Journal of Computational Physics 335 (2017) 21–49

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Conservative handling of arbitrary non-conformal interfaces using an efficient supermesh

Horacio J. Aguerre [a,b,*], Santiago Márquez Damián [a,c], Juan M. Gimenez [a,d], Norberto M. Nigro [a,d]

[a] CIMEC, Centro de Investigación de Métodos Computacionales, UNL, CONICET, Colectora Ruta Nacional 168 s/n, Predio Conicet "Dr Alberto Cassano", 3000 Santa Fe, Argentina
[b] Universidad Tecnológica Nacional – Facultad Regional Concepción del Uruguay, UTN-FRCU, Ing. Pereyra 676, 3260 Concepción del Uruguay, Argentina
[c] Universidad Tecnológica Nacional – Facultad Regional Santa Fe, UTN-FRSF, Lavaise 610, 3000 Santa Fe, Argentina
[d] Universidad Nacional del Litoral – Facultad de Ingeniería y Ciencias Hídricas, UNL-FICH, Ciudad Universitaria, 3000 Santa Fe, Argentina

ARTICLE INFO

Article history:
Received 20 July 2016
Received in revised form 31 December 2016
Accepted 12 January 2017
Available online 16 January 2017

Keywords:
Finite volume method
Non-conformal interfaces
Conservative interpolation
Supermesh
Unstructured meshes

ABSTRACT

This work presents a new and efficient strategy to handle non-conformal interfaces with the aim of assuring the conservation of fluxes in Finite Volume problems. A conservative interpolation is developed for general transport equations. Due to the arbitrary connectivity between the interfaces, the interpolations require flux-based weights defining a complex numerical stencil. In this context, a new method is proposed to simplify the coupling at the interface based on the construction of a simplified supermesh. Here, a supermesh is not completely defined, instead, the interface faces are logically duplicated (or multiplied) to generate a one-to-one connectivity between them. The simplified supermesh named pseudo-supermesh eliminates the interpolations and assures the conservation of fluxes based on the trivial connectivity. The area and the geometrical centroid of the new faces are redefined according to the overlapped sector of the original faces using the local supermeshing approach. Since the arbitrary polygons resulting from the face intersections are not generated and introduced into the mesh, computational cost and implementation efforts are saved. The proposed method is tested focusing on the conservation of fluxes and on the accuracy, showing conservation to machine precision and second order convergence as expected. In order to be able to solve large problems, the methodology is designed and implemented to be run in parallel architectures showing an excellent efficiency.

1. Introduction

The use of interfaces between different domain regions is a very useful strategy in the area of Computational Fluid Dynamics (CFD). The interfaces divide the problem into different parts which may be favourable to simplify the meshing process. Moreover, the main advantage is the flexibility to handle moving zones with tangential motion such as turbo-machinery, internal combustion engines or stirred vessels, among others.

* Corresponding author at: CIMEC, Centro de Investigación de Métodos Computacionales, UNL, CONICET, Colectora Ruta Nacional 168 s/n, Predio CONICET "Dr Alberto Cassano", 3000 Santa Fe, Argentina. Fax: +54 342 4511169.
E-mail address: aguerrehoracio@gmail.com (H.J. Aguerre).

http://dx.doi.org/10.1016/j.jcp.2017.01.018
0021-9991/

The primary challenge of an interface method is to perform a numerical coupling without introducing errors in comparison with an analogous one-region problem. In this context, some methods eliminate the interfaces between the subdomains merging them into a single region. Therefore, the interface errors disappear but at the cost of stitching the mesh subdomains which is computationally demanding. Moreover, in moving boundary problems, the stitching procedure should be done frequently exacerbating this issue. A method of this type is the Sliding interfaces technique [1–3] which is designed to attach and detach subdomains in dynamic mesh simulations and is usually applied to solve internal combustion engine problems. Another similar technique is the Buffer layer method of Qin et al. [4] which is improved by Wang et al. to define the Zipper layer method [5]. Here, mesh blocks are joined without interfaces to avoid interpolations but these blocks can only be structured meshes.

On the other hand, many techniques introduce numerical approaches to address the physical communication between the unconnected regions avoiding the remeshing procedure. Therefore, these strategies are computationally more efficient than the previous methods but, as a counterpart, numerical errors may be introduced at the interface. An important and desirable property of these methods is the conservation of the fluxes in general transport problems. A non-conservation issue at the interface may cause loss of accuracy and stability problems in the simulations, especially when discontinuities are close to the interface [6]. Furthermore, some problems, such as Residence Time Distribution (RTD) determination, which are important in chemical processing, or the tracking of species in stirred vessels, require a strict conservation of the variables to obtain good predictions. Otherwise, non-conservation issues would act as mass sources which may invalidate the study and also trigger instabilities.

The problem of conservativeness at non-conformal interfaces was first formally studied by Berger [7] using conservative differencing in two-dimensional grids. Within these strategies, one of the most cited methods is the Patched-grid technique, presented initially by Rai, which introduces a conservative treatment of structured two-dimensional interfaces [8]. This approach is extended to rotor/stator applications in a subsequent work [9]. Lerut and Wu propose a Patched-grid technique which assures stability and conservativeness for multi-domain simulations with implicit Euler solvers [10] trying partially to cover the geometric complexity of real industrial problems. A more recent work establishes the use of Patched-grid in combination with a high-order linear reconstruction remapping using a WENO scheme [11]. Even though the Patched-grid is an adequate technique to handle multi-domain simulations of structured mesh blocks, most of the industrial problems are complex to address with such a mesh and in general come in an unstructured grid format. Inspired by this limitation, Mathur [12] proposes a more general and conservative strategy. This paper emphasises in the replacement of the two non-conformal interfaces by a new one which is considered as an internal patch. However, the centroid computation is omitted regardless the strong influence of such a geometric feature in the accurate computation of the fluxes. Another conservative method is the work of Loh et al. [13], where the data between interfaces is exchanged via surface fluxes but using only an upwind convective scheme. Recently, the work of Ramírez et al. [14] proposes a high-order interface method using moving least squares. However, the fully-conservative version proposed in this paper is computationally expensive due to the necessity of computing face/edge intersections. To solve the last inconvenience, the authors designed a halo-cell sliding mesh which avoids calculating intersections but incurs conservation problems.

On the other hand, non-conservative methods are in general easier to implement due to the simplicity of the interpolation expressions. It is usual to find linear combinations of values to reconstruct the face fluxes, without considering their conservativeness. A simple and effective technique to join non-conformal interfaces is the work of McNaughton et al. [15] which uses halo (ghost) nodes at the interfaces. Here, the interface is considered as a dual boundary condition for each one of the grids sharing the interface. This technique uses a non-conservative interpolation with the primary focus on the efficiency. Other works are included in the group of techniques that use overlapped-area weights to interpolate fields at the interface. In this sense, the work of Beaudoin and Jasak [16] presents the General Grid Interface (GGI) which is designed for turbomachinery applications. Another method of this type, which is implemented in the OpenFOAM(R) suite [17], is the Arbitrary Mesh Interface (AMI). It is similar to GGI, but it is based on the concept of supermesh presented by Farrell et al. [18] and uses the highly efficient local supermeshing approach proposed by Farrell and Maddison [19] to handle dissimilar meshes. Although the AMI is a computationally efficient methodology, the inconvenience lies in the loss of conservativeness that this method shows. Within this type of methods is also the work of Steijl and Barakos [20] which is particularly applied for helicopter aerodynamics.

In general, it can be concluded that conservative methods are easier to be developed using structured meshes. Attempts of conservative approaches for arbitrary meshes have a reduced computational efficiency and a high implementation complexity. On the other hand, unstructured conservative strategies with a simple conception are of low order accuracy. In this context, most of the engineering applications with multi-domain configurations use non-conservative strategies focusing on the combination of simplicity and efficiency. However, as described above, in some problems the conservativeness is a non-negotiable property.

In this sense, this paper presents a new strategy to couple arbitrary non-conformal interfaces in Finite Volume problems conceived to be conservative, computationally efficient and second-order accurate. Here, the main objective is to provide a fully conservative method with a higher computational efficiency than methods based on remeshing the interfaces. The proposed strategy uses the local supermeshing approach of Farrell and Maddison [19] to generate the connectivity information between interfaces but without creating a physical supermesh. Thus, most of the mesh operations at the interfaces are avoided saving computational efforts.

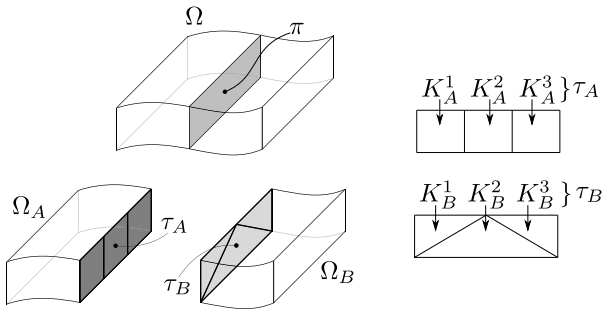

Fig. 1. A simple domain Ω which is divided by a plane π into two parts generating the interfaces τ_A and τ_B. The discretisation of these interfaces results in faces K_A and K_B respectively.

A general interpolation, which is demonstrated to be a sufficient condition to conserve any arbitrary quantity (mass, energy, linear momentum and others) at non-conformal interfaces, is described. This condition is applied to compute the interpolations for the advective and diffusive operators defining a conservative expression for general transport problems. The expression derived introduces a complex numerical coupling at the interfaces requiring flux-based weights and different interpolators. In this regard, a one-to-one connectivity scheme between the interfaces is proposed to implement that conservative coupling without requiring interpolations. The present method is based on the concept of supermesh where each pair of connected faces is represented by a unique one. In this sense, the definition of the fluxes is done identically for both interfaces, and consequently, the conservation is guaranteed. The same strategy is recently followed by Rinaldi et al. [21] who uses a supermesh as the base tool to make a conservative treatment of the fluxes. In contrast to the approach of Rinaldi et al., the proposal of this work does not require to determine the polygons resulting from the intersections. Instead of this, each face intersection is represented by a logical duplication of the original faces. In this sense, the original interfaces are converted in here called pseudo-supermeshes which have a one-to-one connectivity between them. An advantage of this methodology is that the trivial addressing between the interfaces is achieved avoiding to perform complex mesh operations. The area and the geometrical centroid of the pseudo-supermesh faces are computed with the local supermeshing approach without introducing new arbitrary polygons into the mesh. In this work, the replication of pseudo-supermesh faces is trivial since the new faces are born as a copy using the original nodes. Moreover, the new faces are added to the boundary mesh preserving the internal mesh structure unaltered. This new strategy is assessed with examples focused on the conservation of variables and the order of convergence of the error running on a parallel facility.

This work is organised as follows: this section presents an overview of different techniques developed for addressing grids with non-conformal interfaces. In Section 2, an introduction to the local supermeshing approach is presented followed by the description of a sufficient interpolation expression for general transport problems. In Section 3, the pseudo-supermesh strategy is described, and the parallelisation and computational efficiency of the method are discussed. Section 4 includes numerical tests to evaluate the new technique and finally, in Section 5, the paper closes with the main conclusions.

2. Theory

The aim of this section is to present a conservative interpolation expression at interfaces for a general transport equation. In the first place, the fundamentals of the local supermeshing approach are presented followed by the basis of conservative interfaces. Then, a sufficient condition to perform conservative interpolations is described. After that, the sufficient condition is applied on the advective and diffusive operators concluding with a final interpolation expression for a general transport problem.

2.1. Domain interfaces

An example of domain interfaces is depicted in Fig. 1, where an arbitrary domain Ω and an internal plane π are displayed. The plane divides the domain Ω into two subdomains, Ω_A and Ω_B. As shown in the same figure, the plane π can be discretised differently for each subdomain giving place to the surface meshes τ_A and τ_B composed by the faces $K_A^i \in \tau_A$ and $K_B^j \in \tau_B$ respectively. The super-indexes i and j identify each face in its respective interface and τ_A and τ_B are named as the source and target interfaces respectively. In this work, the geometry representations of these interfaces are equal. These geometries can be of an arbitrary shape being defined by the intersection surface between the subdomains Ω_A and Ω_B. For example, in Fig. 1, the interfaces share the plane π ($\Omega_A \cap \Omega_B = \pi$). In this sense, the concept of non-conformal interfaces employed in this paper refers to different discretisations of τ_A and τ_B.

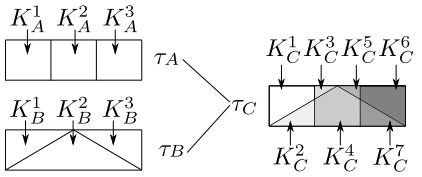

Fig. 2. Combination of interfaces τ_A and τ_B to generate the supermesh τ_C.

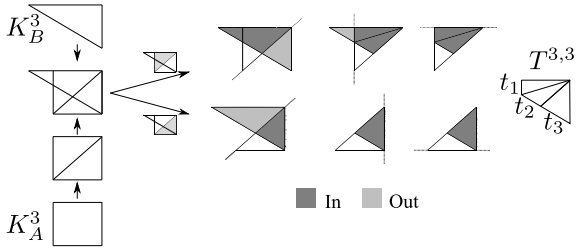

Fig. 3. Construction of a triangle mesh from the intersection of two faces.

2.2. The local supermeshing approach to handle non-conformal interface meshes

In the present work, conservative interpolations are developed to couple numerically non-conformal interfaces. These interpolations use weighting factors based on the overlapped area of each face-to-face intersection of the interface meshes. Therefore, it is necessary to determine the pairs of interface faces that intersect each other and subsequently, the overlapped area of the intersection. These procedures are achieved by the local supermeshing approach of Farrell and Maddison [19] which uses the concept of supermesh [18]. A supermesh is a geometrical conception created by the intersection of two meshes. In this context, the surface meshes τ_A and τ_B are combined to generate another surface mesh τ_C with faces $K_C^k \in \tau_C$ as described in Fig. 2 at right. In the local supermeshing approach, each face of the supermesh K_C^k is identified in turn to generate the addressing information between both interfaces. The intersection identification is performed with the advancing front algorithm which exploits the known connectivity information of the faces within the interfaces τ_A and τ_B to speed up the algorithm. After a pair of connected faces K_A^i and K_B^j are found, the intersection zone is decomposed into a set of triangles $t_n \in T^{i,j}$ which is a triangular mesh of the supermesh face K_C^k. The procedure triangulates each face and then, the intersection between the triangles of both faces are computed. Subsequently, the edges of one of the triangles clip the other generating a new polygon which is again decomposed into triangles. The algorithm finishes when the overlapped sector is fully triangulated. Fig. 3 explains graphically the construction of the set of triangles $T^{3,3}$ resulting from the intersection of the faces K_A^3 and K_B^3.

The connectivity information of the interfaces and the overlapped area for the pairs of intersected faces are stored in two lists which contain the fundamental data used in this work. Two connectivity lists are defined:

$$\begin{aligned} \chi_{AB}\left(K_A^i\right) &= \left\{K_B^j \in \tau_B : A\left(K_A^i \cap K_B^j\right) > 0\right\} \quad \forall \quad K_A^i \in \tau_A, \\ \chi_{BA}\left(K_B^j\right) &= \left\{K_A^i \in \tau_A : A\left(K_B^j \cap K_A^i\right) > 0\right\} \quad \forall \quad K_B^j \in \tau_B, \end{aligned} \tag{2.1}$$

where χ_{AB} and χ_{BA} are the connectivity (or addressing) lists for the source and target sides respectively, and $A\left(K_A^i \cap K_B^j\right)$ is the overlapped area of the faces K_A^i and K_B^j which is calculated as the summation of the area of each triangle belonging to the set $T^{i,j}$,

$$A\left(K_A^i \cap K_B^j\right) = \sum_{t_n \in T^{i,j}} A(t_n). \tag{2.2}$$

The overlapped fraction for each pair of connected faces are stored in the weighting lists ω_{AB} and ω_{BA},

$$\begin{aligned} \omega_{AB}\left(K_A^i\right) &= \left\{ w_B^{i,j} \in \mathcal{R} : w_B^{i,j} = \frac{A\left(K_A^i \cap K_B^j\right)}{A\left(K_A^i\right)} \quad \forall \quad K_B^j \in \chi_{AB}\left(K_A^i\right) \right\}, \\ \omega_{BA}\left(K_B^j\right) &= \left\{ w_A^{j,i} \in \mathcal{R} : w_A^{j,i} = \frac{A\left(K_A^i \cap K_B^j\right)}{A\left(K_B^j\right)} \quad \forall \quad K_A^i \in \chi_{BA}\left(K_B^j\right) \right\}, \end{aligned} \tag{2.3}$$

where $w_B^{i,j}$ and $w_A^{j,i}$ are the weighting factors based on the overlapped area of the faces K_A^i and K_B^j for the source and target interfaces respectively.

