INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2007; **55**:945–963
Published online 19 April 2007 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1502

Parallel numerical simulation for the coupled problem of continuous flow electrophoresis

M. Chau[1], P. Spiteri[1,*,†] and H. C. Boisson[2]

[1]*ENSEEIHT-IRIT, 2 rue Camichel, B.P. 7122, F-31071 Toulouse-Cedex, France*
[2]*IMFT, Avenue du Professeur Camille Soula, F-31400 Toulouse-Cedex, France*

SUMMARY

The performance of parallel subdomain method with overlapping is analysed in the case of the 3D coupled boundary-value problem of continuous flow electrophoresis which is governed by Navier–Stokes equations coupled with convection–diffusion and potential equations. Convergence of parallel synchronous and asynchronous iterative algorithms is studied. Comparison between implemented explicit and implicit schemes for the transport equation is made using these algorithms and shows that both methods provide similar results and comparable performances. Copyright © 2007 John Wiley & Sons, Ltd.

Received 15 November 2006; Revised 1 March 2007; Accepted 11 March 2007

KEY WORDS: parallel computing; continuous-flow electrophoresis; Schwarz alternating method; asynchronous iterations; PISO algorithm; mass transfer

1. INTRODUCTION

Simulation of multi-physics problems requires powerful and efficient algorithms. The continuous flow electrophoresis is a typical example of such problem. It consists in separation of several species in a carrier flow by means of an electrical field. For example, in biological applications it is applied to protein separation. Previous works have been developed on the modelling of this process by coupled boundary-value problems. Among such physical models we refer to the one developed by Clifton, Sanchez and all [1–3] that consists in the coupling of Navier–Stokes equations for the hydrodynamic part with electrical field potential equation and as many convective transport equations as number of proteins to be separated. Discretization of this system of nonlinear partial differential equation leads to the solution of large-scale time-dependent algebraic systems. In order to minimize the computational cost, parallel processing is an appropriate answer. Unfortunately,

*Correspondence to: P. Spiteri, ENSEEIHT-IRIT, 2 rue Camichel, B.P. 7122, F-31071 Toulouse-Cedex, France.
†E-mail: Pierre.Spiteri@enseeiht.fr

Contract/grant sponsor: IDRIS (Institut du Développement et des Ressources en Informatique Scientifique, Paris)

in parallel computations, synchronizations occur during the communications at the beginning of each iteration, when a processor waits for a message sent by the others processors. Thus, synchronizations of concurrent parallel tasks induce idle times in the computation and consequently increase elapsed time. The aim of the present study is to present a particular class of parallel methods, the asynchronous algorithms, that do not require synchronization between processors. Parallel asynchronous algorithms are aimed at suppressing idle times without using load balancing techniques. The processors can work concurrently without any order or synchronization.

Furthermore, subdomains methods are well adapted to parallel computations. Among the classical subdomain methods, one of the most efficient [4] for solving boundary-value problem is the Schwarz alternating method with overlapping between the subdomains. The main difficulty that arises in the implementation of domain decomposition methods is the handling of the computational overhead due to the synchronizations between processors. Thus, in order to solve the electrophoresis problem we propose in the present study to implement and analyse a variant of parallel asynchronous Schwarz alternating method derived from a recent study [5].

The analysis of the behaviour of this class of algorithms is performed in order to show the coherence and consistence of such parallel chaotic methods. It is linked to the specific properties of the discrete operators. In particular, the convergence of asynchronous algorithms, first analysed by Chazan and Miranker for linear systems [6], has been established for nonlinear systems in various theoretical frameworks [7–10]. Moreover, in classical asynchronous algorithms [6], communication cannot occur during the solution of a subproblem. Then, theoretical studies have been carried out on the concept of parallel asynchronous algorithms with flexible communication, which allows communication at any time of the computation. The convergence of this former algorithm has been analysed in various contexts, particularly in the framework of M-matrix and more generally M-functions in the nonlinear case [5, 11].

Due to dominant convection in the transport equation, the associated discretized problem is numerically ill-conditioned. So the transport equation has been solved by using two different numerical schemes the first one implicit and the second one explicit. In the paper both methods are compared.

The paper is organized as follows: Section 2 presents the physical problem, particularly the partial differential equations modelling the electrophoresis phenomenon. In Section 3, we present the discretization of the boundary-value problem and we derive a common property of the discrete operators allowing to study the convergence of the parallel asynchronous Schwarz alternating method. Section 4 is devoted to the analysis of the behaviour of the algorithms, particularly the convergence of the general iterative methods considered in the present study. Finally, in the last section, we draw the main conclusions of sequential and parallel numerical experiments.

2. CONTINUOUS FLOW ELECTROPHORESIS

2.1. Principle

This process takes place in a very thin parallelepipedic cell; a solution flows at low speed through this cell (see Figure 1). The solution constituted by the mixture containing the proteins to separate is injected in the carrier fluid by the face C of the cell in the shape of a sharp liquid filament. An electrical field is created through the cell by two plane electrodes located on both sides of the cell, on the faces E and F. The proteins are transferred by the flow along the cell and due to the effect

DOI: 10.1002/fld

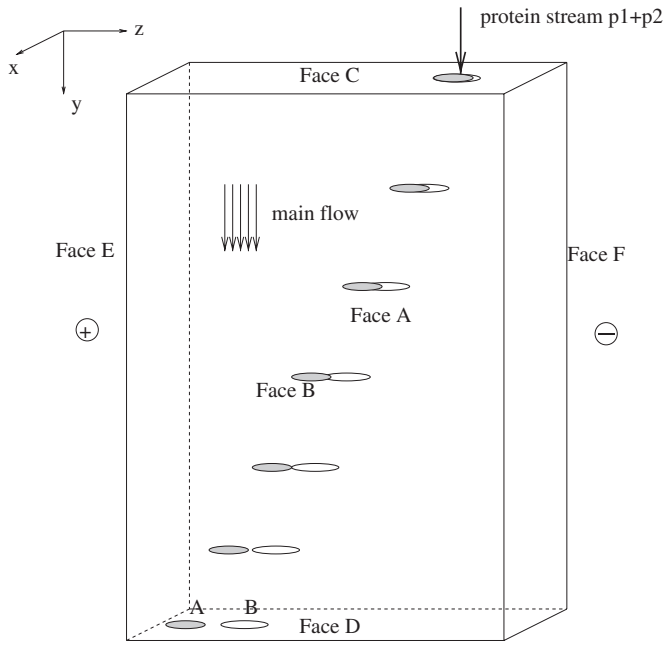

Figure 1. The principle of continuous flow electrophoresis.

of the electrical field they migrate with the speed $\mathbf{W} = \mathbf{V} + \mu\mathbf{E}$, where $\mathbf{V}$ is the flow velocity and $\mu\mathbf{E}$ is the migration velocity. The various species of protein having different electrical mobilities, then can be collected separately on the face D.

In the sequel, the flow is assumed to be isothermal and without chemical reaction; consequently the various physical coefficients arising in the phenomenon are constant. The physical phenomena related to the present study concern:

- the main flow of the fluid in the 3D space, which is perturbated by the effects of electro-hydrodynamics,
- the transport and the migration of the proteins,
- the electrokinetic effect, connected to the spatial changes of the conductivity due to the concentration of the various ionic species.

2.2. *Physical model*

A physical model of the problem was developed in [1–3]. Note that all variables, parameter and equations given in the sequel are non-dimensional in order to simplify the presentation. In the relations governing the electrophoresis flow, the problem can be summarized to compute in the bounded domain Ω included in the 3D space

- the velocity field $\mathbf{V} = (v_1, v_2, v_3)$,
- the pressure p,
- the electrical field $\mathbf{E} = (E_1, E_2, E_3)$,

DOI: 10.1002/fld

- for each protein m, the concentration c_m,
- the electrical potential Φ.

