Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational and Applied Mathematics 176 (2005) 91–103

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

www.elsevier.com/locate/cam

Asynchronous iterations with flexible communication: contracting operators

Didier El Baz[a], Andreas Frommer[b,*,1], Pierre Spiteri[c]

[a] *Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS, 7, avenue du Colonel Roche, F-31077 Toulouse CEDEX 4, France*

[b] *Fachbereich Mathematik und Naturwissenschaften, Universität Wuppertal, Gauss-Strasse 20 D-42097 Wuppertal, Germany*

[c] *Ecole Nationale Supérieure d'Electrotechnique, d'Electronique, d'Informatique et d'Hydraulique de Toulouse, 2, rue Camichel, B.P. 7122, F-31071 Toulouse CEDEX 1, France*

Received 18 November 2003; received in revised form 2 June 2004

Abstract

The concept of flexible communication permits one to model efficient asynchronous iterations on parallel computers. This concept is particularly useful in two practical situations. Firstly, when communications are requested while a processor has completed the current update only partly, and secondly, in the context of inner/outer iterations, when processors are also allowed to make use of intermediate results obtained during the inner iteration in other processors.

In the general case of nonlinear or linear fixed point problems, we give a global convergence results for asynchronous iterations with flexible communication whereby the iteration operators satisfy certain contraction hypotheses. In this manner we extend to a contraction context previous results obtained for monotone operators with respect to a partial ordering.

MSC: 65Y05; 68Q10; 68Q22

Keywords: Asynchronous iterations; Parallel computing; Flexible communication; Fixed point methods

* Corresponding author.

E-mail addresses: elbaz@laas.fr (D. El Baz), frommer@math.uni-wuppertal.de (A. Frommer), pierre.spiteri@enseeiht.fr (P. Spiteri).

[1] The work of this author was partially funded by the French Ministry of Research through a visiting grant at ENSEEIHT.

0377-0427/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2004.07.009

1. Introduction

Parallel computers work efficiently only if the work load for a given computation between two synchronisation points can be distributed evenly among the processors. At a synchronisation point, processors generally need data which have been computed by other processors, so that usually they have to wait until the other processors have finished their computation and, in the case of distributed memory architectures, the communication of the data has been accomplished. There are situations where synchronisation may become a decisive bottleneck. For example, on supercomputers with several thousands of processors, synchronisation can become costly due to technical restrictions. Moreover in many applications, and in particular for nonlinear problems, it can be difficult to predict the computational cost of each parallel task, so that an even distribution of the work load between the processors cannot be achieved *a priori*. A similar situation arises if the parallel computer in use is a cluster of heterogeneous workstations which, in addition, may not be available in dedicated mode. Then, the computational power available on each processor is unpredictable, so that it is again impossible to obtain a fair assignation whereby each parallel task requires equal time between two synchronisation points. In such situations it can be advantageous to use the asynchronous paradigm instead of the synchronous one. We think in particular of iterative processes whereby each iterative step produces an approximation to the solution of a given problem. Then, classically, synchronisation will occur at the beginning of each iterative step where processors build up the value of the current iterate from the data computed in all processors. In the asynchronous case, these synchronisation points are completely skipped. Therefore, if certain processors perform their iterative step faster than others, then the processors will get 'out of phase'. When building up 'their' current iterate, the processors will now use data from other processors which will not correspond to the data used in the synchronized algorithm. In this manner, the asynchronous paradigm tends to eliminate idle times due to synchronisation. On the other hand, the resulting asynchronous iteration is less structured, and there is a need for theoretical results concerning the convergence and the speed of convergence of such methods.

Asynchronous iterations have been studied and implemented by many authors for a variety of different applications. Any attempt to list all relevant publications is beyond the scope of this paper. Instead, we refer to the overview article [11] and the book [5] for references in the case of linear and nonlinear systems of equations and minimization problems and the very recent papers [1–3] for multisplitting ideas and applications to complementarity problems. The recent paper [17] analyses for the first time asynchronous iterations from a stochastic perspective.

In this study, we further develop on a recent and general class of asynchronous iterations for linear and nonlinear fixed point problems which has first been brought forward in a mathematical form in [8,13]. This concept, called 'asynchronous iterations with flexible communication' allows for an even larger degree of freedom on when and how communications are to be performed than the classical model for asynchronous iterations (see [7,12,4,5]). In particular, the flexible communication model is well suited to the following two situations which we call the 'partial update' situation and the 'inner/outer' context.

In the partial update situation, each processor has several components, say a block, of the iterate vector to update, and it may happen that another processor requests data at a moment when this processor has updated only a part of the components of its block. In the classical asynchronous model, communication of data would have to be delayed until the update is completed. This actually introduces an undesirable partial synchronisation together with idle times. In the asynchronous model with flexible communication we avoid this drawback and allow for the possibility to communicate data as soon as it is requested, even if updates are completed only partly.

