OpenFOAM® Journal

Volume 5 [Full Papers], Pages 38–58
ISSN: 2753-8168

Analysis of Coupling Strategies for Conjugate Heat Transfer Problems

Emad Tandis[1,*], Philip Cardiff[2], and Ali Ashrafizadeh[3]

[1]Aston Fluid Group, College of Engineering and Physical Sciences, Aston University, the United Kingdom
Email address: e.tandis@aston.ac.uk

[2]University College Dublin, School of Mechanical and Materials Engineering, Belfield, Ireland

[3]DOS Computational Lab, Faculty of Mechanical Engineering, K. N. Toosi University of Technology, Tehran, Iran

DOI: https://doi.org/10.51560/ofj.v5.92
Results with version(s): foam-extend-4.0, OpenFOAM-v2112, OpenFOAM-9
Repository: https://github.com/tandise/ConjugateHeatTransfer-OpenFOAM

Abstract. Numerical solutions to conjugate heat transfer (CHT) problems present challenges regarding accuracy, stability, and computational cost. The current study presents a detailed analysis of the existing OpenFOAM® CHT solvers, including commentary on the multiple forms of the available governing equations and coupling methods. Subsequently, two modified CHT approaches are proposed: (i) a new partitioned algorithm with improved efficiency, and (ii) the available monolithic CHT solver in foam-extend-4.0 was modified and extended to multiple regions. The performance of the proposed solvers is assessed on several transient test cases with various degrees of coupling strength. The results reveal the superiority of the proposed monolithic approach over the partitioned ones, particularly for problems for more than two regions. It is also found that the new partitioned approach is computationally more efficient than the available partitioned solvers.

1. Introduction

Thermal interaction between fluids and solids, known as conjugate heat transfer (CHT), has been widely studied [1–5]. The primary governing equations for a classical CHT problem consist of fluid equations (conservation of mass, momentum and energy) and one solid equation (conservation of energy). This problem is classified as a multi-physics phenomenon, characterised by at least two distinct computational sub-domains, i.e. fluid and solid regions, whose energy equations (or temperature variables) are coupled at shared boundaries (interfaces). In addition to this inter-region temperature-temperature (TT) coupling, there are usually other types of coupling on the fluid side: (i) the pressure-velocity (pU) coupling ties the momentum and mass equations, and (ii) the pressure-velocity-temperature (pUT) coupling connects the momentum and energy equations. These additional couplings add more complexity to CHT problems and require special care when designing efficient solution algorithms. One objective of this article is to provide a detailed analysis of these types of coupling (Fig. 1): temperature-temperature, pressure-velocity-temperature, and pressure-velocity.

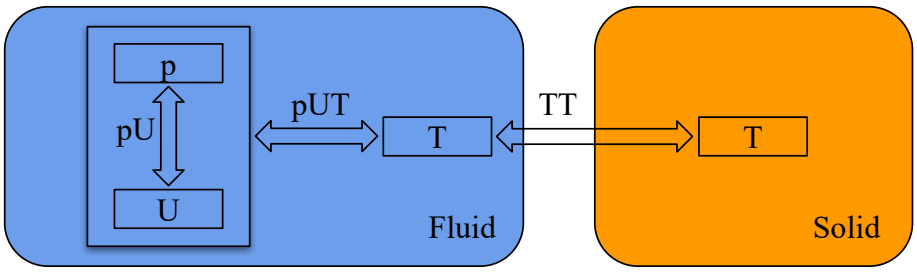


Figure 1. Three types of coupling in the equations of conjugate heat transfer

* Corresponding author
Received: 7 September 2022, Accepted: 20 January 2025, Published: 7 March 2025

Independent of the role of the energy equation, pressure-velocity coupling is present in all classic fluid algorithms. Various studies have investigated different techniques for resolving this type of coupling, with the goal of providing better performance in terms of stability and efficiency [6–11]. Available CHT solvers in OpenFOAM® typically adopt some variant of PISO [12] or SIMPLE [13] algorithm to achieve this, or more generally, by the PIMPLE (merged PISO-SIMPLE) algorithm. As is demonstrated in the test cases section, the solution time for CHT problems can be significantly influenced by the computational expense needed for the pressure-velocity coupling.