2.3. Conservative interfaces

The interfaces τ_A and τ_B shown in Fig. 1 represent the same spatial location in the domain Ω. They are located on the two-dimensional surface spanned by the plane π. In this sense, for any quantity ϕ defined in the domain Ω, its surface integrals at both interfaces must be equal,

$$\int_{\tau_A} \phi(\mathbf{x})\, dA = \int_{\tau_B} \phi(\mathbf{x})\, dA. \tag{2.4}$$

The fulfilment of Eq. (2.4) implies that the interfaces are globally conservative. In this work, the arbitrary quantity $\phi(\mathbf{x})$ is defined as the flux (per unit area) of some conservative variable which crosses the interface. The integrals over the interfaces τ_A and τ_B can be expressed as the summation of the integrals over their respective faces,

$$\sum_{K_A^i \in \tau_A} \int_{K_A^i} \phi(\mathbf{x})\, dA = \sum_{K_B^j \in \tau_B} \int_{K_B^j} \phi(\mathbf{x})\, dA. \tag{2.5}$$

Assuming a linear variation of the flux $\phi(\mathbf{x})$ over the face elements, the integrals of Eq. (2.5) can be calculated using values located at face centroids with a second order error approximation,

$$\sum_{K_A^i \in \tau_A} \phi(K_A^i)\, A(K_A^i) = \sum_{K_B^j \in \tau_B} \phi(K_B^j)\, A(K_B^j), \tag{2.6}$$

where the fluxes $\phi(K_A^i)$ and $\phi(K_B^j)$ are evaluated at face centroids.

2.4. Interface interpolation

The degrees of freedom of a cell-centred co-located Finite Volume Method (FVM) are located at cell centres. On the other hand, the transport of quantities between cells is achieved by face fluxes which require values located at face centroids. In a FVM with a compact stencil formulation, the flux at each internal face is calculated interpolating the values defined in its adjacent cells. In this work, these cells are referred to cell P and cell N. The computing of an internal face flux is given by,

$$\phi(K_{\text{int}}) = \alpha(K_{\text{int}})\, \psi_P(K_{\text{int}}) + \beta(K_{\text{int}})\, \psi_N(K_{\text{int}}) \qquad K_{\text{int}} \in \Omega, \tag{2.7}$$

where K_{int} is an arbitrary internal face, $\psi_P(K_{\text{int}})$ and $\psi_N(K_{\text{int}})$ are the adjacent cell values of any intensive quantity ψ in the cells P and N respectively, and $\alpha(K_{\text{int}})$ and $\beta(K_{\text{int}})$ are linear combination factors which depend on the differential operator and on the discretisation scheme. A face belonging to an interface, K_A^i or K_B^j, is not shared by two cells. It belongs only to the boundary adjacent cell (here defined as cell P). In order to compute the fluxes at the interface, the adjacent cells of the opposite interface are considered. As a consequence, the expression that defines the interface fluxes is,

$$\begin{aligned} \phi(K_A^i) &= f(K_A^i) = f\left[\psi_P(K_A^i), \psi_P(K_B^j)\right] \qquad \forall K_B^j \in \chi_{AB}(K_A^i), \\ \phi(K_B^j) &= g(K_B^j) = g\left[\psi_P(K_A^i), \psi_P(K_B^j)\right] \qquad \forall K_A^i \in \chi_{BA}(K_B^j), \end{aligned} \tag{2.8}$$

where $f(\cdot)$ and $g(\cdot)$ are the interpolation functions for the source and target interfaces respectively. Eq. (2.8) defines the face fluxes ϕ as a relation of boundary-adjacent cells values of ψ. The cells involved in the interpolation are those related by the intersection of faces given by the addressing lists χ_{AB} and χ_{BA}. Therefore, the algebraical coupling at the interface is reduced to local relations between cell values according to the physical communication between the subdomains. A graphical explanation of Eq. (2.8) which is based on the example case is shown in Fig. 4. The functions $f(\cdot)$ and $g(\cdot)$ must be designed properly to satisfy the global conservation statement of Eq. (2.4). Replacing Eq. (2.8) in Eq. (2.6), a conservative relation is defined for the functions $f(\cdot)$ and $g(\cdot)$,

$$\sum_{K_A^i \in \tau_A} f(K_A^i) A(K_A^i) = \sum_{K_B^j \in \tau_B} g(K_B^j) A(K_B^j). \tag{2.9}$$

It is important to highlight that the Eq. (2.9) proposes the global conservation of fluxes ϕ which are located at face centres of the interfaces. Here, the conservative variable ψ, which is the unknown of the numerical problem, is related indirectly via the functions $f(\cdot)$ and $g(\cdot)$. The Eq. (2.9) does not impose a new constraint over the numerical problem. Instead, it

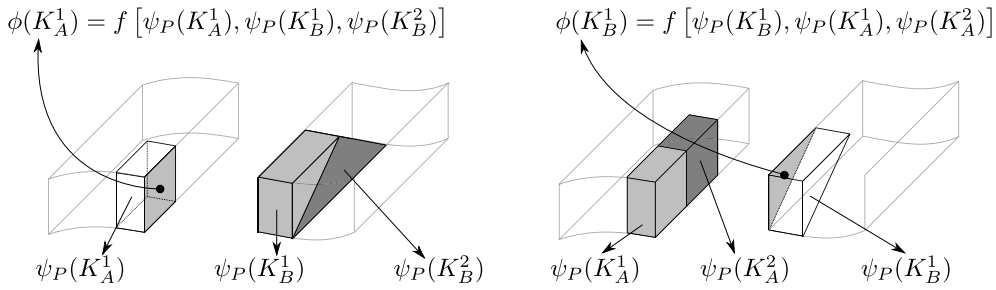

Fig. 4. Interface interpolation using boundary-adjacent cells. The cells involved in the interpolations are highlighted.

defines the matrix coefficients corresponding to each interface face that relates the adjacent cells of both interfaces in their respective conservation balances.

In what follows, the objective is to define the interpolation functions $f(\cdot)$ and $g(\cdot)$ to assure the conservation of the advective and diffusive fluxes of a general transport equation. A sufficient condition for a conservative interpolation of an arbitrary quantity ψ is defined:

Lemma 1. *The following interpolations of boundary-adjacent cell values $\psi_P(K_A^i)$ and $\psi_P(K_B^j)$ define face fluxes $\phi(K_A^i)$ and $\phi(K_B^j)$ which satisfy the Eq. (2.6),*

$$\begin{aligned}
\phi(K_A^i) = f(K_A^i) &= \sum_{w_B^{i,j} \in \omega_{AB}(K_A^i)} w_B^{i,j} \left[\alpha\left(K_A^i, K_B^j\right)\psi_P(K_A^i) + \beta\left(K_A^i, K_B^j\right)\psi_P(K_B^j)\right], \\
\phi(K_B^j) = g(K_B^j) &= \sum_{w_A^{j,i} \in \omega_{BA}(K_B^j)} w_A^{j,i} \left[\alpha\left(K_A^i, K_B^j\right)\psi_P(K_A^i) + \beta\left(K_A^i, K_B^j\right)\psi_P(K_B^j)\right],
\end{aligned} \tag{2.10}$$

where $\alpha\left(K_A^i, K_B^j\right)$ and $\beta\left(K_A^i, K_B^j\right)$ are factors corresponding to the pair of interface faces K_A^i and K_B^j which are based on the spatial discretisation scheme and the differential operator.

Physically, the result of the interpolation defined inside the brackets represents the flux ϕ at a supermesh face K_C^K which is referred here as $\phi(K_A^i, K_B^j)$,

$$\phi(K_A^i, K_B^j) = \alpha\left(K_A^i, K_B^j\right)\psi_P(K_A^i) + \beta\left(K_A^i, K_B^j\right)\psi_P(K_B^j). \tag{2.11}$$

In Eq. (2.11), the factors α and β interpolate the cell values ψ_P to the supermesh face and transform them into fluxes ϕ. In this sense, Lemma 1 proposes to define a set of supermesh fluxes which are merged into the interfaces τ_A and τ_B via an area-based interpolation. Replacing Eq. (2.11) in Eq. (2.10),

$$\begin{aligned}
\phi(K_A^i) = f(K_A^i) &= \sum_{w_B^{i,j} \in \omega_{AB}(K_A^i)} w_B^{i,j} \phi(K_A^i, K_B^j) \\
\phi(K_B^j) = g(K_B^j) &= \sum_{w_A^{j,i} \in \omega_{BA}(K_B^j)} w_A^{j,i} \phi(K_A^i, K_B^j).
\end{aligned} \tag{2.12}$$

Proof. First, the expression of Eq. (2.12) is combined with Eq. (2.6):

$$\begin{aligned}
&\sum_{K_A^i \in \tau_A} \phi(K_A^i)\, A(K_A^i) = \\
&\sum_{K_A^i \in \tau_A} \sum_{w_B^{i,j} \in \omega_{AB}(K_A^i)} w_B^{i,j} A(K_A^i)\, \phi(K_A^i, K_B^j).
\end{aligned} \tag{2.13}$$

Then, the definition of the weights $w_B^{i,j}$ given in Eq. (2.3) is replaced in Eq. (2.13),

$$\sum_{K_A^i \in \tau_A} \phi(K_A^i)\, A(K_A^i) = \sum_{K_A^i \in \tau_A} \sum_{K_B^j \in \chi_{AB}(K_A^i)} A(K_A^i \cap K_B^j)\, \phi(K_A^i, K_B^j). \tag{2.14}$$

The r.h.s. of Eq. (2.14) is a summation over all faces of the supermesh. The external summation sweeps all the faces of τ_A and for each face K_A, the internal summation sweeps all the faces K_B belonging to $\chi_{AB}(K_A^i)$. The summation over all supermesh faces can be done without altering the final result by changing the order of the summations as done in the r.h.s. of next expression,

$$\sum_{K_A^i \in \tau_A} \sum_{K_B^j \in \chi_{AB}(K_A^i)} (\cdot) = \sum_{K_B^j \in \tau_B} \sum_{K_A^i \in \chi_{BA}(K_B^j)} (\cdot), \tag{2.15}$$

where the external summation sweeps all the faces of τ_B and for each face K_B the internal summation sweeps all the faces K_A belonging to $\chi_{BA}(K_B^j)$. Then, the equivalence of Eq. (2.15) can be applied in Eq. (2.14),

$$\sum_{K_A^i \in \tau_A} \phi(K_A^i) A(K_A^i) = \sum_{K_B^j \in \tau_B} \sum_{K_A^i \in \chi_{BA}(K_B^j)} A(K_A^i \cap K_B^j) \phi(K_A^i, K_B^j). \tag{2.16}$$

Now, the second expression of Eq. (2.10) is combined with Eq. (2.6),

$$\sum_{K_B^j \in \tau_B} \phi(K_B^j) A(K_B^j) = \sum_{K_B^j \in \tau_B} \sum_{w_A^{j,i} \in \omega_{BA}(K_B^j)} w_A^{j,i} A(K_B^j) \phi(K_A^i, K_B^j). \tag{2.17}$$

Attending to the definition of the weights $w_A^{j,i}$ given in Eq. (2.3), last expression simplifies in,

$$\sum_{K_B^j \in \tau_B} \phi(K_B^j) A(K_B^j) = \sum_{K_B^j \in \tau_B} \sum_{K_A^i \in \chi_{BA}(K_B^j)} A(K_A^i \cap K_B^j) \phi(K_A^i, K_B^j). \tag{2.18}$$

Finally, combining Eq. (2.16) with Eq. (2.18) results in Eq. (2.6) which is the proof of Lemma 1,

$$\sum_{K_A^i \in \tau_A} \phi(K_A^i) A(K_A^i) = \sum_{K_B^j \in \tau_B} \phi(K_B^j) A(K_B^j). \quad \square \tag{2.19}$$

2.5. Advective interpolation

The quantity ϕ of Eq. (2.4) is now considered as the total advective flux of some scalar variable ψ transported through an interface τ:

$$\phi(\mathbf{x}) \equiv \Phi_c(\mathbf{x}) = \psi(\mathbf{x})\, [\mathbf{v}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x})] \qquad \mathbf{x} \in \tau, \tag{2.20}$$

where $\Phi_c(\mathbf{x})$ is the total advective flux (per unit area), $\mathbf{v}$ is the flow velocity and $\mathbf{n}$ is the unitary area vector of the interface τ. The global conservation of the advective flux on the interfaces must be satisfied. Here, the variable ϕ of Eq. (2.4) is replaced by the definition given in Eq. (2.20),

$$\int_{\tau_A} \psi(\mathbf{x})\, [\mathbf{v}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x})]\, dA = \int_{\tau_B} \psi(\mathbf{x})\, [\mathbf{v}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x})]\, dA. \tag{2.21}$$

Analogously as done with Eq. (2.4), the equality of Eq. (2.21) is discretised as,

$$\sum_{K_A^i \in \tau_A} \psi(K_A^i) \left[\mathbf{v}(K_A^i) \cdot \mathbf{A}(K_A^i)\right] = \sum_{K_B^j \in \tau_B} \psi(K_B^j) \left[\mathbf{v}(K_B^j) \cdot \mathbf{A}(K_B^j)\right], \tag{2.22}$$

where $\mathbf{A}(K)$ is the area vector of the face K. The term between brackets of last equation is referred as the carrier flux φ,

$$\varphi(K) = [\mathbf{v}(K) \cdot \mathbf{A}(K)]. \tag{2.23}$$

Introducing $\varphi(K)$ in Eq. (2.22), the discretised form of the advective flux at the interfaces is obtained,

$$\sum_{K_A^i \in \tau_A} \psi(K_A^i)\,\varphi(K_A^i) = \sum_{K_B^j \in \tau_B} \psi(K_B^j)\,\varphi(K_B^j). \tag{2.24}$$

The equality given in Eq. (2.24) is similar to that of Eq. (2.6) where the interface flux ϕ is considered here as the conservative variable ψ at the interface and the areas $A(K)$ are replaced with the fluxes $\varphi(K)$. The analogy of Eq. (2.6) with Eq. (2.24) is used to define the values of ψ at the interface via a redefinition of the weighting factors. In order to satisfy a conservative interpolation of ψ, the weighting factors defined in Eq. (2.3) must be redefined. In this sense, the weighting factors are defined as done in Eq. (2.3) considering the analogy between $\varphi(K)$ and $A(K)$,

$$\begin{aligned}
\tilde{\omega}_{AB}\left(K_A^i\right) &= \left\{ \tilde{w}_B^{i,j} \in \mathcal{R} : \tilde{w}_B^{i,j} = \frac{\varphi\left(K_A^i \cap K_B^j\right)}{\varphi\left(K_A^i\right)} \quad \forall \quad K_B^j \in \chi_{AB}\left(K_A^i\right) \right\}, \\
\tilde{\omega}_{BA}\left(K_B^j\right) &= \left\{ \tilde{w}_A^{j,i} \in \mathcal{R} : \tilde{w}_A^{j,i} = \frac{\varphi\left(K_A^i \cap K_B^j\right)}{\varphi\left(K_B^j\right)} \quad \forall \quad K_A^i \in \chi_{BA}\left(K_B^j\right) \right\},
\end{aligned} \tag{2.25}$$

where $\varphi\left(K_A^i \cap K_B^j\right)$ is the flux that enters or leaves the face K_A^i, and consequently, leaves or enters the face K_B^j respectively. Based on Lemma 1, the interpolation of the variable ψ at the interface is expressed as,

$$\begin{aligned}
\psi(K_A^i) &= \sum_{\tilde{w}_B^{i,j} \in \tilde{\omega}_{AB}(K_A^i)} \tilde{w}_B^{i,j} \left[\alpha_c\left(K_A^i, K_B^j\right) \psi_P(K_A^i) + \beta_c\left(K_A^i, K_B^j\right) \psi_P(K_B^j) \right], \\
\psi(K_B^j) &= \sum_{\tilde{w}_A^{j,i} \in \tilde{\omega}_{BA}(K_B^j)} \tilde{w}_A^{j,i} \left[\alpha_c\left(K_A^i, K_B^j\right) \psi_P(K_A^i) + \beta_c\left(K_A^i, K_B^j\right) \psi_P(K_B^j) \right],
\end{aligned} \tag{2.26}$$

where the interpolator factors α_c and β_c depend on the convection scheme used in the discretisation. The Eq. (2.26) yields to a global conservation of the total advective flux $\Phi_c(\mathbf{x})$ at the interfaces. It differs from the Eq. (2.10) in that it defines the interface value of ψ instead of the total advective flux Φ_c. The relation between these quantities is,

$$\Phi_c(K) = \frac{\psi(K)\varphi(K)}{A(K)}. \tag{2.27}$$

Note that the interpolation inside the brackets in Eq. (2.26) represents the supermesh value $\psi(K_A^i, K_B^j)$ and not the supermesh flux $\Phi_c(K_A^i, K_B^j)$. Therefore, the merging of conservative variables from the supermesh to the interfaces must be done with a flux-weighted interpolation instead of a overlapped area-based one as done in Eq. (2.10).