In the inner/outer context, we assume that we have an iteration function where some 'inner' iterations have to be performed to approximate the value of the 'outer' iteration function at a given point. The flexible communication model allows intermediate results from the inner iteration in a given processor to be used by the other processors. As a consequence intermediate results are not labelled by an outer iteration index. The basic idea is that such an intermediate value usually represents a better approximation to the solution than the outcome of the previous outer iteration, the only value that would be available for the other processors in the classical asynchronous model. This approach was shown to be very performant in a partial ordering context in [8,13,15] (and also [16] in a multisplitting context). Indeed, in the partial ordering context, the updates as well as the intermediate values converge monotonically to the solution of the problem (see in particular [13]). As a consequence, using intermediate values, i.e. for example, the last available values, corresponds to using a better approximation to the solution. In the case of linear systems of algebraic equations, reference is also made to [18,10] for studies concerning asynchronous iterations with flexible communication (see also, in this case, [9,6,11]).

In the general case of nonlinear or linear fixed point problems, convergence results for asynchronous iterations with flexible communication have been obtained in [8,13] in situations where the iteration function is monotone with respect to the natural partial ordering and where the whole process can, loosely speaking, be described via appropriate monotone 'superfunctions' of the basic iteration function. For further details, the reader is referred to [13]. Reference is also made to [10], for the convergence of two-stage methods in the linear case (see also [9,6]).

The purpose of the present paper is to extend the global convergence results obtained in the partial ordering context for nonlinear and linear fixed point problems and published in [13] to the case where the basic iteration function is contracting. It is important to notice that if one already gets a convergence result in the partial ordering context, then to establish such a result in the case of contracting operators is not straightforward at all. We note in particular that obtaining global convergence results in the case of nonlinear fixed point problems and contracting operators is a difficult issue due to the complexity of the studied parallel iterative schemes and in particular, the lack of synchronization points. We will see in the sequel, that due to the difference of contexts, it has been possible to propose in this paper a mathematical model for asynchronous iterations with flexible communication which is complementary to the one presented in [13] since it does not rely on monotonicity assumptions. Finally, we will also establish the connection with well-known results for the case of classical asynchronous iterations.

The rest of this paper is organized as follows: we describe the different computational and mathematical models for asynchronous iterations in Section 2. We then develop a convergence theory for contracting operators in Section 3. An example is presented in Section 4.

2. Computational and mathematical models

We start by setting the stage for the iterations to be considered. As suggested in [18,10], we will distinguish between computational and mathematical models. A computational model is given in the form of a pseudocode corresponding to the Single Program Multiple Data (SPMD). It explains the implementation of the method. A mathematical model formulates the functional relation between the iterates. It is used to analyze the iteration.

Let $H : \mathbb{R}^n \to \mathbb{R}^n$ denote the basic linear or nonlinear iteration function. We are interested in computing a fixed point x^* of H by parallel variants of the successive approximation scheme:

$$x^{k+1} = H(x^k), \quad k = 0, 1, \ldots, \ (x^0 \text{ given}). \tag{1}$$

We view the space $\mathbb{R}^n$ primarily as a Cartesian product of subspaces which represents blocks. In the simplest case, each block is just one component in $\mathbb{R}^n$, but certain iteration functions H naturally induce larger blocks. For example, the function H could be derived from the solution of several smaller systems, each of these systems being associated with one such block. To be precise, therefore, let $I_i, \ i = 1, \ldots, b$ be a partitioning of the set $\{1, \ldots, n\}$ into non-overlapping blocks, i.e.

$$\bigcup_{i=1}^{b} I_i = \{1, \ldots, n\}, \quad I_i \neq \emptyset \quad \text{for } i = 1, \ldots, b, \quad I_i \cap I_j = \emptyset \quad \text{for } i \neq j.$$

Given a vector $x \in \mathbb{R}^n$, we use subscripts to denote the blocks corresponding to the above partitioning and we will refer to such a part of x as a *component* in the sequel. So, component $x \in \mathbb{R}^n$ is from $\mathbb{R}^{n_i}$ where $n_i = |I_i|$. By using this notation, another way of writing the successive approximation scheme (1) is thus:

$$x_i^{k+1} = H_i(x_1^k, \ldots, x_b^k), \quad i = 1, \ldots, b, \quad k = 0, 1, \ldots, \tag{2}$$

where H_i denotes the ith component of the operator H.

On a parallel computer with p processors, we now assign a set of components to each processor $P_j, \ j=1, \ldots, p$. We therefore have an additional decomposition of $\{1, \ldots, b\}$ into (possibly overlapping) subsets $S_j, \ j = 1, \ldots, p$, of the set $\{1, \ldots, b\}$, and the basic synchronous computational model is given by the following algorithm; the mathematical model being just (2).

