ELSEVIER Journal of Computational and Applied Mathematics 91 (1998) 261–273

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

A general asynchronous block iterative model with related convergence conditions

Yangfeng Su, Amit Bhaya *, Eugenius Kaszkurewicz

Parallel Computing Laboratory, COPPE/UFRJ, P.O. Box 68504, Rio de Janeiro, RJ 21945-970, Brazil

Received 5 November 1996; received in revised form 1 December 1997

Abstract

This paper formulates a general time-varying asynchronous block-iterative model. A convergence condition for asynchronous block-iterations based on this model is given, compared to existing conditions for similar models and shown to be strictly weaker.

AMS classificaiton: 65Y05; 65H10

Keywords: Block asynchronous iterations; Nonlinear equations; Parallel computing; Convergence conditions

1. Introduction

In recent times, with the advent of different kinds of parallel computers and distributed computing systems, there has been a resurgence of interest in asynchronous iterative methods. The seminal paper by Chazan and Miranker [9] that introduced the concept of asynchronism in iterative methods, using the term "chaotic relaxation" was followed by some work by the "French school" [17] in the seventies and again in the eighties by Baudet [4] and El Tarazi [21] who gave a general mathematical formulation and an inductive contraction mapping based convergence proof. These papers were followed by several others, notably [20] and a body of work by the "Russian school" [1, 14] that has largely been ignored in the Western literature. Bertsekas and Tsitsiklis [6], Üresin and Dubois [23] provided an excellent survey of the activity in the West in this area upto the late eighties, and also gave some general theoretical results. In the nineties, with parallel computers widely available, there has been more of an emphasis on applications [3, 7, 22], computable convergence conditions based, for example, on Liapunov theory [13], a large and growing literature on asynchronous multisplitting iterative methods [2, 8, 18, 19], the so-called asynchronous "team algorithms" [3, 20].

* Corresponding author. Fax: (55)(21)290-6626; e-mail: amit.bhaya@na-net.ornl.gov

0377-0427/98/$19.00
PII S0377-0427(98)00053-3

This paper proposes a model that is general enough to represent all the asynchronous iterative schemes referred to above, and gives a convergence condition for this model. Relationships between this condition and the other related conditions available in the literature are examined and it is shown that, theoretically, the new convergence condition is strictly weaker than the others.

2. A block asynchronous iteration model

Let E_i be a normed space, with norm $\|\cdot\|_i$, for $i = 1,\dots,N$. Let E be the Cartesian product $E_1 \times E_2 \times \dots \times E_N$, and given a positive vector $v > 0$, $v \in \mathbb{R}^N$, define a monotone norm on E as follows: for $x = (x_1^{\mathrm{T}},\dots,x_N^{\mathrm{T}})^{\mathrm{T}}$,

$$\|x\|_v = \max_{1\leqslant n\leqslant N} \frac{\|x_n\|_n}{v_n}, \tag{1}$$

and for $x_n \in E_n$, we introduce the notation

$$\|x_n\|_v = \frac{\|x_n\|_n}{v_n},$$

where x_n denotes the nth component of x.

Multisplitting and team algorithms [3, 8, 11, 19, 20] were designed in the specific context of implementation on parallel computers with several processors. Information that is specific to (and available in) a specific processor is termed local and any information that depends on a set of processors is called global. The distinguishing feature of multisplitting and team algorithms is that, during iterative computation of a given vector or subvector, advantage is taken of the fact that more than one processor may be involved in generating the information necessary to compute it. Thus, the new vector is a combination of the old vector and a vector representing some new information. This new information is, in turn, computed from combinations of some old global vectors. To make this verbal description more precise, we need a little terminology.

Given M vectors $x^1,\dots,x^M$ in E, a vector $z \in E$ is called a *combination* of $x^j, 1 \leqslant j \leqslant M$ if there exist M nonnegative diagonal matrices D_j, $1 \leqslant j \leqslant M$ such that

$$z = \sum_{j=1}^{M} D_j x^j, \qquad \sum_{j=1}^{M} D_j = I.$$

In the following, we denote vectors in E as x, y, z; the vector x_n denotes the nth (block) component of vector x. B is an integer that represents the upper bound on time delays; $S_1,\dots,S_k$ is a sequence of subsets of set $\{1,\dots,N\}$. $D_{n,j,k}$ denotes a diagonal matrix (in general nonnegative). G (with subscripts) is an iterative operator from one (sub)space to another (sub)space. k is used to denote the iteration index and i,j,n are used to denote the indices of components of a vector.