2.5.1. Advective flux

The total advective flux at an interface source face K_A^i is computed replacing the first expression of Eq. (2.26) in Eq. (2.20) and scaling it with the face area:

$$\begin{aligned}
\Phi(K_A^i)_c\, A(K_A^i) &= \psi(K_A^i)\,\varphi(K_A^i) \\
&= \varphi(K_A^i) \left\{ \sum_{\tilde{w}_B^{i,j} \in \tilde{\omega}_{AB}(K_A^i)} \tilde{w}_B^{i,j} \left[\alpha_c\left(K_A^i, K_B^j\right) \psi_P(K_A^i) + \beta_c\left(K_A^i, K_B^j\right) \psi_P(K_B^j) \right] \right\} \\
&= \varphi(K_A^i) \left[\sum_{\tilde{w}_B^{i,j} \in \tilde{\omega}_{AB}(K_A^i)} \tilde{w}_B^{i,j}\, \alpha_c\left(K_A^i, K_B^j\right) \right] \psi_P(K_A^i) \\
&\quad + \\
&\quad \varphi(K_A^i) \left[\sum_{\tilde{w}_B^{i,j} \in \tilde{\omega}_{AB}(K_A^i)} \tilde{w}_B^{i,j}\, \beta_c\left(K_A^i, K_B^j\right) \psi_P(K_B^j) \right].
\end{aligned} \tag{2.28}$$

This last equation is simplified introducing the coefficients $\mathcal{C}_A$, $\mathcal{C}_B$ and the advective interpolator $\mathcal{I}_\mathcal{C}$:

$$\begin{aligned}
\mathcal{C}_A(K_A^i) &= \varphi(K_A^i)\left[\sum_{\tilde{w}_B^{i,j}\in\tilde{\omega}_{AB}(K_A^i)} \tilde{w}_B^{i,j}\,\alpha_c\left(K_A^i, K_B^j\right)\right],\\
\mathcal{C}_B(K_A^i) &= \varphi(K_A^i),\\
\mathcal{I}_\mathcal{C}(K_A^i) &= \sum_{\tilde{w}_B^{i,j}\in\tilde{\omega}_{AB}(K_A^i)} \tilde{w}_B^{i,j}\,\beta_c\left(K_A^i, K_B^j\right)\psi_P(K_B^j).
\end{aligned} \tag{2.29}$$

Replacing the previous definitions in Eq. (2.28),

$$\Phi(K_A^i)_c\,A(K_A^i) = \mathcal{C}_A(K_A^i)\,\psi_P(K_A^i) + \mathcal{C}_B(K_A^i)\,\mathcal{I}_\mathcal{C}(K_A^i). \tag{2.30}$$

The Eq. (2.30) expresses the total advective flux of a source face K_A^i as a linear combination of the cell value $\psi_P(K_A^i)$ and the advective interpolator $\mathcal{I}_\mathcal{C}(K_A^i)$. A similar expression can be obtained for the advective flux at a target face K_B^j.

2.6. Diffusive interpolation

Now, the quantity ϕ of Eq. (2.4) is considered as the total diffusive flux of a scalar quantity ψ:

$$\phi(\mathbf{x}) \equiv \Phi_d(\mathbf{x}) = \gamma(\mathbf{x})\left[\nabla\psi(\mathbf{x})\cdot\mathbf{n}(\mathbf{x})\right], \tag{2.31}$$

where $\Phi_d(\mathbf{x})$ is the total diffusive flux per unit area, $\gamma(\mathbf{x})$ is the scalar diffusivity field and $\mathbf{n}(\mathbf{x})$ is the unitary area vector. The global conservation of the diffusive flux at the interfaces must be satisfied. Thus, the arbitrary quantity ϕ of Eq. (2.4) is replaced by the definition given in Eq. (2.31),

$$\int_{\tau_A} \gamma(\mathbf{x})\left[\nabla\psi(\mathbf{x})\cdot\mathbf{n}(\mathbf{x})\right]dA = \int_{\tau_B} \gamma(\mathbf{x})\left[\nabla\psi(\mathbf{x})\cdot\mathbf{n}(\mathbf{x})\right]dA. \tag{2.32}$$

The term between brackets of Eq. (2.32) is redefined as,

$$\left[\nabla\psi(\mathbf{x})\cdot\mathbf{n}(\mathbf{x})\right] \equiv \nabla\psi_n(\mathbf{x}), \tag{2.33}$$

where $\nabla\psi_n(\mathbf{x})$ is the derivative of ψ in the direction given by the surface normal. The introduction of Eq. (2.33) in Eq. (2.32) yields to the global conservation expression of the diffusive flux,

$$\int_{\tau_A} \gamma(\mathbf{x})\nabla\psi_n(\mathbf{x})\,dA = \int_{\tau_B} \gamma(\mathbf{x})\nabla\psi_n(\mathbf{x})\,dA. \tag{2.34}$$

The equality of Eq. (2.34) is discretised as done in Eq. (2.6),

$$\sum_{K_A^i\in\tau_A} \gamma(K_A^i)\nabla\psi_n(K_A^i)\,A(K_A^i) = \sum_{K_B^j\in\tau_B} \gamma(K_B^j)\nabla\psi_n(K_B^j)\,A(K_B^j). \tag{2.35}$$

Considering $\gamma(K)\nabla\psi_n(K)$ as the diffusive flux $\Phi_d(K)$ and using Lemma 1, the following interpolation expression is derived which guarantees the global conservation of the diffusive flux,

$$\begin{aligned}
\gamma(K_A^i)\nabla\psi_n(K_A^i) &= \sum_{w_B^{i,j}\in\omega_{AB}(K_A^i)} w_B^{i,j}\,\Phi_d(K_A^i, K_B^j) = \sum_{w_B^{i,j}\in\omega_{AB}(K_A^i)} w_B^{i,j}\left[\gamma(K_A^i, K_B^j)\nabla\psi_n(K_A^i, K_B^j)\right],\\
\gamma(K_B^j)\nabla\psi_n(K_B^j) &= \sum_{w_B^{i,j}\in\omega_{AB}(K_A^i)} w_B^{i,j}\,\Phi_d(K_A^i, K_B^j) = \sum_{w_A^{j,i}\in\omega_{BA}(K_B^j)} w_A^{j,i}\left[\gamma(K_A^i, K_B^j)\nabla\psi_n(K_A^i, K_B^j)\right],
\end{aligned} \tag{2.36}$$

where $\gamma(K_A^i, K_B^j)$ is the diffusivity value and $\nabla\psi_n(K_A^i, K_B^j)$ is the magnitude of the normal gradient at a supermesh face. It is important to clarify that $\gamma(K_A^i, K_B^j)$ and $\nabla\psi_n(K_A^i, K_B^j)$ are not computed as a linear combination of $\left[\gamma(K_A^i), \gamma(K_B^j)\right]$ and $\left[\nabla\psi_n(K_A^i), \nabla\psi_n(K_B^j)\right]$ respectively. These supermesh quantities will be defined in the next few paragraphs. The diffusive flux balances given in Eq. (2.36) must be expressed as a function of the variable ψ instead of gradients $\nabla\psi$. Therefore, the following equivalences are defined,

$$\begin{aligned}\nabla\psi_n(K_A^i, K_B^j) &= \frac{\psi_P(K_B^j) - \psi_P(K_A^i)}{\|d_n(K_A^i, K_B^j)\|},\\ \nabla\psi_n(K_A^i) &= \frac{\psi(K_A^i) - \psi_P(K_A^i)}{\|d_n(K_A^i)\|},\\ \nabla\psi_n(K_B^j) &= \frac{\psi(K_B^j) - \psi_P(K_B^j)}{\|d_n(K_B^j)\|},\end{aligned} \tag{2.37}$$

where $\mathbf{d_n}(K_A^i, K_B^j)$ is the vector that joins the centroids of the adjacent cells of the faces K_A^i and K_B^j, projected in the direction of the surface normal $\mathbf{n}$. Analogously, $\mathbf{d_n}(K_A^i)$ and $\mathbf{d_n}(K_B^j)$ are the vectors that join the centroids of the adjacent cells of the faces K_A^i and K_B^j with their centroids respectively, projected in the direction of the surface normal $\mathbf{n}$. Next, the expressions of Eq. (2.37) are introduced in Eq. (2.36),

$$\begin{aligned}\gamma(K_A^i)\left[\frac{\psi(K_A^i) - \psi_P(K_A^i)}{\|d_n(K_A^i)\|}\right] &= \sum_{w_B^{i,j} \in \omega_{AB}(K_A^i)} w_B^{i,j}\left\{\gamma(K_A^i, K_B^j)\left[\frac{\psi_P(K_B^j) - \psi_P(K_A^i)}{\|d_n(K_A^i, K_B^j)\|}\right]\right\},\\ \gamma(K_B^j)\left[\frac{\psi(K_B^j) - \psi_P(K_B^j)}{\|d_n(K_B^j)\|}\right] &= \sum_{w_A^{j,i} \in \omega_{BA}(K_B^j)} w_A^{j,i}\left\{\gamma(K_A^i, K_B^j)\left[\frac{\psi_P(K_B^j) - \psi_P(K_A^i)}{\|d_n(K_A^i, K_B^j)\|}\right]\right\}.\end{aligned} \tag{2.38}$$

Then, Eq. (2.38) is reordered to obtain the interface values of ψ,

$$\begin{aligned}\psi(K_A^i) &= \sum_{w_B^{i,j} \in \omega_{AB}(K_A^i)} w_B^{i,j}\left\{\frac{\gamma(K_A^i, K_B^j)}{\gamma(K_A^i)}\frac{\|d_n(K_A^i)\|}{\|d_n(K_A^i, K_B^j)\|}\left[\psi_P(K_B^j) - \psi_P(K_A^i)\right]\right\} + \psi_P(K_A^i),\\ \psi(K_B^j) &= \sum_{w_A^{j,i} \in \omega_{BA}(K_B^j)} w_A^{j,i}\left\{\frac{\gamma(K_A^i, K_B^j)}{\gamma(K_B^j)}\frac{\|d_n(K_B^j)\|}{\|d_n(K_A^i, K_B^j)\|}\left[\psi_P(K_B^j) - \psi_P(K_A^i)\right]\right\} + \psi_P(K_B^j).\end{aligned} \tag{2.39}$$

The interface values $\psi(K_A^i)$ and $\psi(K_B^j)$ are a function of the supermesh diffusivity $\gamma(K_A^i, K_B^j)$. The definition of this value is determined by adding to the system of Eq. (2.39) the equality of the surface integrals of ψ at the interface,

$$\int_{\tau_A} \psi(\mathbf{x})\, dA = \int_{\tau_B} \psi(\mathbf{x})\, dA, \tag{2.40}$$

which is discretised as,

$$\sum_{K_A^i \in \tau_A} \psi(K_A^i)\, A(K_A^i) = \sum_{K_B^j \in \tau_B} \psi(K_B^j)\, A(K_B^j). \tag{2.41}$$

Two new coefficients are defined,

$$\begin{aligned}\Gamma_A(K_A^i, K_B^j) &= \frac{\gamma(K_A^i, K_B^j)}{\gamma(K_A^i)}\frac{\|d_n(K_A^i)\|}{\|d_n(K_A^i, K_B^j)\|},\\ \Gamma_B(K_A^i, K_B^j) &= \frac{\gamma(K_A^i, K_B^j)}{\gamma(K_B^j)}\frac{\|d_n(K_B^j)\|}{\|d_n(K_A^i, K_B^j)\|}.\end{aligned} \tag{2.42}$$

Introducing the last definitions in Eq. (2.39) and reordering,

$$\begin{aligned}\psi(K_A^i) &= \sum_{w_B^{i,j} \in \omega_{AB}(K_A^i)} w_B^{i,j}\left\{\left[1 - \Gamma_A(K_A^i, K_B^j)\right]\psi_P(K_A^i) + \Gamma_A(K_A^i, K_B^j)\,\psi_P(K_B^j)\right\},\\ \psi(K_B^j) &= \sum_{w_A^{j,i} \in \omega_B(K_B^j)} w_A^{j,i}\left\{\Gamma_B(K_A^i, K_B^j)\,\psi_P(K_A^i) + \left[1 - \Gamma_B(K_A^i, K_B^j)\right]\psi_P(K_B^j)\right\}.\end{aligned} \tag{2.43}$$

Based on Lemma 1, Eq. (2.41) would be fulfilled if the Γ constants satisfy the following constraint,

$$\Gamma_A(K_A^i, K_B^j) = \left[1 - \Gamma_B(K_B^j, K_A^i)\right]. \tag{2.44}$$

Explicitly,

$$\frac{\gamma(K_A^i, K_B^j)}{\gamma(K_A^i)} \frac{\|d_n(K_A^i)\|}{\|d_n(K_A^i, K_B^j)\|} = 1 - \frac{\gamma(K_A^i, K_B^j)}{\gamma(K_B^j)} \frac{\|d_n(K_B^j)\|}{\|d_n(K_A^i, K_B^j)\|}. \tag{2.45}$$

From Eq. (2.45), a definition for the diffusivity $\gamma(K_A^i, K_B^j)$ is obtained,

$$\gamma(K_A^i, K_B^j) = \frac{\gamma(K_A^i)\,\gamma(K_B^j)\,\|d_N(K_A^i, K_B^j)\|}{\gamma(K_A^i)\,\|d_N(K_B^j)\| + \gamma(K_B^j)\,\|d_N(K_A^i)\|}. \tag{2.46}$$

Introducing the supermesh diffusivity value given in Eq. (2.46) in Eq. (2.43), an expression is obtained that assures the fulfilment of the two constraints of the diffusive operator given in Eq. (2.41) and Eq. (2.36).

2.6.1. Diffusive flux

The total diffusive flux at a source face K_A^i is computed replacing the first expression of Eq. (2.36) in Eq. (2.35),

$$\begin{aligned}
\Phi_d(K_A^i)\,A(K_A^i) &= \gamma(K_A^i)\,\nabla\psi_n(K_A^i)\,A(K_A^i) \\
&= A(K_A^i) \sum_{w_B^{i,j} \in \omega_{AB}(K_A^i)} w_B^{i,j} \left\{ \gamma(K_A^i, K_B^j) \left[\frac{\psi_P(K_B^j) - \psi_P(K_A^i)}{\|d_n(K_A^i, K_B^j)\|} \right] \right\} \\
&= A(K_A^i) \left[\sum_{w_B^{i,j} \in \omega_{AB}(K_A^i)} \frac{-w_B^{i,j}\,\gamma(K_A^i, K_B^j)}{\|d_n(K_A^i, K_B^j)\|} \right] \psi_P(K_A^i) \\
&\quad + A(K_A^i) \left[\sum_{w_B^{i,j} \in \omega_{AB}(K_A^i)} \frac{w_B^{i,j}\,\gamma(K_A^i, K_B^j)}{\|d_n(K_A^i, K_B^j)\|} \psi_P(K_B^j) \right].
\end{aligned} \tag{2.47}$$

The coefficients $\mathcal{D}_A(K_A^i)$, $\mathcal{D}_B(K_A^i)$ and the diffusive interpolator $\mathcal{I}_{\mathcal{D}}(K_A^i)$ are defined,

$$\begin{aligned}
\mathcal{D}_A(K_A^i) &= -A(K_A^i) \sum_{w_B^{i,j} \in \omega_{AB}(K_A^i)} \frac{w_B^{i,j}\,\gamma(K_A^i, K_B^j)}{\|d_n(K_A^i, K_B^j)\|}, \\
\mathcal{D}_B(K_A^i) &= A(K_A^i), \\
\mathcal{I}_{\mathcal{D}}(K_A^i) &= \sum_{w_B^{i,j} \in \omega_{AB}(K_A^i)} \frac{w_B^{i,j}\,\gamma(K_A^i, K_B^j)}{\|d_n(K_A^i, K_B^j)\|} \psi_P(K_B^j).
\end{aligned} \tag{2.48}$$

Replacing Eq. (2.48) in Eq. (2.47):

$$\Phi_d(K_A^i)\,A(K_A^i) = \mathcal{D}_A(K_A^i)\,\psi_P(K_A^i) + \mathcal{D}_B(K_A^i)\,\mathcal{I}_{\mathcal{D}}(K_A^i). \tag{2.49}$$

Eq. (2.49) expresses the total diffusive flux at a source face K_A^i in terms of its adjacent cell value $\psi_P(K_A^i)$ and the diffusive interpolator $\mathcal{I}_{\mathcal{D}}$. An analogous expression can be obtained for a target face K_B^j.

2.7. Total flux crossing the interface

The sum of the advective and diffusive fluxes which are defined in Eq. (2.30) and Eq. (2.49) respectively, is the total flux that crosses the face K_A^i in a general transport problem:

$$\left[\Phi_c(K_A^i) + \Phi_d(K_A^i)\right] A(K_A^i) = \left[\mathcal{C}_A(K_A^i) + \mathcal{D}_A(K_A^i)\right] \psi_P(K_A^i) + \mathcal{C}_B(K_A^i)\mathcal{I}_{\mathcal{C}}(K_A^i) + \mathcal{D}_B(K_A^i)\mathcal{I}_{\mathcal{D}}(K_A^i). \tag{2.50}$$

The expression given in Eq. (2.50) has some features that are challenging to implement in a Finite Volume code. The principal difficulties are,

- The local supermeshing approach does not generate a physical supermesh. Therefore, additional containers and functions must be implemented to define the supermesh values. For example, the supermesh flux $\varphi(K_A^i, K_B^j)$ or the supermesh diffusivity $\gamma(K_A^i, K_B^j)$.
- The partial contribution of cells located on the opposite side to the total interface flux requires the use of different interpolators for each differential operator of the transport equation. This forces one to use a sophisticated boundary condition which may be difficulty to implement in CFD codes based on modular programming.