Algorithm 1. (Basic synchronous computational model, pseudocode for processor P_t; each processor running the same pseudocode):

```
repeat until convergence
  wait for all processors to have finished previous sweep
    through loop
  for i = 1, ..., b
    read x_i from processor P_r (where i ∈ S_r)
  for i ∈ S_t
    compute x_i ← H_i(x_1, ..., x_b)
```

Remark 1. We note that if two subsets S_r and $S_{r'}$ overlap and $i \in S_r, \ i \in S_{r'}$ then the values for x_i computed by processors P_r and $P_{r'}$ are equal.

In situations where H_i is given only implicitly, one has to perform another, 'inner' iteration to evaluate approximately $H_i(x_1, \ldots, x_b)$. At this point we just modify Algorithm 1 by introducing an inner loop computing an approximation for $H_i(x)$.

Algorithm 2. (Inner/outer synchronous computational model, pseudocode for processor P_t)

```
repeat until convergence
  wait for all processors to have finished previous sweep
    through loop
  for i = 1, ..., b
    read x_i from processor P_r (where i ∈ S_r)
  set x̂ = (x_1, ..., x_b)
  for i ∈ S_t
    repeat until precise enough
      compute new approximation ẋ_i for H_i(x̂)
    set x_i = ẋ_i
```

Here, the stopping criterion 'precise enough' for the inner iteration means that the inner iterate $\dot{x}_i$ is sufficiently close to $H_i(x)$. The mathematical model for this iteration is given as follows:

$$x^{k+1} = G^k(x^k), \quad k = 0, 1, \ldots \tag{3}$$

where, for each k, the function $G^k : \mathbb{R}^n \to \mathbb{R}^n$ gives the result of the inner loop, and for $i = 1, \ldots, b$ the component $G_i^k(x^k)$ approximates $H_i(x^k)$. In a practical situation, the functions G_i^k might be obtained for example via several iterations, $q_i(k)$ say, of a successive approximation scheme or some other iterative method which approximates $H_i(x^k)$, starting, with x_i^k. If we denote the corresponding iteration function by $T_{i,x^k} : \mathbb{R}^{n_i} \to \mathbb{R}^{n_i}$, we then have

$$y_{i,k}^0 = x_i^k, \quad y_{i,k}^{q+1} = T_{i,x^k}(y_{i,k}^q), \quad q = 0, \ldots, q_i(k) - 1; \quad G_i^k(x^k) = y_{i,k}^{q_i(k)}. \tag{4}$$

The 'classical' asynchronous counter-part to the synchronous inner/outer method corresponding to Algorithm 2 arises by just skipping the wait statement.

Algorithm 3. (Inner/outer asynchronous computational model, pseudocode for processor P_t)

```
repeat until convergence
  for i = 1, ..., b
    read x_i from processor P_r (where i ∈ S_r)
  set x̂ = (x_1, ..., x_b)
  for i ∈ S_t
    repeat until precise enough
      compute new approximation ẋ_i for H_i(x̂)
    set x_i = ẋ_i
```

Remark 2. If two subsets S_r and $S_{r'}$ overlap and $i \in S_r$, $i \in S_{r'}$ then processor P_t can take the most recent value of x_i.

Since processors may perform their iterations at different speeds, a mathematical model for this method reads as follows, where G^k is as in (3). For all $i \in \{1, \ldots, b\}$ and $k = 0, 1, \ldots$

$$x_i^{k+1} = \begin{cases} G_i^k(x_1^{s_1(k)}, \ldots, x_b^{s_b(k)}) & \text{if } i \in J_k, \\ x_i^k & \text{otherwise,} \end{cases} \tag{5}$$

where the indices $s_j(k)$ denote appropriate previous time steps. The iteration index k must now be interpreted as a global counter which is stepped by one every time *any* processor starts a new sweep through the outer loop. The additional superscripts $s_j(k)$ model the fact that the data used may come from different steps of the global iteration. Finally, for each k, the set $J_k \subseteq \{1, \ldots, b\}$ corresponds to the set S_t in the computational model assuming that iteration k is performed by processor P_t.

The following weak assumptions have become standard in asynchronous convergence theory (see [4]). Virtually, they are fulfilled in any practical implementation. For all $i \in \{1, \ldots, b\}$:

$$0 \leqslant s_i(k) \leqslant k, \quad k = 0, 1, \ldots \tag{6}$$

$$\lim_{k \to \infty} s_i(k) = +\infty, \tag{7}$$

$$\text{the set } \{k \mid i \in J_k\} \text{ is unbounded.} \tag{8}$$

If, in addition, we want to make the inner iterates more generally the current value of any component of the iterate vector available to the other processors, we end up with asynchronous iterations with flexible communication. To describe the computational model we just replace the `read` x_i statement by `read` $\tilde{x}_i$ in Algorithm 3 to indicate that a processor may now read at any time the current result of the inner iteration or more generally the current value of any component x_i from the other processors.

