Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 187 (2003) 683–715
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Efficient parallel method for implicit upwind schemes approximating the Euler equations in gas dynamics ☆

Zi-Niu Wu *, Hui Zou

Department of Engineering Mechanics, Tsinghua University, Beijing 100084, PR China

Received 13 July 2001; received in revised form 27 September 2002; accepted 12 February 2003

Abstract

A grid-overlapping parallel method for some three-point implicit schemes has been proposed by Wu and Zou [J. Comput. Phys. 157 (2000) 2]. In this paper we conduct a theoretical analysis of the convergence speed to steady state for this parallel method when multipoint one-sided upwind schemes are used. The theoretical parallel efficiency is found to be near 100% for the totally time-lagging interface treatment. Numerical experiments for compressible flows are conducted to confirm the linear theory. Moreover, numerical experiments will also be demonstrated for schemes which are not completely upwind.

Keywords: Multipoint one-sided schemes; Convergence speed; Parallel computing

1. Introduction

Parallel computation is a powerful method in computational fluid dynamics (CFD) for handling large-scale problems. A CFD user wishing to do parallel computation usually has the following requirement:

1. The parallel algorithm should be simple enough to be understood.
2. The parallel algorithm should be easy enough to be efficiently incorporated into an existing code.
3. The parallel algorithm should not involve a fundamental change of the solver for sequential computation.
4. The parallel efficiency should be high enough for the parallel computation to be meaningful (in increasing the size of problem or in speeding up the calculation).

Parallelization in CFD problems can be conveniently realized by domain decomposition [7,10,12]. Explicit schemes are defined pointwisely and are thus inherently parallel. Implicit schemes have spatial coupling and their parallelization traditionally requires additional iterations or modification of the implicit

☆ This work was supported by Chinese National Natural Science Foundations (Contract No. 10025210) and by the China NKBRSF project (Contract No. 2001CB409600).
* Corresponding author. Tel.:/fax: +86-10-627-73794.
E-mail address: ziniuwu@tsinghua.edu.cn (Z.-N. Wu).

0021-9991/03/$ - see front matter
doi:10.1016/S0021-9991(03)00155-4

solvers [13] in order to be parallelized. When solving the difference equations of each subdomain in parallel, we must know the values on the subdomain boundaries at the new time level.

There were several methods proposed to overcome such a difficulty for general partial differential equations. The classic Schwartz algorithm requires subiterations at each time step [1]. The Schur complement method requires the solution of a subsystem [1]. The implicit parallel solver for tridiagonal system requires modification of the Thomas algorithm [13]. The explicit/implicit method [2,3,8] uses an explicit scheme (with multiple time steps or with a large mesh size) at the interface to find the interface values.

In [14,15], a parallel method is proposed based on grid overlapping. The method does not require additional iterations at each time step, and modification of the implicit solvers. The whole computational domain is decomposed into several subdomains according to the number of processors. There is an overlap at each interface. The interface condition (matching condition) is simply done by a time-lagging interpolation, that is, we define interface conditions for the newest time step by using the solutions at the old time step. Such a time lagging treatment would decrease the convergence speed to steady state and this shortcoming is overcome through grid overlapping.

For steady-state problems using dissipative difference schemes, if the time-lagging decreases the convergence speed with respect to the single domain treatment, then increasing the overlapping width allows us to recover the convergence rate of the single domain treatment. This is similar to the classical Schwartz algorithm for elliptic or parabolic problems. But the CFD problems most often have a nature dominated by hyperbolicity so that we have new features such as finite wave speed and optimal overlapping width. It was found that the optimal overlapping width for best parallel efficiency is determined by the CFL (Courant–Friedrichs–Lewy) number for the implicit scheme of Lerat.

In this paper we specially consider one-sided upwind schemes for hyperbolic problems. In [15] only two-point upwind schemes have been studied and the convergence speed is studied only by numerically computing the related eigenvalue problem. The preliminary consideration of a upwind scheme in [15] does not give a general result. Here, we make a systematic study for multipoint one-sided upwind schemes through non-numerical analysis of the eigenvalue systems. Non-one-sided upwind schemes, which are not covered by analysis, will be considered only in numerical experiments.

This paper will be organized as follows. In Section 2, we describe the interface difference approximation suitable for parallel computation and the reduced problem suitable for convergence analysis.

In Section 3, we give a remark on eigenvalue analysis, that caution should be paid to avoid the loss of important eigenvalues for upwind schemes.

Both Sections 4 and 5 are devoted to analysis for parallel efficiency of upwind schemes on parallel computers. Section 5 is concerned with totally time-lagging interface condition (that is, the interface values for both the implicit stage and the explicit stage lag in time, or the interface condition for each level is simply obtained by a translation of the interface condition for a previous level). Section 6 is concerned with partially time-lagging interface condition (that is, we only lag the interface values for the implicit stage and use the time accurate interface condition for the explicit stage). Mathematicians would implicitly assume a totally time-lagging interface condition, while engineers would readily use the partially time-lagging interface condition.

Numerical experiments for the Euler equations in gas dynamics will be presented in section 6. The non-trivial numerical experiments will confirm the linear theory. Besides, some upwind schemes which are not totally one-sided and which are not covered by the theory will also be tested.

The main conclusions will be summarized in Section 7.

2. Interface difference approximations

When using dimensional splitting, a multidimensional problem can be considered as a combination of several one dimensional problems. Thus we only analyze the method in one dimension.

2.1. General presentation

Consider the following system of hyperbolic conservation laws:

$$w_t + h(w)_x = 0, \quad t \in \mathbf{R}^+, \ -1 < x < 1, \tag{1}$$

with initial data,

$$w(x, t = 0) = w_0(x), \quad x \in \mathbf{R} \tag{2}$$

and suitable boundary conditions at $x = \pm 1$. By hyperbolic assumption, the Jacobian matrix $C(w) = \mathrm{d}h(w)/\mathrm{d}w$ has real eigenvalues $\lambda^{(i)}(w)$ and is diagonalizable.

When only two subdomains are considered, the computational domain is split as $\mathscr{D}_u = \{x : x < \frac{1}{2}l_o\}$, $\mathscr{D}_v = \{x : -\frac{1}{2}l_o < x\}$ with an overlapping length l_o. The boundaries $x = -\frac{1}{2}l_o$ and $x = \frac{1}{2}l_o$ of the overlap are called interfaces. A uniform mesh size of δx is assumed in each subdomain, so that the cell centers in the left and right subdomains are respectively given by: $x_j^{(u)} = \frac{1}{2}l_o + (j - 0.5)\delta x$, $x_j^{(v)} = -\frac{1}{2}l_o + (j + 0.5)\delta x$. The overlap $(-\frac{1}{2}l_o, \frac{1}{2}l_o)$ contains L_o grid points for both subdomains. We will call L the overlapping width (in terms of the number of grid points). The case of more subdomains can be similarly described. For convenience, the analysis is essentially based on two subdomains.

The numerical solutions are denoted by $u_j^n = w(x_j^{(u)}, n\delta t)$ $(j \leqslant 0)$ in $\mathscr{D}_u$ and $v_j^n = w(x_j^{(v)}, n\delta t)$ $(j \geqslant 0)$ in $\mathscr{D}_v$, where δt is the time step. In each subdomain, the system (1) is approximated by a difference scheme in conservation form:

$$I\Delta u_j^{n+1} = -\sigma(f_{j+1/2}^n - f_{j-1/2}^n), \quad j \leqslant -1, \tag{3}$$

$$I\Delta v_j^{n+1} = -\sigma(g_{j+1/2}^n - g_{j-1/2}^n), \quad j \geqslant 1. \tag{4}$$

Here $\Delta u_j^{n+1} = u_j^{n+1} - u_j^n$, $\Delta v_j^{n+1} = v_j^{n+1} - v_j^n$ denote the time increments, $f_{j+1/2}$, $g_{j+1/2}$ are numerical fluxes consistent with the exact flux function $h(w)$, σ is the ratio between δt and δx, and I is an implicit operator. For parallel computation, we always use the same scheme in each subdomain.

In order to independently solve the difference equations in each subdomain as required by parallel computation, time-lagging interface conditions are used. This will be detailed in the following subparagraphs.

2.2. Model problem for convergence analysis

Before doing eigenvalue analysis, we must ensure that each boundary or interface treatment is stable in the sense of Gustafsson et al. [6]. We follow the eigenvalue analysis as considered by Gustafsson [5] to do convergence analysis. The eigenvalue analysis has been considered to give a rigorous measure of convergence speed for symmetric operators, but it is also practically useful for non-symmetric operators.

Since the problem considered is hyperbolic, and the interface condition is invariant under any linear transformation, the interface problem can be diagonalized so that each characteristic component can be treated separately. Hence the convergence speed can be studied simply through the use of the following scalar transport equation:

$$u_t + u_x = 0, \quad -1 < x < 1. \tag{5}$$

We consider one-sided upwind schemes

$$\sum_{l=0}^{M} a_{-l} u_{j-l}^{n+1} = \sum_{l=0}^{M} b_{-l} u_{j-l}^n, \quad -N \leqslant j \leqslant -1, \tag{6}$$

$$\sum_{l=0}^{M} a_{-l} v_{j-l}^{n+1} = \sum_{l=0}^{M} b_{-l} v_{j-l}^{n}, \quad 1 \leqslant j \leqslant N, \tag{7}$$

where $M \geqslant 1$.

First let $M = 1$ (two-point upwind scheme). Then we need a Dirichlet condition at the left boundary

$$u_{-N-1}^{n+1} = u_0 \tag{8}$$

and an interface condition

$$u_0^{n+1} = v_L^n, \quad v_0^{n+1} = u_{-L}^n, \tag{9}$$

where L is the *overlapping length*, defined as the number of grid points in the overlap. In fact, the first condition in (9) is not used in the one-sided scheme (6).

The corresponding single domain scheme for $M = 1$ is defined by

$$\sum_{l=0}^{1} a_{-l} u_{j-l}^{n+1} = \sum_{l=0}^{1} b_{-l} u_{j-l}^{n}, \quad -N \leqslant j \leqslant N, \tag{10}$$

$$u_{-N-1}^{n+1} = u_0. \tag{11}$$

The simplest upwind scheme is given by

$$\Delta u_j^n + \beta\sigma a(\Delta u_j^n - \Delta u_{j-1}^n) = -\sigma a(u_j^n - u_{j-1}^n). \tag{12}$$

For $M > 1$, we need more boundary conditions at the left boundary. These can be written as

$$QY_0 = D, \tag{13}$$

where Q is a non-singular $M \times M$ matrix, D is a column vector in R^M depending on the boundary data, and Y_0 is a column vector for the boundary point value of u. There are two ways to put the boundary points.

1. In the first method, the boundary locates at $j = -N - 1$ so that $u_{-N-1}^{n+1} = u_0$. Then we use a two-point upwind scheme at $j = -N$, a three-point upwind scheme at $j = -N + 1$, etc., so that (13) has the following explicit form:

$$\begin{pmatrix} q_1^{(1)} & 0 & 0 & & & & 0 \\ q_2^{(2)} & q_2^{(1)} & 0 & & & & \\ q_3^{(3)} & q_3^{(2)} & q_3^{(1)} & 0 & & & \\ \vdots & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & 0 \\ & & & & \ddots & \ddots & \ddots & 0 \\ q_M^{(M)} & & \cdots & & \cdots & q_M^{(3)} & q_M^{(2)} & q_M^{(1)} \end{pmatrix} \begin{pmatrix} u_{-N-1} \\ u_{-N} \\ u_{-N+1} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ u_{-N+M-2} \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ d_M \end{pmatrix}, \tag{14}$$

   i.e., the matrix Q is a lower triangular one.

