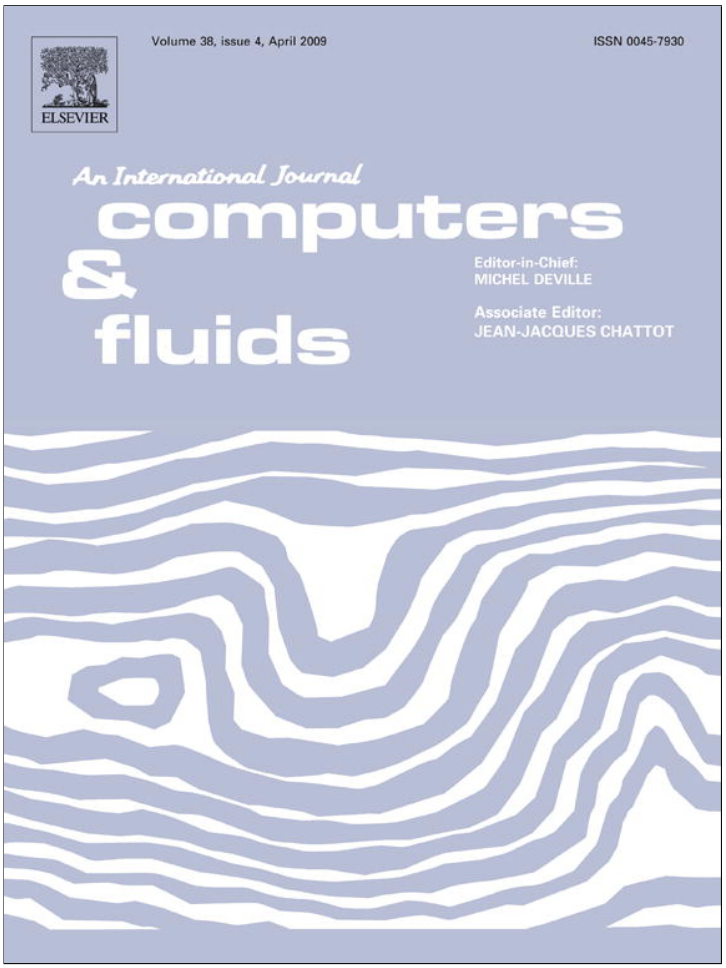

Computers & Fluids 38 (2009) 875–882

Contents lists available at ScienceDirect

Computers & Fluids

journal homepage: www.elsevier.com/locate/compfluid

A hybrid method for unsteady inviscid fluid flow ☆

Jan Nordström [a,b,c,*], Frank Ham [d], Mohammad Shoeybi [d], Edwin van der Weide [e,g], Magnus Svärd [d,f], Ken Mattsson [a,d], Gianluca Iaccarino [d], Jing Gong [a]

[a] Department of Information Technology, Scientific Computing, Uppsala University, SE-751 05 Uppsala, Sweden
[b] Department of Aeronautical and Vehicle Engineering, KTH, The Royal Institute of Technology, SE-100 44 Stockholm, Sweden
[c] Department of Computational Physics, FOI, The Swedish Defence Research Agency, SE-164 90 Stockholm, Sweden
[d] Center for Turbulence Research, CTR, Building 500, Stanford University, Stanford, CA 94305-3035, USA
[e] Department of Aeronautics and Astronautics, Stanford University, Stanford, CA 94305-4035, USA
[f] Center of Mathematics for Applications, University of Oslo, P.B. 1053 Blindern N-0316 Oslo, Norway
[g] Faculty of Engineering Technology, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands

ARTICLE INFO

Article history:
Received 27 November 2007
Received in revised form 17 September 2008
Accepted 18 September 2008
Available online 17 October 2008

ABSTRACT

We show how a stable and accurate hybrid procedure for fluid flow can be constructed. Two separate solvers, one using high order finite difference methods and another using the node-centered unstructured finite volume method are coupled in a truly stable way. The two flow solvers run independently and receive and send information from each other by using a third coupling code. Exact solutions to the Euler equations are used to verify the accuracy and stability of the new computational procedure. We also demonstrate the capability of the new procedure in a calculation of the flow in and around a model of a coral.

1. Introduction

The generation and transportation of vortexes from wing-tips, rotors and wind turbines, and the generation and propagation of sound from aircraft, cars and submarines require methods that can handle locally highly nonlinear phenomena in complex geometries as well as efficient and accurate signal transportation in domains with smooth flow and geometries. This technique can also be used in adapting an essentially structured mesh to a curved shock.

The combination of finite volume methods on unstructured grids (for the part with nonlinear phenomena and complex geometries) and high-order finite difference methods on structured grids (for the wave propagation part) meet these demands. In many cases separate stand-alone codes using these methods also exist. In this paper we will show how to combine the finite volume and finite difference method and the related codes into a practical procedure.

1.1. Background, main ideas and previous results

There are essentially two different types of hybrid methods. The most common one employs different governing equations in different parts of the computational domain. A typical example is noise generated in an isolated part of the flow, considered as the sound source. The nonlinear phenomenon in the complex geometry is often computed by the Euler or Navier–Stokes equations. The sound propagation to the far field is considered governed by the linear wave equation with source terms from the Euler or Navier–Stokes calculation. This type of hybrid method is discussed in [1,2].

In this paper we consider another type of hybrid method that use the same governing equations (in this case the compressible Euler equations) in the whole computational domain, not just close to the source. The word hybrid refers to the use of different numerical methods in different parts of the computational domain. Examples of this type of hybrid method can be found in [3,4]. In this type of coupling procedure (provided that accurate data are known) a stable and accurate numerical procedure does suffice for convergence to the true solution.

Many of the flow phenomena that we are interested in last for long times and information propagate over long distances. Strict stability which prevents error growth on realistic mesh sizes, is very important for calculations over long times. We have derived and studied strictly stable unstructured finite volume methods (see [5–7]) and high-order finite difference methods (see [8–13]) for both hyperbolic, parabolic and incompletely parabolic problems. These methods employ so-called summation-by-parts operators and impose the boundary conditions weakly (see [5,14]).