Algorithm 4. (Asynchronous computational model with flexible communication, pseudocode for processor P_t)

```
repeat until convergence
  for i = 1, ..., b
    read x̃_i from processor P_r (where i ∈ S_r)
  set x̂ = (x̃_1, ..., x̃_b)
  for i ∈ S_t
    repeat until precise enough
      compute new approximation ẋ_i for H_i(x̂)
    set x_i = ẋ_i
```

Note that the last statement which defines x_i is not necessary for the algorithm to work; however, the statement is helpful to formulate the mathematical model.

Remark 3. If two subsets S_r and $S_{r'}$ overlap and $i \in S_r$, $i \in S_{r'}$ then processor P_t can take the most recent value of $\tilde{x}_i$. If it is not possible to determine what is the most recent value, then the decision can be taken arbitrarily on the basis of the last received data.

Mathematically, an asynchronous iterative sequence with flexible communication $\{x^k\}$ is described by the following modification of (5). For all $i \in \{1, \ldots, b\}$ and $k = 0, 1, \ldots$

$$x_i^{k+1} = \begin{cases} G_i^k(\tilde{x}_1^k, \ldots, \tilde{x}_b^k) & \text{if } i \in J_k, \\ x_i^k & \text{otherwise.} \end{cases} \tag{9}$$

In order to establish convergence results, it is necessary to somehow restrict the possible values $\tilde{x}_i^k$. This was done in [13] and [8] in the context of partial ordering, see also [15]. In the present paper, we take a different, complementary approach, and impose the following norm constraint.

$$\|\tilde{x}_i^k - x_i^*\|_i / u_i \leqslant \|x^{s(k)} - x^*\|_u \quad \text{for all } i = 1, \ldots, b, \tag{10}$$

where $x^{s(k)}$ denotes the vector $(x_1^{s_1(k)}, \ldots, x_b^{s_b(k)})$.

Remark 4. It is important to note that in the above model of asynchronous iterations with flexible communication (9)–(10), where we assume that (6)–(8) are true, we do not need any monotonicity condition concerning the access to the components of the iterate vector, such as, for example, the hypothesis (3.9) of [13]. This is because our convergence results will be based on contraction and not an partial ordering techniques. As a consequence, the class of parallel iterative methods considered in this paper is in some sense complementary to those considered in [13] and even more general.

3. Convergence results

In the convergence theory for asynchronous iterations, weighted maximum norms play a prominent role.

Definition 1. Let $u \in \mathbb{R}^b$ be a vector, all components u_i of which are positive. For $i = 1, \ldots, b$ let $\|\cdot\|_i$ be some norm $\mathbb{R}^{n_i}$. Then the weighted maximum norm $\|\cdot\|_u$ in $\mathbb{R}^n = \mathbb{R}^{n_1} \times \cdots \times \mathbb{R}^{n_b}$ is defined as follows

$$\|x\|_u = \max_{i=1}^{b} \|x_i\|_i / u_i.$$

For the standard asynchronous iteration given by Algorithm 3 and the mathematical model (5), the following global convergence result is basically contained in [11].

Theorem 1. *Assume that in* (5) *all operators G^k have a common fixed point in the sense that there exists $x^* \in \mathbb{R}^n$ such that for all $i \in \{1, \ldots, b\}$ we have*

$$G_i^k(x^*) = x_i^*, \quad k = 0, 1, \ldots .$$

In addition, assume that the following uniform contraction hypothesis with respect to x^* is fulfilled: *there exists a weighted maximum norm $\|\cdot\|_u$ and $\alpha \in [0, 1)$ such that for all k we have*

$$\|G^k(x) - x^*\|_u \leqslant \alpha \cdot \|x - x^*\|_u \quad \textit{for all } x \in \mathbb{R}^n.$$

Moreover, suppose that assumptions (6)–(8) *are satisfied. Then the iterates* x^k *of the asynchronous iteration* (5) *converge to* x^*:

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

Since synchronous iterations can be viewed as special asynchronous iterations (take, for example, $s_j(k) = k$ and $J_k = \{1, \ldots, b\}$), the result of Theorem 1 also holds for the iteration (3). Actually, in this case the result could be rephrased using any norm rather than just a weighted maximum norm. For the asynchronous case, the weighted maximum norm is necessary: it was already shown in [7] that for an affine operator $G^k(x) = Ax + c$ for all k, which is not contracting with respect to a weighted maximum norm, there exists an asynchronous iteration satisfying (6)–(8) such that its iterates do not converge.

We now develop a global convergence result similar to Theorem 1 for the case of an asynchronous iteration with flexible communication.