2. In the second method, the boundary locates at $j = -N - 1$ so that $u_{-N-1}^{n+1} = u_0$. But we add $M - 1$ additional boundary points at $j = -N - 2$, $j = -N - 3, \ldots, j - N - M$. Then, we can perform Taylor expansion to establish extrapolation boundary conditions at $j = -N - 2$, $j = -N - 3, \ldots, j - N - M$. See [4] for more details.

The interface condition (only for v) can be written as

$$\begin{aligned} v_0^{n+1} &= u_{-L}^n, \\ v_{-1}^{n+1} &= u_{-L-1}^n, \\ &\vdots \\ v_{-M+1}^{n+1} &= u_{-L-M+1}^n. \end{aligned} \tag{15}$$

3. Remark for convergence speed analysis

Here, we use the single domain two-point upwind scheme to show that caution should be paid in convergence speed analysis. It is very easy to loss an important eigenvalue and get the wrong conclusion.

Insert $u_j^n = z^n \phi_j$ into the problem defined by (10) and (11), we obtain

$$zAY = BY, \tag{16}$$

where A and B are two real matrices, and Y is a column vector of components $\phi_j,\ j = 0, \pm 1, \pm 2, \ldots, \pm N$. There are in total $2N + 12$ eigenvalues z_σ. The convergence rate is characterized by the spectral radius

$$\rho = \max(|z_\sigma|).$$

When this spectral radius is strictly less than one, the problem converges as $n \to \infty$. The number of time iterations required to reach a residual R can be (roughly) estimated by

$$n^{\mathrm{c}} = \frac{\ln R}{\ln \rho}. \tag{17}$$

Thus n^{c} can be defined as the convergence rate (number of time iterations required to reach a prescribed residual).

The explicit forms of the matrices A and B are

$$A = \begin{pmatrix} 1 & & & & & & \\ a_{-1} & a_0 & & & & & \\ & a_{-1} & a_0 & & & & \\ & & \ddots & \ddots & & & \\ & & & \ddots & \ddots & & \\ & & & & & a_{-1} & a_0 \end{pmatrix}$$

and

$$B = \begin{pmatrix} 0 & & & & & & \\ b_{-1} & b_0 & & & & & \\ & b_{-1} & b_0 & & & & \\ & & \ddots & \ddots & & & \\ & & & \ddots & \ddots & & \\ & & & & & b_{-1} & b_0 \end{pmatrix}.$$

Let y_j be the jth component of Y. Then from (16) we have

$$(a_{-1}z - b_{-1})y_{j-1} + (a_0 z - b_0)y_j = 0, \tag{18}$$

and the general solution of (18) is given by

$$y_j = \kappa^j \overline{y}, \tag{19}$$

where κ is the root of the characteristic equation

$$(a_{-1}z - b_{-1}) + (a_0 z - b_0)\kappa = 0. \tag{20}$$

Inserting (19) in the boundary condition (11) yields

$$P(z)\overline{y} = 0, \tag{21}$$

so $z = z_\sigma$ is an eigenvalue if and only if $\det P(z_\sigma) = 0$.

The two eigensystems (16) and (21) are equivalent for schemes with constant coefficients (see for instance [5] for single domain treatment). Each of them has some advantages:

1. System (16) remains valid even for variable coefficients and is easily solvable for sufficiently small values of N. However it is not suitable for qualitative study and its solution is time consuming for large values of N.
2. System (21) is convenient for qualitative study especially for large values of N, but it is limited to constant coefficients and is rather difficult to solve because $\det P(z)$ is a complicated nonlinear function.

The characteristic equation (20) has only one root given by

$$\kappa = \frac{a_{-1}z - b_{-1}}{a_0 z - b_0}.$$

Introducing the solution

$$y_j = \left(\frac{a_{-1}z - b_{-1}}{a_0 z - b_0}\right)^j \overline{y} \tag{22}$$

into the only boundary condition (11) yields

$$P(z)\overline{y} = 0, \tag{23}$$

with $P(z) = 1$. Thus we always have $\overline{y} = 0$ for any z. This means that any z would be an eigenvalue, or the problem would have no determined eigenvalue. However, in practice we have a determined finite convergence speed for the upwind scheme.

The paradox comes from the loss of an important eigenvalue in the analysis. In deriving (23) we have implicitly assumed that

$$\frac{a_{-1}z - b_{-1}}{a_0 z - b_0}$$

is finite so that

$$\left(\frac{a_{-1}z - b_{-1}}{a_0 z - b_0}\right)^0 = 1. \tag{24}$$

In fact (24) is true only when

$$\frac{a_{-1}z - b_{-1}}{a_0 z - b_0} < \infty.$$

We must look for non-trivial solutions in the eigenvalue analysis. Nontrivial solutions here only occurs when $a_0 z - b_0 = 0$ so that y_j does not vanish even for $\overline{y} = 0$, since in this case

$$\frac{a_{-1}z - b_{-1}}{a_0 z - b_0} \to \infty.$$

Thus $z = b_0/a_0$ is an eigenvalue which determines the convergence speed.

4. Parallel efficiency analysis

4.1. Parallel aspects

Let n^{c} be the number of iterations required to reach a prescribed convergence. Next, we will show how to estimate n^{c} through eigenvalue analysis. For convenience, we use $n_{\mathrm{s}}^{\mathrm{c}}$ and $n_{\mathrm{o}}^{\mathrm{c}}$ to distinguish between the single domain case and multidomain case.

Ignoring the communication time, the CPU time T_{CPU} for parallel computations is proportional to $n_{\mathrm{o}}^{\mathrm{c}}$ and the number of mesh points in each subdomain

$$T_{\mathrm{CPU}}^{o} = n_{\mathrm{o}}^{\mathrm{c}}\left(\frac{N_{\mathrm{s}}}{p} + L\right),$$

where N_{s} is the number of mesh points in the single domain treatment, p is the number of processors, and L is the overlapping width defined as the number of mesh points in the overlap (for each subdomain).

The total CPU time for the corresponding single domain treatment using the same implicit scheme is given by

$$T_{\mathrm{CPU}}^{\mathrm{s}} = n_{\mathrm{s}}^{\mathrm{c}} N_{\mathrm{s}}.$$

Thus, the speed up is given by

$$S = \frac{T_{\mathrm{CPU}}^{\mathrm{s}}}{T_{\mathrm{CPU}}^{o}} = \frac{n_{\mathrm{s}}^{\mathrm{c}}}{n_{\mathrm{o}}^{\mathrm{c}}}\frac{N_{\mathrm{s}}}{(N_{\mathrm{s}}/p) + L} = pCD, \tag{25}$$

where $C = n_{\mathrm{s}}^{\mathrm{c}}/n_{\mathrm{o}}^{\mathrm{c}}$ and $D = N_{\mathrm{s}}/N_{\mathrm{s}} + pL$.

The parallel efficiency is

$$E = \frac{S}{p} = CD. \tag{26}$$

Due to unavoidable simplification in the interface treatment, the multidomain treatment generally requires more iterations to converge than the single domain treatment. [1] Thus it is natural that

$$C = \frac{n_{\mathrm{s}}^{\mathrm{c}}}{n_{\mathrm{o}}^{\mathrm{c}}} \lessapprox 1.$$

We know that in elliptic problems solved by the Schwartz algorithm, the convergence speed is an increasing function of the overlapping width [1]. We shall see that, for hyperbolic problems, this remains true in the sense that $n_{\mathrm{o}}^{\mathrm{c}}$ is at least a non-decreasing function of L. Hence C is a non-increasing function of L.

[1] We have the surprising example that the multidomain treatment needs less iterations to converge. See [14].

It is obvious that $D = N_s/(N_s + pL)$ is a decreasing function of L satisfying

$$D = \frac{N_s}{N_s + pL} < 1 \tag{27}$$

for $L > 0$.

It is hopefully that

$$n_o^c = a + bL^{-\alpha} \tag{28}$$

for some positive parameters a, b and α.

Lemma 1. *If the following inequality holds*

$$\alpha > \frac{1 + (a/b)}{1 + (N_s/p)} > 0, \tag{29}$$

then there exits an overlapping width $\infty > L > 1$ such that the parallel efficiency is maximum.

Proof. Using (27) and (28), we can rewrite (26) as

$$E = \frac{N_s}{N_s + pL} \frac{n_s^c}{a + bL^{-\alpha}}.$$

The optimal overlapping efficiency occurs at $L = L_{opt}$ such that

$$\frac{dE}{dL} = 0,$$

which yields

$$\alpha \frac{N_s}{p} L^{-1} - \frac{a}{b} L^{\alpha} = 1 - \alpha. \tag{30}$$

Now for $1 < L < \infty$, (30) leads to (29). □

The above result means that it is possible to have an optimal overlapping width satisfying $1 < L < \infty$ if α is sufficiently large, that is, if n_c decays strongly with L.

If n_c depends quite weakly on L such that $\alpha \to 0$, we have

$$E = \frac{N_s}{N_s + pL} \frac{n_s^c}{a + bL^{-\alpha}} \to \frac{N_s}{N_s + pL} \frac{n_s^c}{a + b},$$

so that E takes its maximum value at $L = 1$.

Lemma 2. *For $\alpha \to 0$, the optimal overlapping width is equal to* 1.

4.2. Parallel efficiency for two-point upwind scheme

The convergence rate is determined from the corresponding eigenvalue problem, which is obtained by introducing

$$u_j^n = z^n \phi_j^{(u)}, \quad v_j^n = z^n \phi_j^{(v)}, \qquad z \in \mathbf{C} \tag{31}$$

into the problem defined by (6)–(9), yielding

$$zAY = BY, \tag{32}$$

where A and B are two real matrices, and Y is a column vector of components $\phi_j^{(u)},\ j = 0, -1, -2, \ldots, -N$ and $\phi_j^{(v)},\ j = 0, 1, 2, \ldots, N$. There are in total $2N + 2$ eigenvalues z_σ. The convergence rate is characterized by the spectral radius $\rho = \max(|z_\sigma|)$. The number of time iterations required to reach a residual R can be estimated by

$$n^{\mathrm{c}} = \frac{\ln R}{\ln \rho}. \tag{33}$$

Thus n^{c} can be defined as the convergence rate (number of time iterations required to reach a prescribed residual).