In this paper, we introduce the following block asynchronous iteration model. Consider a parallel computer consisting of N groups of processors. In the nth group there are s_n processors and associated to each processor, there is an operator $G_{n,j,k} : E \to E_i$, which depends on n (the group number), j (the processor number in this group) and k (the iteration counter). The nth component of the iterate will be updated by the nth group. For many reasons, for example, the speed of each processor, etc., at the kth iteration, only some groups (we use S_k to denote the set of these groups) will update their components of the iterate x^k. In group n, there are s_n different copies of x_n, one copy in each processor; at the kth iteration, only some processors (we use $U_{n,k}$ to denote the set of these processors) will enter the updating of the nth component. Mathematically,

$$x_n^{k+1} = \begin{cases} D_{n,0,k} y_n^k + \sum_{j \in U_{n,k}} D_{n,j,k} G_{n,j,k}(z^{n,j,k}) & \text{if} n \in S_k, \\ x_n^k & \text{if} n \notin S_k, \end{cases} \tag{2}$$

$$k = 0, 1, \ldots,$$

where $D_{n,j,k}$ are nonnegative diagonal matrices satisfying

$$D_{n,0,k} + \sum_{j \in U_{n,k}} D_{n,j,k} = I_n,$$

$\sum_{j \in U_{n,k}} D_{n,j,k}$ nonsingular, y^k and $z^{n,j,k}$ are combinations of $x^{k-B+1}, \ldots, x^k$, and the integer B is an upper bound on the delays.

This is a very general model. To understand it better, we first write down one of its simpler cases.

$$x_n^{k+1} = \begin{cases} G_{n,k}(z^{n,k}) & \text{if } n \in S_k, \\ x_n^k & \text{if } n \notin S_k, \end{cases} \tag{3}$$

where $z^{n,k}$ are combinations of $x^{k-B+1}, \ldots, x^k$, $G_{n,k}$ are maps: from E to E_n, $n = 1, \ldots, N$, and S_k are nonempty subsets of $\{1, \ldots, N\}$.

Consider a parallel computer consisting of N processors, then a parallel implementation of (3) can be described as follows: the nth component resides in nth processor, at the k iteration, some processors have their approximations $z^{n,k}$ for $n \in S_k$. The vector $z^{n,k}$ has the form

$$z^{n,k} = \begin{pmatrix} x_1^{k-k_1} \\ \vdots \\ x_N^{k-k_N} \end{pmatrix},$$

where $x_i^{k-k_i}$ comes from the ith processor. The integer k_i represents the iteration steps that this component needs to transfer from the ith processor to the n-th processor. Utilizing the vector $z^{n,k}$, the nth processor will form its new local approximation x_n^{k+1} according to (3). Other components whose indices are not in S_k will remain unchanged in this iteration step.

In the simplified model (3), if the operators $G_{n,k} = G_n$, i.e. $G_{n,k}$ do not vary with iteration number, we recover the bounded delay version of El Tarazi's model [21] which, in turn (without the bounded delay assumption) generalizes the models of Chazan and Miranker [9] and Baudet [4]. The bound on delays is assumed in this paper mainly in order to simplify notation. It is not difficult to generalize the model and the corresponding results to the case called 'total asynchronism' [6] or 'regular' asynchronism in the Russian literature [14] in which the delays need not necessarily be bounded but must satisfy a certain regularity assumption. It should also be noted that, in most practical implementations of asynchronous iteration algorithms, a uniform bound on time delays is either satisfied or enforceable, and therefore it is a realistic assumption.

Now consider *Multisplitting Asynchronous Iterations* (MAI). Multisplitting of a matrix was first introduced by O'Leary and White [16] and was generalized to nonlinear case by Frommer, see [11], and also studied in [2, 8, 18, 19], etc. An MAI can be written as

$$x^{k+1} = \Big(I - \sum_{l\in\widetilde{S}_k} E_l\Big)x^k + \sum_{l\in\widetilde{S}_k} E_l T_{k,l}(z^{k,l}), \tag{4}$$

where $T_{k,l} : \mathbb{R}^N \to \mathbb{R}^N$ are iteration operators, E_l are L nonnegative diagonal weighting matrices such that the matrix $\sum_{l=1}^{L} E_l$ nonsingular, $\widetilde{S}_k$ is a nonempty subset of $\{1,\dots,L\}$. The parallel implementation of an MAI is as follows: suppose a parallel computer consisting of $L+1$ processors, named as P0, P1, …, PL. For each Pl, $1 \leqslant l \leqslant L$, the following procedure is repeated: get a global approximation $z^{k,l}$ of the solution, which is a combination of $x^k, x^{k-1}, \dots$, from P0, compute the local approximation $E_l T_{k,l}(z^{k,l})$ and then send it to P0. P0 repeats the following procedure: check new local approximation from any other processors, if there are one or more, get it (them) and form a new global approximation according to (4).