Theorem 2. *Assume that there exists a weighted maximum norm* $\|\cdot\|_u$, *a contraction constant* $\alpha \in [0, 1)$ *and a common fixed point* x^* *for all operators* G^k *such that for* $k = 0, 1, \ldots$

$$\|G^k(x) - x^*\|_u \leqslant \alpha \cdot \|x - x^*\|_u \quad \text{for all } x \in \mathbb{R}^n. \tag{11}$$

Assume that in the asynchronous iteration with flexible communication $\{x^k\}$ *given in* (9) *the usual conditions* (6)–(8) *are satisfied as well as* (10). *Then,* $\lim_{k\to\infty} x^k = x^*$.

Proof. The proof follows the lines of the standard proof in the non-flexible case. We show that there exists a sequence of integers k_p, such that we have

$$\|x^k - x^*\|_u \leqslant \alpha^p \cdot \|x^0 - x^*\|_u \quad \text{for all } k \geqslant k_p. \tag{12}$$

We proceed by induction and start by showing that (12) is true for $p = 0$ with $k_0 = 0$, i.e. we show

$$\|x^k - x^*\|_u \leqslant \|x^0 - x^*\|_u \quad \text{for } k \geqslant k_0 = 0. \tag{13}$$

Trivially, (13) is true for $k = 0$. Assume that (13) holds up to some k. For $k + 1$ we then have by (9) and (11):

$$\begin{aligned} \|x_i^{k+1} - x_i^*\|_i/u_i &= \|x_i^k - x_i^*\|_i/u_i, && \text{for } i \notin J_k, \\ \|x_i^{k+1} - x_i^*\|_i/u_i &= \|G_i^k(\tilde{x}_1^k, \ldots, \tilde{x}_b^k) - x_i^*\|_i/u_i \\ &\leqslant \alpha \cdot \|(\tilde{x}_1^k, \ldots, \tilde{x}_b^k) - x^*\|u, && \text{for } i \in J_k. \end{aligned} \tag{14}$$

From (14) and (10) we get, in the case where $i \in J_k$

$$\begin{aligned} \|x_i^{k+1} - x_i^*\|_i/u_i &\leqslant \|(\tilde{x}_1^k, \ldots, \tilde{x}_b^k) - x^*\|_u = \max_{j=1}^{b} \|\tilde{x}_j^k - x_j^*\|_j/u_j \\ &\leqslant \|x^{s(k)} - x^*\|_u \leqslant \|x^0 - x^*\|_u, \end{aligned}$$

where the last inequality is due to the induction hypothesis. We therefore see that (13) is true for $k + 1$. We continue the proof of (12) by induction on p; the validity for $p = 0$ just having been shown. Assume that (12) is true up to some p and define k_{p+1} as follows. Let q_{p+1} be the smallest positive integer such

that $s_j(k) > k_p$ for all $j = 1, \ldots, b$ and for all $k \geqslant q_{p+1}$. Such a value exists because of (7). Let k_{p+1} be the minimal value for k such that

$$\bigcup_{\ell=q_{p+1}}^{k} J_\ell = \{1, \ldots, b\},$$

which exists by (8). Then for $k \geqslant k_{p+1}$ each component i has been updated at least once at some iteration ℓ with $\ell \geqslant q_{p+1}$, and the corresponding indices $s_j(\ell)$ satisfy

$$s_j(\ell) \geqslant k_p \quad \text{for all } j \in \{1, \ldots, b\}. \tag{15}$$

Thus, we can write

$$x_i^k = x_i^{k-1} = \cdots = x_i^{\ell+1} = G_i^\ell(\tilde{x}^\ell).$$

From (11) we therefore have

$$\begin{aligned} \|x_i^k - x_i^*\|_i/u_i &= \|G_i^\ell(\tilde{x}^\ell) - x_i^*\|_i/u_i \leqslant \alpha\|\tilde{x}^\ell - x^*\|_u \\ &= \alpha \max_{j=1}^{b} \|\tilde{x}_j^\ell - x^*\|_j/u_j \leqslant \alpha \max_{j=1}^{b} \|x^{s_j(\ell)} - x^*\|_u, \end{aligned}$$

where the last inequality is due to (10). But by (15) we have $s_j(\ell) \geqslant k_p$ for all j, so that the induction hypothesis yields $\max_{j=1}^{b} \|x^{s_j(\ell)} - x^*\|_u \leqslant \alpha^p \|x^0 - x^*\|_u$. This proves (12) for $p + 1$.

In the case where the functions G_i^k correspond to some inner iteration according to (4), it is of interest to discuss when the assumptions on the G_i^k in Theorems 1 and 2 are fulfilled. This is formulated in the following lemma.