Lemma 3. *Consider the upwind scheme* (12), *the convergence rate for the single domain case is given by*

$$n_{\mathrm{s}}^{\mathrm{c}} = \{\ln R\} \Big/ \left\{ \ln \left| \frac{1 + (\beta - 1)\lambda}{1 + \beta\lambda} \right| \right\} \tag{34}$$

Proof. For the upwind scheme (12), the coefficients in (10) are

$$\begin{aligned} &a_{-1} = -\beta\lambda, \quad b_{-1} = (1 - \beta)\lambda, \\ &a_0 = 1 + \beta\lambda, \quad b_0 = 1 - (1 - \beta)\lambda, \\ &a_{+1} = 0, \quad b_{+1} = 0. \end{aligned}$$

Hence the corresponding matrices A and B as appeared in (32) are given by

$$A = \begin{pmatrix} 1 & & & & & & \\ -\beta\lambda & 1 + \beta\lambda & & & & & \\ & -\beta\lambda & 1 + \beta\lambda & & & & \\ & & \ddots & \ddots & & & \\ & & & -\beta\lambda & 1 + \beta\lambda & & \\ & & & & -\beta\lambda & 1 + \beta\lambda & \end{pmatrix}$$

and

$$B = \begin{pmatrix} 0 & & & & & & \\ (1 - \beta)\lambda & 1 - (1 - \beta)\lambda & & & & & \\ & (1 - \beta)\lambda & 1 - (1 - \beta)\lambda & & & & \\ & & \ddots & \ddots & & & \\ & & & (1 - \beta) & 1 - (1 - \beta)\lambda & & \\ & & & & (1 - \beta)\lambda & 1 - (1 - \beta)\lambda & \end{pmatrix}.$$

There is no difficulty to obtain the following

$$\det(Az - B) = \{(1 + \beta\lambda)z - [1 - (1 - \beta)\lambda]\}^{2N},$$

so that

$$z_\sigma = \frac{1 - (1 - \beta)\lambda}{1 + \beta\lambda}$$

and

$$\rho = \max(|z_\sigma|) = \left|\frac{1-(1-\beta)\lambda}{1+\beta\lambda}\right|.$$

Finally, by (33) we obtain (34). □

From the above theorem, it is readily derived, by requiring $n_c \to 0$, that the best convergence rate occurs at

$$\lambda = \frac{1}{1-\beta}.$$

Hence for the explicit scheme with $\beta = 0$, the best convergence occurs at $\lambda = 1$.

Lemma 4. *Consider the upwind scheme* (12), *the convergence rate for the multidomain case is given by*

$$n^c = \{\ln R\} \Big/ \left\{ \ln \left|\frac{1+(\beta-1)\lambda}{1+\beta\lambda}\right| \right\} \tag{35}$$

independently of the overlapping width L and thus equal to the convergence rate of the single domain case.

Proof. The second boundary condition in (8) and the first interface condition in (9) are not needed for the upwind scheme. When this remark is taken into account, the corresponding matrices A and B as appeared in (32) are found to be

$$A = \begin{pmatrix} 1 & & & & & & \\ -\beta\lambda & 1+\beta\lambda & & & & & \\ & \ddots & \ddots & & & & \\ & & 0 & 1 & & & (\text{line}\, N+1) \\ & & & -\beta\lambda & 1+\beta\lambda & & \\ & & & & \ddots & \ddots & \end{pmatrix}$$

and

$$B = \begin{pmatrix} 0 & & & & & & \\ (1-\beta)\lambda & 1-(1-\beta)\lambda & & & & & \\ & \ddots & \ddots & & & & \\ & & U & 0 & & & (\text{line}\, N+1) \\ & & & (1-\beta)\lambda & 1-(1-\beta)\lambda & & \\ & & & & \ddots & \ddots & \end{pmatrix},$$

where the line vector U is given by

$$U = (\ldots, 0, \overbrace{1, \ldots, 0}^{L}).$$

The existence of the special line $(N+1)$ does not make new trouble, and we obtain

$$\det(Az - B) = z\{(1+\beta\lambda)z - [1-(1-\beta)\lambda]\}^{2N}$$

The final result then follows directly. □

Now we are ready to give an important result.

Theorem 5. *For the two-point upwind scheme, the optimal overlapping width is*

$$L_{\text{opt}} = 1$$

and the optimal parallel efficiency is

$$E_{\text{opt}} = \frac{N_s}{N_s + p} \approx 1$$

for sufficiently large N_s/p.

Proof. Since for the upwind scheme, the convergence speed is independent of the overlapping width (Lemma 4), we have $\alpha = 0$ in the formula (28). Thus the optimal overlapping width is equal to 1 according to Lemma 2. Now considering the parallel efficiency at optimal overlapping length $L = 1$

$$E = \frac{N_s}{N_s + pL}\frac{n_s^c}{n_o^c} = \frac{N_s}{N_s + p}\frac{n_s^c}{n_o^c}.$$

According to Lemma 4, the overlapping grid method has the same convergence rate as the single domain one, so that $n_s^c/n_o^c = 1$ and

$$E_{\text{opt}} = \frac{N_s}{N_s + p} \to 1. \qquad \square$$

4.3. Parallel efficiency for multipoint one-sided upwind schemes

We only consider the second method for defining the boundary condition. The conclusion with the first method will be the same. Inserting (31) into (6), (7), (13) and (15), we still obtain (32), now with A and B given by

$$A = \begin{pmatrix} Q & O \\ A_L & A_R \end{pmatrix}, \quad B = \begin{pmatrix} O & O \\ B_L & B_R \end{pmatrix}, \tag{36}$$

where Q is the boundary matrix as appeared in (13), O is a $M \times M$ matrix with all elements being zero, and

$$(A_L, A_R) = \begin{pmatrix} A_{-N}^{(1)} & & \\ A_{-N+1}^{(1)} & & \\ \vdots & & \\ A_{-1}^{(1)} & & \\ & I_1 & \\ & I_2 & \\ & \vdots & \\ & I_M & \\ & A_1^{(2)} & \\ & A_2^{(2)} & \\ & \vdots & \\ & A_N^{(2)} & \end{pmatrix} \quad (B_L, B_R) = \begin{pmatrix} B_{-N}^{(1)} & & \\ B_{-N+1}^{(1)} & & \\ \vdots & & \\ B_{-1}^{(1)} & & \\ & J_1 & \\ & J_2 & \\ & \vdots & \\ & J_M & \\ & B_1^{(2)} & \\ & B_2^{(2)} & \\ & \vdots & \\ & B_N^{(2)} & \end{pmatrix}, \tag{37}$$

where

$$A_{-l}^{(1)} = (\overbrace{0,\ldots,0}^{N-l}, a_{-M}, a_{-M+1}, \ldots, a_0, \overbrace{0,\ldots,0}^{N+l}), \quad l = N, N-1, \ldots, 1,$$

$$I_l = (\overbrace{0,\ldots,0}^{N-1+l}, 1, \overbrace{0,\ldots,0}^{N+M-l+1}), \quad l = 1, \ldots, M,$$

$$A_{-l}^{(2)} = (\overbrace{0,\ldots,0}^{N-1+M+l}, a_{-M}, a_{-M+1}, \ldots, a_0, \overbrace{0,\ldots,0}^{N-M-l+1}), \quad l = 1, \ldots, N$$

and

$$B_{-l}^{(1)} = (\overbrace{0,\ldots,0}^{N-l}, b_{-M}, b_{-M+1}, \ldots, b_0, \overbrace{0,\ldots,0}^{N+l}), \quad l = N, N-1, \ldots, 1$$

$$J_l = (\overbrace{0,\ldots,0}^{N-1+l-L}, 1, \overbrace{0,\ldots,0}^{N+M-l+1+L}), \quad l = 1, \ldots, M,$$

$$B_{-l}^{(2)} = (\overbrace{0,\ldots,0}^{N-1+M+l}, b_{-M}, b_{-M+1}, \ldots, b_0, \overbrace{0,\ldots,0}^{N-M-l+1}), \quad l = 1, \ldots, N.$$

Hence, we have

$$\det(Az - B) = z^M \{a_0 z - b_0\}^{2N} \det Q. \tag{38}$$

For single domain treatment, we still have the relations (36) with

$$(A_L, A_R) = \begin{pmatrix} A_{-N}^{(1)} \\ A_{-N+1}^{(1)} \\ \vdots \\ A_{-1}^{(1)} \\ A_1^{(2)} \\ A_2^{(2)} \\ \vdots \\ A_N^{(2)} \end{pmatrix} \quad (B_L, B_R) = \begin{pmatrix} B_{-N}^{(1)} \\ B_{-N+1}^{(1)} \\ \vdots \\ B_{-1}^{(1)} \\ B_1^{(2)} \\ B_2^{(2)} \\ \vdots \\ B_N^{(2)} \end{pmatrix}$$

and

$$A_{-l}^{(1)} = (\overbrace{0,\ldots,0}^{N-l}, a_{-M}, a_{-M+1}, \ldots, a_0, \overbrace{0,\ldots,0}^{N+l}), \quad l = N, N-1, \ldots, 1,$$

$$A_{-l}^{(2)} = (\overbrace{0,\ldots,0}^{N-1+l}, a_{-M}, a_{-M+1}, \ldots, a_0, \overbrace{0,\ldots,0}^{N+1-l}), \quad l = 1, \ldots, N,$$

$$B_{-l}^{(1)} = (\overbrace{0,\ldots,0}^{N-l}, b_{-M}, b_{-M+1}, \ldots, b_0, \overbrace{0,\ldots,0}^{N+l}), \quad l = N, N-1, \ldots, 1,$$

$$B_{-l}^{(2)} = (\overbrace{0,\ldots,0}^{N-1+l}, b_{-M}, b_{-M+1}, \ldots, b_0, \overbrace{0,\ldots,0}^{N+1-l}), \quad l = 1,\ldots,N.$$

Hence, we have

$$\det(Az - B) = \{a_0 z - b_0\}^{2N} \det Q. \tag{39}$$

The spectral radius determined by (38) is the same as that of (39) and is given by

$$z_{\max} = \max\left(\frac{b_0}{a_0}, \rho(Q)\right).$$

In consequence, we have established the following result.

Theorem 6. *For the general multipoint upwind scheme, the convergence rate for both single domain and multidomain treatments is given by*

$$n_{\mathrm{o}}^{\mathrm{c}} = n_{\mathrm{s}}^{\mathrm{c}} = \frac{\ln R}{\ln z_{\max}},$$

with

$$z_{\max} = \max\left(\frac{b_0}{a_0}, \rho(Q)\right), \tag{40}$$

where $\rho(Q)$ is the spectral radius of Q. Consequently, the optimal overlapping width is

$$L_{\mathrm{opt}} = 1$$

and the optimal parallel efficiency is

$$E_{\mathrm{opt}} = \frac{N_{\mathrm{s}}}{N_{\mathrm{s}} + p} \approx 1$$

for sufficiently large N_{s}/p.

From (40), we see that the convergence speed is determined by the diagonal element of the scheme and the boundary matrix Q. Thus in order to have the highest convergence speed, one should devise the scheme and the boundary condition so that both b_0/a_0 and $\rho(Q)$ are minimized.