The physical parameters arising in the mathematical model are

- the Reynolds number Re,
- the dielectric permittivity ε,
- the Peclet number associated to the transport of the protein m: Pe
- the electrical conductivity K,
- the electrophoretic mobility of the protein m: μ_m.

2.2.1. Equations. In the present problem we consider the flow of an incompressible viscous fluid in the domain Ω. The main flow is described by the 3D Navier–Stokes equation taking into account the external strength field

$$\frac{\partial v_i}{\partial t} - \frac{1}{Re}\Delta v_i + \sum_{j=1}^{3} v_j \frac{\partial v_i}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \varepsilon \operatorname{div}(E_i \mathbf{E}), \quad i = 1, 2, 3 \tag{1}$$

$$\operatorname{div}(\mathbf{V}) = 0 \tag{2}$$

where

$$\operatorname{div}(E_i \mathbf{E}) = \sum_{j=1}^{3} \frac{\partial}{\partial x_j} E_i E_j$$

The transport equation for a protein m is modelled by the following unsteady convection–diffusion equation

$$\frac{\partial c_m}{\partial t} - \frac{1}{Pe}\Delta c_m + \sum_{i=1}^{3} (v_i + \mu_m E_i) \frac{\partial c_m}{\partial x_i} = 0 \tag{3}$$

The potential Φ is governed by a generalized Poisson equation

$$-\operatorname{div}(K \operatorname{grad} \Phi) = 0 \tag{4}$$

which can also be written as follows:

$$-\sum_{j=1}^{3} \frac{\partial}{\partial x_j} \left(K \frac{\partial \Phi}{\partial x_j} \right) = 0$$

The conductivity K is calculated using the concentrations of the n_p proteins, by the relation

$$K = K_0 + \sum_{m=1}^{n_p} \lambda_m c_m$$

In the sequel, we will consider the migration of only one protein. In this case, the expression of the conductivity becomes

$$K = K_0 + c \tag{5}$$

where c is the concentration of the considered protein; thus, its electrophoretic mobility is noted, μ.

DOI: 10.1002/fld

The equation governing the flow (1) is coupled with the potential equation (4) by the relation

$$\mathbf{E} = -\mathbf{grad}(\Phi) \tag{6}$$

The previous partial differential equations (1)–(4) must be completed by the definition of boundary values induced by physical considerations.

2.2.2. Boundary conditions. The fluid comes in the cell by the upper face C and comes out by the lower face D. So we consider that the velocity fulfils non-homogeneous Dirichlet boundary condition on the face C and homogeneous Neumann boundary condition on the face D: $v_{1/C} = v_{3/C} = 0$, $v_{2/C} = v^0$; $(\partial v_i/\partial n)/D = 0$ for $i = 1, 2, 3$. Furthermore, the velocities v_1 and v_3 are zero on the other four faces; so they fulfil homogeneous Dirichlet boundary conditions on the faces A, B, E and F: $v_{1/A} = v_{3/A} = 0$; $v_{1/B} = v_{3/B} = 0$; $v_{1/E} = v_{3/E} = 0$; $v_{1/F} = v_{3/F} = 0$. The axial velocity v_2 is zero on the faces A and B: $v_{2/A} = v_{2/B} = 0$. On the faces E and F, it fulfils homogeneous Neumann boundary condition: $(\partial v_2/\partial n)/E = (\partial v_2/\partial n)/F = 0$.

The proteins come in the cell by the face C; so on this upper face the concentration is known and it fulfils non-homogeneous Dirichlet boundary condition: $c_{/C} = c^0$. Furthermore, the concentration is free on the other five faces of the cell; so we can consider that on these five faces the concentration fulfils homogeneous Neumann boundary conditions: $(\partial c/\partial n)/A = (\partial c/\partial n)/B = (\partial c/\partial n)/E = (\partial c/\partial n)/F = (\partial c/\partial n)/D = 0$.

The potential fulfils Dirichlet boundary conditions on all the faces. As the potential is known and constant at every points of the electrodes, i.e. on the two lateral faces E and F: $\Phi_{/E} = \Phi^0$; $\Phi_{/F} = 0$. Furthermore, the boundary conditions on faces A and B are obtained by a linear interpolation between the values of the potential defined on the electrodes: $\Phi_{/A} = \Phi_{/B} = (x_1/L)\Phi^0$, where L is the width of the cell. On the horizontal faces C and D, the boundary conditions are obtained by the solution of the potential equation restricted to each upper and lower face; these boundary conditions are preliminarily fixed. As the concentration on the face C is constant, the potential on this face is computed only once: $\Phi_{/C} = \Phi_C$. On the other hand the concentration on the face D changes in process of time. Then, at each time step the potential must be computed on this face in order to obtain the boundary condition: $\Phi_{/D} = \Phi_D(t)$.

3. DISCRETIZATIONS

In the present section, we treat the discretization of the boundary-value problems which describe the evolution of the electrophoresis problem. Then we derive a common property of the discrete operators allowing to study the convergence of the parallel asynchronous domain decomposition methods.

3.1. Fluid flow equations

The incompressible Navier–Stokes equation is solved using pressure-implicit with splitting of operators (PISO) method [12]. The solution process at each time step is split into three separate steps: a predictor step and two corrector steps, where the operations on the velocity are decoupled from those on the pressure.

DOI: 10.1002/fld

Let $(v_i^n)_{i=1,2,3}$ and p^n the values of velocity and pressure at the nth time step. The predictor step consists in computing one time step for Equation (1) by leaving the pressure unchanged. An intermediate velocity $(v_i^{n+1/3})_{i=1,2,3}$ is obtained. Then, each corrector step consists in:

1. solving the mass conservation equation (2) where the pressure's gradient is substituted by the velocity;
2. correcting the former approximate velocity with the computed pressure.

Then the velocities $(v_i^{n+2/3})_{i=1,2,3}$, $(v_i^{n+1})_{i=1,2,3}$ and the pressures $p^{n+1/2}$, p^{n+1} are obtained.

The discretization of the equations involved in the PISO method with finite volume method [13] leads to five linear systems. Note that both matrices are M-matrices [14, 15], i.e. matrices with strictly positive diagonal entries, non-positive off-diagonal entries and strictly diagonally dominant or irreducibly diagonally dominant (see [16]). Note that the centred discretization scheme applied to the convection terms of Equation (1) may not necessarily lead to M-matrices (see [17]).

3.2. Transport equation

If the transport equation is solved with the implicit scheme, analogously the finite difference discretization of the transport equation (3) leads always to M-matrices when upwind scheme is used to discretize the convection terms [15].

Moreover, as diffusion coefficients are very low, false diffusion may occur. That is the reason why the MPDATA algorithm [18] has been implemented to solve the transport equation (3). This method is based on the classical donor-cell explicit scheme; therefore, the use of a solver is pointless.

The first stage of the method consists in computing one step of the former scheme. Then, the expression of false diffusion, given by a second order Taylor development, is transformed to an *antidiffusive* convection velocity. Thus, donor-cell scheme is applied once again, with the antidiffusive velocity in order to counter the effects of false diffusion. Furthermore, this process can be repeated several times to improve the numerical results. In our simulations, second-order MPDATA algorithm have been used.