In [15] it was proved that a specific interface procedure connecting finite difference methods and finite volume methods

☆ This work was carried out while the first and last author were visiting CTR.

* Corresponding author. Address: Department of Information Technology, Scientific Computing, Uppsala University, SE-751 05 Uppsala, Sweden.
E-mail address: Jan.Nordstrom@foi.se (J. Nordström).

0045-7930/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compfluid.2008.09.010

is stable for hyperbolic systems of equations. This study will rely heavily on these results and we will apply the theoretical results to the Euler equations. We will demonstrate that the theoretical results in [15] in combination with two existing efficient codes and a third coupling code will lead to an efficient and practical computational tool.

A three-dimensional code (CDP) that uses the node-centered finite volume method mentioned above has been developed in the Center for Turbulence Research (CTR) at Stanford University, see [16]. Another three-dimensional multi-block code (NSSUS) that uses the finite difference technique discussed above is available at the Department of Aeronautics & Astronautics at Stanford University, see [17]. These codes compute approximations to the Euler or Navier–Stokes equations and are the initial building blocks for the new hybrid method. A third coupling code (Chimps-lite, a simplified version of Chimps [18]) will administer the coupling procedure and make it possible for the two solvers to communicate in an efficient and scalable way.

The rest of this paper will proceed as follows. For completeness, we shortly review the results in [15] in Section 2. In Section 3 we describe the two sets of computational solvers and the specific coupling code. In Section 4 we validate the computational procedure against exact solutions and show the ability to cope with complex geometries and high accuracy requirements. Finally, we draw conclusions and discuss future work in Section 5.

2. Analysis

To introduce our technique (see [15]) we consider the hyperbolic system

$$u_t + Au_x + Bu_y = 0, \qquad -1 \leqslant x \leqslant 1, \quad 0 \leqslant y \leqslant 1 \tag{1}$$

with suitable initial and boundary conditions. A and B are constant symmetric matrices with k rows and columns. We consider a simplified computational domain that is divided into two sub-domains. A so-called node-centered unstructured finite volume method will be used to discretize (1) on sub-domain $[-1,0]\times[0,1]$ with an unstructured mesh, while a high-order finite difference method will be used on sub-domain $[0,1]\times[0,1]$ with a structured mesh, see Fig. 1. The fact that the unknowns in the finite volume and the finite difference methods are located in the nodes and can be co-located at the interface is a key ingredient in the coupling procedure we will present below.

2.1. The node-centered finite volume method

The so-called node-centered finite volume method is used in this paper (see [19–23] for more details). In [5,15] it was shown that the semi-discrete finite volume form of (1) on sub-domain $[-1,0]\times[0,1]$ can be written,

$$\mathbf{u}_t + \left\{D_x^L \otimes A\right\}\mathbf{u} + \left\{D_y^L \otimes B\right\}\mathbf{u} = \mathrm{SAT}_I^L(\mathbf{u}_I - \mathbf{v}_I) + \mathrm{SAT}_O^L. \tag{2}$$

The difference operators and the penalty term that imposes the interface conditions have the form (see [15])

$$D_x^L = (P^L)^{-1}Q_x^L, \quad D_y^L = (P^L)^{-1}Q_y^L, \quad \mathrm{SAT}_I^L = \left[(P^L)^{-1}(E_I^L)^T Y_I\right] \otimes \Sigma^L.$$

SAT_O^L imposes the outer boundary conditions weakly. $\mathbf{u}_I$ and $\mathbf{v}_I$ are vectors which represent $\mathbf{u}$ and $\mathbf{v}$ ($\mathbf{v}$ is the discrete finite difference solution that will be presented below) on the interface, respectively. E_I^L is a projection matrix which maps $\mathbf{u}$ to $\mathbf{u}_I$ such that $\mathbf{u}_I = (E_I^L \otimes I_k)\mathbf{u}$. The non-zero components of E_I^L have the value 1 and appear at the interface. I_k is the $k\times k$ identity matrix. $Y_I \otimes \Sigma^L$ is a penalty matrix that will be determined below by stability requirements.

P^L is a positive diagonal $m\times m$ matrix with the control volumes Ω_i on the diagonal. Q_x^L and Q_y^L are almost skew symmetric $m\times m$ matrices and satisfy

$$Q_x^L + (Q_x^L)^T = Y, \qquad Q_y^L + (Q_y^L)^T = X, \tag{3}$$

where the non-zero elements in Y and X are Δy_i, $-\Delta x_i$ and correspond to the boundary points. For the definition of Δx_j and Δy_j, see Fig. 2. The part of the penalty term SAT_I^L denoted by Y_I is the restriction of Y to the interface. For more details on the SBP properties of the finite volume scheme and how to implement the outer boundary conditions weakly, see [5].

2.2. The high-order finite difference method

The high-order finite difference method used in this paper is described in [8–13]. Consider the sub-domain $[0,1]\times[0,1]$ with a structured mesh of $n\times l$ points. The finite difference approximation of u at the grid point (x_i,y_j) is a $k\times 1$ vector denoted $\mathbf{v}_{ij}$. We organize the solution in the global vector $\mathbf{v} = [\mathbf{v}_{11},\ldots,\mathbf{v}_{1l},\mathbf{v}_{21},\ldots,\mathbf{v}_{2l},\ldots,\mathbf{v}_{n1},\ldots,\mathbf{v}_{nl}]^T$. $\mathbf{v}_x$ and $\mathbf{v}_y$ are approximations of u_x and u_y and are approximated using the high-order accurate SBP operators for the first derivative that were constructed in [24,25].