In MAI, if $E_l E_j = 0$ for $l \neq j$, we say that there is no overlapping between the weighting matrices E_l, otherwise, we say that the weighting matrices are overlapping. An MAI with nonoverlapping weighting matrices is equivalent to the simplified version (3) of block asynchronous model (2). If there is overlapping between weighting matrices, split these weighting matrices such that either there is no overlapping between two split weighting matrices or two split weighting matrices overlap completely. For instance, $L = 2$, $N = 3$, $x = (x_1, x_2, x_3)^{\mathrm{T}}$, $E_1 = \mathrm{diag}(1,\alpha,0)$, $E_2 = \mathrm{diag}(0,\beta,1)$, $\alpha\beta \neq 0$, in this case there is an "overlapping" computation of the vector x_2 because the two weight matrices E_1 and E_2 are overlapping. Now we split

$$E_1 = D_1 + D_{2,1} \equiv \mathrm{diag}(1,0,0) + \mathrm{diag}(0,\alpha,0),$$

$$E_2 = D_{2,2} + D_3 \equiv \mathrm{diag}(0,\beta,0) + \mathrm{diag}(0,0,1).$$

Now there is no overlapping between any two D_1, $D_{2,1}$, D_3, while $D_{2,1}$ and $D_{2,2}$ overlap completely. Construct a block asynchronous iteration with three blocks which is equivalent to the MAI (4) as follows. Let

$$S_k = \begin{cases} \{1,2\} & \text{if } \widetilde{S}_k = \{1\}, \\ \{1,2,3\} & \text{if } \widetilde{S}_k = \{1,2\}, \\ \{2,3\} & \text{if } \widetilde{S}_k = \{2\}, \end{cases}$$

and in case that $S_k = \{1,2,3\}$,

$$x_2^{k+1} = (1-\alpha-\beta)x_2^k + \alpha G_{k,2,1}(z^{k,1}) + \beta G_{k,2,2}(z^{k,2}), \tag{5}$$

where $G_{k,2,l}$ is the component of $T_{k,l}$, $l = 1,2$, respectively. Therefore, MAI is a special case of the block asynchronous model (2).

One can see that if we allow the operators $G_{n,j,k}$ in (2) and the multisplitting operators $T_{k,l}$ in (4) to be identity operators, the block asynchronous iterative model (2) is equivalent to MAI. But, as we will see in the next section, from the point of view of convergence, we require that every iterative operator mentioned above have some contracting property, while the identity operator is never contracting. In Section 4, we give an example of a Team Algorithm, which can be written in the form of our proposed block asynchronous model (2), but not in the form of MAI. Thus, it may be said that model (2) is, strictly speaking, more general than MAI.

3. Convergence results

In this section, we give some convergence results for block asynchronous models (2) and (3).

Definition. A vector $z \in E$ is called a x^*-**combination** of the vectors $x^1,\dots,x^B$ if z is a combination of $x^1,\dots,x^B$ and for $1 \leqslant n \leqslant N$,

$$\min_{1\leqslant d\leqslant B} \|x_n^d - x_n^*\|_n \leqslant \|z_n - x_n^*\|_n \leqslant \max_{1\leqslant d\leqslant B} \|x_n^d - x_n^*\|_n. \tag{6}$$

We recall that z is a combination of $x^1,\dots,x^B$ if there exist B diagonal nonnegative matrices $D_1,\dots,D_B$ such that

$$z = \sum_{j=1}^{B} D_j x^j, \qquad \sum_{j=1}^{B} D_j = I. \tag{7}$$

Obviously, if $\|\cdot\|_n$ is a monotone norm or for all j: $1 \leqslant j \leqslant B$, $D_j = \alpha_j I$ with scalar α_j, then for any x^*, z is a combination of $x^1,\dots,x^B$ if and only if z is a x^*-combination. But this is not always true. For instance, choosing $N = 1, E = \mathbb{R}^2$ with Euclidean norm, $B = 2, x^1 = (1,0)^{\mathrm{T}}$, $x^2 = (0,1)^{\mathrm{T}}$, $D_1 = \mathrm{diag}(1,0)$, $D_2 = \mathrm{diag}(0,1)$, $z = D_1x^1 + D_2x^2$, $x_n^* = 0$, then $\|z\| = \sqrt{2} > \max\{\|x_n^1\|, \|x_n^2\|\} = 1$. So z is not a x^*-combination although it is a combination of $x^1,\dots,x^B$

In what follows, all combinations refer to x^*-combinations, where x^* is the fixed point of iteration operators involved. Our first theorem is for the model (3).