3.3. Potential equation

The potential equation is discretized with a finite difference scheme. As the conductivity K in the term $-(\partial/\partial x_i)(K\,\partial\Phi/\partial x_i)$ is not constant, the actual discretization scheme is the mean of *forward–backward* and *backward–forward* scheme. For example, consider the 1D potential equation $-(\partial/\partial x)(K\,\partial\Phi/\partial x)$ and a uniform mesh where h is the discretization step. The forward–backward scheme leads to

$$-\frac{\partial}{\partial x}\left(K\frac{\partial\Phi}{\partial x}\right)=\frac{1}{h}\left(K(x_{i+1})\frac{\Phi(x_{i+1})-\Phi(x_i)}{h}-K(x_i)\frac{\Phi(x_i)-\Phi(x_{i-1})}{h}\right)$$

and the backward–forward scheme leads to

$$-\frac{\partial}{\partial x}\left(K\frac{\partial\Phi}{\partial x}\right)=\frac{1}{h}\left(K(x_i)\frac{\Phi(x_{i+1})-\Phi(x_i)}{h}-K(x_{i-1})\frac{\Phi(x_i)-\Phi(x_{i-1})}{h}\right)$$

Finally, it can be shown that the considered numerical scheme leads also to an M-matrix [15, 17].

DOI: 10.1002/fld

4. SOLUTION OF LINEAR SYSTEMS BY THE PARALLEL ASYNCHRONOUS SCHWARZ ALTERNATING METHOD

4.1. Parallel synchronous and asynchronous Schwarz algorithms

Domain decomposition methods, such as the Schwarz algorithm introduced by Lions [19–21], are well suited to the parallel solution of boundary-value problems (see [4]). In these kind of methods, in order to parallelize the computation, the domain of a partial derivative equation is splitted into parallelepiped subdomains. Thus, sequences of smaller subproblems are solved on each processor of a parallel computer in order to compute a solution of the global problem; practically more accuracy is obtained. Consider a boundary-value problem $\Lambda \cdot u = f$ on a domain Ω with boundary condition $B \cdot u = g$ on $\partial\Omega$. For the sake of simplicity, we consider a decomposition in two subdomains Ω_1 and Ω_2.

Parallel asynchronous Schwarz algorithms for two processors consists in solving at each iteration:

$$\begin{cases} \Lambda_1 \cdot u_1^{r+1} = f_1 \text{ on } \Omega_1 \\ B_1 \cdot u_1^{r+1} = g_1 \text{ on } \partial\Omega \cap \Omega_1 \\ \quad u_1^{r+1} = \tilde{u}_2^r \text{ on } \partial\Omega_1 \cap \Omega_2 \end{cases} \quad \text{and} \quad \begin{cases} \Lambda_2 \cdot u_2^{r+1} = f_2 \text{ on } \Omega_2 \\ B_2 \cdot u_2^{r+1} = g_2 \text{ on } \partial\Omega \cap \Omega_2 \\ \quad u_2^{r+1} = \tilde{u}_1^r \text{ on } \partial\Omega_2 \cap \Omega_1 \end{cases} \tag{7}$$

where $\tilde{u}_1^r$ and $\tilde{u}_2^r$ denote the available values of the components of the iterate vector (u_1, u_2) at the current iteration. In the synchronous algorithm, $\tilde{u}_1^r = u_1^r$ and $\tilde{u}_2^r = u_2^r$. Besides, in the classical asynchronous algorithm (see [6, 7]), these components may be delayed as follows: $\tilde{u}_1^r = u_1^{\rho_1(r)}$ and $\tilde{u}_2^r = u_2^{\rho_2(r)}$, with $\rho_i(r) \geqslant 0,\ i = 1, 2$. Finally, in the case of asynchronous algorithm with flexible communication (see [5, 11]), $\tilde{u}_i^r$ are not necessarily associated with components that are labelled by an outer iteration number as communication may occur at any time. Then, in this class of method, partial updates, i.e. the current value of any component of the iterate vector, can be used at any time in the computation. Thus, flexible data exchanges between processors are allowed; as a consequence, the coupling between communication and computation can be improved.

In the sequel, we will focus on cases where Λ is a linear operator. Then, the discretization matrix of the operator Λ will be denoted by A, the right-hand side of the linear system derived from discretization by F and its associated discrete solution by X.

4.2. Numerical behaviour of the parallel algorithms

In the previous section, we have shown that the linearization and the discretization of the 3D continuous flow electrophoresis problem leads to the solution of seven linear algebraic systems; furthermore, the matrices arising in these seven linear systems are all M-matrices. According to previous defined notations, we consider the following system of algebraic equations:

$$A \cdot X = F \tag{8}$$

where $A \in \mathscr{L}(\mathbb{R}^n)$, $X \in \mathbb{R}^n$ and $F \in \mathbb{R}^n$. Furthermore, assume that

$$A \text{ is an M-matrix} \tag{9}$$

DOI: 10.1002/fld

The numerical solution of (8) by the Schwarz alternating method is equivalent to the solution of the following system:

$$\tilde{\mathscr{A}} \cdot \tilde{\mathscr{X}} = \tilde{\mathscr{F}} \tag{10}$$

where $\tilde{\mathscr{A}}$, $\tilde{\mathscr{X}}$ and $\tilde{\mathscr{F}}$ are derived from the augmentation process of the Schwarz alternating method [22]. This process is a theoretical model that represents the solution of the algebraic system (8) by a Schwarz domain decomposition method. In the implementation of the algorithms, $\tilde{\mathscr{A}}$ and $\tilde{\mathscr{F}}$ are not explicitly computed. According to a result of Evans and Deren [22], the matrix $\tilde{\mathscr{A}}$ is also an M-matrix. So, system (10), derived from the augmentation process, has the same property as in the initial algebraic system (8).

Let $\alpha \in \mathbb{N}$ be a positive integer and consider now the following block decomposition of problem (10) into α subproblems

$$\sum_{i=1}^{\alpha} \tilde{\mathscr{A}}_{ij} \cdot \tilde{\mathscr{X}}_j = \tilde{\mathscr{F}}_i \quad \forall i \in \{1, \ldots, \alpha\} \tag{11}$$

where $\tilde{\mathscr{X}}_i \in \mathbb{R}^{n_i}$ and $\tilde{\mathscr{F}}_i \in \mathbb{R}^{n_i}$, where n_i denotes the size of the ith block of the previous vectors and $\tilde{\mathscr{A}} = (\tilde{\mathscr{A}}_{ij})_{1 \leqslant i,j \leqslant \alpha}$, according to the associated block decomposition.

Consider now the solution of subproblems (11) by the asynchronous parallel iteration which can be written as follows:

$$\begin{aligned} \tilde{\mathscr{A}}_{ii} \cdot \tilde{\mathscr{X}}_i^{r+1} &= \tilde{\mathscr{F}}_i - \sum_{j \neq i} \tilde{\mathscr{A}}_{ij} \cdot \tilde{\mathscr{W}}_j \quad \text{if } i \in s(r) \\ \tilde{\mathscr{X}}_i^{r+1} &= \tilde{\mathscr{X}}_i^{r} \quad \text{if } i \notin s(r) \end{aligned} \tag{12}$$

where $\{\tilde{\mathscr{W}}_1, \tilde{\mathscr{W}}_2, \ldots, \tilde{\mathscr{W}}_\alpha\}$ are the available values of the components $(\tilde{\mathscr{X}}_j)_{j \neq i}$, which will be specified more precisely in the sequel, and $\mathscr{S} = \{s(r)\}_{r \in \mathbb{N}}$ is a sequence of non-empty subsets of $\{1, 2, \ldots, \alpha\}$. In other words, $s(r)$ is the subset of the subscripts of the components updated at the rth iteration. In the sequel, we will also consider the vector

$$\mathscr{R} = \{\rho_1(r), \ldots, \rho_\alpha(r)\}_{r \in \mathbb{N}}$$

denoting a sequence of integer vectors from $\mathbb{N}^\alpha$. In order to take into account the asynchronism between the processors, $\mathscr{R}$ models the delays between the parallel updates of each component, at the rth iteration. Furthermore, $\mathscr{S}$ and $\mathscr{R}$ verify the following assumptions:

$$\begin{aligned} &\forall i \in \{1, 2, \ldots, \alpha\} \quad \text{the set } \{r \in \mathbb{N} | i \in s(r)\} \text{ is unbounded} \\ &\forall i \in \{1, 2, \ldots, \alpha\} \quad \forall r \in \mathbb{N}, \ \ 0 \leqslant \rho_i(r) \leqslant r \\ &\forall i \in \{1, 2, \ldots, \alpha\} \quad \lim_{r \to \infty} \rho_i(r) = +\infty \end{aligned}$$

