 1, 2, \ldots, 8, \]

we can write \( A \) as

\[
A = \begin{bmatrix}
A_{11} & -I & & & \\
-I & A_{22} & -I & & \\
& -I & \ddots & \ddots & \\
& & -I & \ddots & -I \\
& & & -I & A_{88}
\end{bmatrix},
\]

where \( A_{jj} = \text{tridiag}[-I, C, -I], \quad j = 1, 2, \ldots, 8 \). We construct the matrices \( F_j = \text{diag}[a_{11}, a_{22}, \ldots, L_j, \ldots, a_{nn}], \quad j = 1, 2, \ldots, 8 \) where \( L_j \) is the lower triangular part of \( A_{jj} \). The matrices \( E_j, \quad j = 1, 2, \ldots, 8 \), are partitioned in the same way as the matrices \( F_j \), with the \( j \)th block equal to the identity and 0's elsewhere.

It is easy to see that the splittings are \( H \)-compatible and that \( A \) is an \( H \)-matrix; hence Theorems 3.1 and 3.2 assure the convergence for the synchronous and asynchronous algorithms, respectively. The stopping criterion used was \( \sum_{k=1}^{n} |x_k^{(l)} - x_k^{(l-1)}| < 5 \times 10^{-10} \).

We ran experiments with different matrix sizes. Generally, we observed a better behavior of the nonstationary models than that of the stationary ones (or classic parallel models). Nevertheless, we obtain the best results when the bandwidth of the matrix \( A \) is small, i.e., when the size of the blocks \( C \) is small compared with the size of the matrix \( A \). This is due to the fact that, in the nonstationary models, the larger the blocks \( C \), the larger the subsystem that each processor solves in each local iteration.

We have observed differences depending upon the bandwidth of \( A \), and the results are similar for all tested matrices with a similar bandwidth. Thus, we discuss the results for two representative matrices of size 4096, each with a different bandwidth. We refer to the matrix with smaller bandwidth as the matrix 4096 – \( S \) and to the matrix with larger bandwidth as the matrix 4096 – \( T \). The matrix 4096 – \( S \) has 64 blocks \( C \) of size 64, the size of \( A_{jj} \) is 1024 for \( j = 1, 2 \); 512 for \( j = 3, 4 \); and 256 for the rest. The matrix 4096 – \( T \) has 16 diagonal blocks \( C \) of size 256 and the distribution of the tasks assigned to each processor, i.e., the sizes of the matrices \( A_{jj}, \quad j = 1, 2, \ldots, 8 \), are the same as before.

For our experiments, we considered different synchronous and asynchronous nonstationary iterations depending on the nonstationary parameters \( q(l,j) \) and for each, we recorded the CPU time in seconds (for the concurrent compilation) as a function of different relaxation parameters.
Figs. 1 and 2 show the results for the two matrices described, in the synchronous case. The notation for the nonstationary parameters $1^2_2^3_4$ indicates that the first two processors update once its subvector ($q(l,1) = q(l,2) = 1$, $l = 1, 2, \ldots$), the next two processors update two times its subvector, ($q(l,3) = q(l,4) = 2$, $l = 1, 2, \ldots$), and the last four processors update the corresponding vector three times, in any global iteration ($q(l,5) = q(l,6) = q(l,7) = q(l,8) = 3$, $l = 1, 2, \ldots$). This notation is similar for all figures.
One can observe that for the matrix of size 4096 – S, the nonstationary iterations \((q(l,j) > 1)\) considered are better than the stationary one \((q(l,j) = 1)\) or classic parallel method. For the matrix of size 4096 – T (larger bandwidth) there are some good nonstationary iterations but not all are better than the stationary case. These considerations are also true for the nonstationary asynchronous schemes, as in Figs. 3 and 4. On the other hand, comparing Figs. 1 with 3 and 2 with 4, one observes that the asynchronous schemes are always better than the respective synchronous iterations.
We thank and appreciate the comments of an anonymous referee, which helped improve our presentation. J. M. was supported by the ESPRIT III basic research programme of the EC under Contract 9072 (project GEPPCOM). V. M. and J. P. were supported by Spanish CICYT Grant TIC91-1157-C09-01. D. B. S. was supported by National Science Foundation grants DMS-9201728 and INT-9521226.

REFERENCES


20 V. Migallón, Modelos Iterativos Caóticos Síncronos y Asíncronos para la Resolución de Sistemas Lineales, Ph.D. thesis, Departamento de Tecnología Informática y Computación, Universidad de Alicante, November 1993. [In Spanish.]


Received 28 November 1994; final manuscript accepted 15 July 1995