Lemma 1. *Assume that the functions H_i are contracting with respect to x_i^* in the norm $\|\cdot\|_i$, and assume that all operators $T_{i,x}$ from* (4) *are contracting with respect to $H_i(x)$ in the norm $\|\cdot\|_i$. This means that we have numbers $\beta \in [0, 1)$ and $\beta_{i,x} \in [0, 1)$ such that*

$$\|H_i(x) - x_i^*\|_i \leqslant \beta \cdot \|x_i - x_i^*\|_i \quad \textit{for all } x \in \mathbb{R}^n \textit{ and for all } i, \tag{16}$$

$$\|T_{i,x}(x_i) - H_i(x)\|_i \leqslant \beta_{i,x}\|x_i - H_i(x)\|_i \quad \textit{for all } x \in \mathbb{R}^n \textit{ and for all } i. \tag{17}$$

Then for all $i \in \{1, \ldots, b\}$,

$$\|G_i^k(x) - x^*\|_i/u_i \leqslant (\beta_{i,x}^{q_i(k)}(\beta + 1) + \beta) \cdot \|x - x^*\|_u.$$

In particular, if for each k and i we take $q_i(k)$ sufficiently large such that $(\beta_{i,x}^{q_i(k)}(\beta + 1) + \beta) \leqslant \alpha < 1$, then assumption (11) *is satisfied.*

Proof. First note that from (4) we immediately get, by using (17)

$$\|G_i^k(x) - H_i(x)\|_i \leqslant \beta_{i,x}^{q_i(k)}\|x_i - H_i(x)\|_i.$$

Together with (16) this yields

$$\begin{aligned}
\|G_i^k(x) - x_i^*\|_i &\leqslant \|G_i^k(x) - H_i(x)\|_i + \|H_i(x) - x_i^*\|_i \\
&\leqslant \beta_{i,x}^{q_i(k)} \cdot \|x_i - H_i(x)\|_i + \|H_i(x) - x_i^*\|_i, \\
&\leqslant \beta_{i,x}^{q_i(k)} \cdot (\|x_i - x_i^*\|_i + \|x_i^* - H_i(x)\|_i) + \|H_i(x) - x_i^*\|_i, \\
&\leqslant \beta_{i,x}^{q_i(k)} \cdot \|x_i - x_i^*\|_i + (1 + \beta_{i,x}^{q_i(k)}) \cdot \|H_i(x) - x_i^*\|_i, \\
&\leqslant \beta_{i,x}^{q_i(k)} \cdot \|x_i - x_i^*\|_i + (1 + \beta_{i,x}^{q_i(k)}) \cdot \beta \cdot \|x_i - x_i^*\|_i, \\
&= (\beta_{i,x}^{q_i(k)}(\beta + 1) + \beta) \cdot \|x_i - x_i^*\|_i.
\end{aligned}$$

Dividing by u_i, and majorizing $\|x_i - x_i^*\|_i/u_i$ by $\|x - x^*\|_u$ proves the lemma. □

4. Application to linear systems

We consider a linear system

$$Ax = c, \tag{18}$$

where $A = (a_{ij}) \in \mathbb{R}^{n\times n}$ is non-singular. Recall that A is called a nonsingular M-matrix if $a_{ij} \leqslant 0$ for $i \neq j$ and $A^{-1} \geqslant 0$, where this inequality is to be understood componentwise. Nonsingular M-matrices arise in a variety of applications, particularly in finite difference discretizations of elliptic boundary value problems; see e.g., [14].

We decompose $\mathbb{R}^n = \mathbb{R}^{n_1} \times \cdots \times \mathbb{R}^{n_b}$ into b blocks (with $\sum_{i=1}^b n_i = n$) and partition A, x, c accordingly into blocks A_{ij}, x_i, c_i. The solution x^* of (18) can be characterized as the fixed point of H, the components H_i of which are given as

$$H_i(x) = A_{ii}^{-1}\left(c_i - \sum_{j=1, j\neq i}^{b} A_{ij}x_j\right), \quad i = 1, \ldots, b.$$

Note that together with A the diagonal blocks A_{ii} are nonsingular M-matrices, too (see for e.g., [10]), so that the inverses A_{ii}^{-1} all exist.

Evaluating H_i is costly because one has to solve systems with the matrix A_{ii}. We therefore introduce additional splittings

$$A_{ii} = B_i - C_i, \quad i = 1, \ldots, b,$$

and perform $q_i(x)$ steps of the iteration

$$\begin{aligned}
&y_i^0 = x_i \\
&\text{for } q = 0, \ldots, q_i(x) - 1 \\
&\text{solve } B_i y_i^{q+1} = C_i y_i^q + (c_i - \textstyle\sum_{j=1, j\neq i}^{b} A_{ij}x_j).
\end{aligned}$$

Let us denote

$$\mathscr{G}_i^{q_i(x)}(x) = y_i^{q_i(x)}, \tag{19}$$

the result of this iteration which should be an approximation to $H_i(x)$.

We now consider an asynchronous iteration with flexible communication (9), where in each iteration k the function G_i^k is taken to be $\mathscr{G}_i^{q(k)}$ with some non-zero value for $q(k)$. For the block components $\tilde{x}_\ell^k$ in (9) we assume

$$\tilde{x}_\ell^k \in \{\mathscr{G}_\ell^0(x^{s_\ell(k)}), \ldots, \mathscr{G}_\ell^{q(k)}(x^{s_\ell(k)})\}, \tag{20}$$

which means that $\tilde{x}_\ell^k$ may be any of the *intermediate* results y_ℓ^q, $q = 0, \ldots, q(k)$ of the iteration above which evaluates $G_\ell^k(x^k)$.