Remark 1

Classically, when $\rho_i(r) = r$ for all $i \in \{1, \ldots, \alpha\}$ and for all $r \in \mathbb{N}$, then (12) models a synchronous Schwarz alternating method.

DOI: 10.1002/fld

For the solution of system (8), the parallel asynchronous Schwarz alternating method with flexible communications corresponds to the more general model of parallel asynchronous iterations. In such a case, the access to the available values of the iterate vector $\tilde{\mathscr{W}}_j$ is obtained by the following norm constraint [5]:

$$\|\tilde{\mathscr{W}}_j - X_j^\star\|_{j,\infty} \leqslant \|W^{\rho(r)} - X^\star\|_\infty \quad \forall j \in \{1, \ldots, \alpha\} \tag{13}$$

where $W^{\rho(p)} = (W_1^{\rho_1(p)}, \ldots, W_\alpha^{\rho_\alpha(p)})$ and $\|\cdot\|_\infty$ denotes a suitable weighted uniform norm [5, 23] and $\|\cdot\|_{j,\infty}$ an analogous weighted uniform norm defined in $\mathbb{R}^{n_j}$, $j = 1, \ldots, \alpha$. It can be noted that, in the present context, the values of the components of the iterate vector generated by the other process, can be accessed while the computations are still in progress; so, we have the following result derived from [5]:

Proposition 1
Assume that (9) is verified. Then, $\tilde{\mathscr{W}} = \{\tilde{\mathscr{W}}_1, \ldots, \tilde{\mathscr{W}}_\alpha\}$ being defined according to (13), the numerical solution of problem (8) by the parallel asynchronous Schwarz alternating method with flexible communications associated to (12) and starting from any initial guess X^0, converges to the solution of $A \cdot X = F$.

Proof
As previously said, since A is an M-matrix, the augmented matrix $\tilde{\mathscr{A}}$, derived from the augmentation process of the Schwarz alternating method, has the important property of being an M-matrix. Then, the convergence of the parallel asynchronous Schwarz alternating method with flexible communications, obtained by contraction techniques, is derived from the result presented in [5], in the linear case. □

Remark 2
Another possibility to analyse the behaviour of parallel asynchronous Schwarz alternating method with flexible communications, consists in using partial ordering techniques linked with the discrete maximum principle [11]. This approach, defined in a different mathematical background, is nevertheless more limiting than the one considered in Proposition 1. This method consists in starting the iteration (12) with an initial guess X^0 such that

$$A \cdot X^0 - F \geqslant 0 \tag{14}$$

In this case, for $r \geqslant 1$, the vector $\tilde{\mathscr{W}} = \{\tilde{\mathscr{W}}_1, \tilde{\mathscr{W}}_2, \ldots, \tilde{\mathscr{W}}_\alpha\}$ belongs to the order interval $\langle \mathscr{X}^r, \min(\mathscr{X}^{\rho(r)}, \mathscr{X}^q)\rangle$, where $\mathscr{X}^{\rho(r)}$ denotes the vector with block components $\mathscr{X}_j^{\rho_j(r)}$, $j \in \{1, 2, \ldots, \alpha\}$ and $q = \max_{k \in K_{s(r)}^r}(k)$, where the set K_i^r contains all the iteration numbers lower than r associated with the computation of the ith block component (see [11]). Note that the values of the components $\tilde{\mathscr{W}}_j$ are also the available values of the iterate vector. In this case, if (14) is satisfied, it can be proved that the sequence of iterate vectors satisfy the maximum discrete principle; indeed $(X^r)_{r \geqslant 0}$ satisfies $X \leqslant \cdots \leqslant X^r \leqslant \cdots \leqslant X^0$.

Remark 3
A particular class of asynchronous Schwarz alternating method corresponds to the one where the updates are performed at the end of any relaxation and defined by $\tilde{\mathscr{W}}_j = \tilde{\mathscr{X}}_j^{\rho_j(r)}$. This case corresponds to the classical parallel asynchronous iterations as defined in [7, 24]. Then, since A is an

DOI: 10.1002/fld

M-matrix, according to a result of [9], it can be proved by a straightforward way, since the uniform weighted norm of the error at the step r is decreasing, that, for any subdomain decomposition, the numerical solution of problem (8) by the classical parallel asynchronous Schwarz alternating method defined by (12) and starting from any initial guess X^0 converges to the solution of $A \cdot X = F$.

Remark 4
Sometimes, the implementation of parallel algorithms requires the use of lexicographical ordering or red–black ordering. The previous convergence results still hold, in both cases of natural and red–black ordering of the mesh points, and moreover, for similar ordering of the subdomains. Let us denote, respectively, by A and $\bar{A}$ the corresponding matrices associated with the two considered ordering. Assume that A is an M-matrix, then $\bar{A}$ is also an M-matrix since $\bar{A}$ is obtained from A by a permutation matrix P which preserves the sign of the entries and particularly the sign of the diagonal entries. Furthermore, we have $\bar{A} = P \cdot A \cdot P^{\mathrm{t}}$; thus we have $\bar{A}^{-1} = P \cdot A^{-1} \cdot P^{\mathrm{t}}$. The matrix A being an M-matrix, it follows that A^{-1} is a non-negative matrix [16]. Thus, $\bar{A}^{-1}$ is also a non-negative matrix obtained from A^{-1} by the same permutation as the one considered for A. Then $\bar{A}$ is an M-matrix. This proves that the asynchronous parallel methods converge in the context of the red–black ordering of both mesh points and subdomains.

From the previous results, we can infer the following obvious corollary.

Corollary 1
The solution of the seven discretized systems $AX = F$ associated with the Navier–Stokes equation, the transport equation of the proteins and the potential equation by the parallel synchronous and asynchronous Schwarz alternating method with or without flexible communications converge to the solution of the considered discretized boundary-value problem for any initial guess X^0 and for any ordering of the subdomains.

5. NUMERICAL EXPERIMENTS

5.1. Sequential simulations

The domain size is $30 \times 0.5 \times 10$ cm and a regular mesh of $400 \times 150 \times 20$ points is defined corresponding to 1 200 000 points. In the sequential simulations 900 time steps have been performed. The domain is decomposed into 128 overlapping subdomains numbered using a red–black ordering. Such an ordering turns out to be well adapted to parallel computation. The flow is assumed to be laminar and the entry corresponds to a developed parabolic velocity profile. Only one protein is injected in the entry section by prescribing the initial concentration (Dirichlet boundary condition). The physical parameters are the following: Reynolds number is equal to 250, electrical field is equal to $950\ \mathrm{V\,m^{-1}}$, Electrophoretic mobility is equal to $526 \times 10^{-8}\ \mathrm{m^2\,V^{-1}\,s^{-1}}$. The filament radius is equal to 1.5 mm.