5. Partially time-lagging interface treatment

5.1. Partially time-lagging interface condition

An implicit scheme involves an explicit stage (depending on values at n and lower) and an implicit stage (depending on values at $n+1$). One can define the interface values separately for each stage, i.e., one can use different ways to define the interface values (such as u_0^{n+1} and v_0^{n+1}) required by the implicit stage, and the interface values (such as u_0^n and v_0^n) required by the explicit stage. One has to lag in time the interface values at $n+1$. But for the explicit stage, one can use the value at n (no time lagging) or the value at $n-1$ (time lagging). This leads to several combinations. The simplest case would be to use a partial time-lagging

condition: a time-accurate interface condition for the explicit stage, and a time-lagging interface condition for the implicit stage:

$$\begin{aligned} u_0^n &= v_L^n, \quad u_0^{n+1} = v_L^n, \\ v_0^n &= u_{-L}^n, \quad v_0^{n+1} = u_{-L}^n. \end{aligned} \tag{41}$$

This interface condition would be frequently used in practice. However, the above condition is not totally time-lagging. In the totally time-lagging definition, the interface value at time $n+1$ is obtained from the value at time n augmented by the time increment. Precisely, after defining Δu_0^{n+1} for the solution of the implicit stage, u_0^{n+1} (which is to be used in the next time step) should be defined as

$$u_0^{n+1} = u_0^n + \Delta u_0^{n+1}.$$

The totally time-lagging interface condition is thus defined by

$$\begin{aligned} u_0^n &= v_L^{n-1}, \quad u_0^{n+1} = v_L^n, \\ v_0^n &= u_{-L}^{n-1}, \quad v_0^{n+1} = u_{-L}^n. \end{aligned} \tag{42}$$

For multipoint one-sided schemes, the partial time-lagging interface condition is defined by

$$\begin{aligned} &v_0^n = u_{-L}^n, \quad v_0^{n+1} = u_{-L}^n, \\ &v_{-1}^n = u_{-L-1}^n, \quad v_{-1}^{n+1} = u_{-L-1}^n, \\ &\quad \vdots \\ &v_{-M+1}^n = u_{-L-M+1}^n, \quad v_{-M+1}^{n+1} = u_{-L-M+1}^n. \end{aligned} \tag{43}$$

5.2. Parallel efficiency analysis

We must transform the partially time-lagging interface treatment to an equivalent totally time-lagging one before doing eigenvalue analysis.

The equivalent totally time-lagging interface treatment is defined as follows: (1) it is totally time-lagging and (2) it yields the same solution as the partially time-lagging interface condition. For the partially time-lagging interface condition (43), the equivalent totally time-lagging one can be obtained by combining (43) with the interior difference Eq. (7) defined at $j = 1, 2, \ldots, M$

$$\sum_{l=0}^{M} a_{-l} v_{j-l}^{n+1} = \sum_{l=0}^{M} b_{-l} v_{j-l}^n, \quad 1 \leqslant j \leqslant M. \tag{44}$$

Inserting (43) into (44) yields

$$\begin{aligned} \sum_{l=0}^{0} a_{-l} v_{1-l}^{n+1} &= \sum_{l=1}^{M} (b_{-l} - a_{-l}) u_{1-l-L}^n + \sum_{l=0}^{0} b_{-l} v_{1-l}^n, \\ \sum_{l=0}^{1} a_{-l} v_{2-l}^{n+1} &= \sum_{l=2}^{M} (b_{-l} - a_{-l}) u_{2-l-L}^n + \sum_{l=0}^{1} b_{-l} v_{2-l}^n, \\ &\vdots \\ \sum_{l=0}^{M-1} a_{-l} v_{M-l}^{n+1} &= \sum_{l=M}^{M} (b_{-l} - a_{-l}) u_{M-l-L}^n + \sum_{l=0}^{M-1} b_{-l} v_{M-l}^n. \end{aligned} \tag{45}$$

Subtracting (44) for j from the jth relation in (45), we obtain the following equivalent totally time-lagging interface condition

$$\begin{aligned}
&\sum_{l=1}^{M} a_{-l} v_{1-l}^{n+1} = \sum_{l=1}^{M} c_{-l} u_{1-l-L}^{n} + \sum_{l=1}^{M} b_{-l} v_{1-l}^{n},\\
&\sum_{l=2}^{M} a_{-l} v_{2-l}^{n+1} = \sum_{l=2}^{M} c_{-l} u_{2-l-L}^{n} + \sum_{l=2}^{M} b_{-l} v_{2-l}^{n},\\
&\qquad\vdots\\
&\sum_{l=M}^{M} a_{-l} v_{M-l}^{n+1} = \sum_{l=M}^{M} c_{-l} u_{M-l-L}^{n} + \sum_{l=M}^{M} b_{-l} v_{M-l}^{n},
\end{aligned} \tag{46}$$

where $c_{-l} = a_{-l} - b_{-l}$.

Inserting (31) into (6), (7), (13) and (46), we still obtain (32), (36) and (37), with I_l and J_l here defined by

$$\begin{aligned}
I_1 &= (\overbrace{0,\ldots,0}^{N}, \overbrace{a_{1-M},\ldots,a_{-1},a_0}^{M}, \overbrace{0,\ldots,0}^{N+1}),\\
I_2 &= (\overbrace{0,\ldots,0}^{N+1}, \overbrace{a_{2-M},\ldots,a_{-1},a_0}^{M-1}, \overbrace{0,\ldots,0}^{N+1}),\\
&\vdots\\
I_M &= (\overbrace{0,\ldots,0}^{N+M-1}, a_0, \overbrace{0,\ldots,0}^{N+1})
\end{aligned}$$

and

$$\begin{aligned}
J_1 &= (\overbrace{0,\ldots,0}^{N-L-M}, \overbrace{c_{1-M},\ldots,c_{-1},c_0}^{M}, \overbrace{0,\ldots,0}^{L}, \overbrace{b_{1-M},\ldots,b_{-1},b_0}^{M}, \overbrace{0,\ldots,0}^{N+1}),\\
J_2 &= (\overbrace{0,\ldots,0}^{N-L-M+2}, \overbrace{c_{2-M},\ldots,c_{-1},c_0}^{M-1}, \overbrace{0,\ldots,0}^{L}, \overbrace{b_{2-M},\ldots,b_{-1},b_0}^{M-1}, \overbrace{0,\ldots,0}^{N+1}),\\
&\vdots\\
J_M &= (\overbrace{0,\ldots,0}^{N-L+M-2}, c_0, \overbrace{0,\ldots,0}^{L}, b_0, \overbrace{0,\ldots,0}^{N+1}).
\end{aligned}$$

The correspondent value of $\det(Az - B)$ is found to be

$$\det(Az - B) = \{a_0 z - b_0\}^{2N} \det Q \det P, \tag{47}$$

where $P = Rz - S$ with

$$R = \begin{pmatrix}
a_{1-M} & a_{2-M} & \cdots & & a_0\\
0 & a_{2-M} & & & a_0\\
\vdots & 0 & \ddots & & \vdots\\
 & & & & \\
 & & \cdots & 0 & a_0
\end{pmatrix}$$

and

$$R = \begin{pmatrix} b_{1-M} & b_{2-M} & \cdots & & b_0 \\ 0 & b_{2-M} & & & b_0 \\ \vdots & 0 & \ddots & & \vdots \\ & & \cdots & 0 & b_0 \end{pmatrix}.$$

Hence, from $\det P = 0$ we obtain the following eigenvalues:

$$z_l = \frac{b_{l-1}}{a_{l-1}}, \quad l = 1, 2, \ldots, M. \tag{48}$$

Let us define

$$\rho_a = \left| \frac{b_0}{a_0} \right|, \tag{49}$$

$$\rho_b = \max_{2 \leqslant l \leqslant M} \left| \frac{b_{l-1}}{a_{l-1}} \right|, \tag{50}$$

$$\rho_c = \rho(Q). \tag{51}$$

Obviously, ρ_a is due to the interior scheme, ρ_b is due to the coupling between the partially time-lagging interface treatment with the interior scheme and ρ_c is due to boundary treatment.

Lemma 7. *For the problem defined by* (6), (7), (13) *and* (43), *the spectral radius is given by*

$$\rho = \max \left(\left| \frac{b_0}{a_0} \right|, \max_{2 \leqslant l \leqslant M} \left| \frac{b_{l-1}}{a_{l-1}} \right|, \rho(Q) \right). \tag{52}$$

Due to the existence of ρ_b, it is not ensured that the multidomain case converges as rapid as the single domain one. To see this more clearly, let us consider the case of two-point scheme (12) for which

$$\begin{aligned} \rho_c &= 0, \\ \rho_a &= \left| \frac{b_0}{a_0} \right| = \left| \frac{1 - (1-\beta)\lambda}{1 + \beta\lambda} \right|, \\ \rho_b &= \left| \frac{b_{-1}}{a_{-1}} \right| = \left| \frac{(1-\beta)\lambda}{-\beta\lambda} \right|. \end{aligned}$$

Since

$$\frac{\rho_b}{\rho_a} = \left| \frac{(1-\beta)\lambda}{-\beta\lambda} \frac{1 + \beta\lambda}{1 - (1-\beta)\lambda} \right|,$$

we have

$$\frac{\rho_b}{\rho_a} = \begin{cases} 0, & \beta = 1, \\ \frac{1}{3} \left| \frac{4+3\lambda}{4-\lambda} \right|, & \beta = \frac{3}{4}, \\ \left| \frac{2+\lambda}{2-\lambda} \right| > 1, & \beta = \frac{1}{2}. \end{cases}$$

Hence, for $\beta = 1$, $\rho_b/\rho_a = 0$ so that

$$\rho = \left| \frac{1-(1-\beta)\lambda}{1+\beta\lambda} \right|,$$

that is, the multidomain problem converges as rapid as the single domain case.

For $\beta = 1/2$, we always have $\rho_b/\rho_a > 1$, so that

$$\rho = \left| \frac{(1-\beta)\lambda}{-\beta\lambda} \right| = 1$$

and the multidomain problem does not converge.

The above result is still independent of the overlapping width. Thus we have proved the following theorem.

Theorem 8. *For the two-point upwind scheme, the convergence rate for the single domain treatment is given by*

$$n_s^c = \{\ln R\} \Big/ \left\{ \ln \left| \frac{1-(1-\beta)\lambda}{1+\beta\lambda} \right| \right\}$$

and, independently of the overlapping width, the convergence rate for the multidomain treatment with the partially time-lagging interface condition is given by

$$n_o^c = \frac{\ln R}{\ln |(1-\beta)\lambda/-\beta\lambda|}.$$

Consequently, the optimal overlapping width is $L_{opt} = 1$ *and, for sufficiently large* N_s/p*, the optimal parallel efficiency is*

$$E_{opt} = \frac{N_s}{N_s+p} \left\{ \ln \left| \frac{(1-\beta)\lambda}{-\beta\lambda} \right| \right\} \Big/ \left\{ \ln \left| \frac{1-(1-\beta)\lambda}{1+\beta\lambda} \right| \right\} \approx \left\{ \ln \left| \frac{(1-\beta)\lambda}{-\beta\lambda} \right| \right\} \Big/ \left\{ \ln \left| \frac{1-(1-\beta)\lambda}{1+\beta\lambda} \right| \right\},$$

so that $E_{opt} \approx 1$ *for* $\beta = 1$ *and* $E_{opt} \approx 0$ *for* $\beta = \frac{1}{2}$.

Thus, the use of a slightly different interface condition may significantly reduce the theoretical parallel efficiency.

It is therefore recommended to use the totally time-lagging interface condition.

6. Numerical experiments for the compressible Euler equations

6.1. Numerical methods

We perform numerical experiments for the two-dimensional compressible Euler equations

$$w_t + f(w)_x + g(w)_y = 0, \tag{53}$$

with

$$w = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix}, \quad f(w) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (\rho E + p)u \end{pmatrix}, \quad g(w) = \begin{pmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ (\rho E + p)v \end{pmatrix}$$

and, for a perfect gas,

$$p = (\gamma - 1)\rho[E - \tfrac{1}{2}(u^2 + v^2)].$$

Here ρ, p, u, v and E denote, respectively, the density, the pressure, the Cartesian components of the velocity vector, and the total energy, and γ, with $\gamma = 1.4$ for the present purpose, is the specific heat ratio.