Numerical approaches for solving CHT problems and, more generally, multi-physics problems can be classified in terms of the coupling strategy: (i) partitioned or (ii) monolithic. While the former allows for an independent specialised solver in each region and offers the advantage of code reuse, the latter is typically superior in efficiency and stability, albeit with the disadvantage of a larger linear system. The fluid and solid solvers are called sequentially in the partitioned algorithms, and information is exchanged explicitly [14–17]. Consequently, accurate enforcement of the interface conditions necessitates a predictor-corrector loop, which entails additional computational costs. In contrast, the monolithic approach simultaneously solves the energy equation throughout the entire domain (fluid and solid) [18,19]. Such monolithic CHT solvers (e.g. `conjugateHeatFoam` in foam-extend-4.0) avoid additional predictor-corrector loops for the inter-region temperature-temperature coupling; however, iterative loops are still required to enforce the pressure-velocity and pressure-velocity-temperature couplings.

Similar to pressure-velocity coupling, the numerical challenges of the pressure-velocity-temperature coupling have attracted significant interest. Oliveira et al. [20] proposed a modified version of the PISO algorithm for buoyancy-driven incompressible fluid problems, where two momentum predictors, two pressure equations, and one temperature predictor are solved each time step. They used their method to solve a free-convection problem using the finite difference method and reported speedups of 2.1-4.1 compared to a standard SIMPLEC algorithm. In another study, Moraga et al. [21] presented the PSIMPLER algorithm for handling pressure-velocity-temperature coupling in liquid metal solidification problems. When applied to both natural and mixed convection problems, the PSIMPLER algorithm was shown to outperform the SIMPLE algorithm in terms of stability, especially for problems with strong pressure-velocity-temperature coupling. In addition, the PSIMPLER algorithm allowed the use of velocity under-relaxation values close to one, which reduced the computational cost to half that the SIMPLE solver required. In the OpenFOAM® terminology, SIMPLE is a solution algorithm for fluid problems in steady-state conditions; hence, in the authors' view, the SIMPLE algorithm in these studies refers to the PIMPLE algorithm in OpenFOAM® .

The temperature-temperature coupling between the fluid and solid regions appears in the form of two interface continuity conditions: (i) temperature continuity and (ii) heat flux continuity. How these interface conditions are enforced significantly impacts the accuracy and efficiency of CHT procedures, especially for strongly coupled problems. Various techniques have been proposed to enforce interface conditions, with differing characteristics in terms of (i) accuracy and conservation, (ii) robustness, and (iii) efficiency [16, 22–24]. He and Oldfield [25] identified disparate time scales in fluid-solid heat transfer as a challenge for unsteady CHT problems. Their proposed remedy involves a novel hybrid approach coupling time-domain fluid with frequency-domain solid conduction, enabled by a semi-analytical harmonic interface condition. Giles [26] demonstrated that in a straightforward one-dimensional model featuring constant material properties, the imposition of a Dirichlet boundary condition on the fluid side and a Neumann condition on the solid side yields greater stability compared to other boundary-type configurations. For partitioned methods, the Dirichlet-Robin coupling approach, whose optimal coupling coefficients have been the subject of several studies [23, 27], is frequently suggested to optimise the accuracy and robustness. This interface coupling technique applies fixed temperature (Dirichlet) conditions for the fluid and a linear combination of the temperature and the heat flux (Robin) conditions for the solid boundaries, where their corresponding values are updated during the predictor-corrector loop. While Dirichlet-Robin coupling is widely used, it is important to recognise a potential issue: the Dirichlet condition on the interface of the fluid region ensures a fixed temperature value. However, the Robin condition on the interface of the solid region is more flexible, permitting temperature and temperature derivative adjustments. Consequently, the method may exhibit a limitation in certain scenarios due to the lack of strict enforcement of temperature continuity and heat flux conservation. Meng et al. [16] showed that the mixed Robin-Robin approach presents attractive convergence properties, especially when adjacent materials have similar properties. Scholl et al. [24] corroborated these findings. In all variants of OpenFOAM® , the Robin-Robin boundary condition is available for the partitioned solvers. Interested readers are referred to the codes for the `solidWallMixedTemperatureCoupledFvPatchScalarField` and

`solidWallMixedTemperatureCoupledFvPatchScalarField` classes, both of which are derived from the `mixedFvPatchScalarField` class within foam-extend-4.0.