We will now show that under the given M-matrix assumptions and under suitable assumptions on the splittings $A_{ii} = B_i - C_i$, the crucial conditions (11) and (10) for Theorem 2 are all fulfilled, i.e. we have convergence of the asynchronous iteration with flexible communication.

For this purpose, let us assume that all splittings $A_{ii} = B_i - C_i$, are weak regular, i.e. we have

$$B_i^{-1} \geqslant 0, \quad B_i^{-1} C_i \geqslant 0, \quad i = 1, \ldots, b. \tag{21}$$

Note that this condition is fulfilled for the most important standard splittings like the Jacobi and the Gauss-Seidel splitting, since A_{ii} is a nonsingular M-matrix. This condition applies also to ILU factorizations.

We need the following auxiliary result.

Lemma 2. *Let A be a non-singular M-matrix and $A_{ii} = B_i - C_i$ be weak regular according to* (21). *Let $D = \mathrm{diag}(B_i)$ be the block diagonal matrix with diagonal blocks B_i and let $A = D - F$, $T - D^{-1}F$. Moreover, let $e = (1, \ldots, 1)^{\mathrm{T}}$ be the vector in R^n with all components equal to one and let $v = A^{-1}e$. Then*

(i) $v > 0$, $T \geqslant 0$ *and*

$$Tv \leqslant \alpha v \text{ for some } \alpha \in [0, 1). \tag{22}$$

(ii) *Defining the weighted max norm $\|\cdot\|_v$ in R^n as*

$$\|x\|_v = \max_{i=1}^{b} \frac{|x_i|}{v_i} = \|V^{-1}x\|_\infty, \quad V = \mathrm{diag}(v),$$

we have

$$\|Tx\|_v \leqslant \alpha \|x\|_v, \tag{23}$$

and thus

$$\|T\|_v = \max_{i=1}^{n} \frac{1}{v_i} \sum_{j=1}^{n} |t_{ij}| \cdot v_j = \|V^{-1}TV\|_\infty \leqslant \alpha.$$

(iii) *Using weighted max norms on the blocks x_i and $T_{ij} \in \mathbb{R}^{n_i \times n_j}$ defined through*

$$\|x_i\|_i = \|V_i^{-1}x\|_\infty, \tag{24}$$

$$\|T_{ij}\|_{ij} = \|V_i^{-1}T_{ij}V_j\|_\infty, \quad V_i, V_j \text{ diagonal blocks of } V, \tag{25}$$

we have for $i = 1, \ldots, b$

$$\sum_{j=1}^{b} \|T_{ij}\|_{ij} = \|B_i^{-1}C_i\|_{ii} + \sum_{j=1, j\neq i}^{b} \|B_i^{-1}A_{ij}\|_{ij} \leqslant \alpha. \tag{26}$$

In particular, for $i = 1, \ldots, b$

$$\|B_i^{-1}C_i\|_{ii} < 1. \tag{27}$$

Proof. Since A is a nonsingular M-matrix and the splittings are weak regular, we have $T \geqslant 0$. The inverse of A is a nonnegative matrix without zero rows. This shows that all components of $v = A^{-1}e$ have to be positive. Using $T = I - D^{-1}A$, where, again, D is nonnegative without nonzero rows, we see that

$$Tv = v - D^{-1}e < v,$$

which finishes the proof for (22) in (i). Since T is nonnegative, part (ii) is just a reformulation of (22). This is also the case for (26), which restates (23) in a block oriented manner.

With these preparations we can now prove the following result. It improves upon Lemma 1, because it shows that with (20) conditions (11) and (10) are satisfied without further restriction.

Theorem 3. *Let* A *and* $A_{ii} = B_i - C_i$ *be as in Lemma 2 and take* α *from* (22). *Moreover, define the norms* $\|\cdot\|_i$ *on the blocks as in Lemma* 2. *Then we have, for* $\mathscr{G}^q$ *defined in* (19), $q \geqslant 1$,

$$\|\mathscr{G}^q(x) - x^*\|_e \leqslant \alpha\|x - x^*\|_e, \tag{28}$$

and, in particular,

$$\|\mathscr{G}_i^q(x) - x^*\|_i \leqslant \alpha\|x - x^*\|_e, \quad q = 0, 1, \ldots .$$