System (53) is approximated by the implicit version of Roe's upwind scheme [11]:

$$\tilde{f}_{i+1/2,j} = \frac{1}{2}\left(f^{(\mathrm{L})}_{i+\frac{1}{2},j} + f^{(\mathrm{R})}_{i+\frac{1}{2},j}\right) + \frac{1}{2}\left|A^{(\mathrm{Roe})}\right|_{i+\frac{1}{2},j}\left(w^{(\mathrm{L})}_{i+\frac{1}{2},j} - w^{(\mathrm{R})}_{i+\frac{1}{2},j}\right),$$

$$\tilde{g}_{i,j+1/2} = \frac{1}{2}\left(g^{(\mathrm{L})}_{i,j+\frac{1}{2}} + f^{(\mathrm{R})}_{i,j+\frac{1}{2}}\right) + \frac{1}{2}\left|B^{(\mathrm{Roe})}\right|_{i,j+\frac{1}{2}}\left(w^{(\mathrm{L})}_{i,j+\frac{1}{2}} - w^{(\mathrm{R})}_{i,j+\frac{1}{2}}\right),$$

$$\Delta w^{\mathrm{expl}}_{i,j} = -\Delta t(\delta_1 \tilde{f}/\Delta x + \delta_2 \tilde{g}/\Delta y)_{i,j},$$

$$\Delta w^{*}_{i,j} + \tfrac{1}{2}(\Delta t/\Delta x)\left\{\delta_1[|\ A\ |^n\ \delta_1(\Delta w^*)]_{i,j} + \delta_1[A^n \mu_1(\Delta w^*)]_{i,j}\right\} = \Delta w^{\mathrm{expl}}_{i,j},$$

$$\Delta w_{i,j} + \tfrac{1}{2}(\Delta t/\Delta y)\left\{\delta_2[|\ B\ |^n\ \delta_2(\Delta w)]_{i,j} + \delta_2[B^n \mu_2(\Delta w)]_{i,j}\right\} = \Delta w^{*}_{i,j},$$

$$w^{n+1}_{i,j} = w^{n}_{i,j} + \Delta w_{i,j},$$

where $A^{(\mathrm{Roe})}$ and $B^{(\mathrm{Roe})}$ are the well-known Roe averages of the Jacobian matrices $A = \mathrm{d}f(w)/\mathrm{d}w$ and $B = \mathrm{d}g(w)/\mathrm{d}w$, and δ_s, μ_s for $s = 1, 2$ are spatial operators such that for $\phi_{i,j}$ defined at the mesh point $x = i\Delta x$ and $y = j\Delta y$:

$$(\delta_1\phi)_{i,j} = \phi_{i+1/2,j} - \phi_{i-1/2,j}, \quad (\delta_2\phi)_{i,j} = \phi_{i,j+1/2} - \phi_{i,j-1/2},$$

$$(\mu_1\phi)_{i,j} = \tfrac{1}{2}(\phi_{i+1/2,j} + \phi_{i-1/2,j}), \quad (\mu_2\phi)_{i,j} = \tfrac{1}{2}(\phi_{i,j+1/2} + \phi_{i,j-1/2}).$$

The states for left or right side of cell boundary are defined with the MUSCL (monotonic upstream scheme for conservation laws) method [9]:

$$w^{(\mathrm{L})}_{i+1/2,j} = w_{i,j} + \tfrac{1}{4}[(1-k)(\delta_1 w)_{i-1/2,j} + (1+k)(\delta_1 w)_{i+1/2,j}],$$

$$w^{(\mathrm{R})}_{i+1/2,j} = w_{i+1,j} - \tfrac{1}{4}[(1-k)(\delta_1 w)_{i+3/2,j} + (1+k)(\delta_1 w)_{i+1/2,j}],$$

$$w^{(\mathrm{L})}_{i,j+1/2} = w_{i,j} + \tfrac{1}{4}[(1-k)(\delta_2 w)_{i,j-1/2} + (1+k)(\delta_2 w)_{i,j+1/2}],$$

$$w^{(\mathrm{R})}_{i,j+1/2} = w_{i,j+1} - \tfrac{1}{4}[(1-k)(\delta_2 w)_{i,j+3/2} + (1+k)(\delta_2 w)_{i,j+1/2}].$$

Three schemes will be considered.

1. The second-order fully one-sided ($k = -1$) scheme. We note that the case of first-order scheme has been tested in [15] and will not be repeated here.
2. The third-order upwind-biased ($k = 1/3$) scheme. In this case, the scheme is no longer one-sided. The purpose is to see how a slight destruction of total upwinding (one-sided upwinding) alters the convergence speed.

3. The second-order fully one-sided ($k = -1$) scheme with a slope limiter. The limiter alters the scheme if the solution is not smooth, and makes the scheme no longer totally one-sided. The interpolation formulations with limiters can be written as:

$$w^{(\mathrm{L})}_{i+1/2,j} = w_{i,j} + \tfrac{1}{4}[(1-k)(\overline{\overline{\delta}}_1 w)_{i-1/2,j} + (1+k)(\overline{\delta}_1 w)_{i+1/2,j}],$$

$$w^{(\mathrm{R})}_{i+1/2,j} = w_{i+1,j} - \tfrac{1}{4}[(1-k)(\overline{\delta}_1 w)_{i+3/2,j} + (1+k)(\overline{\overline{\delta}}_1 w)_{i+1/2,j}],$$

$$w^{(\mathrm{L})}_{i,j+1/2} = w_{i,j} + \tfrac{1}{4}[(1-k)(\overline{\overline{\delta}}_2 w)_{i,j-1/2} + (1+k)(\overline{\delta}_2 w)_{i,j+1/2}],$$

$$w^{(\mathrm{R})}_{i,j+1/2} = w_{i,j+1} - \tfrac{1}{4}[(1-k)(\overline{\delta}_2 w)_{i,j+3/2} + (1+k)(\overline{\overline{\delta}}_2 w)_{i,j+1/2}],$$

$$(\overline{\delta}_1 w)_{i+1/2,j} = \Phi\Big((\delta_1 w)_{i+1/2,j}, \omega(\delta_1 w)_{i-1/2,j}\Big),$$

$$(\overline{\overline{\delta}}_1 w)_{i+1/2,j} = \Phi\Big((\delta_1 w)_{i+1/2,j}, \omega(\delta_1 w)_{i+3/2,j}\Big),$$

$$(\overline{\delta}_2 w)_{i,j+1/2} = \Phi\Big((\delta_2 w)_{i,j+1/2}, \omega(\delta_2 w)_{i,j-1/2}\Big),$$

$$(\overline{\overline{\delta}}_2 w)_{i,j+1/2} = \Phi\Big((\delta_2 w)_{i,j+1/2}, \omega(\delta_2 w)_{i,j+3/2}\Big),$$

where ω is a parameter between 1 and $3-k/1-k$, and $\Phi(d_+, d_-)$ is a limiter. In this paper, the minmod limiter will be used ($s = \mathrm{sign}(d_+)$):

$$\Phi(d_+, d_-) = \mathrm{minmod}(d_+, d_-) = \begin{cases} s\min(|d_+|, |d_-|), & d_+ d_- > 0, \\ 0, & d_+ d_- \leqslant 0. \end{cases}$$

The scheme has been implemented on a structured mesh by using a finite-volume formulation. On a rigid wall, the slip condition is applied and the pressure is computed from a linear combination of the discrete form of the x and y-momentum equations to obtain a conservative approximation of the normal momentum equation. On an external subsonic inflow boundary, we prescribe the free-stream direction, the entropy and the enthalpy. On an external subsonic outflow boundary we prescribe the pressure.

Domain splitting is done automatically. A structured grid can be split into $n_x \times n_y$ subdomains with an overlapping of L_o mesh points normal to each interface. Since an approximate factorization is used, the computation of the implicit part in each direction is similar to a one-dimensional problem. As a result, the interface conditions can be straightforwardly realized as in the one-dimensional case.

In the single domain case, the grid is a 247×65 C mesh. In multidomain computations, the computational domain is split into 2×1 , 4×1, 8×1, 6×2 or 8×2 subdomains.

For both transonic and subsonic computations, we use the totally time-lagging interface conditions recommended in the previous section.

In order to center on the parallel aspect, we just consider a bidimensional symmetric flow around a fixed NACA0012 airfoil which is either transonic with a free-stream Mach number $M_\infty = 0.85$, subsonic with a free-stream Mach number $M_\infty = 0.536$, or supersonic with $M_\infty = 1.2$.

Note that the overlapping width L defined in the previous sections is different from the overlapping width L_o defined in [15]. The reason is to abbreviate the notations. Our numerical results will be displayed using the old width L_o, which is related to L by

$$L_o = L + 1.$$

Both sequential and parallel computations will be conducted.

The sequential computation is for purpose of testing the convergence speed of the overlapping grid method. First we must ensure that the multidomain approach is correctly realized: the multidomain and single domain computations must yield the same solution.

Since the multidomain treatments have the same convergence speed as the single domain one, the only factor which alters the parallel efficiency is the communication time.

The parallel computation is done on a mixed shared and distributed parallel computers using PVM (parallel virtual machines, see [10] for details and for references). We have a parallel machine with 19 double-CPU processors.

As usual, we will measure the parallel performance by using the parallel efficiency defined as

$$E_p = \frac{\mathrm{CPU}(1)}{n\mathrm{CPU}(n)},$$

where $\mathrm{CPU}(k)$ is the CPU time (including the communication time) computed with k processors.

The parallel performance can also be measured by the speedup $\mathrm{CPU}(1)/\mathrm{CPU}(n)$. It is also possible to use the wall clock time T_{wc} to measure parallel efficiency:

$$\overline{E}_p = \frac{T_{wc}(1)}{nT_{wc}(n)}.$$

Normally some of the processors of the parallel machine are occupied by different tasks (different users may occupy a part of the processors), thus the wall clock time necessarily contains an extra part due to synchronization. As a result, we must have the relation

$$E_p > \overline{E}_p, \tag{54}$$

and this relation will be confirmed by our numerical experiments.

Normally it is the CPU time (not the wall clock time) that is charged in the cost. The use of CPU times (including computation and communication) to measure parallel efficiency is more appropriate here to study the numerical efficiency of the algorithm. But we will display results based on both CPU times and wall clock times.

In each case we give the parallel efficiency and the CPU time or wall clock time required to reach a prescribed convergence.

6.2. One-sided upwind scheme

The case of first-order upwind scheme has already been studied in [15]. In this paragraph we just consider a second-order scheme (totally one-sided scheme with $k = -1$).

6.2.1. Subsonic flow

We take $M_\infty = 0.536$ for the subsonic case.

First we perform sequential computation.

The Mach contours obtained by single domain computation and multidomain computations (with $8 \times 2 = 16$) are displayed in Figs. 1 and 2, respectively. Both yield the same results, showing that there is no error in implementation.

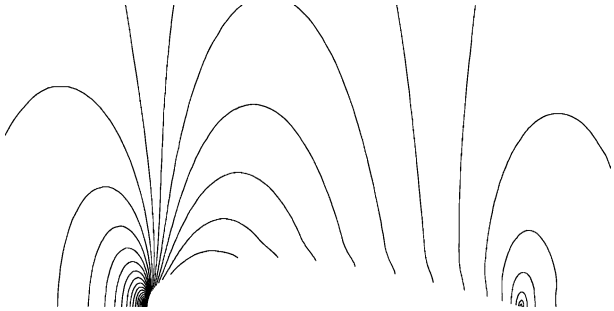

Fig. 1. Mach contours for single domain computation for subsonic ($M\infty = 0.536$), with one-sided Roe scheme ($k = -1$).