In order to implement the interpolation expression given in Eq. (2.50) and to avoid the implementation difficulties described above, a simplified supermesh is proposed in the next section of this paper. On the other hand, in pure advective or diffusive problems, only one interpolator is required and then, the implementation of the interface expressions becomes simpler. For example, the Eq. (2.49) can be used to couple fluid and solid domains in conjugate heat transfer problems.

3. The pseudo-supermesh approach

In the first part of this section, the supermesh strategy is defined and its advantages are described concerning the implementation of Eq. (2.50). Secondly, a simplified version of a supermesh is presented being named pseudo-supermesh. After that, the advective and diffusive interpolations are redefined considering the new approach.

3.1. Supermesh definition

The combination of the interfaces τ_A and τ_B can be represented with a unique interface τ_C named "supermesh". An example of a supermesh is shown in Fig. 2. Here, a formal definition is presented as done in the work of Farrel *et* al. [18]: an element of the supermesh $K_C^k \in \tau_C$ is defined with the following statements,

- $\mathcal{N}_C \supseteq \mathcal{N}_A \cup \mathcal{N}_B$
- $A(K_C^k \cap K) \in \left\{0,\ A(K_C^k)\right\} \forall\, K_C^k \in \tau_C,\ K \in \tau,\ \tau \in \{\tau_A,\ \tau_B\}$

where $\mathcal{N}_C$, $\mathcal{N}_A$ and $\mathcal{N}_B$ are the sets of nodes of the interfaces τ_C, τ_A and τ_B respectively. The supermesh τ_C includes all the nodes of the original interfaces τ_A and τ_B, and each face of the supermesh K_C^k is completely included within a face of the original interfaces. The construction of a supermesh, solves the obstacles found in the implementation of Eq. (2.50):

- Each face of the supermesh is a container for the values corresponding to each pair of connected interface faces. For example,

$$\varphi\left(K_A^i, K_B^j\right) \rightarrow \varphi\left(K_C^k\right). \tag{3.1}$$

- Each face of the supermesh links only two cells belonging to each subdomain respectively. Therefore, the supermesh face is considered as an internal face which is shared by two cells. Thus, the interpolation of Eq. (2.7) is applied straightforward. As a consequence, the interpolators required for the advective and diffusive fluxes given in Eq. (2.50) are avoided:

$$\mathcal{C}_B(K_A^i)\mathcal{I}_{\mathcal{C}}(K_A^i) + \mathcal{D}_B(K_A^i)\mathcal{I}_{\mathcal{D}}(K_A^i) \rightarrow \left[\mathcal{C}_B(K_C^k) + \mathcal{D}_B(K_C^k)\right]\psi_P\left(K_C^k\right). \tag{3.2}$$

The definition of a supermesh allows performing conservative interpolations without the implementation barriers described before. The key concept of the supermesh is that it generates a one-to-one addressing between the interface faces. Therefore, the telescopic property of the FVM can be applied on the interface. The flux that leaves a supermesh face from one side of the interface is equal to the flux that enters into the other side. It is important to remark that the topology of a supermesh face K_C^K is not necessary in a second order cell-centred collocated FVM, where an extensive quantity on a face is a function of its area and its centroid. These geometrical features can be directly computed from the triangles belonging to the set $T^{i,j}$ defined by the local supermeshing approach without determining the arbitrary polygon of the intersection. In this sense, this work presents a new supermesh conception. The objective of the new proposal is to perform a flux-conservative treatment of arbitrary non-conformal interfaces. The present technique employs the local supermeshing approach to determine the interface addressing, and the area and the geometrical centroid of the overlapped sector defined by each pair of connected faces.

3.2. Definition of the pseudo-supermesh

The new strategy is based on redefining the number of faces of the source and target interfaces τ_A and τ_B, respectively, with the aim of constructing a new pair of interfaces with a one-to-one connectivity. In this sense, each face $K_A^i \in \tau_A$ ($K_B^j \in \tau_B$) is duplicated "n" times according to the connectivity with the opposite interface given by the addressing set

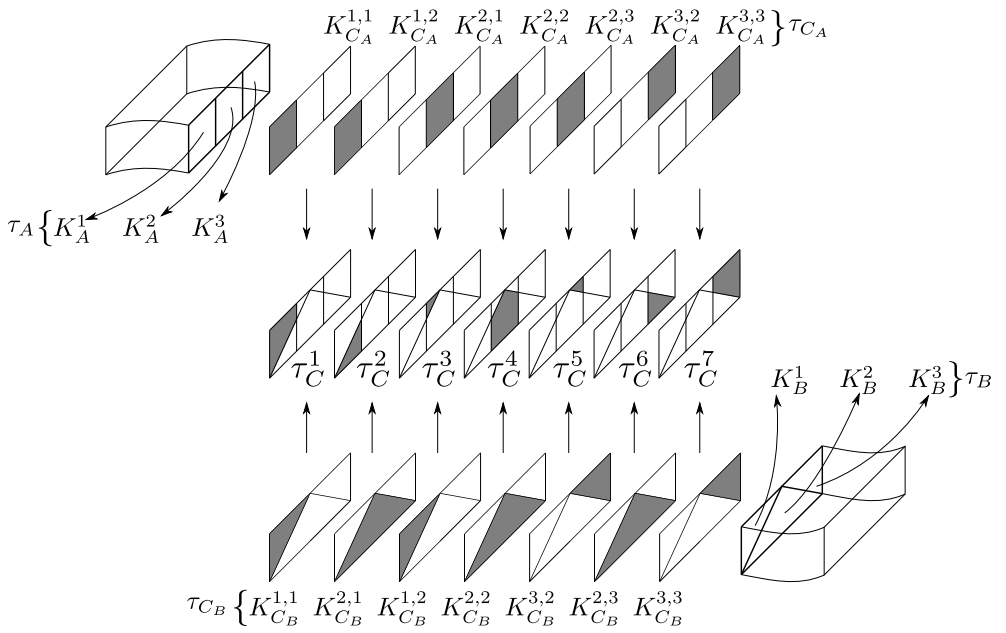

Fig. 5. Example of pseudo-supermesh interfaces. The original interface faces are multiplied to define a one-to-one connectivity.

$\chi_{AB}(\chi_{BA})$. The new faces share the points with their parent faces $K_A^i(K_B^j)$ so new points are not required. The new interfaces are named τ_{C_A} and τ_{C_B} for the source and target side respectively, and they are referred as pseudo-supermesh interfaces. The definition of the pseudo-supermeshes is:

Source side: the pseudo-supermesh τ_{C_A} of τ_A with faces $K_{C_A}^{i,j}$ is a mesh of $\Omega \in \mathcal{R}^2$ such that:

- $\mathcal{N}_{C_A} = \mathcal{N}_A$
- $\forall\, K_A^i \in \tau_A \quad \wedge \quad \forall\, K_B^j \in \chi_{AB}(K_A^i) \quad \exists!\, K_{C_A}^{i,j} \in \tau_{C_A} : A(K_{C_A}^{i,j}) = A(K_A^i \cap K_B^j) \quad \wedge \quad \mathcal{N}_{K_{C_A}^{i,j}} = \mathcal{N}_{K_A^i}$

Target side: the pseudo-supermesh τ_{C_B} of τ_B with faces $K_{C_B}^{j,i}$ is a mesh of $\Omega \in \mathcal{R}^2$ such that:

- $\mathcal{N}_{C_B} = \mathcal{N}_B$
- $\forall\, K_B^j \in \tau_B \quad \wedge \quad \forall\, K_A^i \in \chi_{BA}(K_B^j) \quad \exists!\, K_{C_B}^{j,i} \in \tau_{C_B} : A(K_{C_B}^{j,i}) = A(K_A^i \cap K_B^j) \quad \wedge \quad \mathcal{N}_{K_{C_B}^{j,i}} = \mathcal{N}_{K_B^j}$

where $\mathcal{N}_K$ is the set of nodes of the face K. The first assertion expresses that the nodes of each pseudo-supermesh interface are the same of its parent mesh respectively. The second statement affirms that for each pair of connected faces of the original interfaces exists one and only one face in the pseudo-supermesh. The supermesh face area matches to the overlapped area of the original ones and the pseudo-supermesh face has the same nodes as its parent face. A graphical example of the pseudo-supermeshes is shown in Fig. 5. For each face of a parent interface (τ_A or τ_B) there are "n" identical faces in its child interface (τ_{C_A} or τ_{C_B}) respectively. The number "n" is equal to the number of faces from the opposite side overlapping the parent face. As for each connection between the parent interfaces τ_A and τ_B exists a unique face in each of the pseudo-supermesh interfaces, the addressing between τ_{C_A} and τ_{C_B} is one-to-one. In this sense, the addressing set $\chi_{C_AC_B}$ contains for each source face $K_{C_A}^{i,j}$ in τ_{C_A} the corresponding face $K_{C_B}^{j,i}$ in the target interface τ_{C_B},

$$\chi_{C_AC_B}(K_{C_A}^{i,j}) = K_{C_B}^{j,i} \in \tau_{C_B} \qquad \forall K_{C_A}^{i,j} \in \tau_{C_A}, \tag{3.3}$$

similarly,

$$\chi_{C_BC_A}(K_{C_B}^{j,i}) = K_{C_A}^{i,j} \in \tau_{C_A} \qquad \forall K_{C_B}^{j,i} \in \tau_{C_B}. \tag{3.4}$$

According to the definition of the pseudo-supermesh interfaces, the addressing between the parent and child faces can be directly defined. In this context, the addressing χ_{AC_A} has for each parent face K_A^i a set containing the child faces $K_{C_A}^{i,j}$,

$$\chi_{AC_A}(K_A^i) = \left\{ K_{C_A}^{i,j} \in \tau_{C_A} : K_B^j \in \chi_{AB}(K_A^i) \right\} \qquad \forall K_A^i \in \tau_A, \tag{3.5}$$

$$\chi_{BC_B}(K_B^j) = \left\{ K_{C_B}^{j,i} \in \tau_{C_B} : K_A^i \in \chi_{BA}(K_B^j) \right\} \qquad \forall K_B^j \in \tau_B. \tag{3.6}$$

3.2.1. Area of the pseudo-supermesh faces

The initial geometrical area of each pseudo-supermesh face is equal to the area of its respective parent face since it is born as a copy. These areas must be redefined to accomplish the real intersection between the pair of source and target faces. The modified area of each pseudo-supermesh face is calculated by scaling the area of its parent face with the weight of the intersection,

$$\begin{aligned} A\left(K_{C_A}^{i,j}\right) &= w_B^{i,j} A\left(K_A^i\right) \qquad \forall K_{C_A}^{i,j} \in \tau_{C_A}, \\ A\left(K_{C_B}^{j,i}\right) &= w_A^{j,i} A\left(K_B^j\right) \qquad \forall K_{C_B}^{j,i} \in \tau_{C_B}, \end{aligned} \tag{3.7}$$

which satisfy,

$$\begin{aligned} A\left(K_A^i\right) &= \sum_{K_{C_A}^{i,j} \in \chi_{AC_A}(K_A^i)} A\left(K_{C_A}^{i,j}\right), \\ A\left(K_B^j\right) &= \sum_{K_{C_B}^{j,i} \in \chi_{BC_B}(K_B^j)} A\left(K_{C_B}^{j,i}\right), \end{aligned} \tag{3.8}$$

and,

$$A\left(K_{C_A}^{i,j}\right) = A\left(K_{C_B}^{j,i}\right). \tag{3.9}$$

3.2.2. Centroid of the pseudo-supermesh faces

The pseudo-supermesh faces are geometrically equal to their parent ones, and consequently, they share the same face centroid. In other words, the pair of pseudo-supermesh faces $K_{C_A}^{i,j}$ and $K_{C_B}^{j,i}$ inherit the centroids of the faces K_A^i and K_B^j respectively. The original centroids may be different to that one of the overlapped sector resulting from the intersection. Some numerical schemes employ the position of the face centroid to define the face values. An example is the second order upwind scheme [22,23] which computes a value at a face K as,

$$\phi(K) = \phi^u(K) + \nabla\phi^u(K) \cdot \left[\mathbf{x}_c(K) - \mathbf{x}_c^u(K)\right], \tag{3.10}$$

where $\phi^u(K)$ and $\nabla\phi^u(K)$ are the value and the gradient of the upwind cell of the face K respectively, $\mathbf{x}_c(K)$ is the centroid of K and $\mathbf{x}_c^u(K)$ is the centroid of the upwind cell. If the face centroid is not correctly defined, an error is computed in the discretisation. In this context, the centroids of the pseudo-supermesh faces are redefined to avoid errors in centroid-dependent numerical schemes. The pseudo-supermesh centroids are calculated according to the geometrical centroid of the intersection,

$$\mathbf{x}_c(K_{C_A}^{i,j}) = \mathbf{x}_c(K_{C_B}^{j,i}) = \mathbf{x}_c(K_A^i \cap K_B^j) \tag{3.11}$$

The local supermeshing approach generates for each pair of the intersected faces $(K_A^i \cap K_B^j)$ a set of triangles $t_n \in T^{i,j}$. Therefore, the centroid of the intersection zone is,

$$\mathbf{x}_c(K_A^i \cap K_B^j) = \frac{\sum_{t_n \in T^{i,j}} \mathbf{x}_c(t_n)\, A(t_n)}{\sum_{t_n \in T^{i,j}} A(t_n)} \tag{3.12}$$

where the centroid of each triangle $\mathbf{x}_c(t_n)$ is the average of its points coordinates.

3.3. Interpolation with the pseudo-supermesh approach

The pseudo-supermesh interfaces have a one-to-one connectivity which guarantees an identical definition of the interface fluxes at both interfaces,

$$\phi(K_{C_A}^{i,j}) = \phi(K_{C_B}^{j,i}). \tag{3.13}$$

Similarly as defined in Eq. (2.8), the fluxes are obtained by interpolation of adjacent cell values. In this case, there is only one cell on the opposite side due to the trivial addressing,

$$\begin{aligned} \phi\left(K_{C_A}^{i,j}\right) &= \alpha\left(K_{C_A}^{i,j}\right) \psi_P(K_{C_A}^{i,j}) + \beta\left(K_{C_A}^{i,j}\right) \psi_N(K_{C_A}^{i,j}), \\ \phi\left(K_{C_B}^{j,i}\right) &= \alpha\left(K_{C_B}^{j,i}\right) \psi_N(K_{C_B}^{j,i}) + \beta\left(K_{C_B}^{j,i}\right) \psi_P(K_{C_B}^{j,i}), \end{aligned} \tag{3.14}$$

where the cell P of each pseudo-supermesh face is the same as its respective parent face,

$$\begin{aligned}\phi_P\left(K_{C_A}^{i,j}\right) &= \phi_P\left(K_A^i\right) \qquad \forall K_{C_A}^{i,j} \in \chi_{AC_A}\left(K_A^i\right),\\ \phi_P\left(K_{C_B}^{j,i}\right) &= \phi_P\left(K_B^j\right) \qquad \forall K_{C_B}^{j,i} \in \chi_{BC_B}\left(K_B^j\right),\end{aligned} \tag{3.15}$$

and the cell N of each pseudo supermesh face is the cell P of the of the opposite face determined with the one-to-one addressing,

$$\begin{aligned}\phi_N\left(K_{C_A}^{i,j}\right) &= \phi_P\left(K_{C_B}^{j,i}\right) \qquad K_{C_B}^{j,i} \in \chi_{C_AC_B}\left(K_A^{i,j}\right),\\ \phi_N\left(K_{C_B}^{j,i}\right) &= \phi_P\left(K_{C_A}^{i,j}\right) \qquad K_{C_A}^{i,j} \in \chi_{C_BC_A}\left(K_B^{j,i}\right).\end{aligned} \tag{3.16}$$

3.3.1. Total flux crossing the pseudo-supermesh interfaces

The total flux contribution described in Eq. (2.50) is expressed here for a source pseudo-supermesh face $K_{C_A}^{i,j}$,

$$\left[\Phi_c(K_{C_A}^{i,j}) + \Phi_d(K_{C_A}^{i,j})\right] A(K_{C_A}^{i,j}) = \left[\mathcal{C}_A(K_{C_A}^{i,j}) + \mathcal{D}_A(K_{C_A}^{i,j})\right] \psi_P(K_{C_A}^{i,j}) + \left[\mathcal{C}_B(K_{C_A}^{i,j}) + \mathcal{D}_B(K_{C_A}^{i,j})\right] \psi_N(K_{C_A}^{i,j}), \tag{3.17}$$

where the coefficients $\mathcal{C}_A$,$\mathcal{C}_B$ and $\mathcal{D}_A$,$\mathcal{D}_B$ are defined analogously to the definition given in Eq. (2.29) and Eq. (2.48) respectively for the advective and diffusive terms,

$$\begin{aligned}\mathcal{C}_A(K_{C_A}^{i,j}) &= \varphi(K_{C_A}^{i,j})\,\alpha_c(K_{C_A}^{i,j}),\\ \mathcal{C}_B(K_{C_A}^{i,j}) &= \varphi(K_{C_A}^{i,j})\,\beta_c(K_{C_A}^{i,j}),\\ \mathcal{D}_A(K_{C_A}^{i,j}) &= -A(K_{C_A}^{i,j})\,\frac{\gamma(K_{C_A}^{i,j})}{\|d_n(K_{C_A}^{i,j})\|},\\ \mathcal{D}_B(K_{C_A}^{i,j}) &= A(K_{C_A}^{i,j})\,\frac{\gamma(K_{C_A}^{i,j})}{\|d_n(K_{C_A}^{i,j})\|}.\end{aligned} \tag{3.18}$$

The coefficients $\mathcal{C}$ and $\mathcal{D}$ are grouped in two factors collecting the values from both sides of the interface,

$$\begin{aligned}P(K_{C_A}^{i,j}) &= \mathcal{C}_A(K_{C_A}^{i,j}) + \mathcal{D}_A(K_{C_A}^{i,j}),\\ N(K_{C_A}^{i,j}) &= \mathcal{C}_B(K_{C_A}^{i,j}) + \mathcal{D}_B(K_{C_A}^{i,j}).\end{aligned} \tag{3.19}$$

Replacing the new factors of Eq. (3.19) in Eq. (3.17),

$$\left[\Phi_c(K_{C_A}^{i,j}) + \Phi_d(K_{C_A}^{i,j})\right] A(K_{C_A}^{i,j}) = P(K_{C_A}^{i,j})\,\psi_P(K_{C_A}^{i,j}) + N(K_{C_A}^{i,j})\,\psi_N(K_{C_A}^{i,j}). \tag{3.20}$$

Eq. (3.20) expresses the total contribution of the advective and diffusive fluxes at a pseudo-supermesh face $K_{C_A}^{i,j}$. The expression is analogous to the internal face interpolation given in Eq. (2.7) where the interface interpolators $\mathcal{I}_\mathcal{C}$ and $\mathcal{I}_\mathcal{D}$ of Eq. (2.50) are avoided. The numerical coupling at the interface is achieved with a one-to-one a addressing using the maps $\chi_{C_AC_B}$ and $\chi_{C_BC_A}$.