Theorem 1. *Let $\{S_k\}_k$ be admissible, i.e., for any integer $K > 0$,*

$$\bigcup_{k=K}^{\infty} S_k = \{1,\dots,N\}.$$

If there exists a constant q, $q < 1$, independent of k, such that for any $x \in B(x^*,\delta) \equiv \{x \mid \|x - x^*\|_v \leqslant \delta\}$,

$$\|G_{n,k}(x) - x_n^*\|_v \leqslant q\|x - x^*\|_v, \tag{8}$$

then $\{x^k\}$ defined by Eq. (3) converges to x^* as $k \to \infty$.

A variant of Theorem 1 can also be found in [12]. We include the proof here for completeness and also in order to point out that the proofs of Theorem 2 and 3 which follow are very similar in concept, but more complicated in terms of notation.

Proof. Set $x^{-B+1} = x^{-B+2} = \cdots = x^{-1} = x^0$. At first, we prove that for given K, $n \in S_K$,

$$\|x_n^k - x_n^*\|_v \leqslant q\max\{\|x^{K-B+1} - x^*\|_v,\dots,\|x^K - x^*\|_v\}, \quad k \geqslant K+1 \tag{9}$$

holds for $k \geqslant K+1$.

For $k = K+1$, from (8), we have

$$\|x_n^k - x_n^*\|_v \leqslant q\|z^{k,n} - x^*\|_v,$$

and $z^{k,n}$ is a combination of $x^{K-B+1},\dots,x^K$, by (3), so that (9) holds.
Suppose for $K+1 \leqslant k \leqslant K'$ (9) holds, consider $k = K'+1$. If $n \in S_{K'}$,

$$\begin{aligned}\|x_n^k - x_n^*\|_v &\leqslant q\|z^{K',n} - x^*\|_v \\ &\leqslant q\max\{\|x^{K'-B+1} - x^*\|_v,\dots,\|x^{K'} - x^*\|_v\} \\ &\leqslant q\max\{\|x^{K-B+1} - x^*\|_v,\dots,\|x^{K} - x^*\|_v\},\end{aligned}$$

and if $n \notin S_{K'}$,

$$\|x_n^k - x_n^*\|_v = \|z^{K',n} - x^*\|_v.$$

So (9) always holds.
Since $\{S_k\}_k$ is admissible, we can construct an integer sequence $\{K_j\}_j$ as follows: $K_0 = 0$, K_{j+1} is the smallest integer $\widetilde{K}$ such that

$$\bigcup_{k=K_j+B+1}^{\widetilde{K}} S_k = \{1,\dots,N\}.$$

Now it is easy to verify that for all $k > K_j$,

$$\|x^k - x^*\|_v \leqslant q^j\|x^0 - x^*\|_v.$$

From here we know that x^k converges to x^*. □

From (8), we always have

$$G_{n,k}(x^*) = x_n^*,$$

so the fixed point condition is given implicitly in the conditions of this theorem.
Now we give two convergence theorems for the general asynchronous iteration (2).

Theorem 2. *If $\{S_k\}_k$ is admissible, and there exists $q < 1$, independent of k, such that for any $x \in \boldsymbol{B}(x^*,\delta)$,*

$$\|G_{n,j,k}(x) - x_n^*\|_v \leqslant q\|x - x^*\|_v, \tag{10}$$

where $\|\cdot\|_n$ (see (1)) is a monotone norm and if

$$\alpha I_n \leqslant \sum_{j\in U_{n,k}} D_{n,j,k} \leqslant \frac{2}{1+q} I_n, \tag{11}$$

then, for any initial iterate $x^0 \in \mathrm{B}(x^*,\delta)$, $\lim_{k\to\infty} = x^*$.

In this theorem, we can see that a relaxation factor can also be included in model (2) by properly choosing the weighting matrices $D_{n,j,k}$. The factor $2/(1+q)$ is a well-known bound for the relaxation factor in nonnegative matrix iteration, see for example, [5, 19].

Theorem 3. *Let $\{S_k\}_k$ be admissible, and suppose that for every $G_{n,j,k}$, there exists a $q_{n,j,k}$ such that for any $x \in \boldsymbol{B}(x^*, \delta)$,*

$$\|G_{n,j,k}(x) - x_n^*\|_v \leqslant q_{n,j,k}\|x - x^*\|_v, \tag{12}$$

and also that there exists $q < 1$, independent of k, for $D_{n,j,k} = \beta_{n,j,k}I$, $\beta_{n,j,k} > 0$, $\beta_{n,0,k} = 1 - \sum_{j \in U_{n,k}} \beta_{n,j,k}$, such that

$$\sum_{j \in U_{n,k}} \beta_{n,j,k} q_{n,j,k} + |\beta_{n,0,k}| < q. \tag{13}$$

Then, for any initial iterate $x^0 \in \boldsymbol{B}(x^*, \delta)$,

$$\lim_{k \to \infty} x^k = x^*.$$

In (12), we do not require that $q_{n,j,k} < 1$ holds for each triple n, j, k, while this is so in (10). In the next section we will give an example in which there exist some $q_{n,j,k}$ greater than 1.