Remark 5
The mesh size has been carefully selected after long test series. For the incompressible Navier–Stokes equation the mesh size has been diminished in order to satisfy the analytical solution for the channel flow and to obtain a consistent behaviour for the established jet without any interaction with concentration or electrical field (see Reference [14]). The mesh size was also tested separately for

DOI: 10.1002/fld

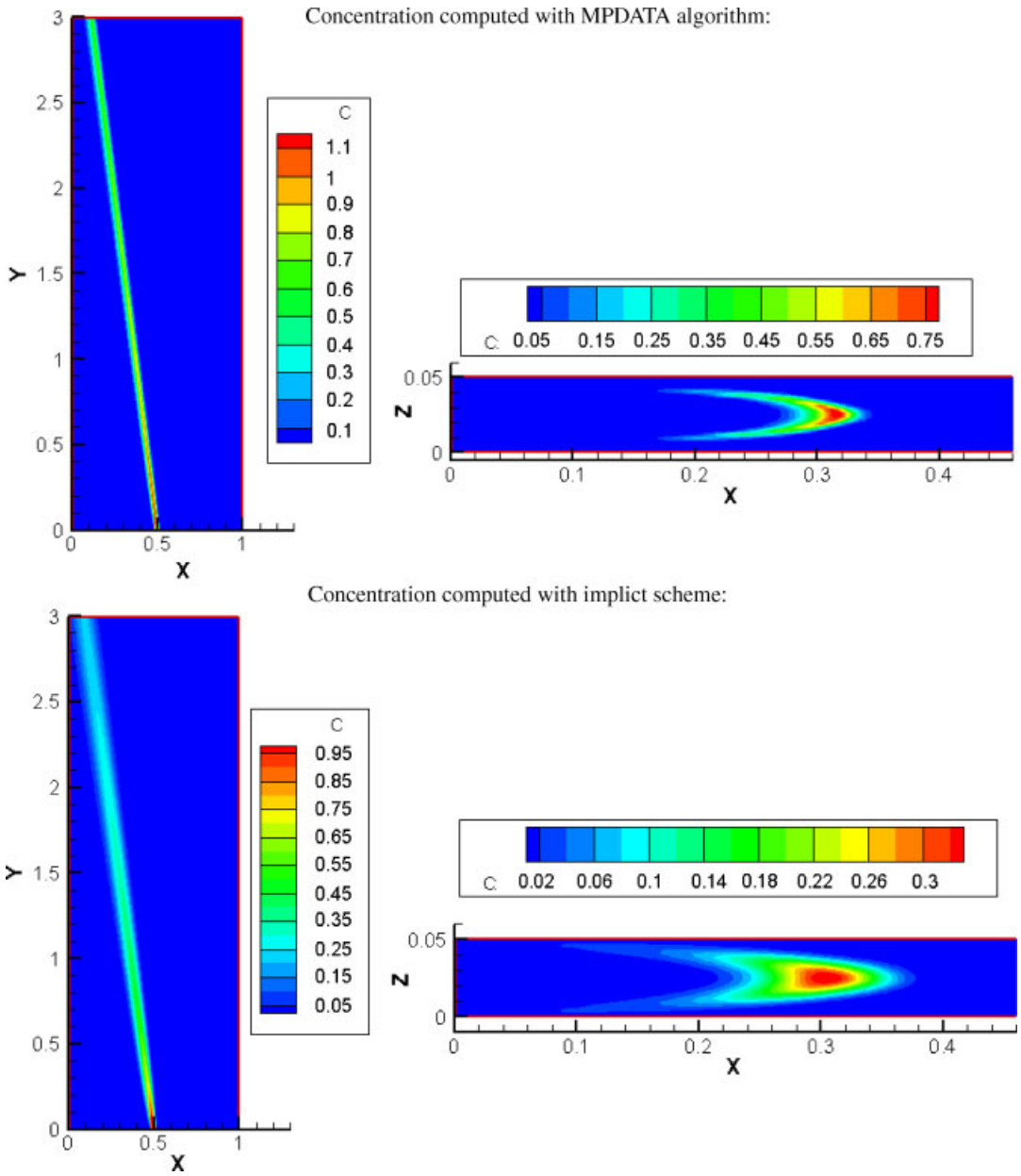

Figure 2. Concentration fields (face C at bottom).

the concentration equation for constant convection coefficients and different diffusion coefficients in order to fix the mesh reducing artificial diffusivity of the computation (see Reference [25]).

The first numerical experiments have been carried out with a 2.8 GHz Pentium 4 personal computer. Figure 2 shows the concentration fields computed with MPDATA and implicit scheme.

DOI: 10.1002/fld

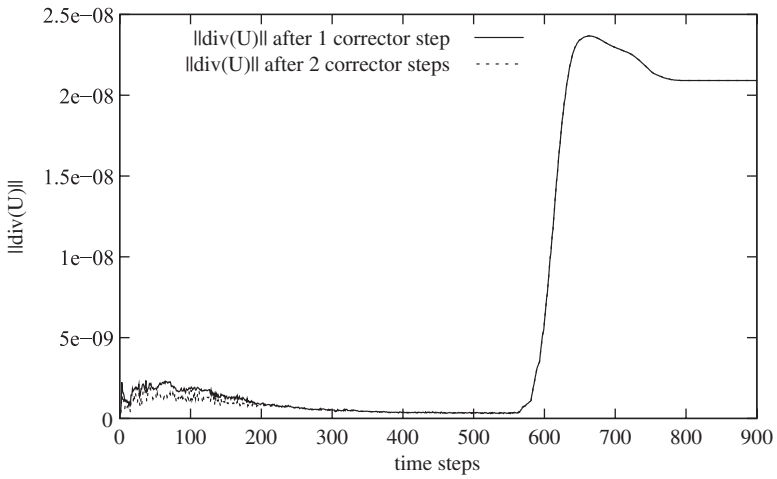

Figure 3. Effect of the corrector steps of PISO on $\|\mathrm{div}(\mathbf{U})\|$ (MPDATA).

In the transport process, the fluid velocity is lower near the surfaces thus the particles near the faces A and B stay longer inside the cell. Therefore, they migrate more than those situated in the centre of the cell. This explains why the filament is distorted near the faces A and B as shown in Figure 2.

A comparison between the two simulations on Figure 2 clearly shows the perturbation due to false diffusion; the filament is wider and the concentration at the exit of the cell is much lower with MPDATA algorithm. However, both numerical results are similar and consistent with the physical phenomenon. Therefore, it is meaningful to compare both algorithms.

Both sequential simulations take at least 40 h. So the parallelization is necessary to improve performance.

5.2. Stability of sequential numerical algorithms

In sequential numerical simulations, PISO algorithm and implicit scheme are unconditionally stable [12, 26]. The stability of PISO algorithm can be experimentally evaluated by checking $\|\mathrm{div}(\mathbf{V})\|$ after each step of the algorithm. Figures 3 and 4 show $\|\mathrm{div}(\mathbf{V})\|$ after each corrector step for the simulations with, respectively, MPDATA algorithm and implicit scheme. In both case, $\|\mathrm{div}\,\mathbf{V}\|$ is bounded and small. Note that our experimental stability study focuses on the corrector steps because at those stages of the algorithm, the velocity updates are explicit operations, whereas the computation of the velocities in the predictor step is implicit (see [12]).

However, the stability of MPDATA algorithm depends on the Courant number, which corresponds to the worst case over the whole domain and must be less than 1 (see [18]). Figure 5 shows the Courant number during the two steps of the second-order MPDATA algorithm, in order to check if Courant–Friedrich–Levy (CFL) condition is verified. The time step has been adjusted in order to obtain Courant numbers inferior to 0.5.