3.4. Algorithm

In the following algorithm, all the steps required to construct the pseudo-supermeshes are detailed. The algorithm is conceived to solve a dynamic mesh problem where the pseudo-supermeshes are removed and regenerated for every time-step of the simulation.

1. The local supermeshing approach is performed on the original interfaces τ_A and τ_B to generate the addressing and weighting lists χ_{AB}, χ_{BA} and ω_{AB}, ω_{BA} respectively. The supermesh centroids are also computed.
2. According to the information given by the lists χ_{AB} and χ_{BA}, the new faces K_{C_A} and K_{C_B} are generated and consequently the new pseudo-supermesh interfaces τ_{C_A} and τ_{C_B} are created.
3. The new addressing lists $\chi_{C_AC_B}$, $\chi_{C_BC_A}$ and χ_{AC_A}, χ_{BC_B} are defined.
4. The geometrical areas of the faces K_{C_A} and K_{C_B} are scaled and the pseudo-supermesh centroids are redefined.
5. The numerical problem is solved.
6. The pseudo-supermeshes are removed and thus, the original interfaces τ_A and τ_B are recovered.
7. The dynamic mesh action is achieved generating a new state of the interfaces, τ'_A and τ'_B.
8. The algorithm is restarted from step 1 with the new interfaces ($\tau_A \to \tau'_A$ and $\tau_B \to \tau'_B$).

The algorithm is explained schematically in Fig. 6.

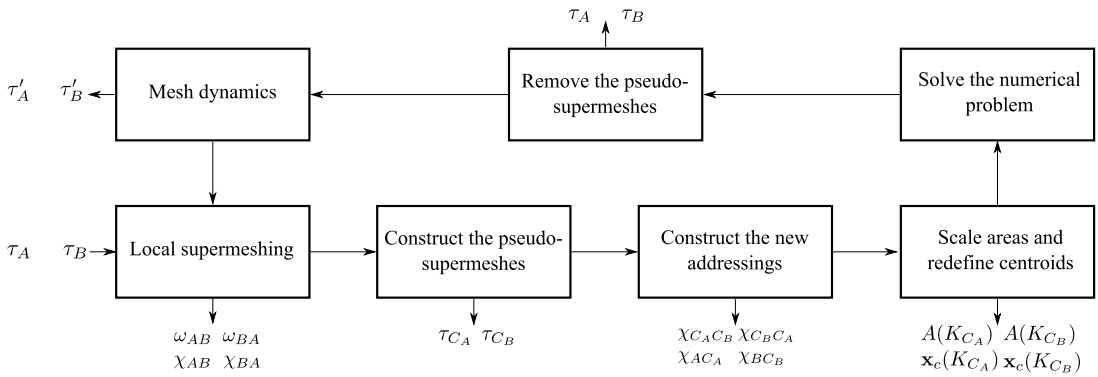

Fig. 6. Different steps performed in a dynamic mesh simulation when using the pseudo-supermesh strategy.

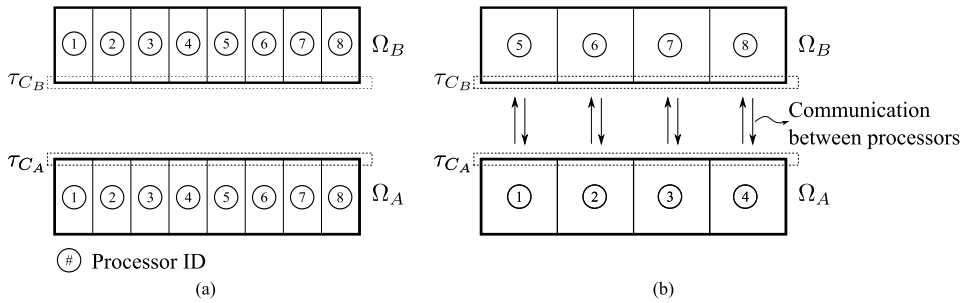

Fig. 7. Two examples of domain decomposition. Both cases are decomposed in eight processors but with different partitioning schemes; (a) partitioning scheme with minimum inter-processor communication; (b) partitioning scheme with maximum inter-processor communication.

3.5. The pseudo-supermesh approach in parallel

The pseudo-supermesh strategy is implemented in parallel using a distributed memory architecture. There is no restriction in the domain decomposition procedure. The source and target inter-connected faces may lay on the same processor or not, and each pseudo-supermesh can be distributed over multiple processors. It is worth mentioning that for turbomachinery applications, the possibility of positioning the source and target interfaces in different processors is crucial since the rotating motion generates a variable connectivity between the processors (subdomains). Furthermore, for interfaces with many faces, if the strategy is not parallelised, the construction of the pseudo-supermesh by only one processor will induce a load imbalance in the whole simulation.

In Fig. 7, two examples are presented where the domain is partitioned over eight processors with different schemes for each case respectively. In example (a), the pseudo-supermesh interfaces are divided into eight parts with each processor owning the target and source interfaces. In this case, the pseudo-supermesh construction and the flux assembling are performed without communication between processors. On the other hand, in example (b), the pseudo-supermesh interfaces are divided into four parts where each processor owns either the source or the target interface. Therefore, inter-processor communication is required to perform the local supermeshing, to construct the pseudo-supermeshes and to assemble the fluxes. Both examples are limit situations, a practical case would be a combination of both.

A test is performed to evaluate the scalability of the parallel implementation. The time required to remove and reconstruct the pseudo-supermeshes is evaluated using the two partitioning schemes shown in Fig. 7. The resulting CPU time is plotted against the number of processors in Fig. 8. The results show a very good efficiency of the parallel implementation.

3.5.1. Implementation

A brief description of the parallel implementation of the pseudo-supermesh approach is detailed by the following steps:

1. Each processor removes the pseudo-supermesh faces recovering the original interfaces (pseudo-supermesh removal).
2. Each processor receives from the others those target faces which are inside a bounding box determined by its source faces (target faces communication).
3. Processors with source faces (and target faces) perform the local supermeshing defining the local addressing and weighting lists. Processors without source faces are idle in this step (local supermeshing).
4. Local addressing and weighting lists are sent to all processors. Inter-processor maps are created for future communication (local supermesh communication).

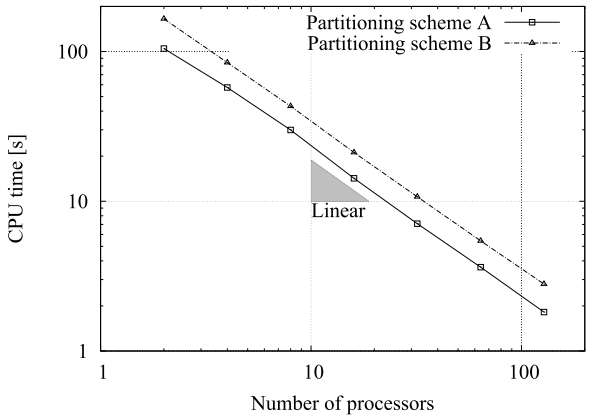

Fig. 8. Evaluation of the pseudo-supermesh parallel implementation. The CPU time of two partitioning schemes with minimum (scheme A) and maximum (scheme B) inter-processor communication are plotted varying the number of processors. The pseudo-supermesh interfaces used in the tests are composed of more than 6 million faces. The test are executed in a cluster with processors Intel(R) Xeon(R) E5-1620 v2 connected with Infiniband QDR.

Table 1
Relative time consumed by each step of the parallel implementation using the partitioning scheme A with two processors.

Steps	Relative time [%]
(1) Pseudo-supermesh removal	24.9
(2) Target faces communication	2.6
(3) Local supermeshing	39.9
(4) Local supermesh communication	2.1
(5) Pseudo-supermesh construction	24.5
(6) Maps communication	1.3
(7) Pseudo-supermesh connectivity	1.4
(8) Pseudo-supermesh communication	3.3

5. Each processor creates and introduces into the mesh its new pseudo-supermesh faces. Maps between new and old faces are computed (pseudo-supermesh construction).
6. Local face-maps are sent to all processors (maps communication).
7. Each processor generates the new local addressing lists by relating the old connectivity with the face-maps (pseudo-supermesh connectivity).
8. Local addressing and weighting lists of the new pseudo-supermesh interfaces are sent to all processors. New inter-processor maps are created for future communication (pseudo-supermesh communication).

Steps 1, 3, 5 and 7 are parallelised trivially without requiring inter-processor communication. On the contrary, steps 2, 4, 6 and 8 involve communication and are not required when running in serial mode. However, the total time consumed by these steps is of a minor order of magnitude as shown in Table 1 which details the relative time consumed by each step of the parallel implementation. From Table 1 it is concluded that the steps 1, 3, 5 and 7 consume the 90.4 percent of the total time. An important detail is that the local supermeshing (step 3) is achieved only by the processors who own source faces. Therefore, the source side of the interface should be distributed over all processors to optimally balance the algorithm. The incidence of this point is clearly appreciated in Fig. 8 where the partitioning scheme A has a better performance than the scheme B.

3.6. Computational efficiency

This test evaluates the computational cost of the proposed algorithm as a function of the total number of original interface faces. Two different tests are introduced. The first one consists in measuring the CPU time required to couple and decouple two static and planar interfaces which are meshed with a non-conformal mesh. On the other hand, the second test achieves a dynamic mesh simulation using a rotor–stator configuration with a cylindrical interface. Here, the CPU time required to complete a whole revolution of the rotor is analysed using 10 time-steps. At each time-step, the interfaces are coupled and decoupled. A graphical description of the tests and results are shown in Fig. 9 and Fig. 10 respectively. As a reference, an implementation of the Sliding Interfaces technique [1–3] is also evaluated.[1] The pseudo-supermesh performance

[1] The denominations “sliding interfaces” and “sliding meshes” are commonly used to describe the general framework of these strategies. Here, “Sliding Interfaces” is the name of a particular technique within this group of methodologies.

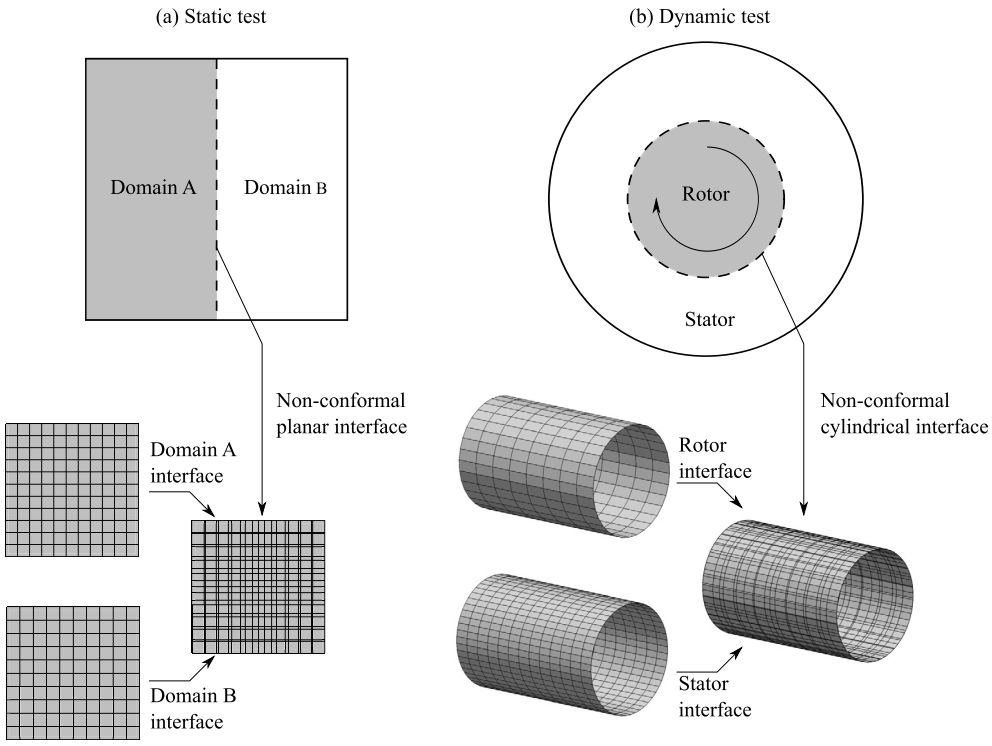

Fig. 9. Description of the examples solved for the computational efficiency tests: (a) static test with non-conformal planar interfaces; (b) dynamic test with non-conformal cylindrical interfaces.

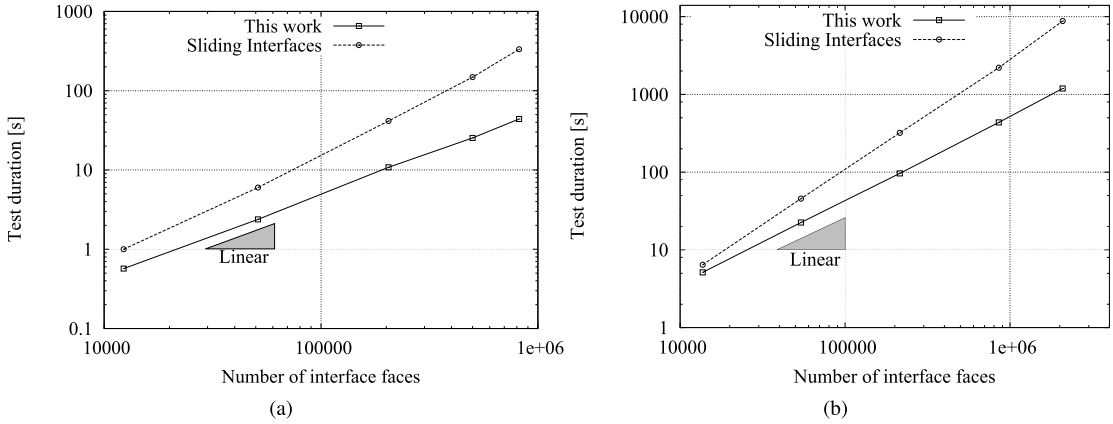

Fig. 10. Computational time required to couple the interfaces as a function of the number of faces: (a) static test; (b) dynamic test. Both tests are executed in serial mode with an i7-3770K CPU.

scales linearly with the size of the interface. In this context, the original order of the local supermeshing approach is preserved. Moreover, when the interface faces are on the order of one million, the current pseudo-supermesh implementation is approximately an order of magnitude faster than the Sliding Interfaces implementation used as a reference.