Here, $e = (1, \ldots, 1)^{\mathrm{T}} \in R^b$, i.e. $\|\cdot\|_e$ *denotes the maximum norm built up from norms* $\|\cdot\|_i$ *on the blocks as*

$$\|x\|_e = \max_{i=1}^{b} \|x_i\|_i.$$

Proof. The solution x^* of (18) is a fixed point of $\mathscr{G}^q$ for all q, so that we have

$$\begin{aligned}\mathscr{G}_i^q(x) - x_i^* &= \mathscr{G}_i^q(x) - \mathscr{G}_i^q(x^*),\\ &= B_i^{-1}C_i(\mathscr{G}_i^{q-1}(x) - x_i^*) - B_i^{-1}\left(\sum_{j=1, j\neq i}^{b} A_{ij}(x_j - x_j^*)\right).\end{aligned}$$

This gives

$$\begin{aligned}V_i^{-1}(\mathscr{G}_i^q(x) - x_i^*) &= V_i^{-1}B_i^{-1}C_iV_i(V_i^{-1}(\mathscr{G}_i^{q-1}(x) - x_i^*))\\ &\quad - \sum_{j=1, j\neq i}^{b} V_i^{-1}B_i^{-1}A_{ij}V_j(V_j^{-1}(x_j - x_j^*)),\end{aligned}$$

which, by using the norms previously defined, yields

$$\|\mathscr{G}_i^q(x) - x_i^*\|_i \leqslant \|B_i^{-1}C_i\|_{ii} \cdot \|\mathscr{G}_i^{q-1}(x) - x_i^*\|_i + \sum_{j=1, j\neq i}^{b} \|B_i^{-1}A_{ij}\|_{ij} \cdot \|x_j - x_j^*\|_j.$$

Via induction on q and by using (26) as well as (22) we obtain

$$\|\mathscr{G}_i^q(x) - x_i^*\|_i \leqslant \alpha \cdot \|x - x^*\|_e,$$

for all $q \geqslant 1$ and i and thus (28). Therefore, the asynchronous iterative method with flexible communication converges for linear systems with nonsingular M-matrices. □

Remark 5. Similar results can be obtained for H-matrix by using H-splittings, see [10].

References

[1] Z. Bai, Experimental study of the asynchronous multisplitting relaxation methods for the linear complementarity problems, J. Comput. Math. 20 (2002) 561–574.
[2] Z. Bai, D.J. Evans, Matrix multisplitting methods with applications to linear complementarity problems: parallel asynchronous methods, Internat. J. Comput. Math. 79 (2002) 205–232.
[3] Z. Bai, Y. Huang, A class of asynchronous parallel multisplitting relaxation methods for large sparse linear complementarity problems, J. Comput. Math. 21 (2003) 773–790.
[4] G. Baudet, Asynchronous iterative methods for multiprocessors, J. Assoc. Comput. Mach. 25 (1978) 226–244.
[5] D. Bertsekas, J. Tsitsiklis, Parallel and Distributed Computation, Numerical Methods, Prentice-Hall, Englewood Cliffs, NJ, 1989.
[6] R. Bru, V. Migallón, J. Penadés, D. Szyld, Parallel synchronous and asynchronous two-stage multisplitting methods, Electronic Trans. Numer. Anal. 3 (1995) 24–38.
[7] D. Chazan, W. Miranker, Chaotic relaxation, Linear Algebra Appl. 2 (1969) 199–222.
[8] D. El Baz, P. Spiteri, J.C. Miellou, D. Gazen, Asynchronous iterative algorithms with flexible communication for nonlinear network flow problems, J. Parallel Distrib. Comput. 38 (1996) 1–15.
[9] A. Frommer, D. Szyld, Asynchronous two-stage iterative methods, Numer. Math. 69 (1994) 141–153.
[10] A. Frommer, D. Szyld, Asynchronous iterations with flexible communication for linear systems, Calculateurs Parallèles, Réseaux Systèmes Répartis 10 (1998) 421–429.
[11] A. Frommer, D. Szyld, On asynchronous iterations, J. Computat. Appl. Math. 123 (2000) 201–216.
[12] J.C. Miellou, Algorithmes de relaxation chaotique à retards, RAIRO Anal. Numér. R1 (1975) 55–82.
[13] J.C. Miellou, D. El Baz, P. Spiteri, A new class of iterative algorithms with order intervals, Math. Comp. 67 (1998) 237–255.
[14] J. Ortega, W. Rheinboldt, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York, 1970.
[15] P. Spiteri, J. Miellou, D. El Baz, Asynchronous Schwarz alternating method with flexible communication for the obstacle problem, Calculateurs Parallèles, Réseaux et Systèmes Répartis 13 (2001) 47–66.
[16] P. Spiteri, J. Miellou, D. El Baz, Parallel Schwarz and multisplitting methods for a nonlinear diffusion problem, Numer. Algorithms 33 (2003) 461–474.
[17] J.C. Strikwerda, A probabilistic analysis of asynchronous iteration, Linear Algebra Appl. 349 (2002) 125–154.
[18] D.B. Szyld, Different models of parallel asynchronous iterations with overlapping blocks, Comp. Appl. Math. 17 (1998) 101–115.