DOI: 10.1002/fld

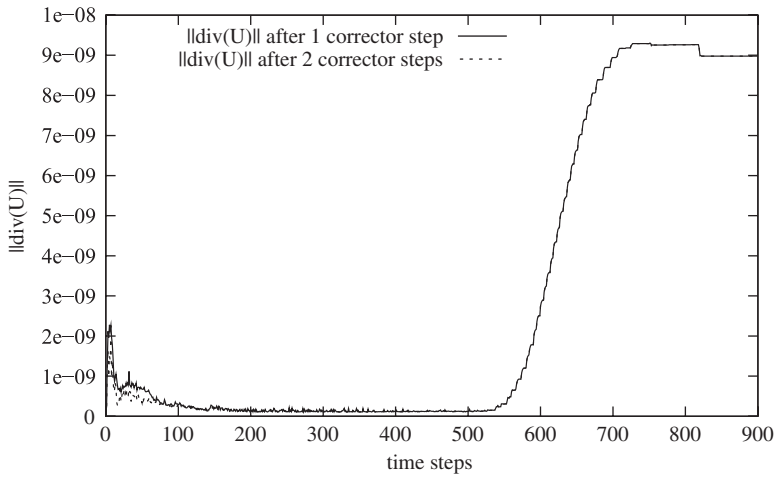

Figure 4. Effect of the corrector steps of PISO on $\|\mathrm{div}(\mathbf{U})\|$ (implicit scheme).

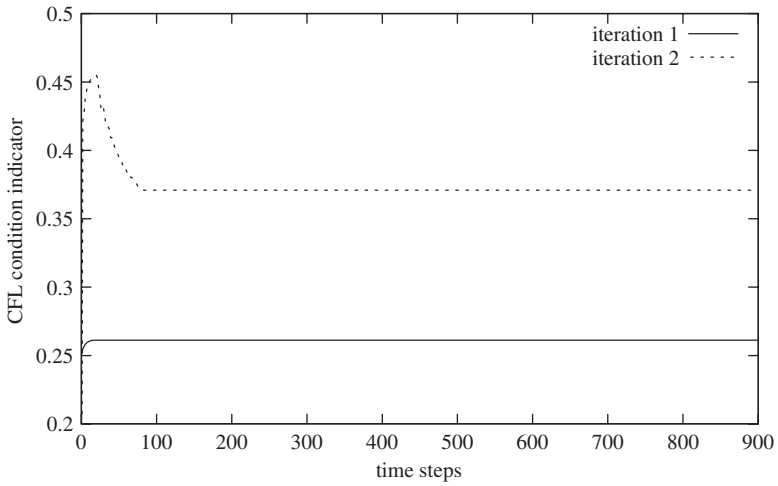

Figure 5. Courant number for MPDATA algorithm.

5.3. Parallel simulations

Parallel simulations have been performed on an IBM Regatta p690+ of IDRIS (Institut du Développement et des Ressources en Informatique Scientifique). This machine is a SMP containing 32 Power4 processors (1.3 GHz). The parallel simulations have been performed with the same parameters as those used in the experiments presented in Section 5.1. Due to large duration of computation, only 20 time steps have been performed. Moreover, according to the size of the linear systems to be solved and due to the limitation of computation means in dedicated exploitation, the number of processors used in the parallel computations has been limited to 16.

DOI: 10.1002/fld

Remark 6
The algorithm is divided in matrix coefficient updates which are sequential and linear system inversions that are solved by iterative parallel algorithms. The differences between synchronous and asynchronous methods are found only in the parallel computation. Preconditioning is not retained for practical reasons: the linear system matrix coefficients are updated at each time step and preconditioning could be time consuming and not systematically efficient. This choice was not made. Since there is no preconditioning phase in the present procedure, both these sequential and parallel parts are encountered since the first time step and thus initial phase of parallel simulation is not burdened by expensive sequential processing. We have limited our comparisons of the methods to the first 20 time steps for obvious reasons of excessive computation costs and also the policy of the computational centre when dedicated exploitation mode is made. But it is observed that the computational effort for solving the iterative procedure is much higher in this transient case than in the quasi-steady convergence process at the end of the computation. Therefore, in the first time steps the asynchronous algorithm is submitted to a more severe challenge. The good results obtained show the robustness and the efficiency of the procedure that will also occur in the rest of the computation.

Both parallel synchronous and asynchronous iterative schemes of computation have been implemented using message passing interface (MPI) facilities; the reader is referred to [17] for more details. Furthermore, in order to get as close as possible of a multiplicative Schwarz alternating method's behaviour (see Lions [19–21]), several subdomains are assigned to each processor. Furthermore, it can be noted that parallel Schwarz alternating method is more efficient if two or more subdomains are treated by each processor. The subdomains are also treated cyclically according to a red–black ordering; in this case, the convergence results of all parallel iterative schemes studied in Section 4 can be applied.

There is no point in attempting to parallelize MPDATA since its computational cost is very low with regard to the costs of the solution of linear systems. Note also that owing to the use of staggered meshes for the velocities, the parallelization of MPDATA algorithm is difficult to implement. Indeed, the computation of the velocities requires interpolations between the components of v_i, $i = 1$–3 defined on each mesh. Therefore, MPDATA algorithm is not worth being parallelized. Then the sequential processing is more important when MPDATA is used.

The number of processors varies up to 16. Experimental results of parallel simulations are presented in Figures 6 (elapsed times), 7 (speed-up) and 8 (efficiency). Elapsed times are also given in Table I. In that table, elapsed times of parallelized sections of the program are indicated in parenthesis. Experiments show that removing synchronizations in the parallel solution of the linear systems is an efficient method. Asynchronous simulations do have better speed-up and efficiencies than synchronous ones, except when two processors are used. In this last case, it can be noted that the overhead due to parallelization is very low. Moreover, as the number of processors increases, the efficiency of synchronous algorithm decreases faster than the efficiency of asynchronous algorithm. The lack of synchronization and the use of current values of the iterate vector's components lead to a better efficiency for parallel Schwarz alternating methods. The efficiencies of both parallel simulations collapse when 16 processors are used (see Figure 8).

In addition, Figures 9 and 10 display the speed-up and efficiencies of the parallelized part of the code, namely the solution of linear systems. The fact that asynchronous iterations are more efficient than synchronous ones is confirmed here. The efficiency collapse noted in Figure 8 is