3.6.1. Advantages of the pseudo-supermesh approach

Regarding the computational efficiency, the main advantage of the pseudo-supermesh approach is that nodes and faces which are already present in the original interfaces are not modified. This feature is beneficial to optimise computational cost and to save memory space. Several computations which are required to define a full supermesh are avoided using the pseudo-supermesh approach. If a full supermesh is required, nodes of the triangulated intersection described in Fig. 3 must be reordered to define a valid polygon. Then, the labels of the supermesh nodes which are already existing in the mesh must be searched from the original ones and, in addition, new labels must be assigned to the new supermesh nodes. Furthermore, new nodes must be introduced into the mesh and removed when recovering the original interfaces. Moreover,

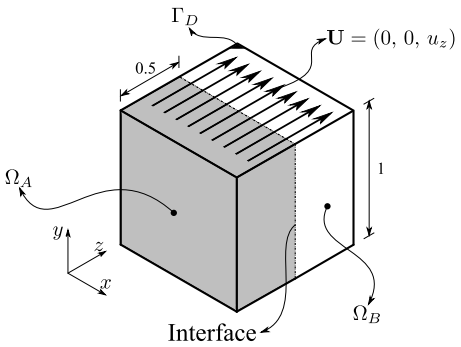

Fig. 11. Description of the pure advective transport inside a cavity flow. The interface separates the region filled with the marker.

each supermesh face must be explicitly defined using the new nodes which implies modifying the original interface faces. In contrast to this, the pseudo-supermesh approach re-uses the original faces as new pseudo-supermesh faces without modifying their definition, only their areas and centroids are modified. In practical applications, the amount of re-used faces are in order of 25–50% of the total pseudo-supermesh faces depending on the interface connectivity.

Additionally to the comparison shown in Fig. 10, another quantitative estimation of the computational efficiency is performed. The aim of this new test is evaluating the CPU time increment as a result of introducing new nodes into the mesh as it is done when constructing a full supermesh. In this sense, a modified version of the pseudo-supermesh approach is evaluated where not only faces are replicated or multiplied but also each node of the pseudo-supermesh faces is considered as a new node which must be introduced into the mesh. In contrast to the comparison achieved with the Sliding Interfaces technique, the methods compared in this analysis use a similar implementation except for the one which adds new nodes. The test is analogous to the static problem shown in Fig. 10(a). In this case, two interfaces of approximately 3 million faces are coupled and decoupled. Here, the implementation which explicitly introduces new nodes into the mesh shows a CPU time overhead of approximately 30%. The inclusion of all steps required to construct a full-supermesh will require even more CPU resources.

4. Numerical examples

In this section, a series of numerical tests are performed to evaluate the pseudo-supermesh interfaces. The first examples are conceived to test the conservation property of the pseudo-supermesh strategy. In the second group of examples, problems with an analytical solution are solved with the objective of evaluating the accuracy of the numerical method when using the pseudo-supermesh interfaces. Finally, a three-dimensional Navier–Stokes problem is solved comparing the pseudo-supermesh results with a conformal mesh solution and with a reference found in the literature.

4.1. Conservation examples

Two scalar transport problems are solved with the aim of evaluating the global conservation of the transported quantity. The first problem is an unsteady and pure advective scalar transport problem inside a cubic cavity flow. The second example is an unsteady pure diffusive problem in a cubic cavity.

4.1.1. Pure advective transport in a cavity flow

An unsteady Navier–Stokes problem is solved inside a cubic cavity of a 1-meter side. Half of the domain is filled with a scalar marker having a value of $\phi = 200$, and the remaining sector is left in $\phi = 0$. The flow is at rest at the beginning of the problem. On the upper side of the cavity, a fixed velocity value in the z component is imposed to induce the flow motion. The fluid kinematic viscosity ν has a value of 0.01 m^2/s and the velocity magnitude of the upper side is set up in 10 m/s to define a Reynolds number of $Re = 1000$. The total simulation time is set up in 50 s. A schematic representation of the problem is shown in Fig. 11 and the mathematical description is presented for the flow and scalar transport problems respectively as follows,

$$\begin{aligned}
\frac{\partial\,[\mathbf{U}(\mathbf{x},t)]}{\partial t} + \nabla\cdot[\mathbf{U}(\mathbf{x},t)\,\mathbf{U}(\mathbf{x},t)] &= \frac{-\nabla p(\mathbf{x},t)}{\rho} + \nu\,\Delta\,[\mathbf{U}(\mathbf{x},t)] && \mathbf{x}\in\Omega \\
\nabla\cdot[\mathbf{U}(\mathbf{x},t)] &= 0 && \mathbf{x}\in\Omega \\
\mathbf{U}(\mathbf{x},t) &= (0,\ 0,\ u_z) && \mathbf{x}\in\Gamma_D \\
\mathbf{U}(\mathbf{x},t) &= (0,\ 0,\ 0) && \mathbf{x}\in\{\,\Gamma-\Gamma_D\} \\
\mathbf{U}(\mathbf{x},0) &= (0,\ 0,\ 0) && \mathbf{x}\in\Omega,
\end{aligned} \tag{4.1}$$

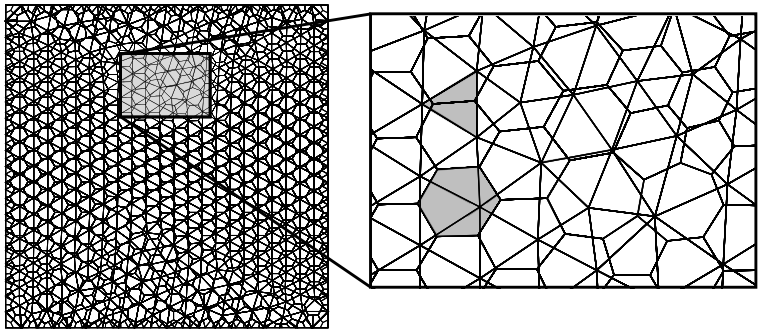

Fig. 12. Triangular and a hexagonal dominant interfaces used in the cavity problem.

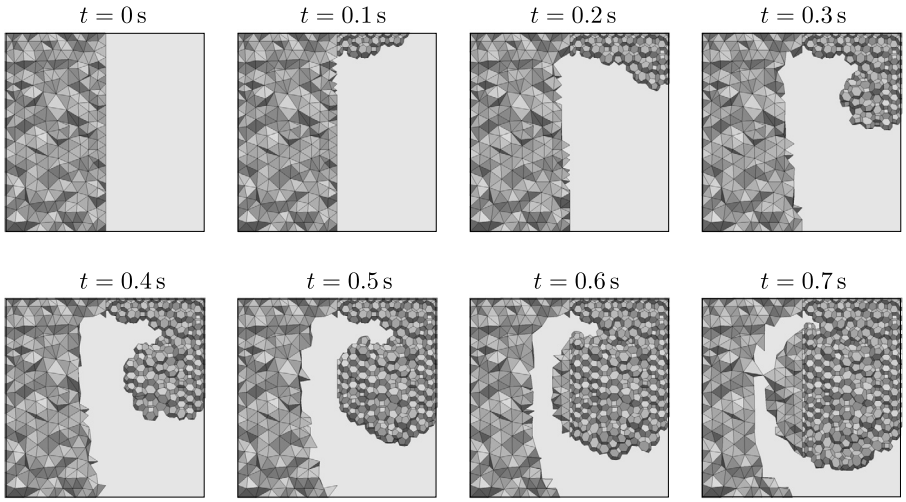

Fig. 13. Threshold of cells located at a transversal plane $x = 0.5$ with values of the scalar marker $\phi \geq 50$ for the first phase of the simulation.

$$\begin{aligned} \frac{\partial \phi(\mathbf{x},t)}{\partial t} + \nabla \cdot [\mathbf{U}(\mathbf{x},t)\,\phi(\mathbf{x},t)] &= 0 \quad && \mathbf{x} \in \Omega \\ \phi(\mathbf{x},0) &= 200 \quad && \mathbf{x} \in \Omega_A \\ \phi(\mathbf{x},0) &= 0 \quad && \mathbf{x} \in \Omega_B \\ \frac{\partial \phi}{\partial n}(\mathbf{x},0) &= 0 \quad && \mathbf{x} \in \Gamma, \end{aligned} \tag{4.2}$$

where $\mathbf{U}(\mathbf{x},t)$ is the velocity, p is the pressure, ρ is the fluid density assumed as constant with a value of 1 kg/m^3, Ω_A and Ω_B are the subdomains separated by the interface, Γ_D is the upper boundary of the cube and Γ is the whole domain boundary.

The subdomains Ω_A and Ω_B are meshed with different discretisation patterns in order to generate an arbitrary non-conformal interface. The first subdomain is meshed with tetrahedral cells (21173 cells) and the remaining subdomain with polyhedra (6831 cells). The non-conformal discretisation patterns generate a triangular and an hexagonal dominant interfaces as shown in Fig. 12. The problem is solved using the merged SIMPLE-PISO algorithm [24,25]. The time-step length is configured with a value of 0.01 s defining a maximum Courant number of $Co \approx 14$. The second order backward differencing scheme is used for the time discretisation and upwind schemes are adopted for the advective terms. The diffusion terms are discretised using a linear scheme.

In this problem, the flow mixes the scalar marker along the whole domain as shown in Fig. 13 where the flow and the scalar marker cross the interfaces continuously. Therefore, to conserve the global integral of the scalar marker in the cavity it is important to solve both problems conservatively, the flow and the scalar transport ones. The problem is also solved by interpolating the fields at the interface with area-based weights, without pseudo-supermeshes. The results are presented in Fig. 14 showing the global average of the scalar quantity over time. The pseudo-supermesh approach exhibits conservation to machine precision (10^{-16}) of the scalar marker quantity over the whole simulation time. On the other hand, the simulation based on area-weighted interpolations has a non-negligible conservation issue. For this solution, the global average grows indefinitely over the time indicating a stability problem of this non-conservative interpolation. The standard deviation of the scalar marker is also evaluated as an indicator of mixing. As a reference, a one-region simulation (without

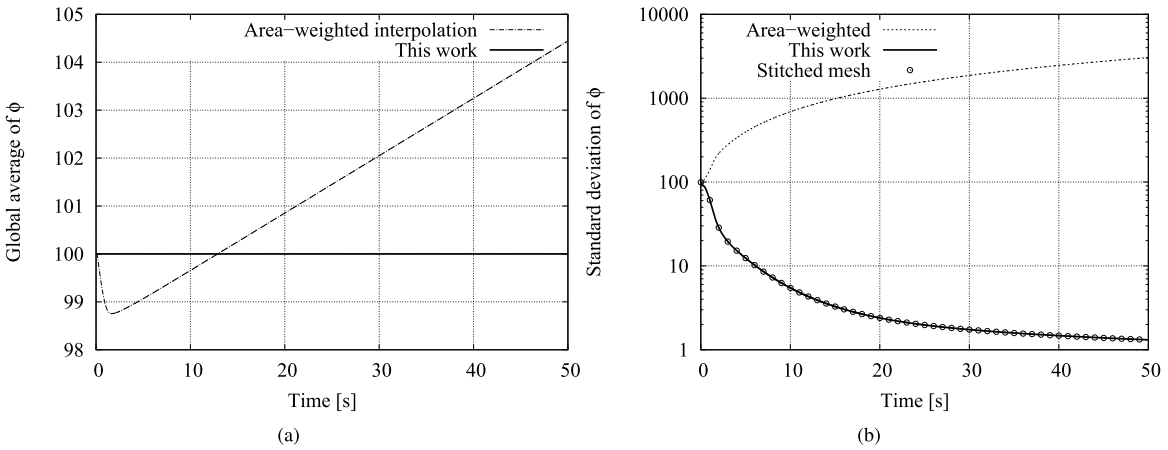

Fig. 14. Results of the simulations using pseudo-supermeshes and a method based on overlapped-area interpolations; (a) evolution over time of the global average of ϕ; (b) evolution over time of the standard deviation of ϕ.

interfaces) is solved. The evolution of the standard deviation against the time is plotted in Fig. 14. The standard deviation of ϕ for the pseudo-supermesh approach is equivalent to the results of the one-region problem. In contrast, the results of the non-conservative method show an unphysical behaviour of the standard deviation.

4.1.2. Pure diffusive problem

A pure diffusive problem is proposed to evaluate the conservative property of the diffusive discretisation using pseudo-supermesh interfaces. The test analyses the mixing through diffusion of an initial scalar marker distribution inside a cubic cavity. The definition of the problem is as follows,

$$\begin{aligned} \frac{\partial \phi(\mathbf{x},t)}{\partial t} + \nabla \cdot [\alpha(\mathbf{x}) \nabla \phi] &= 0 \\ \phi(\mathbf{x},0) &= 200 \qquad \mathbf{x} \in \Omega_A \\ \phi(\mathbf{x},0) &= 0 \qquad \mathbf{x} \in \Omega_B \ , \\ \frac{\partial \phi}{\partial n}(\mathbf{x},0) &= 0 \qquad \mathbf{x} \in \Gamma \\ \alpha(\mathbf{x}) &= 1 + 10000\, x^2 y^2 z^2 \end{aligned} \tag{4.3}$$

where ϕ is the transported variable, $\alpha(\mathbf{x})$ is the diffusivity coefficient, Γ is the domain boundary and Ω_A and Ω_B are the subdomains. They are meshed with hexahedral cells using different grid refinements in order to generate a non-conformal interface. The problem is schematically presented in Fig. 15. A non-homogeneous scalar diffusivity is defined to amplify the potentially non-conservation issues. For reference, three arrows are located on the cavity sides indicating the direction in which the diffusivity increases. Furthermore, references to the minimum and maximum values of the diffusivity are detailed for the nodes $(0,0,0)$ and $(1,1,1)$. The problem is solved with an unsteady diffusive solver where the Laplacian operator is discretised with a linear scheme and the temporal derivative is discretised with an Euler scheme. The total simulation time is configured to 1 s and the time-step length is set to 0.001 s. The global average of the scalar marked is plotted over the time in Fig. 16. The evolution of the global average of ϕ indicates a correct conservation of the scalar quantity with the method presented in this work. Alternatively, as done in the previous test, a simulation based on area-weighted interpolations is included. This simulation does not conserve correctly the scalar marker increasing the initial global quantity of ϕ on the order of 1%. The significant imbalance of ϕ takes place in the initial phase of the simulation where the gradient of the scalar marker at the interface is high.

4.2. Analytic examples

The next tests evaluate the accuracy of the pseudo-supermesh approach. In particular, the first example analyses the performance of the present approach with an advective problem. Similarly, the second test evaluates the accuracy of the diffusive discretisation showing the influence of the non-orthogonal corrections. Finally, the last two examples combine the two differential operators by solving two different Navier–Stokes problems.

4.2.1. Pure advection of a Gaussian profile

In this test, a scalar marker with a Gaussian profile is transported along a square duct by a constant flow. The domain Ω is divided into two subdomains which are meshed with different discretisation patterns and coupled with the pseudo-supermesh interfaces. The problem is defined in Eq. (4.4) and schematically presented in Fig. 17.

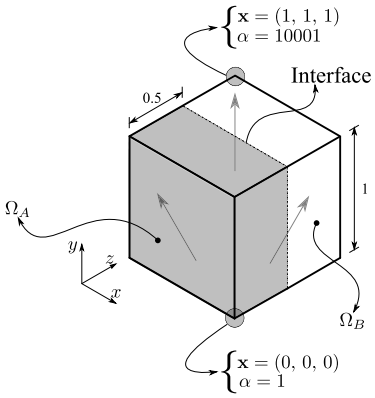

Fig. 15. Cubic cavity with an interface located at the plane $z = 0.5$. A non-uniformity diffusivity field is defined in the cavity. The diffusivity field has its minimum value at point $\mathbf{x} = (0, 0, 0)$ and its maximum in point $\mathbf{x} = (1, 1, 1)$.

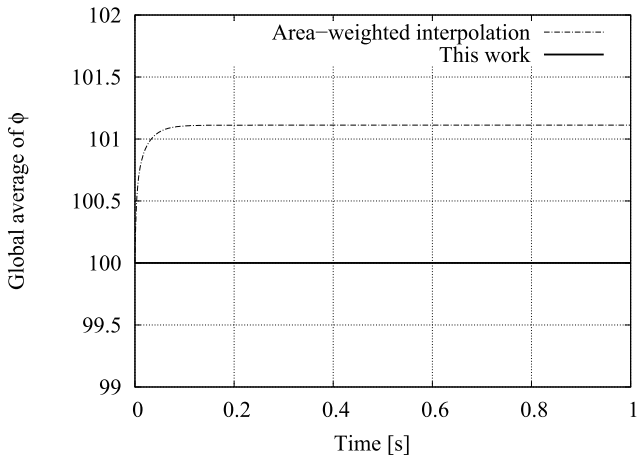

Fig. 16. Global average of the scalar marker over simulation time for the pure diffusive problem.