Proofs of Theorem 2 and Theorem 3 are very similar to the proof of Theorem 1. We only state the main fact that requires proof here: namely, there exists a $\tilde{q}$, $\tilde{q} < 1$, independent of k, such that for $n \in S_k$,

$$\|x_n^{k+1} - x_n^*\|_v \leqslant \tilde{q} \max_{0 \leqslant d \leqslant B-1} \|x^{k-d} - x^*\|_v. \tag{14}$$

4. An example: Team algorithm

In this section, we present an example of a Team Algorithm (TA). It will be shown below that this example can be written in the form of the block model (2), however it cannot be written in the form of an MAI. It is also an example to which only Theorem 3 can be applied while Theorem 2 cannot.

For most problems, there are usually many solution methods. Sometimes one method works well, sometimes another. Occasionally none of the methods will work well alone, but certain combinations of the methods can be effective. Such combinations are referred to as team algorithms, see [20]. Team algorithms were generalized in [3] and here we consider the so-called "TA without administrator". Suppose that we have two processors P1 and P2, and that vector x is partitioned as $x = (x_1^{\mathrm{T}}, x_2^{\mathrm{T}}, x_3^{\mathrm{T}})^{\mathrm{T}}$. Component x_1 is updated by P1, x_2 is updated by P2, x_3 is updated by both processors and there exist two different versions of x_3, one resides in P1, and is denoted by x_{31}, the other resides in P2, and is denoted by x_{32}. If P1 needs x_2, it will get the latest available x_2 from P2; if P2 needs x_1, it will get the latest available x_1 from P1. Thus the TA can be written as:

At the k-th iteration, P1 forms a new approximation as follows:

$$\begin{bmatrix} x_1^{k+1} \\ \chi_{31}^{k+1} \\ x_{31}^{k+1} \end{bmatrix} := \begin{bmatrix} G_{11}(z^{k,1}) \\ G_{31}(z^{k,1}) \\ c_1 x_{31}^k + \omega_{11}\chi_{31}^k + \omega_{21}\chi_{32}^{k1} \end{bmatrix} \tag{15}$$

Similarly, P2 forms a new approximation:

$$\begin{bmatrix} x_2^{k+1} \\ \chi_{32}^{k+1} \\ x_{32}^{k+1} \end{bmatrix} := \begin{bmatrix} G_{22}(z^{k,2}) \\ G_{32}(z^{k,2}) \\ c_2 x_{32}^k + \omega_{12}\chi_{31}^{k2} + \omega_{22}\chi_{32}^k \end{bmatrix}. \tag{16}$$

In (15), $z^{k,1}$ is the latest available approximate solution in P1, and is expressed as

$$z^{k,1} = \begin{pmatrix} x_1^k \\ x_2^{k-d(k,1)} \\ x_{31}^k \end{pmatrix},$$

χ_{31} is an intermediate vector which serves to form a new x_{31}. $c_1, \omega_{11}, \omega_{12}$ are scalars (possible time-varying), χ_{32}^{k1} is the latest available value of χ_{32} at the kth iteration in P1, which comes from P2 and is formed at time k_1. Similar notation is used in (16).

Now we write TA (15) and (16) in the form of the general block asynchronous iteration (2). We construct a vector sequence $\{y^l\}_{l=0,1,\ldots}$ as follows.

(1) $y_1^0 = x_1^0, y_2^0 = x_2^0, y_3^0 = x_3^0, l := 0$;

(2) For $k = 0, 1, 2, \ldots$ do

if only P1 updates its local approximation at time k,

$$y_1^{l+1} := x_1^{k+1}, \quad y_2^{l+1} := y_2^l, \quad y_3^{l+1} := x_{31}^{k+1}; \quad l := l+1, \tag{17}$$

if only P2 updates its local approximation at time k,

$$y_1^{l+1} := y_1^l, \quad y_2^{l+1} := x_2^{l+1}, \quad y_3^{l+1} := x_{32}^{k+1}; \quad l := l+1, \tag{18}$$

if both P1 and P2 update local approximations at time k,

$$y_1^{l+1} := x_1^{k+1}, \quad y_2^{l+1} := y_2^l, \quad y_3^{l+1} := x_{31}^{k+1}; \tag{19}$$

$$y_1^{l+2} := y_1^{l+1}, = \quad y_2^{l+2} := x_2^{k+1}, \quad y_3^{l+2} := x_{32}^{k+1}; \quad l := l+2. \tag{20}$$