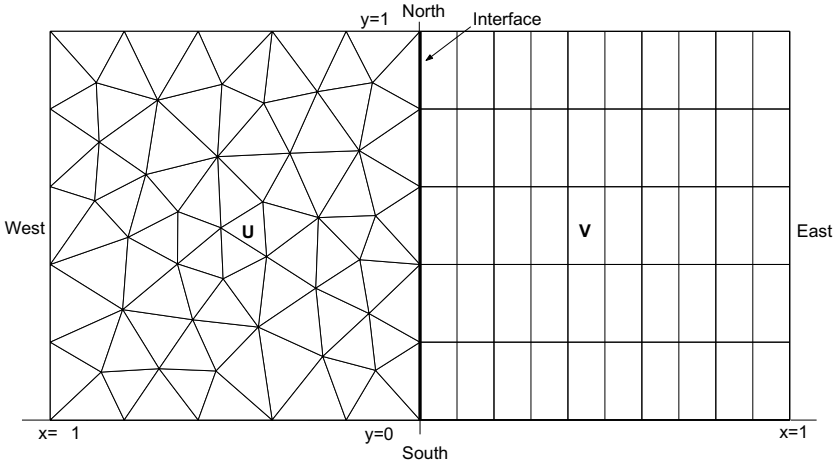

Fig. 1. The hybrid mesh on the computational domain.

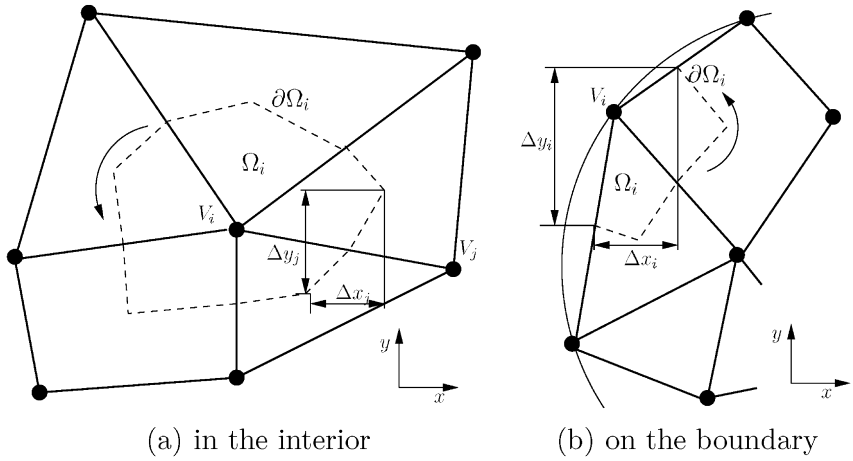

Fig. 2. The grid (solid lines) and the dual grid (dashed lines).

The semi-discrete approximation of (1) on $[0,1]\times[0,1]$ can be written,

$$\mathbf{u}_t + \left\{D_x^R \otimes A\right\}\mathbf{u} + \left\{D_y^R \otimes B\right\}\mathbf{u} = \mathrm{SAT}_I^R(\mathbf{u}_I - \mathbf{v}_I) + \mathrm{SAT}_O^R. \tag{4}$$

The difference operators and the penalty term that imposes the interface conditions have the form (see [15])

$$D_x^R = \left[(P_x^R)^{-1}Q_x^R\right]\otimes I_y^R, \quad D_y^R = I_x^R \otimes \left[(P_y^R)^{-1}Q_y^R\right],$$
$$\mathrm{SAT}_I^R = \left[(P_x^R \otimes P_y^R)^{-1}(E_I^R)^T\right]P_y^R \otimes \Sigma^R.$$

SAT_O^R imposes outer boundary conditions weakly. The identity matrices I_x^R and I_y^R are $n\times n$ and $l\times l$, respectively. E_I^R is a projection matrix which maps $\mathbf{v}$ to $\mathbf{v}_I$, that is, $\mathbf{v}_I = (E_I^R \otimes I_k)\mathbf{v}$. Σ^R is a penalty matrix that will be determined below by stability requirements. Details on how to implement the outer boundary conditions weakly for the finite difference method are given in [8].

Furthermore, $(P_x^R)^{-1}Q_x^R$ and $(P_y^R)^{-1}Q_y^R$ are SBP operators since matrices P_x^R and P_y^R are symmetric and positive definite and

$$Q_x^R + \left(Q_x^R\right)^T = B_x^R, \quad Q_y^R + \left(Q_y^R\right)^T = B_y^R. \tag{5}$$

B_x^R, B_y^R are diagonal matrices with the structure $(-1,0,.0,1)$ and size $n\times n$ and $l\times l$, respectively.

Remark. The approach we use in this paper above is closely related to the interior penalty technique used in finite element and discontinuous Galerkin methods. The penalty terms in Eqs. (2) and (4) are often denoted lifting operators in that context, see [26] for more details.

2.3. Stable interface treatment

Define the norms $N^L = P^L \otimes I_k$ and $N^R = (P_x^R \otimes P_y^R)\otimes I_k$, where $N^L = (N^L)^T > 0$ and $N^R = (N^R)^T > 0$. We apply the energy method by multiplying (2) and (4) with $\mathbf{u}^T N^L$ and $\mathbf{v}^T N^R$, respectively. We also use (3) and (5) and assume that the terms including $\mathbf{u}_B$, $\mathbf{v}_E$, $\mathbf{v}_S$, $\mathbf{v}_N$ at the outer boundaries are precisely canceled by the SAT terms (see [8,9]). This yields the energy estimate

$$\frac{d}{dt}(\|u\|_{N^L}^2 + \|u\|_{N^R}^2) = [\mathbf{u}_I, \mathbf{v}_I]^T M_I [\mathbf{u}_I, \mathbf{v}_I], \tag{6}$$

$$M_I = \begin{bmatrix} -Y_I \otimes A + Y_I \otimes (\Sigma^L + (\Sigma^L)^T) & -Y_I \otimes \Sigma^L - P_y^R \otimes \Sigma^R \\ -Y_I \otimes \Sigma^L - P_y^R \otimes \Sigma^R & P_y^R \otimes A + P_y^R \otimes (\Sigma^R + (\Sigma^R)^T) \end{bmatrix}.$$

The penalty matrices Σ^L and Σ^R have the form $X^T\Sigma^L X = \Lambda^L$ and $X^T\Sigma^R X = \Lambda^R$. X is the orthogonal matrix that diagonalizes A, i.e. $X^T A X = \Lambda$. λ_i^L and λ_i^R are the ith diagonal components in Λ^L and Λ^R, respectively.