An additional concern, which can significantly affect the overall solution efficiency and robustness, is the priority assigned to each type of coupling (pressure-velocity, pressure-velocity-temperature, or temperature-temperature) when formulating the solution algorithm. To the authors' knowledge, the effect of these sequences/priorities on the computational cost of CHT solvers is not well-addressed in the literature. In particular, answering the following question is not trivial: *"Which of the following orders results in greater computational efficiency?"*:

(a) Enforcing pressure-velocity coupling within one internal loop, temperature-temperature in another internal loop, and finally, pressure-velocity-temperature via an outer loop;
(b) Enforcing pressure-velocity within an internal loop and then the pressure-velocity-temperature and temperature-temperature within the outer loop.

Approach (a), first proposed by Tandis et al. [28] and which we will refer to as the *separate loop approach*, applies a separate iterative loop for enforcing each type of coupling, while approach (b), which we will refer to as the *integrated loop approach*, enforces two types of coupling in a single outer loop. In addition to these two approaches, other combinations can be easily imagined, each potentially leading to different performance. It should be noted that algorithm SIPM-IP (the integrated loop approach) is implemented within the `chtMultiRegionFoam` solvers in all the main OpenFOAM® forks (OpenFOAM.com, OpenFOAM.org and foam-extend). In the recent OpenFOAM-v2112 release, an option has been added to `chtMultiRegionFoam`, which allows for an approach similar to the algorithm SIPM-SP.

Examining the latest versions of the main OpenFOAM® forks, several CHT solvers are found (Tab. 1). All three forks (OpenFOAM-v2112, OpenFOAM-9, and foam-extend-4.0) provide the partitioned solver, `chtMultiRegionFoam`, whereas foam-extend-4.0 is the only fork to supply a monolithic solver, `conjugateHeatFoam`. For the partitioned solver `chtMultiRegionFoam`, one difference among the three forks is the form of the energy equation, where the primary unknown may be temperature, specific internal energy, or specific enthalpy. Furthermore, in OpenFOAM-v2112, `chtMultiRegionFoam` can solve two-phase flow and an additional energy loop; the benefit of this additional temperature loop is discussed further in the test cases section. The monolithic solver in foam-extend-4.0 (`conjugateHeatFoam`) simultaneously solves the energy equation in both the solid and fluid regions and uses a PISO loop to enforce pressure-velocity coupling. As no outer iterations are performed, strict enforcement of pressure-velocity-temperature coupling is not ensured. In other words, the `conjugateHeatFoam` solver implements *loose* pressure-velocity-temperature coupling. Additionally, this solver, whose superiority of its solution algorithm over other solvers is demonstrated later, is limited to incompressible flow and lacks advanced features such as two-phase and reaction modelling.

The current article presents a detailed analysis of the CHT solvers in the main OpenFOAM® forks. In addition, modifications to the standard partitioned and monolithic solvers are proposed:

(i) The inclusion of the *separate loop approach* in the `chtMultiRegionFoam` partitioned solver, similar to the additional temperature loop that was recently added to OpenFOAM-v2112;
(i) An outer loop is added to the `conjugateHeatFoam` solver to create a strongly coupled pressure-velocity-temperature approach; additionally, the solver is generalised for multiple regions (more than one solid and one fluid).
(i) Both monolithic and partitioned (with *separate and integrated loop*) algorithms are all unified within one solver, called `multiChtFoam`, where the set-up of the case determines the solution algorithm.

The performance of the existing solvers and newly proposed ones are assessed on CHT problems with various coupling strengths. Conclusions are provided on choosing the most efficient solver, particularly for cases with multiple regions.

2. Mathematical formulation

This section reviews the mathematical formulation for a CHT problem, including the fluid and solid governing equations and interface conditions.