DOI: 10.1002/fld

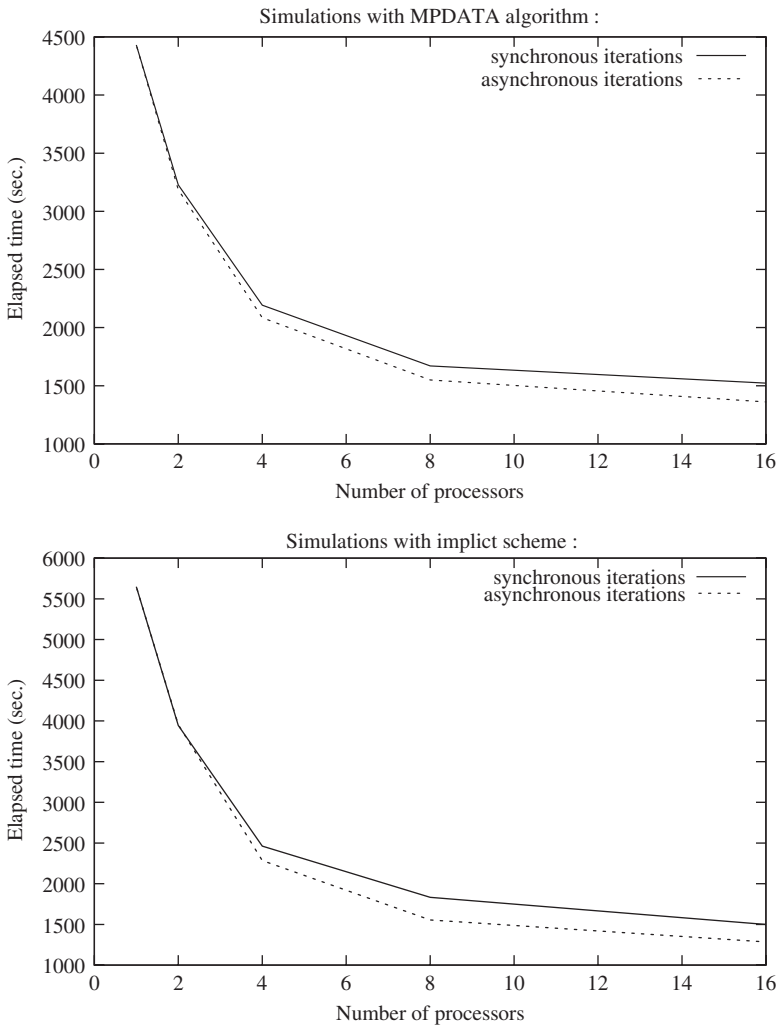

Figure 6. Elapsed times of parallel simulations performed on p690+.

linked to a lack of efficiency that lies in the parallel solution with 16 processors.

- In the asynchronous case, the speed-up increases slower and the efficiency decreases faster above eight processors.
- In the synchronous case, a significant breakdown of efficiency can be noted above two processors.

Lastly, the current efficiency analysis can be complemented with the number of relaxations (update of components) performed by the sequential, synchronous and asynchronous algebraic solvers. The data are displayed in Table II. Two relevant facts should be noted:

1. When synchronous algorithm is used, the number of relaxations varies as the number of processors increases. Actually, as the number of subdomains is bound to 128, the more

DOI: 10.1002/fld

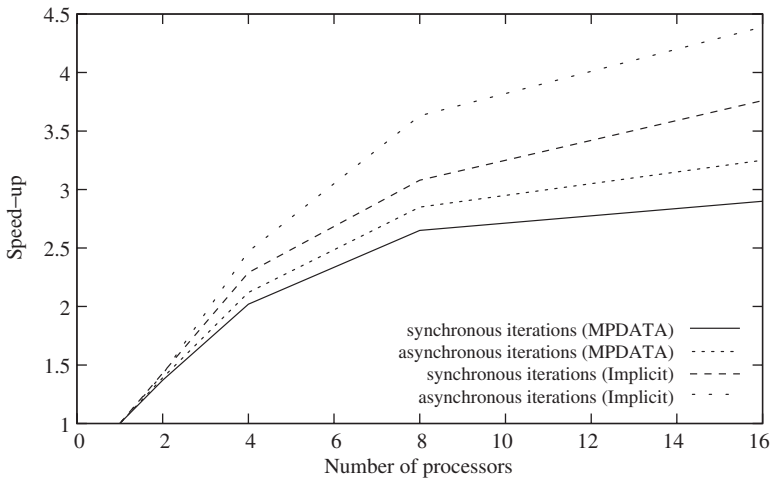

Figure 7. Speed-up of parallel simulations performed on p690+.

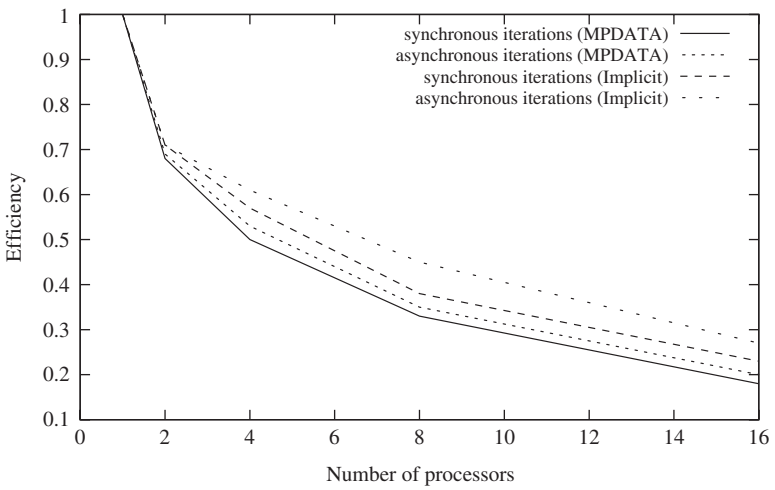

Figure 8. Efficiency of parallel simulations performed on p690+.

Table I. Elapsed times of parallel simulations performed on p690+.

	Implicit scheme		MPDATA algorithm	
Number of proc.	Sync.	Async.	Sync.	Async.
1	5649 (s)		4431 (s)	
2	3844 (3135)	3954 (3146)	3227 (2218)	3186 (2177)
4	2461 (1670)	2285 (1497)	2193 (1201)	2085 (1095)
8	1832 (1041)	1555 (769)	1670 (679)	1550 (559)
16	1499 (709)	1285 (499)	1525 (532)	1361 (371)

DOI: 10.1002/fld

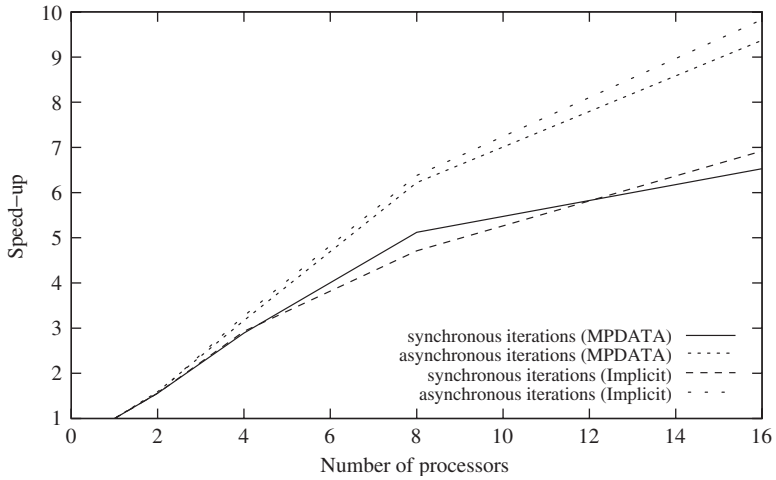

Figure 9. Speed-up of the parallelized parts of the simulations on p690+.

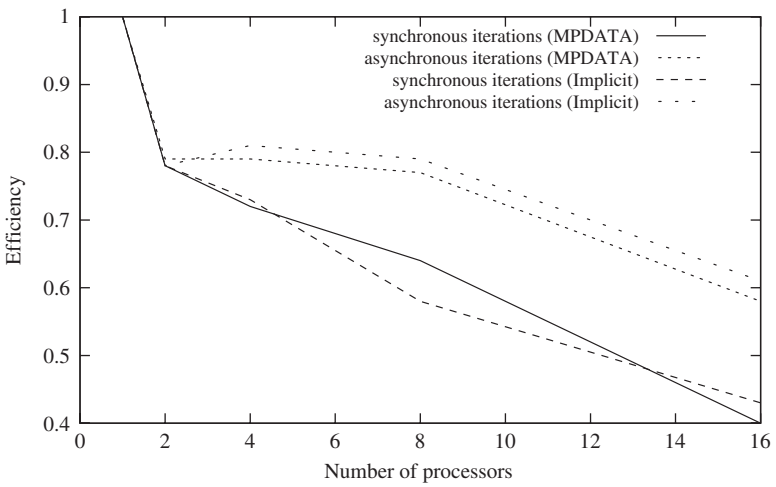

Figure 10. Efficiencies of the parallelized parts of the simulations on p690+.