In [15] it was shown that M_I was negative semi-definite if

$$Y_I = P_y^R = P_y, \qquad \Sigma^L = (\Sigma^L)^T, \qquad \Sigma^R = (\Sigma^R)^T, \tag{7}$$

$$\lambda_i^R = \lambda_i^L - \lambda_i, \qquad \lambda_i^L \leqslant \lambda_i/2, \qquad i = 1,\ldots,k \tag{8}$$

holds. Negative semi-definiteness of M_I leads directly to

Proposition 2.3.1. *If the conditions (7) and (8) hold, (6) leads to a bounded energy and (2) and (4) have a stable and conservative interface treatment.*

For more details on how to implement the interface conditions weakly for the coupled hybrid method, see [15].

Remark. The conditions $\lambda_i^R = \lambda_i^L - \lambda_i$ in (8) in combination with (7) lead to a conservative interface treatment. The conservation conditions are obtained by multiplying (2) and (4) with $\boldsymbol{\phi}^T N^L$ and $\boldsymbol{\phi}^T N^R$, respectively. By using the continuity of the smooth grid function $\boldsymbol{\phi}$ at the interface together with (3), (5) and (7) and requiring that all terms vanish at the interface leads to $\lambda_i^R = \lambda_i^L - \lambda_i$. For more details, see [8–10,15].

The specific SBP operators based on diagonal norms are given in [11,25]. When we use the second-order diagonal norm on the right sub-domain, we do not need to change the control volume since $Y_I = P_y^R$ automatically. But the standard fourth- and sixth-order diagonal norms are

$$h\cdot\mathrm{diag}\left(\frac{17}{48}, \quad \frac{59}{48}, \quad \frac{43}{48}, \frac{49}{48}, \quad 1, \quad 1,1,1,\ldots,\right)$$

$$h\cdot\mathrm{diag}\left(\frac{13{,}649}{43{,}200}, \quad \frac{12{,}013}{8640}, \quad \frac{2711}{4320}, \quad \frac{5359}{4320}, \quad \frac{7877}{8640}, \quad \frac{43{,}801}{43{,}200}, 1,1,\ldots,\right)$$

respectively. In both cases we need to modify the control volume for the finite volume method at the points on the interface to guarantee $Y_I = P_y^R$. The old dual grid for the points at the interface consists of the lines between the center of the triangles and the midpoints of the edges. In order to match Y_I and P_y^R, the new lines will connect the center of the triangles and the points at the interface which correspond to the P_y^R, see Fig. 3.

Remark. The hybrid method described above can be extended in a straightforward manner to three dimensions by interfacing hexa-

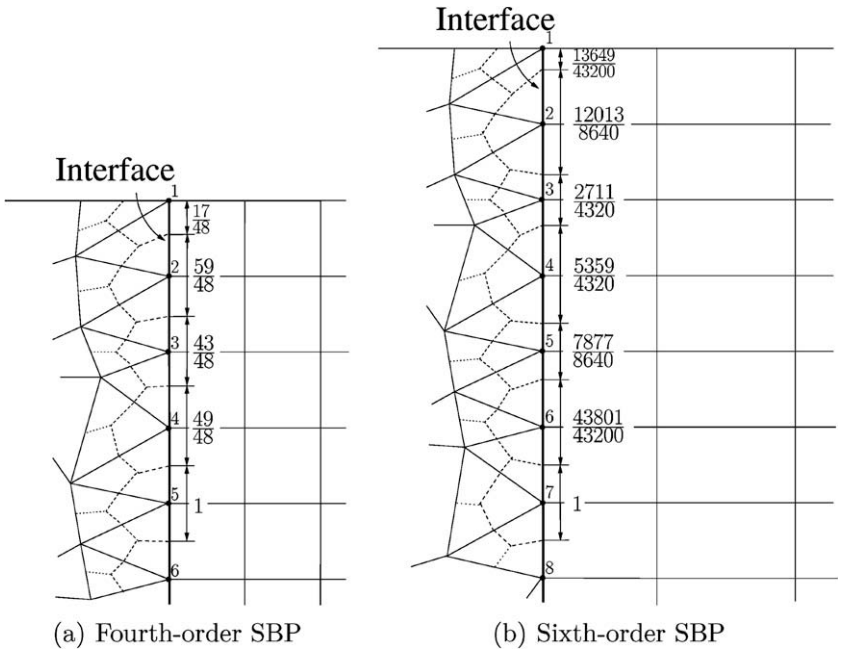

Fig. 3. The modified control volumes for the points on the interface.

hedra from the structured side with pyramids on the unstructured side. Stability will be obtained by modifying the corresponding two-dimensional finite volume norm (choose the dual grid properly) to match the two-dimensional finite difference norm.

3. Computational tools

The node-centered finite volume code (CDP) and the high order finite difference code (NSSUS) are the initial building blocks for the new hybrid method. Both CDP and NSSUSS are stand-alone codes that computes approximations to the Euler or Navier–Stokes equations. The codes are node-based and use SBP operators and penalty techniques for imposing the boundary and interface conditions weakly. This numerical technique enables coupling of the two codes by sending the value of the dependent variables in the nodes located on the interface to the other code and at the same time receiving the co-located data at the interface from the other code. Each code provides boundary data to the other code.

A third coupling code (Chimps-lite) will administer the coupling procedure and make it possible for the two solvers to communicate in a correct way. Chimps-lite identifies co-located nodes in a preprocessing step and during the execution it manages the exchange of data between CDP and NSSUS, at each stage in the explicit time-integration procedure [27]. See Fig. 4 for a schematic illustration.