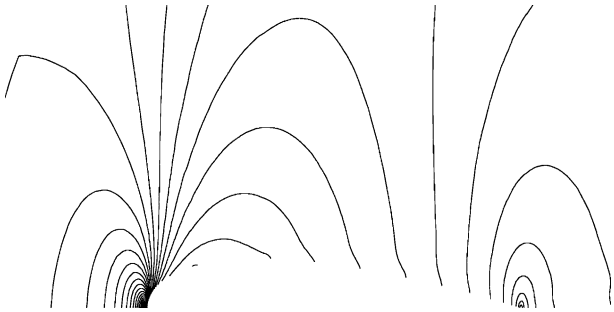

Fig. 2. Mach contours for $8 \times 2 = 16$ subdomains computation for subsonic ($M\infty = 0.536$), with one-sided Roe scheme ($k = -1$).

The convergence curves (the time evolution of the root-mean-square residual R_2 for the discrete density equation) of the overlapping computation compared with the single domain one are displayed in Fig. 3 for $8 \times 1 = 8$ subdomains and Fig. 4 for $8 \times 2 = 16$ subdomains. Though slight difference can be observed, there is no essential difference for the convergence speeds of multidomain treatments and the

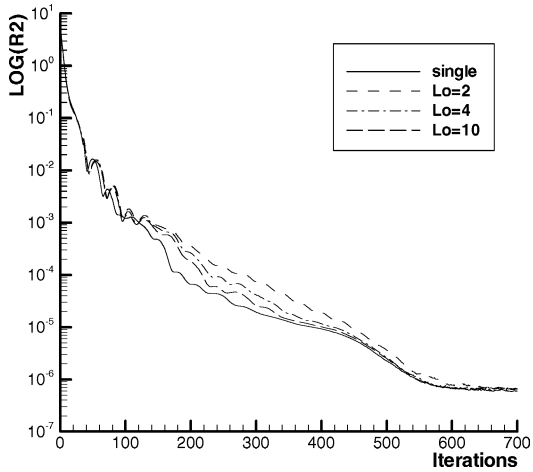

Fig. 3. Convergence curve for $M_\infty = 0.536$ with $8 \times 1 = 8$ subdomains. One-sided Roe scheme ($k = -1$).

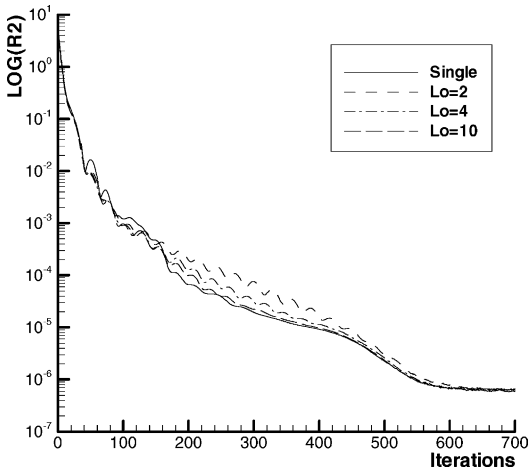

Fig. 4. Convergence curve for $M_\infty = 0.536$ with $8 \times 2 = 16$ subdomains. One-sided Roe scheme ($k = -1$).

single domain one, regardless of the overlapping width or the number of subdomains, as predicted by the theory.

Now we perform parallel computation.

The results are displayed in Tables 1 and 2. From Table 1, which displays the parallel efficiency based on the CPU time, we see that the optimal overlapping width is always equal to 2, for which the parallel efficiency is the highest. The larger the overlapping width is, the lower the parallel efficiency becomes. Table 2 gives the results based on the wall clock time. The parallel efficiencies based on the wall clock time are

Table 1
Parallel efficiency E_p (based on the CPU time) vs overlapping width L_o

L_o	Two domains		Four domains		Eight domains		Twelve domains		Sixteen domains	
	CPU	E_p	CPU	E_p	CPU	E_p	CPU	E_p	CPU	E_p
2	287	0.960	146	0.944	74	0.927	52	0.875	40	0.854
4	287	0.958	148	0.930	77	0.885	55	0.822	42	0.802
10	295	0.932	156	0.882	87	0.791	73	0.622	56	0.608

$M_\infty = 0.536$, computed with fully one-sided ($k = -1$) Roe scheme, and CFL = 20.
For single domain CPU = 551.

Table 2
Parallel efficiency E_p (based on the wall clock time) vs overlapping width L_o

L_o	Two domains		Four domains		Eight domains		Twelve domains		Sixteen domains	
	CPU	E_p	CPU	E_p	CPU	E_p	CPU	E_p	CPU	E_p
2	288	0.958	149	0.920	82	0.831	55	0.824	42	0.813
4	288	0.955	153	0.897	83	0.824	57	0.793	44	0.767
10	297	0.929	165	0.835	99	0.695	79	0.579	60	0.566

$M_\infty = 0.536$, computed with fully one-sided ($k = -1$) Roe scheme, and CFL = 20.
For single domain CPU = 551.

slightly smaller than those based on the CPU time, as can be expected from (54). In any case, the parallel efficiency is a decreasing function of the number of subdomains, due to the increase of communication time since the total number of mesh points is kept fixed.

6.2.2. Transonic flows

In the transonic flow case, we have chosen $M_\infty = 0.85$.

The Mach contours obtained by single domain computation and multidomain computations (with $8 \times 2 = 16$) are displayed in Figs. 5 and 6, respectively. No difference is observed, as is more clear in the C_p distribution displayed in Fig. 7.

The convergence curves of the overlapping computation compared with the single domain one are displayed in Fig. 8 for $8 \times 1 = 8$ subdomains and Fig. 9 for $8 \times 2 = 16$ subdomains. The multidomain treatments (using the totally time-lagging interface conditions) converge as rapidly as the single domain one regardless of the overlapping width or the number of subdomains, as predicted by the theory.

Now consider parallel computations. The parallel efficiencies (based on the CPU time and the wall clock time) are displayed in Tables 3 and 4. The results are very similar to the case of subsonic flows. The optimal overlapping width is always $L_o = 2$ for which the parallel efficiency is the highest. And the parallel efficiencies are decreased with larger overlapping width.

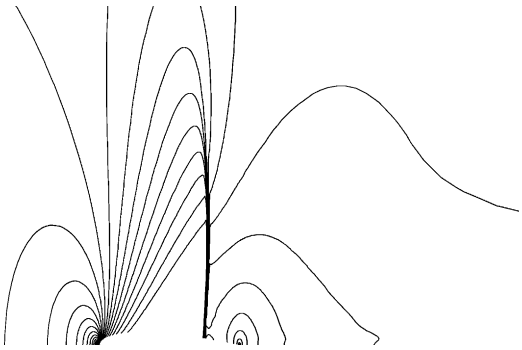

Fig. 5. Mach contours for single domain computation ($M_\infty = 0.85$), with one-sided Roe scheme ($k = -1$).

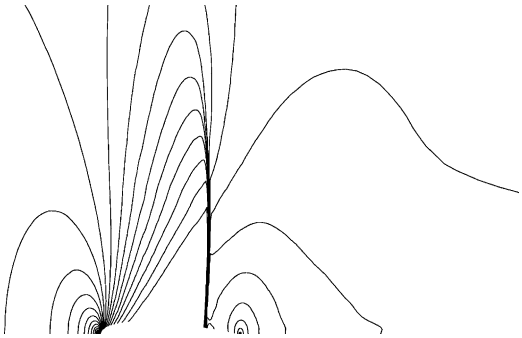

Fig. 6. Mach contours for multidomain domain computation ($M_\infty = 0.85$) with $8 \times 2 = 16$ subdomains), with one-sided Roe scheme ($k = -1$).

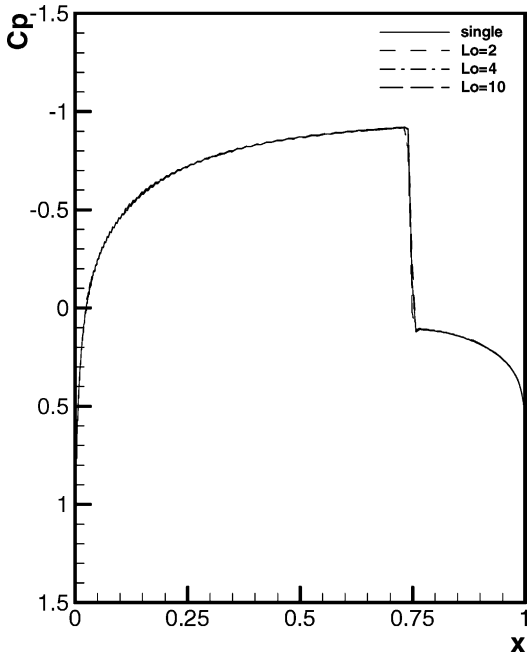

Fig. 7. Pressure coefficient distributions on the wall ($M_\infty = 0.85$). Comparison of the single domain and $8 \times 2 = 16$ subdomains treatment for fully one-upwind Roe scheme ($k = -1$).

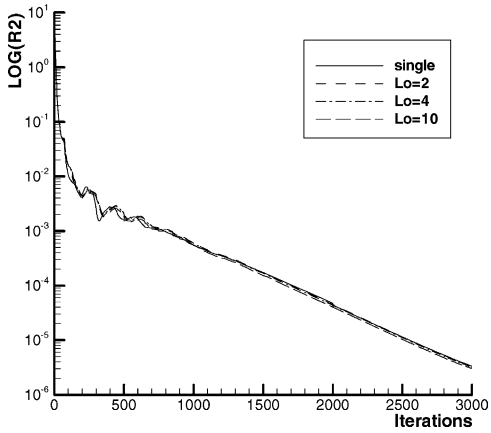

Fig. 8. Convergence curve for $M_\infty = 0.85$ with $8 \times 1 = 8$ subdomains. One-sided Roe scheme ($k = -1$).

6.3. Second- and third-order schemes which are not completely upwind

Since totally upwind schemes of order higher than two are unstable, it is interesting to see schemes which are not completely upwind. The first case is the second-order Roe scheme ($k = -1$) with a minmod limiter. The second case is the third-order scheme ($k = 1/3$).

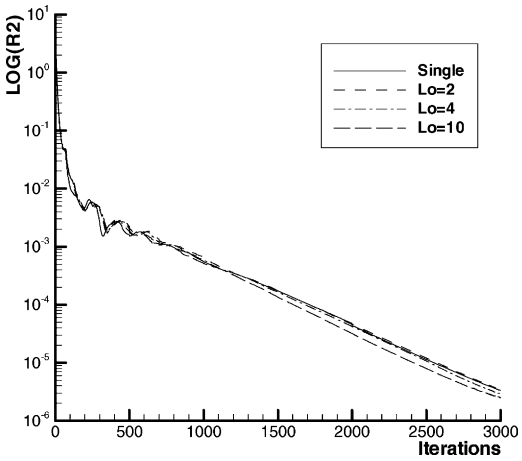

Fig. 9. Convergence curve for $M_\infty = 0.85$ with $8 \times 2 = 16$ subdomains. One-sided Roe scheme ($k = -1$).