We note that every term x_1^k, x_2^k, x_{31}^k and x_{32}^k in (15) and (16) (if it exists) appears as one component of some vector y^l. In case of (17) (Eqs. (18)–(20) can be interpreted similarly),

$$y_1^{l+1} = G_{11}(u^{l,1}), \tag{21}$$

$$y_3^{l+1} = c_1 w_3^l + \omega_{11} G_{31}(u^{l,3,1}) + \omega_{12} G_{32}(u^{l,3,2}), \tag{22}$$

where $u^{l,1}, w^l, u^{l,3,1}$ and $u^{l,3,2}$ are in the form of

$$\begin{pmatrix} y_1^{l-d(l,1)} \\ y_2^{l-d(l,2)} \\ y_3^{l-d(l,3)} \end{pmatrix}. \tag{23}$$

Any vector in the form of (23) is the combination of $y_1^{l-d(l,1)}, y_2^{l-d(l,2)}$ and $y_3^{l-d(l,3)}$. Block component w_3^l equals y_3^l if $l-1$ corresponds to $t(l-1)$ and is equal to y_3^{l-d} for some $d \geqslant 1$, otherwise. So (21) and (22) are in the form of (2), hence a TA is a special case of block asynchronous iteration (2).

In most cases, we can prove that the operators $G_{n,j,k}$ are strictly contracting, cf. condition (10), thus we can apply Theorem 2 to obtain convergence. If the identity operator is viewed as one of the $G_{n,j,k}$, the strict contraction condition is not satisfied, and unless more condition(s), such as (13), are satisfied, convergence cannot be proved. In other words, in the example, the term w_3^l cannot be replaced by y_3^l, so the Team Algorithm can not be written in the form of Multisplitting Asynchronous Iteration mentioned above.

Talukdar et al. [20] gave an example as follows. Suppose that there is only one block component, i.e., the dimension of the first and the second block component is zero. Consider the problem of solving the nonlinear equation

$$F(x) = 0.$$

Two possible algorithms are the Newton–Raphson (NR) algorithm and an optimization approach (minimize some norm of the residual), for example, the steepest-descent (SD) method.

$$x^{k+1} = x^k + \omega_{\mathrm{NR}}^k \Delta x_{\mathrm{NR}}^k + \omega_{\mathrm{SD}}^k \Delta x_{\mathrm{SD}}^k, \quad k = 0, 1, 2, \ldots, \tag{24}$$

where Δx_{NR}^k and Δx_{SD}^k are the NR and SD directions, respectively; ω_{NR}^k and ω_{SD}^k are the stepsizes, which are given by the TA. There are some numerical experiments in [20] showing that sometimes the NR method works well and sometimes the SD method works well, but the TA is always better than each individual method. In case the NR method or the SD method do not work, i.e. the corresponding iterative operators are not contracting, we need more conditions, e.g., condition (13), and then Theorem 3 may be applied to prove convergence.

5. Discussion of the convergence conditions

In the previous convergence theorems, we gave a contraction condition (8) based on a monotone norm to get the convergence of the block asynchronous iteration. The contraction conditions in Theorems 2 and 3 are given in the light of condition (8). In this section we compare this condition with those given in earlier papers, first listing some of them.

$$\begin{aligned} &\|G_{n,k}(x) - G_{n,k}(y)\|_n \leqslant \sum_{j=1}^{N} h_{n,j}\|x_j - y_j\|_j, \quad \rho(H) < 1, H \geqslant 0, \\ &\text{for all } x, y \in \boldsymbol{B}(x^*, \delta), k, n \in S(k), \end{aligned} \tag{25}$$

$$\|G_{n,k}(x) - x^*\|_n \leqslant \sum_{j=1}^{N} h_{n,j}\|x_j - x_j^*\|_j, \quad \rho(H) < 1, H \geqslant 0, \\ \text{for all } x \in \boldsymbol{B}(x^*, \delta), k, n \in S(k), \tag{26}$$

$$\|G_{n,k}(x) - G_{n,k}(y)\|_v \leqslant q\|x - y\|_v, \quad q < 1, \\ \text{for all } x, y \in \boldsymbol{B}(x^*, \delta), k, n \in S(k), \tag{27}$$

$$\|G_{n,k}(x) - x_n^*\|_v \leqslant q\|x - x^*\|_v, \quad q < 1, \\ \text{for all } x \in \boldsymbol{B}(x^*, \delta), k, n \in S(k), \tag{28}$$

$$\|G^k(x) - x^*\|_v \leqslant q\|x - x^*\|_v, \quad q < 1, \\ \text{for all } x \in \boldsymbol{B}(x^*, \delta), k. \tag{29}$$

Condition (25) is a generalization of the one proposed by Miellou in [15]. Condition (29), with $G^k = G$, reduces to the one given by Baudet in [4]. If G^k belongs to a finite set, this was used by Su in [19]. The following proposition shows the relations between the above conditions.