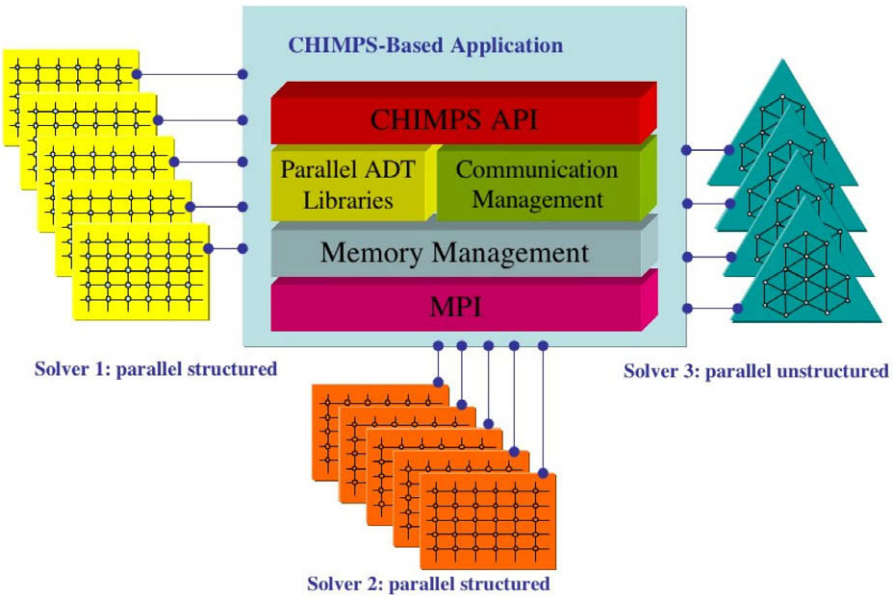

Fig. 4. Schematic interface communication.

The development of Chimps-lite is an essential new ingredient that will take the coupling idea from a theoretical concept to a practically useful tool for fluid flow investigations. It will be discussed in some detail below.

3.1. Chimps-lite and parallel implementation consideration

In addition to the mathematical and numerical foundation presented previously, the development of a massively parallel (say 1000+ processor) hybrid simulation capability requires a fast, scalable model for the regular exchange of data between the various solvers.

One option is to write a new hybrid solver that merges the desired solver capabilities and includes an additional layer of communication associated with the interfaces. While this option will allow us to continue to run in the single-program-multiple-data (SPMD) mode that has emerged as the dominant model for large-scale parallel computation, it has the down-side of requiring major modifications to both codes. In some cases the codes will be written in different languages. There may be global name-space conflicts that prevent us from simply writing a common "main" that calls the appropriate solver as a "subroutine". The required re-coding will invariably introduce additional bugs that must be corrected. When the stand-alone codes have had 10s or even hundreds of man-years of development, verification, and validation, this level of intervention is normally unacceptable.

An alternative approach is to run both codes in a stand-alone mode under one multiple-program-multiple-data (MPMD) session, and use an additional library of routines to handle the interface communication. This approach requires minimal modifications to the codes, and is the one we follow in the present implementation using a simplified version of the CHIMPS coupling library [18] called CHIMPS-lite.

In this model, only two minor modifications are required to the participating codes – one associated with the initialization of MPI and the other with exchanging data prior to application of the interface conditions. These modifications are described in the following subsections. Note that the MPI commands in the next section are shell dependent.

3.2. Communicator splitting

Many MPI environments support running MPMD from the command line. For example, the command:

```
mpirun -np 10 codel: -np 25 code2
```

will run `codel` on 10 processors, and `code2` on 25 processors (i.e. 35 processors total), all having a common world communicator. For systems that do not support this model, the same thing can be accomplished through modifications to the job submission script, or even by using a simple shell script that parses the processor number normally available as an environment variable. For example:

```
mpirun -np 35 mpmd.sh
```

where `mpmd.sh` is the following:

```
#!/bin/bash
if [$ PROCID -le 9]
then./codel
else./code2
fi
```

In either case, both codes will start up with a common world communicator that must be split before normal execution can continue. This splitting is the first modification required in the codes. It can be done at the point where the routine `MPI_Init()` would normally be called. Instead, both codes should call a common routine in the coupling library that takes a unique key or name from each solver, calls `MPI_Init()`, and then splits the communicator based on the key, returning a unique communicator to each solver. All local MPI routines in the solvers should then use this split communicator for their local MPI calls, and not `MPI_COMM_WORLD`. This call will of course be collective and blocking.

3.3. Data exchange

In the context of SBP/SAT codes, the application of interface conditions is very similar to boundary conditions except that the data used in forming the penalty terms comes from the point-matched interface data associated with the other solver. The second modification to the participating codes is thus an exchange of interface data prior to forming the interface penalty terms. For the present computations involving fixed grids and conformal interfaces, the building of the communication pattern associated with this exchange can be considered a preprocessing step because it remains fixed throughout the simulation. Consequently the scalability of the searches is not critical, however, we have used the scalable search routines described in [18] to locate matching points.

On the first call to the data exchange, both codes provide the list of coordinates of their interface points to CHIMPS-lite, which then proceeds to build the communication pattern associated with the data exchange based on matching point coordinate locations between solvers (within a small tolerance). On subsequent calls, the same communication pattern can be reused, making the cost of each exchange very modest. Fig. 4 illustrates this process schematically. By using the CHIMPS-lite routines to build the communication pattern and manage the data exchanges, it is never necessary for the solvers to have any direct knowledge of each other's partitioning details.

4. Numerical calculations

The coupling procedure applied to the scalar advection equation was extensively tested and validated in [15]. Here we will make sure the theoretical results derived above for the symmetrized constant coefficient system also apply to the fully non-linear Euler equations.

Remark. The stability analysis above was done on a system with constant symmetric matrices. Here we multiply the constant coefficient version of the Euler equations with a symmetrizing matrix from the left. We use the symmetrizer derived in [28]. Once the penalty matrices are obtained, the whole system is transformed back to non-symmetric form and implemented.

4.1. Validation

The coupling procedure applied to the scalar advection equation was extensively tested and validated in [15]. Here we will make sure these results also apply to a non-linear system of equations (the Euler equations). We calculate the propagation of a vortex with constant velocity (an exact solution to the Euler equations) across an interface. A typical mesh for this calculation is shown in Fig. 5. We use the explicit three-stage third-order time-stepping scheme of Le and Moin [27]. The solution is advanced with a global time-step based on the smallest grid size in both domains.