Table 3
Parallel efficiency E^p (based on the CPU time) vs overlapping width L_o

L_o	Two domains		Four domains		Eight domains		Twelve domains		Sixteen domains	
	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p
2	1233	0.942	631	0.921	317	0.917	215	0.900	162	0.896
4	1238	0.939	632	0.920	327	0.888	224	0.863	171	0.846
10	1253	0.928	665	0.874	363	0.799	257	0.753	201	0.722

$M_\infty = 0.85$, computed with fully one-sided ($k = -1$) Roe scheme, and CFL = 20.
For single domain CPU = 2325.

Table 4
Parallel efficiency E^p (based on the wall clock time) vs overlapping width L_o

L_o	Two domains		Four domains		Eight domains		Twelve domains		Sixteen domains	
	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p
2	1241	0.937	637	0.912	332	0.874	227	0.853	185	0.782
4	1242	0.936	647	0.899	347	0.836	244	0.794	199	0.727
10	1257	0.925	695	0.836	403	0.720	287	0.674	240	0.605

$M_\infty = 0.85$, computed with fully one-sided ($k = -1$) Roe scheme, and CFL = 20.
For single domain CPU = 2326.

6.3.1. Supersonic flow case

Now we consider the supersonic case ($M_\infty = 1.2$) with Roe scheme ($k = -1$) plus minmod limiter. The Mach contours are displayed in Figs. 10 and 11 for both single domain and multidomain (8 subdomains) computations. The convergence curves of multidomain treatments with 2 and 8 subdomains compared with single domain computations are displayed in Figs. 12 and 13, respectively. We remarked that the multidomain method converges at the same speed as in the single domain case.

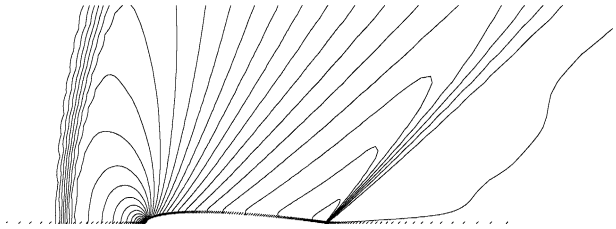

Fig. 10. Mach contours for single domain computation for supersonic ($M\infty = 1.2$), Second-order Roe scheme with minmod limiter ($k = -1$).

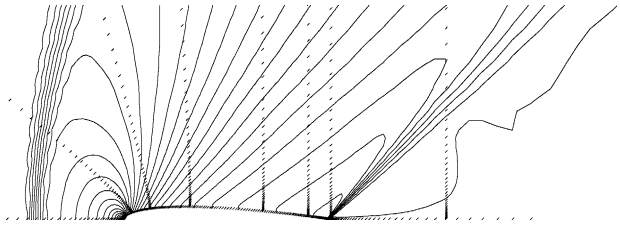

Fig. 11. Mach contours for $8 \times 1 = 8$ subdomains computation for supersonic ($M\infty = 1.2$), Second-order Roe scheme with minmod limiter ($k = -1$).

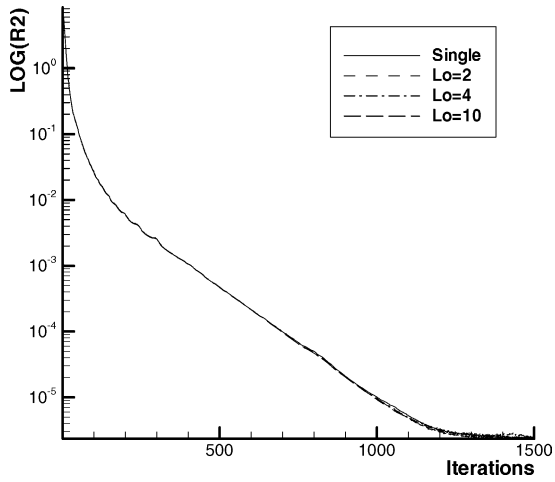

Fig. 12. Convergence curve for $M_\infty = 1.2$ with $2 \times 1 = 2$ subdomains. Second-order Roe scheme with minmod limiter ($k = -1$).

The parallel efficiencies (based on the CPU time and the wall clock time) are displayed in Tables 5 and 6. The optimal overlapping width is also equal to 2 for which the parallel efficiency is the highest.

6.3.2. Transonic flow case

Now consider the Roe scheme ($k = -1$) with minmod limiter and with $M_\infty = 0.85$, the convergence curves of the overlapping computation compared with the single domain one are displayed in Fig. 14 for

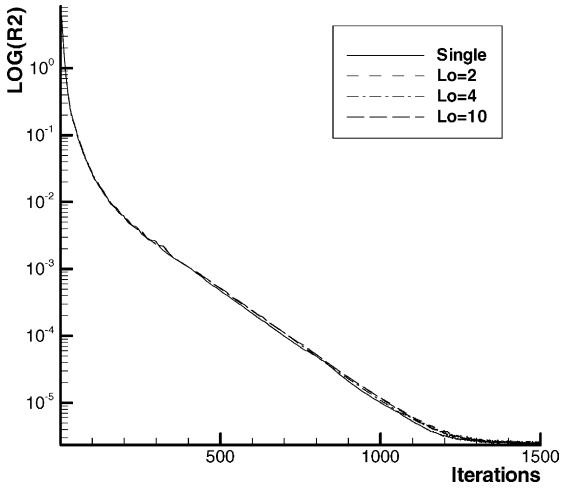

Fig. 13. Convergence curve for $M_\infty = 1.2$ with $8 \times 1 = 8$ subdomains. Second-order Roe scheme with minmod limiter ($k = -1$).

Table 5
Parallel efficiency E^p (based on the CPU time) vs overlapping width L_o

L_o	Two domains		Four domains		Eight domains	
	CPU	E^p	CPU	E^p	CPU	E^p
2	771	0.907	408	0.857	214	0.8148
4	785	0.890	415	0.842	222	0.786
10	821	0.851	463	0.754	252	0.6941

$M_\infty = 1.2$, computed with Roe scheme ($k = -1$) with minmod limiter, and CFL = 20.
For single domain CPU = 1398.

Table 6
Parallel efficiency E^p (based on the wall clock time) vs overlapping width L_o

L_o	Two domains		Four domains		Eight domains	
	CPU	E^p	CPU	E^p	CPU	E^p
2	787	0.888	426	0.820	239	0.731
4	808	0.865	443	0.789	253	0.691
10	862	0.811	508	0.688	301	0.581

$M_\infty = 1.2$, computed with Roe scheme ($k = -1$) with minmod limiter, and CFL = 20.
For single domain CPU = 1398.

$8 \times 1 = 8$ subdomains and in Fig. 15 for $8 \times 2 = 16$ subdomains. Though the theory does not cover this situation, the multidomain treatment does not delay significantly the convergence speed, regardless of the overlapping width or the number of subdomains. The reason is that the limiter is a nonlinear operator which acts on the scheme only when oscillations tend to occur, while the post-period convergence speed is a linear phenomenon.

The parallel efficiencies (based on the CPU time and the wall clock time) are displayed in Tables 7 and 8.

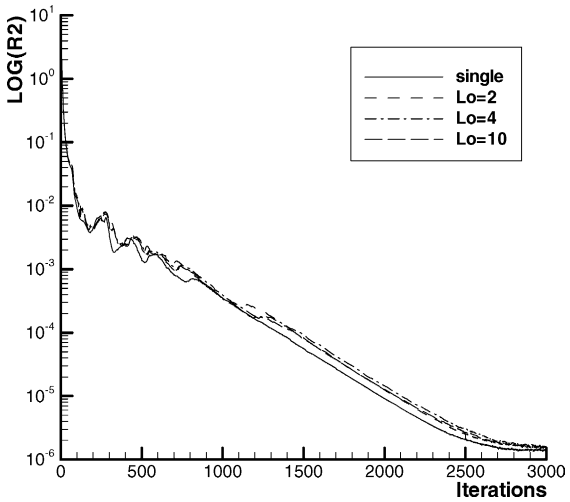

Fig. 14. Convergence curve for $M_\infty = 0.85$ with $8 \times 1 = 8$ subdomains. Second-order Roe scheme with minmod limiter ($k = -1$).

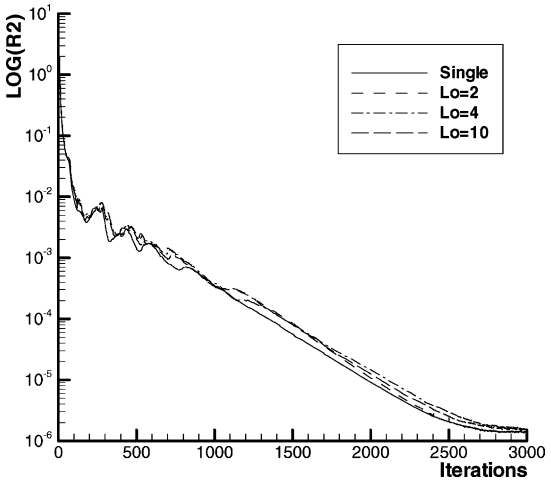

Fig. 15. Convergence curve for $M_\infty = 0.85$ with $8 \times 2 = 16$ subdomains. Second-order Roe scheme with minmod limiter ($k = -1$).

6.3.3. Third-order scheme

For the third-order upwind-biased Roe scheme ($k = 1/3$), the convergence curves of the overlapping computation compared with the single domain one are displayed in Figs. 16 and 17 (subsonic with $M_\infty = 0.536$) and Figs. 18 and 19 (transonic with $M_\infty = 0.85$). This case is not covered by the theory, since the scheme is no longer totally one-sided. In this case the convergence speed is accelerated by multidomain treatments for the transonic flow case, while the difference between single domain treatment and multidomain treatments is not obvious for the subsonic flow case. The superconvergence of the transonic case is very difficult to be explained, but it at least shows that the multidomain treatment with time lagging interface condition works well for high-order upwind schemes.

Table 7
Parallel efficiency E^p (based on the CPU time) vs overlapping width L_o

L_o	Two domains		Four domains		Eight domains		Twelve domains		Sixteen domains	
	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p
2	1473	0.949	733	0.953	388	0.901	267	0.873	194	0.898
4	1495	0.935	757	0.923	412	0.848	276	0.844	216	0.807
10	1514	0.923	796	0.878	458	0.762	335	0.695	259	0.673

$M_\infty = 0.85$, computed with Roe scheme ($k = -1$) with minmod limiter, and CFL = 20.
For single domain CPU = 2796.

Table 8
Parallel efficiency E^p (based on the wall clock time) vs overlapping width L_o

L_o	Two domains		Four domains		Eight domains		Twelve domains		Sixteen domains	
	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p
2	1478	0.946	740	0.944	399	0.874	278	0.836	215	0.812
4	1499	0.932	768	0.910	433	0.807	292	0.797	241	0.723
10	1517	0.922	831	0.841	505	0.691	364	0.639	299	0.583

$M_\infty = 0.85$, computed with Roe scheme ($k = -1$) with minmod limiter, and CFL = 20.
For single domain CPU = 2796.