Table 1. Features of CHT solvers in OpenFOAM® packages

Package	Solver	Features *
foam-extend-4.0	`chtMutiRegionFoam` (`chtMutiRegionSimpleFoam`)	• Transient (steady-state), compressible$^+$ fluids • Temperature-based energy equation for solids and enthalpy-based equation for fluids • Mechanical to thermal conversion, i.e. term $\boldsymbol{\sigma} : \boldsymbol{\nabla}\mathbf{U}$, is neglected. • Arbitrary number of fluid and solid regions
	`conjugateHeatFoam` (`conjugateSimpleHeatFoam`)	• Transient (steady-state), incompressible$^+$ fluids • limited to one region for fluid and one for solid • Temperature-based energy equations for regions • Monolithic solution for energy equations
OpenFOAM-v2112	`chtMutiRegionFoam` (`chtMultiRegionSimpleFoam`)	• Transient (steady-state), compressible fluids • Enthalpy-based energy equation for solids and energy/enthalpy-based equation for fluids • Shear stress work, i.e. term $\boldsymbol{\nabla} \cdot (\boldsymbol{\tau}.\boldsymbol{\nabla}\mathbf{U})$, is neglected. • Arbitrary number of fluid and solid regions • Chemical reaction and radiation models are included
	`chtMultiRegionTwoPhaseEulerFoam`	• In addition to chtMutiRegionFoam, it applies two-phase Euler approach on the fluid region.
OpenFOAM-9	chtMutiRegionFoam	• Transient and steady-state, compressible fluids • Enthalpy-based energy equations for solid and fluid regions • Arbitrary number of fluid and solid regions • Chemical reaction is included

*All solvers include buoyancy-driven and turbulence models
+For incompressible solvers, Boussinesq approxiamtion is applied

The time-dependent equations for describing the conservation of mass, momentum and energy for a continuum are expressed in differential form as

$$\frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \mathbf{U}) = 0 \tag{1}$$

$$\frac{\partial}{\partial t}(\rho \mathbf{U}) + \boldsymbol{\nabla} \cdot (\rho \mathbf{U}\mathbf{U}) = \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} + \rho \boldsymbol{g} + \boldsymbol{S}_{\mathbf{U}} \tag{2}$$

$$\frac{\partial E}{\partial t} + \boldsymbol{\nabla} \cdot (\mathbf{U}E) = -\boldsymbol{\nabla} \cdot \boldsymbol{q} + \rho r + \boldsymbol{\nabla} \cdot (\boldsymbol{\sigma} \cdot \mathbf{U}) + \rho \boldsymbol{g} \cdot \mathbf{U} \tag{3}$$

Here, $\mathbf{U}$ is the velocity vector; p is the pressure; $\boldsymbol{\nabla}$ is the del operator; $\boldsymbol{S}_{\mathbf{U}}$ is the momentum source term; $\boldsymbol{\sigma} = \boldsymbol{\tau} - p\mathbf{I}$ is the stress tensor; $\boldsymbol{\tau}$ is shear stress tensor; $\boldsymbol{g}$ is gravitational acceleration; $E = \rho(e+K)$ is total energy; e is specific internal energy, i.e. total internal energy per mass unit; $K = \frac{\mathbf{U}\cdot\mathbf{U}}{2}$ is the kinetic energy per mass unit; r is a heat generation source per mass unit; $\boldsymbol{q}$ is the heat flux transfer due to conduction. The heat flux, $\boldsymbol{q}$, is typically given in terms of the temperature, T, by Fourier's law: $\boldsymbol{q} = -k\boldsymbol{\nabla}T$. For turbulent flows, Reynolds averaging approaches introduce a turbulent thermal conductivity to account for heat transfer due to turbulent fluctuations.