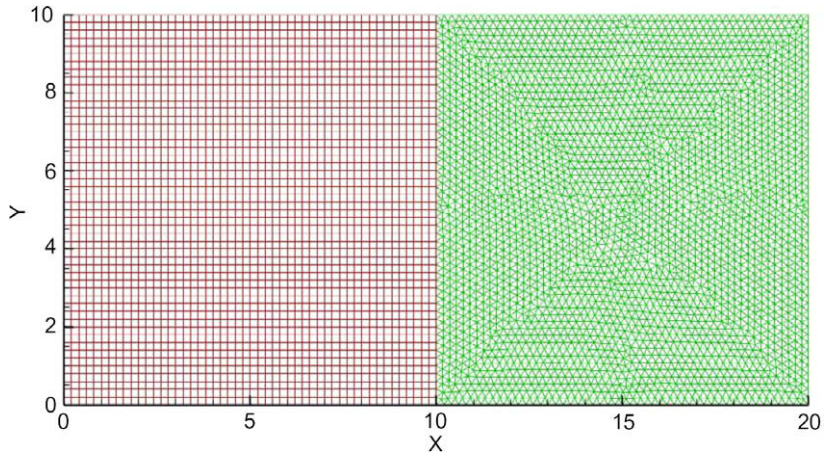

Fig. 5. Mesh for the accuracy validation of the interface procedure.

The accuracy of the coupled procedure for various orders of accuracy of the NSSUS finite difference code is shown in Tables 1–5. The errors are computed when the vortex is centered at the interface. The presence of second-order errors produced by CDP will limit the overall convergence rate to 2 even if NSSUS runs with higher accuracy. Similar results have been produced for various combinations of directions and orders of accuracy in NSSUS and they indicate that the procedure converges with the appropriate rate.

As can be seen in the tables, the highest gain in accuracy is obtained for the case where the vortex propagates from the high-order accurate NSSUS region into the second-order accurate CDP region. We illustrate that in Fig. 6 which shows the solution and the error for the coupling between NSSUS (second and fourth order) and CDP. The error levels in both calculations are very small (of the order 10^{-4}). The error levels for the second-order NSSUS are visible long before the vortex hits the interface. For the fourth-order case, nothing can be seen until the vortex reaches the interface.

Table 1
Error as vortex propagate from second-order NSSUS region intosecond-order CDP region.

N	$\log\left(l_2^{(\mathrm{NSSUS})}\right)$	$q^{(\mathrm{NSSUS})}$	$\log\left(l_2^{(\mathrm{CDP})}\right)$	$q^{(\mathrm{CDP})}$
51×51	−2.88	–	−3.22	–
101×101	−3.50	2.06	−3.77	1.83
201×201	−4.11	2.03	−4.36	1.96
401×401	−4.71	2.00	−4.96	1.99

Table 2
Error as vortex propagate from fourth-order NSSUS into region second-order CDP region.

N	$\log\left(l_2^{(\mathrm{NSSUS})}\right)$	$q^{(\mathrm{NSSUS})}$	$\log\left(l_2^{(\mathrm{CDP})}\right)$	$q^{(\mathrm{CDP})}$
51×51	−4.00	–	−3.82	–
101×101	−4.89	2.96	−4.41	1.98
201×201	−5.63	2.47	−5.01	2.00
401×401	−6.30	2.21	−5.62	2.02

Table 3
Error as vortex propagate from 6th-order NSSUS region into second-order CDP region.

N	$\log\left(l_2^{(\mathrm{NSSUS})}\right)$	$q^{(\mathrm{NSSUS})}$	$\log\left(l_2^{(\mathrm{CDP})}\right)$	$q^{(\mathrm{CDP})}$
51×51	−3.89	–	−3.82	–
101×101	−4.92	3.41	−4.41	1.98
201×201	−5.68	2.53	−5.01	2.00
401×401	−6.31	2.11	−5.62	2.02

Table 4
Error as vortex propagate from second-order CDP region into second-order NSSUS region.

N	$\log\left(l_2^{(\mathrm{CDP})}\right)$	$q^{(\mathrm{CDP})}$	$\log\left(l_2^{(\mathrm{NSSUS})}\right)$	$q^{(\mathrm{NSSUS})}$
51×51	−3.03	–	−3.01	–
101×101	−3.65	2.05	−3.60	1.93
201×201	−4.25	2.00	−4.19	1.98

Table 5
Error as vortex propagate from second-order CDP region into fourth-order NSSUS region.

N	$\log\left(l_2^{(\mathrm{CDP})}\right)$	$q^{(\mathrm{CDP})}$	$\log\left(l_2^{(\mathrm{NSSUS})}\right)$	$q^{(\mathrm{NSSUS})}$
51×51	−3.03	–	−3.06	–
101×101	−3.65	2.05	−3.65	1.95
201×201	−4.25	2.00	−4.25	1.98

4.2. An application

The new hybrid method can handle nonlinear phenomena in complex geometries as well as efficient and accurate signal transportation in domains with smooth flow and geometries. We demonstrate that capability in a calculation of the flow through a two-dimensional model of a coral. In this calculation we use the sixth-order accurate version of NSSUS. The geometry and the corresponding mesh can be seen in Fig. 7. The center of the coral is at $(x, y) = (0, 0)$. The interface is located at $x = 0.6$. The calculation proceeds as follows. First we compute a steady state solution. Next, we take the steady state solution and add the vortex centered at $(x, y) = (-1.5, 0)$. That is our initial solution, see Fig. 8a.

As time passes, the vortex propagates through the coral (in the unstructured finite volume region) and sits at $t = 2.6$ just at the interface leaving the coral, see Fig. 8b. The shape of the vortex is modified by the coral. At $t = 3.6$ the vortex has left the coral region and is now propagating (with sixth-order accuracy) downstream, see Fig. 9a. The vortex seems to return to it's original form. Finally, at $t = 4.4$ the vortex is approaching the right boundary and it is even closer to it's original shape, see Fig. 9b.