Table II. Number of relaxations performed in the parallel solution of algebraic systems.

	Implicit scheme		MPDATA algorithm	
Number of proc.	Sync.	Async.	Sync.	Async.
1	1 346 784		943 984	
2	1 358 484	1 378 204	951 252	958 088
4	1 373 844	1 364 016	962 618	968 300
8	1 359 392	1 395 318	963 166	979 688
16	1 592 278	1 688 674	1 183 576	1 231 690

DOI: 10.1002/fld

processors are being used, the less subdomains are assigned to each one. The order in which boundary values are exchanged between the processors, varies as the assignment of the subdomains changes. This order does have a slight influence on the convergence speed of domain decomposition methods.

2. Asynchronous algorithms perform more relaxations than synchronous ones. This is a well-known drawback of asynchronous iterative schemes, due to asynchronous message passing and to termination issue. In asynchronous domain-decomposition methods, boundary values of subdomains are exchanged with no order: then convergence may be slower. Because of asynchronism, termination occurs a few iterations after the convergence is actually detected (with synchronization, termination occurs as soon as convergence is detected); additional relaxations are performed during this time. These relaxations improve the solution's accuracy.

We must note that despite higher number of relaxations, elapsed time of asynchronous parallel iterations are inferior to synchronous ones. In other words, the withdrawal of synchronization can overcome slower convergence. Asynchronism is an efficient way to deal with communication overhead and load unbalance, which are major issues in parallel computing.

6. CONCLUSION

Due to large computational cost, the 3D electrophoresis problem is a very hard challenge to solve by numerical computations. Indeed, if good accuracy is expected, the sizes of the algebraic systems become very large. So, the solution of the considered problem by numerical ways requires large elapsed times of computations. Due to idle times owing to synchronizations, the parallel synchronous subdomain methods have the disadvantage of reducing the speed of computation. In the present study, we have considered parallel asynchronous subdomain methods. Such methods do not require synchronizations between the parallel process. Moreover, load balancing techniques are not necessary for implementing efficient parallel asynchronous Schwarz alternating methods. So, despite the few time steps considered during the parallel simulations, the computation times and efficiencies of the algorithms are improved. Note also that, in the present experimental study, the separation of only one protein is considered. From a practical point of view, the considered computational approach seems to be very attractive when several species are considered.

REFERENCES

1. Clifton MJ, Roux-de-Balmann H, Sanchez V. Electro-hydrodynamic deformation of the sample stream in continuous-flow electrophoresis with an AC electric field. *The Canadian Journal of Chemical Engineering* 1992; **70**:1055–1062.
2. Clifton MJ, Sanchez V. Continuous-flow electrophoresis: Numerical simulation of electrokinetics and electrohydrodynamics. In *Forty-third Congress of the International Astronautical Federation*, Washington, DC, 28 August–5 September 1992.
3. Clifton MJ. Numerical simulation of protein separation by continuous-flow electrophoresis. *Electrophoresis* 1993; **14**:1284–1291.
4. Hoffman KH, Zou J. Parallel efficiency of domain decomposition methods. *Parallel Computing* 1993; **19**: 1375–1391.
5. El Baz D, Frommer A, Spitéri P. Asynchronous iterations with flexible communication: Contracting operators. *Journal of Computational and Applied Mathematics* 2005; **176**:91–103.
6. Chazan D, Miranker W. Chaotic relaxation. *Linear Algebra and its Applications* 1969; **2**:199–222.
7. Miellou JC. Algorithmes de relaxation chaotique à retards. *RAIRO* 1975; **1**:55–82.

DOI: 10.1002/fld

8. Miellou JC. Itérations chaotiques à retards, étude de la convergence dans le cas d'espaces partiellement ordonnés. *CRAS Paris* 1975; **280**:233–236.
9. Miellou JC, Spitéri P. Un critère de convergence pour des méthodes générales de point fixe. *M2AN* 1985; **19**(4):645–669.
10. Giraud L, Spitéri P. Résolution parallèle de problèmes aux limites non linéaires. *M2AN* 1991; **25**:73–100.
11. Miellou JC, El Baz D, Spitéri P. A new class of iterative algorithms with order intervals. *Mathematics of Computation* 1998; **67**:237–255.
12. Issa RI. Solution of the implicitly discretised fluid flow equations by operator splitting. *Journal of Computational Physics* 1986; **62**:40–65.
13. Patankar SV. *Numerical Heat Transfer and Fluid Flow*. McGraw-Hill: New York, 1980.
14. Spitéri P, Boisson HC. Subdomain predictor–corrector algorithms for solving the incompressible Navier–Stokes equation. In *Asymptotic and Numerical Methods for Partial Differential Equations with Critical Parameters*, Kaper HG, Garbey M (eds). NATO ASI Series, vol. 384. Kluwer Academic Publishers: Dordrecht, 1993; 335–347.
15. Guivarch R. Résolution parallèle de problèmes aux limites couplés par des méthodes de sous-domaines synchrones et asynchrones. *Thèse de Doctorat*, Institut National Polytechnique de Toulouse, Laboratoire d'Informatique et de Mathématiques Appliquées (ENSEEIHT-IRIT), 1997.
16. Ortega JM, Rheinboldt WC. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press: New York, 1970.
17. Chau M. Algorithmes parallèles asynchrones pour la simulation numérique. *Thèse de Doctorat*, Institut National Polytechnique de Toulouse, Laboratoire d'Informatique et de Mathématiques Appliquées (ENSEEIHT-IRIT), 2005.
18. Smolarkiewicz PK. A fully multidimensional positive definite advection transport algorithm with small implicit diffusion. *Journal of Computational Physics* 1984; **54**:325–362.
19. Lions PL. On the Schwarz alternating method I. In *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, Glowinski R, Golub GH, Meurant GA, Periaux J (eds). SIAM: Philadelphia, 1988; 1–42.
20. Lions PL. On the Schwarz alternating method II. In *Domain Decomposition Methods*, Chan T, Glowinski R, Periaux J, Widlund O (eds). SIAM: Philadelphia, 1989; 47–70.
21. Lions PL. On the Schwarz alternating method III: A variant for nonoverlapping subdomains. In *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, Chan T, Glowinski R, Periaux J, Widlund O (eds), vol. 6. SIAM: Philadelphia, 1990; 202–223.
22. Evans DJ, Deren W. An asynchronous parallel algorithm for solving a class of nonlinear simultaneous equations. *Parallel Computing* 1991; **17**:165–180.
23. El Tarazi MN. Some convergence results for asynchronous algorithms. *Numerische Mathematik* 1982; **39**:325–340.
24. Baudet GM. Asynchronous iterative methods for multiprocessor. *Journal of the Association for Computing Machinery* 1978; **2**:226–244.
25. Chau M, El Baz D, Guivarch R, Spitéri P. MPI implementation of parallel subdomain methods for linear and nonlinear convection–diffusion problems. *Journal of Parallel and Distributed Computing* 2007; **67**:581–591.
26. Dautray R, Lions JL. *Analyse mathématique et calcul numérique pour les sciences et les techniques*, vol. 9. Masson: Paris, 1988.

DOI: 10.1002/fld