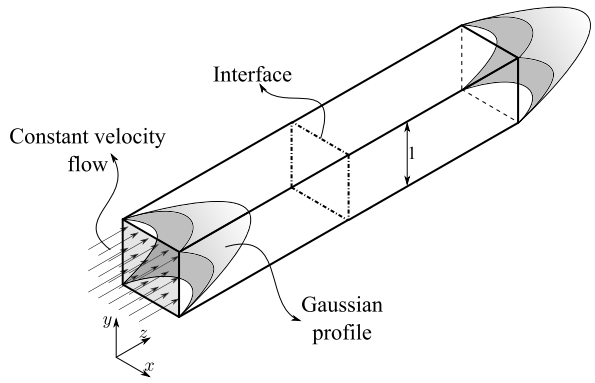

Fig. 17. A Gaussian profile is transported by a uniform flow inside a square duct. The domain discretisation presents a non-conformal interface in the middle of the duct.

$$\begin{aligned} \nabla \cdot [\mathbf{U}(\mathbf{x})\,\phi] &= 0 \\ \mathbf{U}(\mathbf{x}) &= (0,\,0,\,1) \qquad\quad \mathbf{x} \in \Omega \\ \phi(\mathbf{x}) &= \exp\left[-10\,(x^2 + y^2)\right] \quad \mathbf{x} \in \Gamma_D \end{aligned} \quad , \tag{4.4}$$

where $\mathbf{U}(\mathbf{x})$ is the flow velocity, $\phi(\mathbf{x})$ is the transported variable (scalar marker) and Γ_D is the inlet boundary. The solution of the problem is the complete propagation of the inlet boundary condition along the whole domain which generates a constant profile of $\phi(\mathbf{x})$ in the z direction,

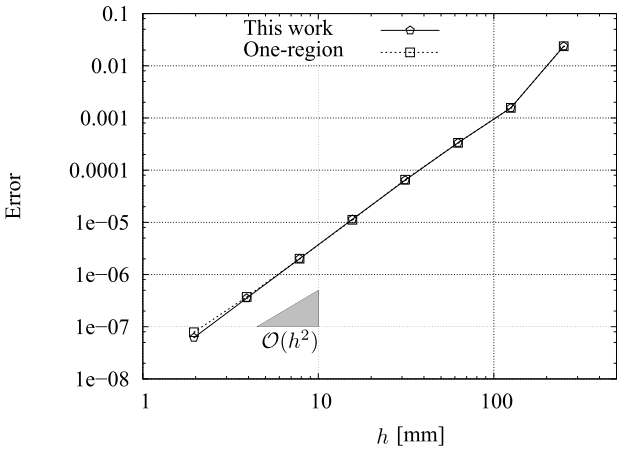

Fig. 18. Convergence of the error for the pure advective problem.

$$\phi(\mathbf{x}) = \exp\left[-10\,(x^2+y^2)\right] \qquad \mathbf{x}\in\Omega. \tag{4.5}$$

The subdomains are meshed with a pure hexahedral mesh with different discretisation patterns to generate dissimilar surface meshes at the interface. Namely, the interfaces are meshed with $[n\times n]$ and $[(n+1)\times(n+1)]$ points per side, where n is varied from $n=4$ to $n=512$. The problem is solved with an unsteady solver starting from $\phi=0$ on the whole domain at time $t=0$. The simulation finishes when the stationary state is reached. The advective terms are discretised with the second order upwind scheme. The mean square error of the outlet profile is plotted as a function of the mesh size in Fig. 18. A one-region domain (without interfaces) is also included in the analysis. The results demonstrate a second order convergence of the solution using the pseudo-supermesh approach. Furthermore, the error magnitude is equal to the one-region case which demonstrates that the pseudo-supermesh approach does not introduce additional error into the solution.

4.2.2. A manufactured diffusive problem

A manufactured diffusive problem on a cubic cavity domain is presented. The main aim of this example is to evaluate the order of convergence of the pseudo-supermesh approach in a pure diffusive problem. In particular, the influence of the non-orthogonal corrections on the error is analysed. The definition of the problem is as follows,

$$\begin{aligned}
\nabla\cdot[\nabla\phi(\mathbf{x})] &= 6 && \mathbf{x}\in\Omega\\
\phi(\mathbf{x}) &= 1+x+y+x^2 && \mathbf{x}\in\Gamma_A=\{\mathbf{x}\in\Gamma : z=0\}\\
\phi(\mathbf{x}) &= 3+x+y+xy+x^2+y^2 && \mathbf{x}\in\Gamma_B=\{\mathbf{x}\in\Gamma : z=1\}\\
\phi(\mathbf{x}) &= 1+y+z+y^2+z^2 && \mathbf{x}\in\Gamma_C=\{\mathbf{x}\in\Gamma : x=0\}\\
\phi(\mathbf{x}) &= 3+y+z+yz+y^2+z^2 && \mathbf{x}\in\Gamma_D=\{\mathbf{x}\in\Gamma : x=1\}\\
\phi(\mathbf{x}) &= 1+x+z+x^2+z^2 && \mathbf{x}\in\Gamma_E=\{\mathbf{x}\in\Gamma : y=0\}\\
\phi(\mathbf{x}) &= 3+x+z+xz+x^2+z^2 && \mathbf{x}\in\Gamma_F=\{\mathbf{x}\in\Gamma : y=1\}
\end{aligned}, \tag{4.6}$$

where $\phi(\mathbf{x})$ is the scalar quantity, Ω is the domain of the problem, Γ is the domain boundary and $\Gamma_A..\Gamma_F$ are the sides of the cavity. The problem has analytical solution:

$$\phi(\mathbf{x}) = 1+x+y+z+xyz+x^2+y^2+z^2 \qquad \mathbf{x}\in\Omega. \tag{4.7}$$

The domain Ω is a cubic cavity of 1 meter per side where a pseudo-supermesh interface is located in the plane $z=0.5$ as shown in Fig. 15. A convergence study is achieved solving the problem given in Eq. (4.6) with different discretisations for both sides of the interfaces as done in the previous test. The results are presented in Fig. 19 where a very good agreement is achieved between the results of the pseudo-supermesh and the one-region simulations. The non-orthogonal corrections improve the accuracy of the method to second order which indicates a proper handling of the non-orthogonal corrections at the pseudo-supermesh interfaces.

4.2.3. Viscous vortex problem

In the previous tests, the convective and diffusive discretisations at the pseudo-supermeshes are tested separately. In this case, the combination of both operators is validated solving the unsteady two-dimensional Navier Stokes equations. The problem solved is the transport of a vortex structure through the interfaces by a uniform mean flow. The initial condition of the velocity $\mathbf{U}=(u,v)$ is,

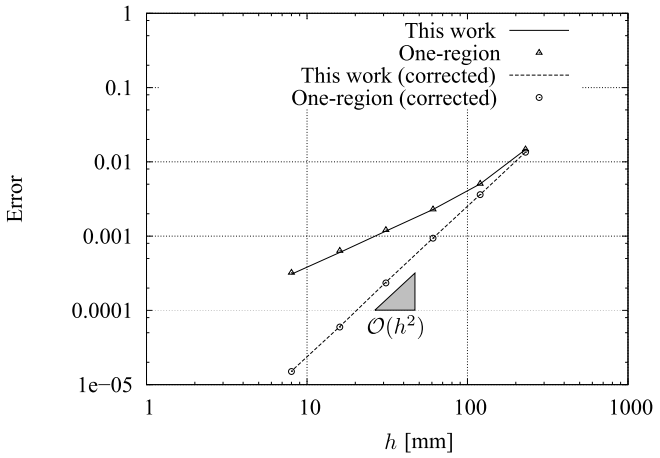

Fig. 19. Convergence of the error in the manufactured diffusive problem. The cases with non-orthogonal corrections and without them are compared.

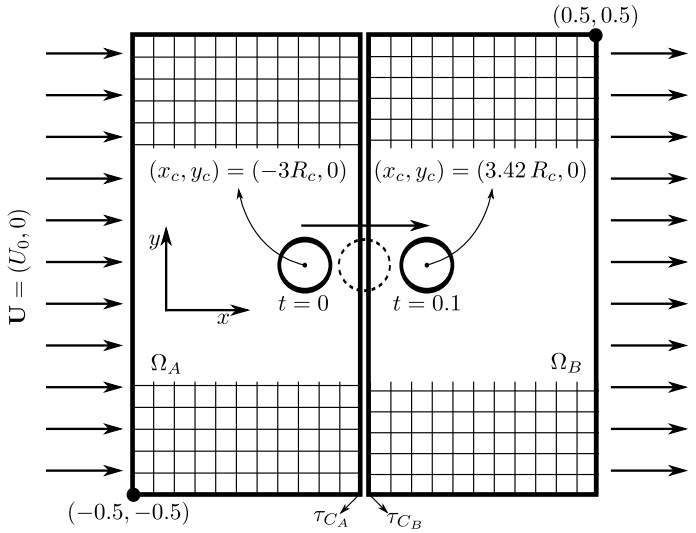

Fig. 20. Convection of a vortex through the pseudo-supermesh interfaces by a uniform flow. The starting and final positions of the vortex (x_c, y_c) are indicated.

$$u(x, y, 0) = U_0 - \frac{\Gamma}{R_c^2}(y - y_c)\exp\left(\frac{-r^2}{2R_c^2}\right), \qquad v(x, y, 0) = \frac{\Gamma}{R_c^2}(x - x_c)\exp\left(\frac{-r^2}{2R_c^2}\right), \tag{4.8}$$

where x_c and y_c are the vortex centre coordinates, $r = \sqrt[2]{(x - x_c)^2 + (y - y_c)^2}$ is the distance to the vortex centre, Γ is the vortex strength, R_c the radius of the vortex and U_0 is the mean flow velocity magnitude. The problem has analytical solution given by,

$$u(x, y, t) = U_0 - \frac{\Gamma}{R_c^2}\frac{(y - y_c)}{\alpha^2}\exp\left(\frac{-r^2}{2\,\alpha\,R_c^2}\right), \qquad v(x, y, t) = \frac{\Gamma}{R_c^2}\frac{(x - x_c)}{\alpha^2}\exp\left(\frac{-r^2}{2\,\alpha\,R_c^2}\right), \tag{4.9}$$

where α is defined by,

$$\alpha = 1 + 2\,\nu\,t, \tag{4.10}$$

being ν the kinematic viscosity.

The configuration of the problem is based on the work of Wang et al. [26]: the vortex strength is set up to $\Gamma = 0.036$, the vortex radius is $R_c = 0.01556$ m and the mean flow velocity is $U_0 = 1$ m/s. A graphical scheme of the domain and problem description is shown in Fig. 20. The vortex is positioned at the time $t = 0$ s in $(x_c, y_c) = (-3R_c, 0)$ and is convected to the position $(x_c, y_c) \approx (3.42R_c, 0)$ at time $t = 0.1$ s. The subdomains Ω_A and Ω_B are meshed with different number of grid points in the y-direction: N_y and $(N_y + 1)$ for the pseudo-supermesh interfaces τ_{C_A} and τ_{C_B} respectively.

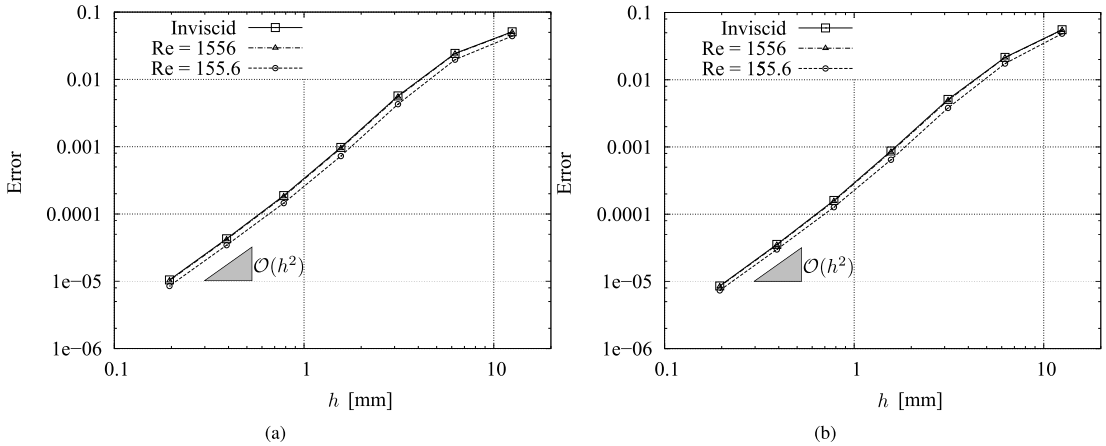

Fig. 21. Root mean square error of the velocity; (a) u component and (b) v component.

The pressure–velocity coupling is solved with the SIMPLE-PISO algorithm. The domain boundaries are located far enough away from the vortex so that the analytic solution there falls below machine precision (10^{-16}). At boundaries, the velocity is set up to the uniform mean flow U_0 and the pressure is fixed to $p = 0$ at the outlet. On the other hand, a null normal gradient is configured for p at the rest of the boundary. The problem is solved with different grid refinements (from $N_y = 40$ to $N_y = 640$), varying the time-step length accordingly to set up a maximum Courant number of $Co \approx 0.95$. Three simulations are performed with different viscosity values, $\nu_1 = 0$, $\nu_2 = 10^{-5}$ and $\nu_3 = 10^{-4}$ m^2/s which define the inviscid case, and viscous problems with $Re = 1556$ and $Re = 155.6$ respectively. The Reynolds number is defined as $Re = U_0\, Rc/\nu$. The advective operator is discretised with the second order upwind scheme and the diffusive operator with a linear interpolation scheme using non-orthogonal corrections. Likewise, the temporal discretisation is addressed with the second order backward differencing scheme. The root mean square error of the solutions is plotted in Fig. 21 as a function of the cell size of the mesh for both velocity components. The h value considered is the left sub-domain cell size. The results demonstrate that the pseudo-supermesh interfaces do not affect the order of accuracy of the discretisation schemes being second order accurate.

4.2.4. Two-dimensional Taylor–Green vortex

The two-dimensional Taylor–Green vortex problem is solved to evaluate the behaviour of the pseudo-supermeshes when using a non-flat interface. The problem configuration is based on the work of Mirkov et al. [27] which has the following analytic solution,

$$\begin{aligned} u(x, y, t) &= -\sin(x)\cos(y)\exp(-2\nu t), \\ v(x, y, t) &= \cos(x)\sin(y)\exp(-2\nu t), \\ p(x, y, t) &= \frac{1}{4}[\cos(2x) + \sin(2y)]\exp(-4\nu t), \end{aligned} \tag{4.11}$$

where ν is the kinematic viscosity. The problem is solved using a square domain with a side length of 2π and setting periodic conditions at boundaries. A circular interface of a diameter equal to $\sqrt{2}\pi$ is centred inside the squared domain defining the subdomains Ω_A and Ω_B. This circular interface is coincident with some vortex centres (points with null velocity magnitude) of the problem as shown in Fig. 22. A convergence study is performed varying the cell size of the domain discretisation. In contrast to the previous problems, a different non-conformal pattern is adopted at the interface. More specifically, for n divisions of the interface τ_{C_A}, the opposite interface τ_{C_b} is meshed with $(1.5n + 1)$ grid points. Using this division scheme, the ratio of characteristic lengths of both interfaces tends to 1.5 when n increases. A graphical description of the interface discretisation is depicted in Fig. 22 where the coarser and finer interfaces have 64 and 97 grid points respectively.

The discretisation schemes of the differential operators are equal to those used in the vortex convection test. The initial condition for velocity and pressure is the analytic solution at time $t = 0$ s. The time step is set up in order to obtain a maximum Courant number of approximately 3.8 and each simulation is run until reaching the total time of 1.28 s. Kinematic viscosity is set in 0.01 m^2/s defining a Re number of 157. The results are presented in Fig. 23 showing the root mean square error of the velocity (the error of u_x and u_y are equal since the problem is symmetric) as a function of the finest interface characteristic length.

Second order convergence of the error is achieved. It is important to remark that the circular interface generates a mesh which has some non-orthogonality and skewness. As a consequence of this, skewness corrections on the cell-face interpolation should be used to obtain a second order convergence.

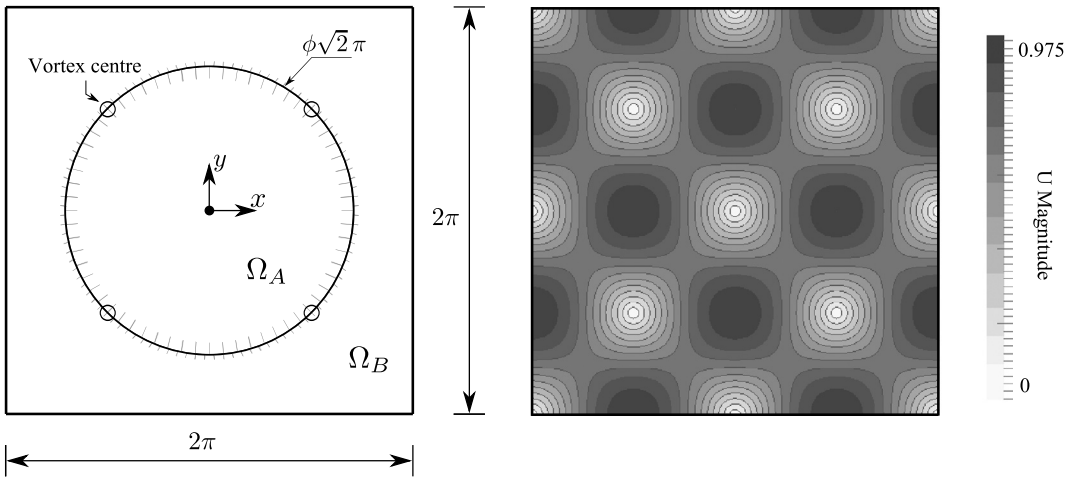

Fig. 22. Description of the Taylor–Green vortex problem. A circular interface with a (n) vs $(1.5n+1)$ discretisation scheme is located in the centre of the domain. On the right side, velocity contours of the simulation results at $t = 1.28$ s are presented.