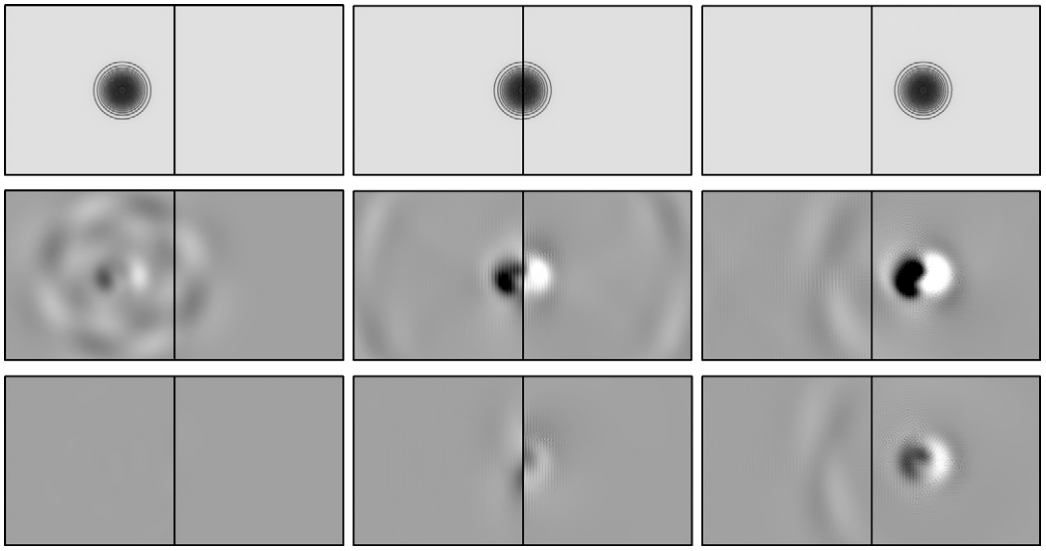

Fig. 6. Vortex propagating over interface. The columns correspond to three different times. The first row show the density distribution. The second and third rows show the density error for the second and fourth-order NSSUS coupled to CDP, respectively.

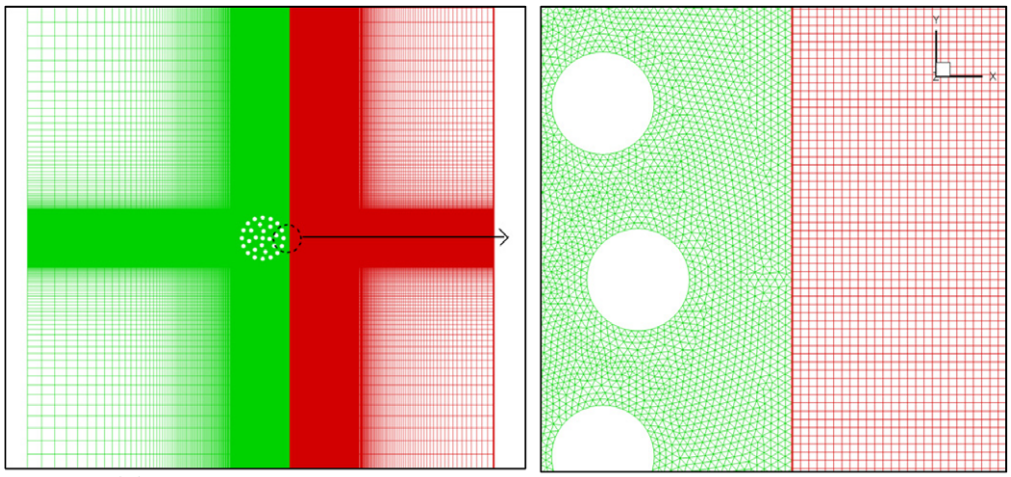

Fig. 7. Geometry and grid topology of hybrid calculation around coral.

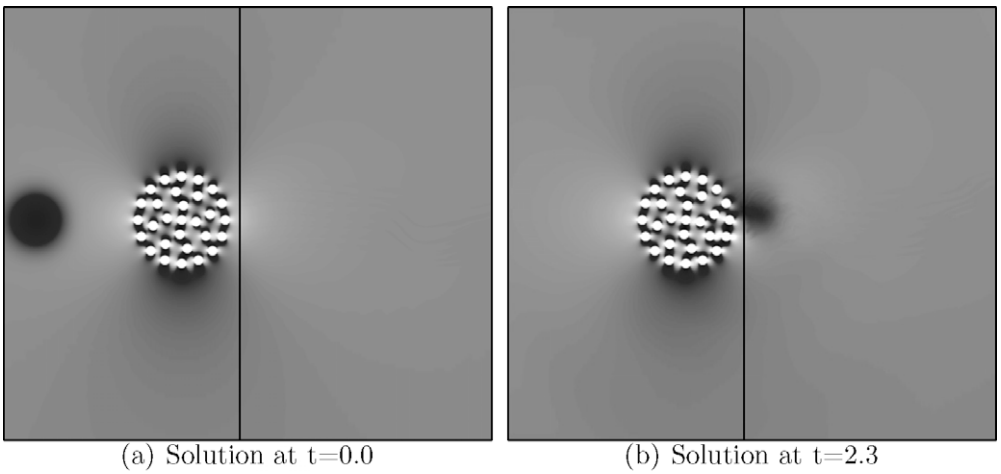

Fig. 8. Time sequence for the vortex–coral interaction, $t = 0.0$ and $t = 2.3$.

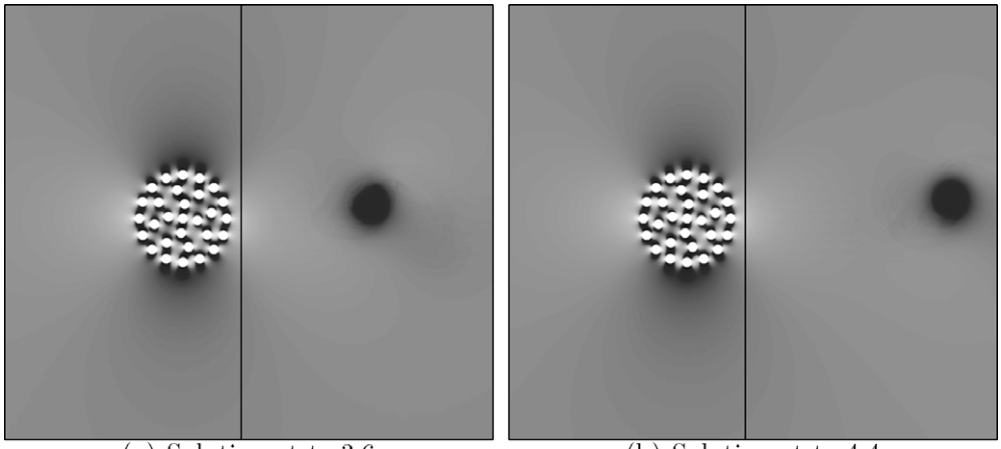

Fig. 9. Time sequence for the vortex–coral interaction, $t = 3.6$ and $t = 4.4$.