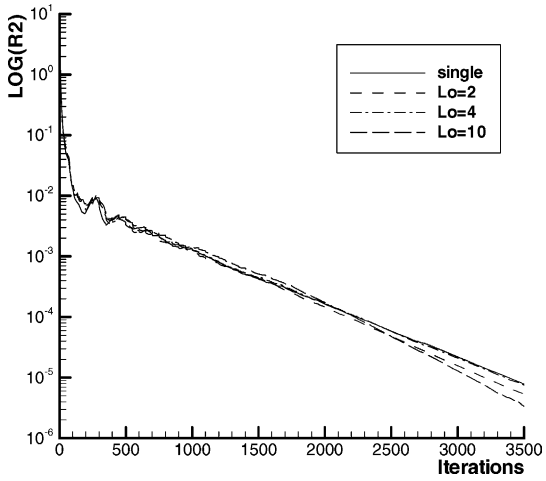

Fig. 16. Convergence curve for $M_\infty = 0.85$ with $8 \times 1 = 16$ subdomains. Third-order scheme ($k = 1/3$).

The parallel efficiencies (based on the CPU time and the wall clock time) are displayed in Tables 9 and 10 for the subsonic flow case (subsonic with $M_\infty = 0.536$). The results are a little different from those of fully one-sided scheme.

Now we consider the transonic case ($M_\infty = 0.85$). The parallel efficiencies (based on the CPU time and the wall clock time) are displayed in Tables 11 and 12. The parallel efficiencies are higher than using second order one-sided scheme.

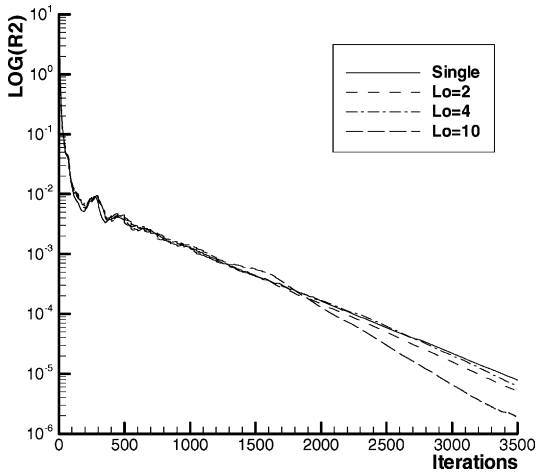

Fig. 17. Convergence curve for $M_\infty = 0.85$ with $8 \times 2 = 16$ subdomains. Third-order scheme ($k = 1/3$).

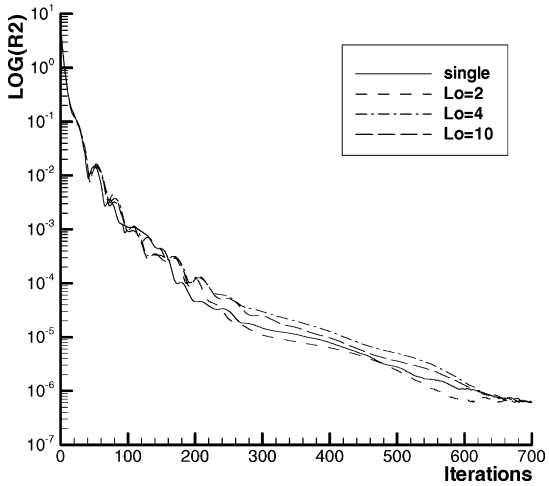

Fig. 18. Convergence curve for $M_\infty = 0.536$ with $8 \times 1 = 8$ subdomains. Third-order scheme ($k = 1/3$).

6.4. Further discussions on high-order upwind schemes and remark on unsteady problems

It is well known that there is no stable one-sided scheme of order of accuracy higher than two. This is why we have not tested one-sided upwind schemes of order higher than 2. Third-order schemes are not completely upwind, but the numerical results presented above show that such schemes together with the time lagging interface condition, though not covered by the present theory, work as well as the lower-order one-sided scheme.

The present paper focuses on convergence to steady state using implicit upwind schemes. Such analysis is not concerned with unsteady flow problems. Unsteady flow problems using the same technique have been studied in the paper [15]. Instead of using an additional inner iteration for each time step, the method

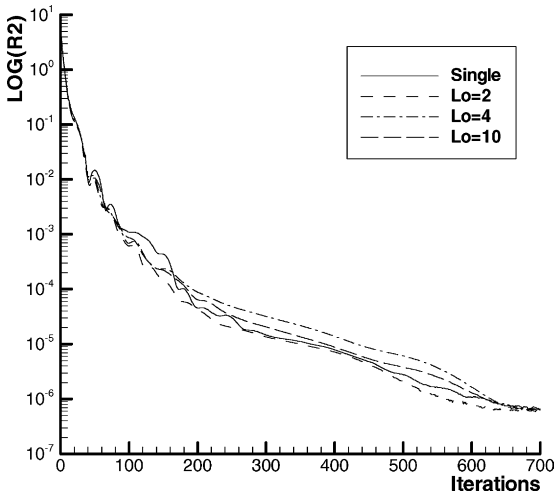

Fig. 19. Convergence curve for $M_\infty = 0.536$ with $8 \times 2 = 16$ subdomains. Third-order scheme ($k = 1/3$).

Table 9
Parallel efficiency E^p (based on the CPU time) vs overlapping width L_o

L_o	Two domains		Four domains		Eight domains		Twelve domains		Sixteen domains	
	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p
2	384	0.893	195	0.877	96	0.890	64	0.883	49	0.870
4	372	0.921	191	0.895	111	0.772	80	0.712	59	0.726
10	388	0.882	195	0.876	124	0.687	85	0.667	79	0.540

$M_\infty = 0.536$, computed with upwind-biased ($k = 1/3$) Roe scheme, and CFL = 20.
For single domain CPU = 686.

Table 10
Parallel efficiency E^p (based on the wall clock time) vs overlapping width L_o

L_o	Two domains		Four domains		Eight domains		Twelve domains		Sixteen domains	
	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p
2	385	0.891	198	0.863	100	0.850	66	0.865	51	0.828
4	373	0.920	196	0.872	118	0.725	85	0.668	61	0.695
10	390	0.880	206	0.830	138	0.621	88	0.643	85	0.501

$M_\infty = 0.536$, computed with third-order ($k = 1/3$) scheme, and CFL = 20.
For single domain CPU = 686.

presented in [15] uses an overlapping width which is proportional to the CFL number. It is based on the following principle.

For unsteady problems, the numerical errors created by time lagging travel a distance bounded by CFL since CFL is the number of grid points that the energy-containing numerical wave travels at each time iterations. Let the overlapping width be 2CFL. The solution over a distance CFL (here distance means number of grid points) close to each boundary of the overlap is polluted. But this polluted points coincide

Table 11
Parallel efficiency E^p (based on the CPU time) vs overlapping width L_o

L_o	Two domains		Four domains		Eight domains		Twelve domains		Sixteen domains	
	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p
2	1926	0.976	969	0.970	482	0.975	329	0.951	244	0.963
4	1965	0.957	1011	0.929	525	0.896	355	0.883	268	0.874
10	2152	0.874	928	1.012	548	0.857	340	0.921	288	0.816

$M_\infty = 0.85$, computed with third-order ($k = 1/3$) scheme, and CFL = 20.
For single domain CPU = 3761.

Table 12
Parallel efficiency E^p (based on the wall clock time) vs overlapping width L_o

L_o	Two domains		Four domains		Eight domains		Twelve domains		Sixteen domains	
	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p	CPU	E^p
2	1934	0.972	979	0.960	502	0.937	349	0.897	274	0.857
4	1974	0.953	1032	0.911	556	0.845	379	0.826	307	0.766
10	2156	0.872	970	0.969	608	0.773	376	0.833	340	0.691

$M_\infty = 0.85$, computed with upwind-biased ($k = 1/3$) Roe scheme, and CFL = 20.
For single domain CPU = 3762.

with a part of the adjacent subdomain that is not yet polluted. Then by projecting the unpolluted solution to the polluted points allows us to minimize the error. See [15] for more details.

7. Conclusions

We have analyzed the convergence speed for multipoint one-sided upwind schemes using time-lagging interface conditions for parallel computation. The conclusion is rather surprising:

1. For the totally time-lagging interface treatment, the multidomain treatment converges as rapidly as the single domain treatment (despite of time lagging at the interface) so that the theoretical parallel efficiency (that is, parallel efficiency without taking into account the communication time) is near 100%.
2. For the partially time-lagging interface treatment, the multidomain treatment converges generally more slowly than the single domain treatment so that the theoretical parallel efficiency is smaller than 100% in most cases.

The numerical experiments based on the two-dimensional Euler equations yield the following conclusions:

1. For the fully one-sided upwind scheme, the optimal overlapping width is always equal to $L_o = 2$ (or $L = 1$), which confirms exactly the linear study.
2. For other kinds of upwind-biased schemes which are not fully one-sided, the optimal overlapping width is still close to $L_o = 2$, though these schemes are not considered in the linear analysis.
3. Since the Thomas algorithm for inverting the implicit tridiagonal system is unchanged with respect to a code for sequential computation, the parallel efficiency E_p or $\overline{E}_p$ we present here is the absolute one. The obtained parallel efficiencies for $L_o = 2$ lie in the range of $(0.8, 1.0)$. This result is very nice since we have used the definition of absolute parallel efficiency, a very simple interface treatment, a very small grid and sufficient subdomains.

References

[1] T.F. Chan, D. Goovearts, Schwartz = Schur, overlapping versus non-overlapping domain decomposition, Technical Reports, CAM 88-21, UCLA, 1988.
[2] Q. Du, M. Mu, Z.N. Wu, Efficient parallel algorithms for parabolic problems, SIAM J. Numer. Anal. 39 (2002) 1469–1487.
[3] C.N. Dawson, Q. Du, T.F. Dupont, A finite difference domain decomposition algorithm for numerical solution of the heat equation, Math. Comput. 57 (1991) 63–71.
[4] M. Goldberg, E. Tadmor, Scheme-independent stability critiria for difference approximations of hyperbolic initial-boundary value problems II, Math. Comput. 36 (1981) 603–626.
[5] B. Gustafsson, The choice of numerical boundary conditions for hyperbolic systems, J. Comput. Phys. 48 (1982) 270–283.
[6] B. Gustafsson, H.-O. Kreiss, A. Sundström, Stability theory of difference approximations for initial boundary value problems. II″, Math. Comput. 26 (1972) 649–686.
[7] D.E. Keyes, Domain decomposition: a bridge between nature and parallel computers, ICASE Report No. 92-44, 1992.
[8] Y. Kuznetsov, New algorithms for approximate realization of implici difference scheme, Soviet J. Numer. Math. Model. 3 (1988) 99–114.
[9] B. van Leer, Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method, J. Comput. Phys. 32 (1979) 101–136.
[10] D. Roose, R.V. Driessche, Parallel computers and parallel algorithms for CFD: an introduction, AGARD-R-807, 1995, pp. 1–23.
[11] P.L. Roe, Approximate Riemann solvers parameter vectors and difference schemes, J. Comput. Phys. 43 (1981) 357–372.
[12] M.L. Sawley, J.K. Tegnér, A comparison of parallel programming models for multiblock flow computations, J. Comput. Phys. 122 (1995) 280–290.
[13] H. Wang, A parallel solver for tridiagonal solvers, ACM Trans. Math. Software 7 (1981) 170–183.
[14] Z.N. Wu, Convergence study of an implicit multidomain method for compressible flow computations, Comput. Fluids 25 (1996) 181–196.
[15] Z.N. Wu, H. Zou, Grid overlapping for implicit parallel computations of compressible flows, J. Comput. Phys. 157 (2000) 2–43.