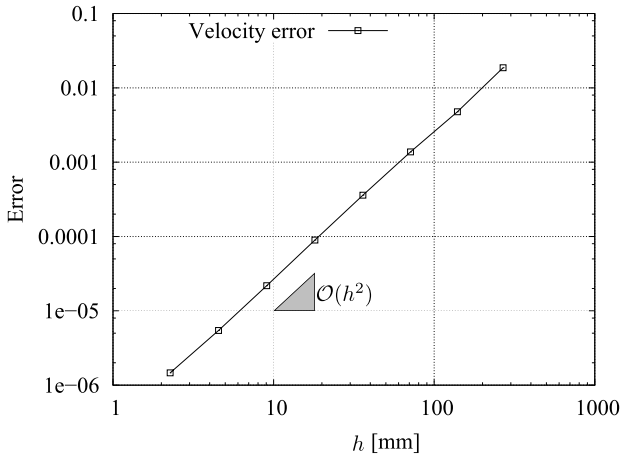

Fig. 23. Convergence of the velocity error (u_x or u_y) for the Taylor–Green vortex problem.

4.3. Three-dimensional Taylor–Green vortex

Finally, the three-dimensional Taylor–Green vortex problem [28] is solved using non-conformal interfaces with the pseudo-supermesh approach. This test starts with an initial vortex flow which evolutes until rest transitioning through turbulence and energy dissipation phases. The aim of this problem is to evaluate the capability of the pseudo-supermesh approach to reproduce the physics of the flow without introducing spurious errors in comparison with a one-region mesh simulation which is used as a validation result. As an additional reference, a pseudo-spectral solution is also included in the analysis [29]. The domain of the problem is a cube of a side length of 2π. In this particular test, the cubic domain is divided into three subdomains: Ω_A, Ω_B and Ω_C. These subdomains generate two pairs of pseudo-supermesh interfaces which are planes perpendicular to the x-axis as shown in Fig. 24.

The problem is defined by a periodic flow which starts with the following initial condition for the velocity components and the pressure,

$$\begin{aligned}
u(x,y,z,t=0) &= V_0 \sin\left(\frac{x}{L}\right)\cos\left(\frac{y}{L}\right)\cos\left(\frac{z}{L}\right),\\
v(x,y,z,t=0) &= -V_0 \cos\left(\frac{x}{L}\right)\sin\left(\frac{y}{L}\right)\cos\left(\frac{z}{L}\right),\\
z(x,y,z,t=0) &= 0,\\
p(x,y,z,t=0) &= p_0 + \frac{\rho_0 V_0^2}{16}\left[\cos\left(\frac{2x}{L}\right)+\cos\left(\frac{2y}{L}\right)\right]\left[\cos\left(\frac{2z}{L}\right)+2\right],
\end{aligned} \tag{4.12}$$

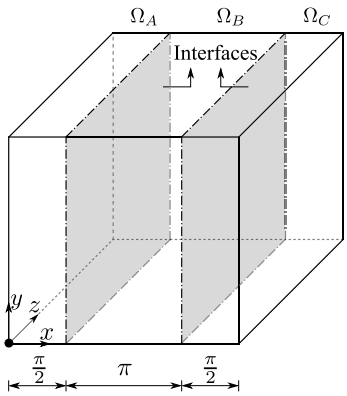

Fig. 24. Cubic region divided into three subdomains along the x-axis defining two pairs of interfaces.

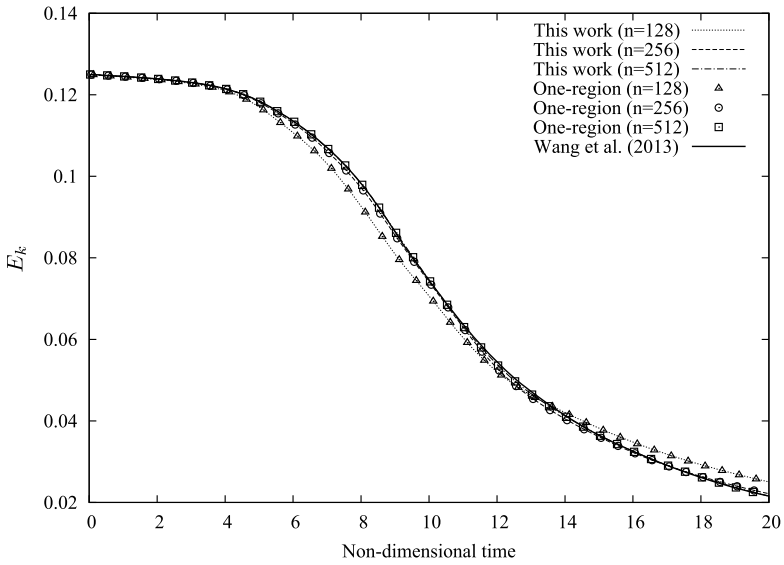

Fig. 25. Temporal evolution of the kinetic energy.

where the velocity V_0, the characteristic length L and the reference density ρ_0 are set up to 1 and the reference pressure p_0 is set up to 0. The viscosity is defined in order to set up a Reynolds number of 1600 ($\mathrm{Re} = V_0 L/\nu$). The problem is studied computing the temporal evolution of the kinetic energy E_k and its derivative $-dE_k/dt$ which represents the kinetic energy dissipation rate. The kinetic energy is calculated as,

$$E_k = \frac{1}{\rho_0 \Omega} \int_{\Omega} \frac{1}{2} (\mathbf{U} \cdot \mathbf{U}) \, d\Omega, \tag{4.13}$$

being $U = (u, v, z)$ the velocity vector.

A mesh convergence study is performed where the subregions Ω_A and Ω_C are meshed with $[n/4 \times n \times n]$ points per side and the middle sector Ω_B with $[(n/2) \times (n+1) \times (n+1)]$ respectively generating non-conformal meshes at the interfaces. Analogously, the one-region domain is discretised with $[n \times n \times n]$ points per side. The set up of the numerical solver is configured as done in the viscous vortex problem with second order discretisation schemes for the convective, diffusive and temporal terms. In particular, the Δt value is set up to define a maximum Courant number of approximately 6. Three different mesh resolutions are simulated ($n = 128$; $n = 256$ and $n = 512$). The temporal evolutions of E_k and $-dE_k/dt$ are presented in Fig. 25 and Fig. 26 for each mesh resolution analysed. From these results, it is concluded that the pseudo-supermesh interfaces do not introduce numerical errors in comparison with the one-region mesh simulations. Furthermore, the results of the current approach have an acceptable agreement with the pseudo-spectral solution chosen as a reference.

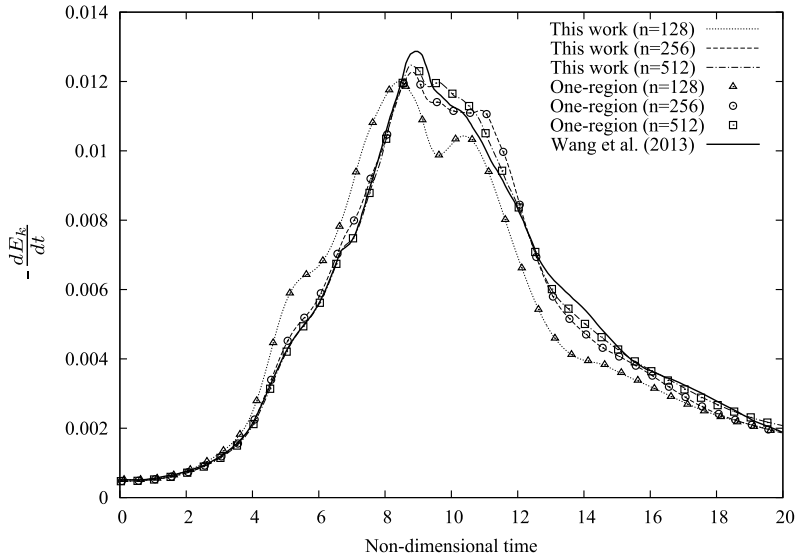

Fig. 26. Temporal evolution of the kinetic energy dissipation rate.

5. Conclusions

A simple strategy to couple non-conformal interfaces has been presented and proved to be conservative, computationally efficient and second order accurate. In order to achieve these objectives, an interpolation expression for a general transport problem with advective and diffusive terms has been derived based on the conservation of fluxes. Due to the arbitrary connectivity of non-conformal grids, the interface values must be calculated by a flux-weighted interpolation. If the interfaces are conformal, the interpolations become trivial and consequently, the flux computation at the interfaces is straightforward. Considering the last feature, the original non-conformal grids have been replaced with a pair of new interfaces (pseudo-supermeshes) with a one-to-one addressing. The new coupling strategy generates copies of the original faces to determine a conformal connectivity between them. A significant advantage of the developed pseudo-supermesh approach is that it is formally conservative. The flux-conservation does not depend on the convergence of the numerical algorithm since the pseudo-supermesh faces are equivalent to the internal ones. Then, the telescopic property of the FVM is fully satisfied.

To obtain second order accuracy, the area and centroid of the pseudo-supermesh faces must be redefined. This task has been achieved with the local supermeshing approach [19] which is a robust and efficient algorithm to handle non-conformal interfaces resulting from complex unstructured meshes. The obtained pseudo-supermesh strategy is computationally efficient and simple to implement since the arbitrary polygons resulting from the grid intersections are not required. Therefore, the addition of new nodes, edges, and faces into the mesh is avoided.

The new strategy has been implemented for parallel computing architectures, showing an excellent performance in a scalability test. Regarding the computational efficiency, the method has been evaluated measuring the CPU time required to couple and decouple the interfaces. A better performance of the pseudo-supermesh strategy has been obtained, in contrast to a stitch-based methodology. Moreover, the linear complexity of the local supermeshing algorithm has been preserved.

In order to evaluate the conservativeness and accuracy of the new technique, a series of numerical tests have been performed. The pseudo-supermesh approach has shown conservation to machine precision of fluxes for advective and diffusive problems. In contrast, the computations done with a method based on area-weighted interpolations (not flux-weighted) have shown conservation and stability problems. The accuracy of the proposed technique has been evaluated using analytical solutions as a reference. The performance of the advective and diffusive operators have been analysed separately and its combination has been tested solving Navier–Stokes problems with analytical solutions. In all the tests, the solutions have converged with a second-order rate. Finally, the pseudo-supermesh approach was tested in a three-dimensional benchmark problem obtaining a very good agreement with the reference solutions.

Taking into account the properties summarised above, this work proposes a very advisable alternative to simulate academic and industrial multi-domain problems in which conservativeness, accuracy and efficiency are desired.

Acknowledgements

The authors wish to acknowledge CONICET, Universidad Nacional del Litoral and ANPCyT for their financial support through grants PIP-2012 GI 11220110100331, CAI+D 2011 501 201101 00435 LI and PICT-2013 0830.

References

[1] T. Lucchini, G. D'Errico, H. Jasak, Z. Tukovic, Automatic Mesh Motion with Topological Changes for Engine Simulation, Tech. rep., SAE technical paper, 2007.
[2] F. Piscaglia, A. Montorfano, A. Onorati, Development of Fully-automatic Parallel Algorithms for Mesh Handling in the OpenFOAM®-2.2.x Technology, Tech. rep., SAE technical paper, 2013.
[3] F. Piscaglia, A. Montorfano, A. Onorati, A moving mesh strategy to perform adaptive large eddy simulation of IC engines in OpenFOAM, in: International Multidimensional Engine Modeling User's Group Meeting, The Detroit Downtown Courtyard by Marriott Hotel, Detroit, MI, USA, 2014.
[4] N. Qin, G. Carnie, A. LeMoigne, X. Liu, S. Shahpar, Buffer layer method for linking two non-matching multi-block structured grids, in: 47th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, Orlando, FL, January 2009, pp. 5–8.
[5] Y. Wang, N. Qin, G. Carnie, S. Shahpar, Zipper layer method for linking two dissimilar structured meshes, J. Comput. Phys. 255 (2013) 130–148.
[6] Z. Wang, N. Hariharan, R. Chen, Recent development on the conservation property of chimera, Int. J. Comput. Fluid Dyn. 15 (4) (2001) 265–278.
[7] M. Berger, On conservation at grid interfaces, SIAM J. Numer. Anal. 24 (5) (1987) 967–984.
[8] M.M. Rai, A relaxation approach to patched-grid calculations with the Euler equations, J. Comput. Phys. 66 (1) (1986) 99–131.
[9] M.M. Rai, Navier–Stokes simulations of rotor/stator interaction using patched and overlaid grids, J. Propuls. Power 3 (5) (1987) 387–396.
[10] A. Lerat, Z. Wu, Stable conservative multidomain treatments for implicit Euler solvers, J. Comput. Phys. 123 (1) (1996) 45–64.
[11] Y. Zhang, H. Chen, S. Fu, Improvement to patched grid technique with high-order conservative remapping method, J. Aircr. 48 (3) (2011) 884–893.
[12] S. Mathur, Unsteady flow simulations using unstructured sliding meshes, in: Fluid Dynamics Conference, 1994, p. 2333.
[13] C.Y. Loh, W.M. To, A. Himansu, A conservative treatment of sliding interface for upwind finite volume methods, in: 47th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, 2009, p. 861.
[14] L. Ramírez, C. Foulquié, X. Nogueira, S. Khelladi, J.-C. Chassaing, I. Colominas, New high-resolution-preserving sliding mesh techniques for higher-order finite volume schemes, J. Comput. Fluids 118 (2015) 114–130.
[15] J. McNaughton, I. Afgan, D. Apsley, S. Rolfo, T. Stallard, P. Stansby, A simple sliding-mesh interface procedure and its application to the CFD simulation of a tidal-stream turbine, Int. J. Numer. Methods Fluids 74 (4) (2014) 250–269.
[16] M. Beaudoin, H. Jasak, Development of a generalized grid interface for turbomachinery simulations with OpenFOAM, in: Open Source CFD International Conference, Berlin, 2008.
[17] H. Weller, G. Tabor, H. Jasak, C. Fureby, A tensorial approach to computational continuum mechanics using object oriented techniques, Comput. Phys. 12 (6) (1998) 620–631.
[18] P. Farrell, M. Piggott, C. Pain, G. Gorman, C. Wilson, Conservative interpolation between unstructured meshes via supermesh construction, Comput. Methods Appl. Mech. Eng. 198 (2009) 2632–2642.
[19] P. Farrell, J. Maddison, Conservative interpolation between volume meshes by local Galerkin projection, Comput. Methods Appl. Mech. Eng. 200 (2011) 89–100.
[20] R. Steijl, G. Barakos, Sliding mesh algorithm for CFD analysis of helicopter rotor–fuselage aerodynamics, Int. J. Numer. Methods Fluids 58 (5) (2008) 527–549.
[21] E. Rinaldi, P. Colonna, R. Pecnik, Flux-conserving treatment of non-conformal interfaces for finite-volume discretization of conservation laws, Comput. Fluids 120 (2015) 126–139.
[22] W. Shyy, A study of finite difference approximations to steady-state, convection-dominated flow problems, J. Comput. Phys. 57 (3) (1985) 415–438.
[23] T.J. Barth, D.C. Jespersen, The design and application of upwind schemes on unstructured meshes, in: 27th AIAA Aerospace Sciences Meeting, Nevada, January 1989.
[24] S. Patankar, Numerical Heat Transfer and Fluid Flow, Hemisphere Publishing Company, 1980.
[25] R.I. Issa, Solution of implicitly discretized fluid flow equations by operator-splitting, J. Comput. Phys. 62 (1986) 40–65.
[26] G. Wang, F. Duchaine, D. Papadogiannis, I. Duran, S. Moreau, L. Gicquel, An overset grid method for large eddy simulation of turbomachinery stages, J. Comput. Phys. 274 (2014) 333–355.
[27] N. Mirkov, B. Rašuo, S. Kenjereš, On the improved finite volume procedure for simulation of turbulent flows over real complex terrains, J. Comput. Phys. 287 (2015) 18–45.
[28] G. Taylor, A. Green, Mechanism of the production of small eddies from large ones, Proc. R. Soc. Lond. Ser. A, Math. Phys. Sci. 158 (895) (1937) 499–521.
[29] Z. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H. Huynh, et al., High-order CFD methods: current status and perspective, Int. J. Numer. Methods Fluids 72 (8) (2013) 811–845.