5. Conclusions and future work

We have developed a hybrid method constructed by the coupling of two stand-alone existing CFD codes. The coupling is administered by a third separate coupling code.

The hybrid method allows for individual development of the stand-alone CFD codes. No development with consideration to the other code is required since the CFD codes only communicate with each other through the third coupling code.

We have demonstrated that the hybrid method is an accurate, efficient and a practically useful computational tool that can handle complex geometries as well as wave propagation phenomena.

The next step involves including the viscous terms. Initial work in that direction is ongoing, see [29]. With an efficient hybrid method for the Navier–Stokes equations, large scale flow problems in complex geometries including wave propagation effects, can be analyzed.

References

[1] Lyrintzis AS. Review: the use of Kirchhoff's method in computational aeroacoustics. J Fluids Eng 1994;116:665–76.
[2] Wells VL, Renaut RA. Computing aerodynamically generated noise. Annu Rev Fluid Mech 1997;29:161–99.
[3] Burbeau A, Sagaut P. A dynamic p-adaptive discontinuous Galerkin method for viscous flow with shocks. Comput Fluids 2005;34(4–5):401–17.
[4] Rylander T, Bondeson A. Stable FEM-FDTD hybrid method for Maxwell's equations. Comput Phys Commun 2000;125:75–82.
[5] Nordström J, Forsberg K, Adamsson C, Eliasson P. Finite volume, methods unstructured meshes and strict stability. Appl Numer Math 2003;45:453–73.
[6] Svärd M, Nordström J. Stability of finite volume approximations for the Laplacian operator on quadrilateral and triangular grids. Appl Numer Math 2004;51:101–25.
[7] Svärd M, Gong J, Nordström J. Strictly stable artificial dissipation for finite volume schemes. Appl Numer Math 2007;56(12):1481–90.
[8] Carpenter MH, Nordstrom J, Gottlieb D. A stable and conservative interface treatment of arbitrary spatial accuracy. J Comput Phys 1999;148:341–6.
[9] Nordström J, Carpenter MH. Boundary and interface conditions for high order finite difference methods applied to the Euler and Navier–Stokes equations. J Comput Phys 1999;148:621–45.
[10] Nordström J, Carpenter MH. High-order finite difference methods, multidimensional linear problems and curvilinear coordinates. J Comput Phys 2001;173:149–74.
[11] Mattsson K, Nordström J. Summation by parts operators for finite difference approximations of second derivatives. J Comput Phys 2004;199:503–40.
[12] Svärd M, Nordström J. On the order of accuracy for difference approximations of initial-boundary value problems. J Comput Phys 2006;218(1):333–52.
[13] Svärd M, Carpenter MH, Nordström J. A stable high-order finite difference scheme for the compressible Navier–Stokes equations: far-field boundary conditions. J Comput Phys 2007;225(1):1020–38.
[14] Carpenter MH, Gottlieb D, Abarbanel S. Time-stable boundary conditions for finite-difference schemes solving hyperbolic systems: methodology and application to high-order compact schemes. J Comput Phys 1994;129(2).
[15] Nordström J, Gong Jing. A stable hybrid method for hyperbolic problems. J Comput Phys 2006;212(2):436–53.
[16] Ham F, Mattsson K, Iaccarino G. Accurate and stable finite volume operators for unstructured flow solvers. Annual Research Briefs, CTR, Stanford University; 2006.
[17] Svard M, Van der Weide E. Stable and high-order accurate finite difference schemes on singular grids. Annual Research Briefs, CTR, Stanford University; 2006.
[18] J.J. Alonso, S. Hahn, F. Ham, M. Herrmann, G. Iaccarino, G. Kalitzin, P. LeGresley, K. Mattsson, G. Medic, P. Moin, H. Pitsch, J. Schluter, M. Svard, E. van der Weide, D. You, and X. Wu, Chimps: A high-performance scalable module for multi-physics simulations, AIAA Paper 2006-5274, 2006.
[19] T. Gerhold, O. Friedrich, and J. Evans, Calculation of complex three-dimensional configurations employing the DLR-τ-code, AIAA Paper 97-0167, 1997.
[20] Haselbacher A, McGuirk JJ, Page GJ. Finite volume discretization aspects for viscous flows on mixed unstructured grids. AIAA J 1999;37(2).
[21] Mavriplis DJ. Accurate multigrid solution of the Euler equations on unstructured and adaptive meshes. AIAA J 1990;28(2).
[22] Mavriplis DJ, Venkatakrishnan V. A unified multigrid solver for the Navier–Stokes equations on mixed element meshes, Technical report, Institute for Computer Applications in Science and Engineering; 1995.
[23] Weiss JM, Maruszewski JP, Smith WA. Implicit solution of preconditioned Navier–Stokes equations using algebraic multigrid. AIAA J 1999;37(1).
[24] Kreiss H-O, Scherer G. Finite element and finite difference methods for hyperbolic partial differential equations. In: De Boor C, editor. Mathematical aspects of finite elements in partial differential equation. New York: Academic Press; 1974.
[25] Strand B. Summation by parts for finite difference approximation for d/dx. J Comput Phys 1994;110(1):47–67.
[26] Arnold DN, Brezzi F, Cockburn B, Marini LD. Unified analysis of discontinuous Galerkin methods for eliptic problems. SIAM J Numer Anal 2002;39(5):1749–79.
[27] Le H, Moin P. An improvement of fractional step methods for the incompressible Navier–Stokes equations. J Comput Phys 1991;92:369–79.
[28] Abarbanel S, Gottlieb D. Optimal time splitting for two- and three-dimensional Navier–Stokes equations with mixed derivatives. J Comput Phys 1981;41:1–43.
[29] Gong J, Nordström J. A stable and efficient hybrid scheme for viscous problems in complex geometries. J Comput Phys 2007;226:1291–309.