Proposition 4. *If condition* (26) *holds, then condition* (8) *holds.*

Proof. From Perron–Frobenius theory [5] since $H > 0, \rho(H) < 1$, there exists a $v \in \mathbb{R}^N, v > 0, q < 1$, such that

$$Hv \leqslant qv. \tag{30}$$

Therefore,

$$\begin{aligned} \|G_{n,k}(x) - G_{n,k}(y)\|_v &= \frac{\|G_{n,k}(x) - G_{n,k}(y)\|_n}{v_n} \\ &\leqslant \frac{1}{v_n}\sum_{j=1}^{N} h_{n,j}\|x_j - y_j\|_j \\ &= \frac{1}{v_n}\sum_{j=1}^{N} h_{n,j}v_j\frac{\|x_j - y_j\|_j}{v_j} \\ &\leqslant \frac{1}{v_n}\sum_{j=1}^{N} h_{n,j}v_j\|x - y\|_v \\ &\leqslant q\|x - y\|_v. \quad \square \end{aligned}$$

This proposition was observed by Miellou [15]. The simple proof given here is also valid for nonstationary operators. With this proposition, we have.

$$(25) \Rightarrow (26) \Rightarrow (8),$$
$$(27) \Rightarrow (28) \Rightarrow (8),$$
$$(29) \Rightarrow (28) \Rightarrow (8).$$

Now we show that the converses of the above relations are not valid.

(26) $\not\Rightarrow$ (25): For example, $N = 1$, $G(x) = \frac{1}{2}\sin(x^2)$, for $x^* = 0$, $q = \frac{1}{2}$, $\delta = \infty$, (26) holds. But there does not exist $H \in \mathbb{R}^{1\times 1}$ such that for all $x, y \in \mathbb{R}$,

$$|g(x) - g(y)| \leqslant H|x - y|.$$

(8) $\not\Rightarrow$ (26): Note that for $N = 1$, (8) and (26) are the same condition. We now give a counterexample for $N = 2$, $q = 0.8$, $\delta = \infty$, $v = (1, 1)^{\mathrm{T}}$, $\|\cdot\|_n = \|\cdot\|_\infty$, and

$$G^k(x) = \begin{cases} Ax, & x_1 \geqslant 0, \\ Bx, & x_1 < 0, \end{cases}$$

where

$$A = qI,$$

$$B = q\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Obviously, (8) holds for $n = 1, 2$.

Suppose there exists an H such that for all $x \in \mathbb{R}^2$,

$$|G_1^k(x)| \leqslant h_{11}|x_1| + h_{12}|x_2|, \tag{31}$$

$$|G_2^k(x)| \leqslant h_{21}|x_1| + h_{22}|x_2|, \tag{32}$$

and if we can prove $\rho(H) > 1$, we obtain the conclusion.

From (31), (32), we have for all $x \in \mathbb{R}^2$,

$$|G^k(x)| \leqslant H|x|. \tag{33}$$

For all $x > 0$, from (33), we have

$$Ax = |G^k(x)| \leqslant Hx, \tag{34}$$

$$Bx = |B(-x)| = |G^k(-x)| \leqslant H|(-x)| = Hx. \tag{35}$$

But (34) and (35) mean that $H \geqslant A$, and $H \geqslant B$, hence,

$$H \geqslant q\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}.$$

From nonnegative matrix theory [5] one has $\rho(H) \geqslant 2q > 1$.

(28) $\not\Rightarrow$ (27): Same example as in (26) $\not\Rightarrow$ (25).

(8) $\not\Rightarrow$ (28): Other components of $|x - x^*|$ may be larger than $|x_n - x_n^*|$.

(28) $\not\Rightarrow$ (29): In (28), there is no restriction on those components of G^k which do not enter the iteration. □

So, theoretically, (8) is a weaker condition than the others that have appeared in the literature so far. In practice, when we need to test whether an operator satisfies the contraction condition, conditions (25)–(29) may be easier to check than (8).

Remark The example $y = \sin(x^2)$ (which is similar to the one used to show that (26) $\not\Rightarrow$ (25)) was used by Elsner et al. in [10] to demonstrate the difference between a paracontracting operator and a strictly nonexpansive operator.

Acknowledgements

This research was partially supported by grants from the Brazilian Government agencies CAPES and CNPq under the auspices of the research productivity and PRONEX programs. The first author is a CNPq Visting Research Fellow at the Parallel Computing Laboratory at COPPE/UFRJ, which is also supported in part by a grant from FINEP. The authors would also like to thank the anonymous reviewers for several valuable suggestions.

References

[1] E.A. Asarin, V.S. Kozyakin, M.A. Krasnosel'skii, N.A Kuznetsov, Stability analysis of desynchronized discrete-event systems, Nauka, Russia, 1992(in Russian).
[2] Z. Bai, D. Wang, D.J. Evans, Models of asynchronous parallel matrix multisplitting relaxed iterations, Parallel Comput. 21 (1995) 565–582.
[3] B. Barán, E. Kaszkurewicz, A. Bhaya, Parallel asynchronous team algorithms: convergence and performance analysis, IEEE Trans. Parallel Distributed Systems 7 (7) (1996) 677–688.
[4] G. Baudet, Asynchronous iterative methods for multiprocessors, J. Assoc. Comput. Mach. 25 (1978) 226–244.
[5] A. Berman, B. Plemmons, Nonnegative Matrices in the Mathematical Sciences, Academic Press, New York, 1979.
[6] D.P. Bertsekas, J.N. Tsitsiklis, Parallel and Distributed Computation, Prentice-Hall, Englewood Cliffs, NJ, 1989.
[7] A. Bhaya, E. Kaszkurewicz, F.C. Mota, Asynchronous block-iterative methods for almost-linear equations, Linear Algebra Appl. 154–156 (1991) 487–508.
[8] R. Bru, L. Elsner, M. Neumann, Models of parallel chaotic iteration methods, Linear Algebra Appl. 103 (1988) 175–192.
[9] D. Chazan, W.L. Miranker, Chaotic relaxation, Linear Algebra Appl. 2 (1969) 190–222.
[10] L. Elsner, I. Koltracht, M. Neumann, Convergence of sequential and asynchronous nonlinear paracontractions, Numer. Math. 62 (1992) 305–319.
[11] A. Frommer, Parallel nonlinear multisplitting methods, Numer. Math. 56 (1989) 269–282.
[12] A. Frommer, D. Szyld, Asynchronous two-stage iterative methods, Numer. Math. 69 (1994) 141–153.
[13] E. Kaszkurewicz, A. Bhaya, D.D. Šiljak, On the convergence of parallel asynchronous block-iterative computations, Linear Algebra Appl. 131 (1990) 139–160.
[14] A.F. Kleptsyn, V.S. Kozyakin, M.A. Krasnosel'skii, N.A. Kuznetsov, Effect of small synchronization errors on stability of complex systems. Parts i, ii, iii, Automation and Remote Control 44 (1983) 861–867; 45 (1984) 309–314; 45 (1984) 1014–1018.
[15] J.C. Miellou, Itérations chaotiques à retards, C.R. Acad. Sci. Paris, 278 (1974) 957–960.
[16] D.P. O'Leary, R.E. White, Multisplittings of matrices and parallel solution of linear systems, SIAM J. Algebraic Discrete Methods 6(4) (1985) 630–640.
[17] F. Robert, Contraction en norme vectorielle: Convergence d'iterations chaotiques pour des equations non lineaires de point fixe a plusieurs variables, Linear Algebra Appl. 13 (1976) 19–35.
[18] Y. Song, D. Yuan, On the convergence of relaxed parallel chaotic iterations for H-matrix, Internat. J. Comput. Math. 52 (1994) 195–209.
[19] Y. Su, Generalized multisplitting asynchronous iteration, Linear Algebra Appl. 235 (1996) 77–92.

[20] S.N. Talukdar, S.S. Pyo, R. Mehrotra, Designing algorithms and assignments for distributed processing, Technical Report EL-3317, Electric Power Research Institute, Carnegie-Mellon University, 1983. Research Project 1764–3.
[21] M. El Tarazi, Some convergence results for asynchronous algorithms, Numer. Math. 39 (1982) 325–340.
[22] P. Tseng, D.P. Bertsekas, J.N. Tsitsiklis, Partially asynchronous parallel algorithms for network flow and other problems, SIAM J. Control Optim. 28(3) (1990) 678–710.
[23] A. Üresin, M. Dubois, Asynchronous iterative algorithms: Models and convergence, in: L. Kronsjö, D. Shumsheruddin (Eds.), Advances in Parallel Algorithms, Oxford, London, 1992.