All solvers in OpenFOAM® forks, except `conjugateHeatFoam` in foam-extend-4.0, adopts a compressible approach and the density is assumed to have a linear dependence on the temperature, $\rho = \rho_0 - \rho_0\beta\,(T - T_0)$, where ρ_0 represents the fluid density at the reference temperature T_0, and β is the thermal expansion coefficient. The `conjugateHeatFoam` solver additionally applies Boussinesq approximation [29], which means that changes in density can be ignored except for when calculating the buoyancy

terms (those containing $\boldsymbol{g}$). The buoyancy-driven effects in the momentum equation (Eqn. (2)) are sometimes treated as a body force [20], i.e. $\rho\boldsymbol{g} = \rho_0\boldsymbol{g} - \rho_0\beta\boldsymbol{g}\,(T - T_0)$. Alternatively, as applied in all OpenFOAM® forks, the buoyancy-driven effect is directly added to the pressure field as $p = p_{rgh} + \rho\boldsymbol{g}\cdot\boldsymbol{z}$, where p_{rgh} is pressure field *without* the effect of the fluid weight, and $\boldsymbol{z}$ is the position vector.

Equation (3) presents the energy equation in the general and conservative form, which accounts for the mechanical work, including work done by surface and body forces (third and fourth terms on the right-hand side, respectively) and heat sources (the second term on the right-hand side). As detailed below, this equation form can be simplified to reach the three forms solved in the available OpenFOAM® solvers.

2.1. **Specific energy as the primary unknown.** To express Fourier's Law in terms of internal energy, the definition of specific heat at constant volume, denoted as $C_v = \dfrac{\partial e}{\partial T}$, is utilised to substitute $\boldsymbol{\nabla}T$ with $\dfrac{\boldsymbol{\nabla}e}{C_v}$, where C_v is generally a function of temperature and pressure. By combining Eqn. (3) with the internal-energy-based Fourier's law and replacing E with $\rho(e + K)$, we obtain:

$$\frac{\partial(\rho e)}{\partial t} + \boldsymbol{\nabla}\cdot(\rho\mathbf{U}e) + \frac{\partial(\rho K)}{\partial t} + \boldsymbol{\nabla}\cdot(\rho\mathbf{U}K) = \boldsymbol{\nabla}\cdot\left(\frac{k}{C_v}\boldsymbol{\nabla}e\right) + \rho r + \underbrace{\boldsymbol{\nabla}\cdot(\boldsymbol{\sigma}\cdot\mathbf{U})}_{-\boldsymbol{\nabla}\cdot(\mathbf{U}p)+\boldsymbol{\nabla}\cdot(\boldsymbol{\tau}\cdot\mathbf{U})} + \rho\boldsymbol{g}\cdot\mathbf{U} \tag{4}$$

Neglecting $\boldsymbol{\nabla}\cdot(\boldsymbol{\tau}\cdot\mathbf{U})$ term from Eqn. (4) gives the final form of the energy equation which is solved in OpenFOAM-v2112 and OpenFOAM-9. In both of these OpenFOAM® forks, the effect of chemical reactions and additional user-defined source terms are modelled through the term ρr. In OpenFOAM-v2112, the radiation effect is also incorporated in the energy equation via this term.

Once the specific energy has been computed, the temperature field is determined based on the fluid's definition and using various thermophysical libraries. Readers interested in further details are encouraged to review the code of `ePsiThermo<MixtureType>::calculate()` within foam-extend-4.0 as a reference.

In addition to this form, an alternative form can be obtained, where the inner product of the momentum equation (Eqn. (2)) with $\mathbf{U}$ is used to cancel out terms related to mechanical energy and kinetic energy in Eqn. (4):

$$\frac{\partial(\rho e)}{\partial t} + \boldsymbol{\nabla}\cdot(\rho\mathbf{U}e) = \boldsymbol{\nabla}\cdot\left(\frac{k}{C_v}\boldsymbol{\nabla}e\right) + \rho r + \underbrace{\boldsymbol{\sigma}:\boldsymbol{\nabla}\mathbf{U}}_{-p\boldsymbol{\nabla}\cdot\mathbf{U}+\boldsymbol{\tau}:\boldsymbol{\nabla}\mathbf{U}} + \rho\boldsymbol{g}\cdot\mathbf{U} \tag{5}$$

This form is later used for deriving an enthalpy-based equation in foam-extend-4.0. The term $\boldsymbol{\sigma}:\boldsymbol{\nabla}\mathbf{U}$ represents work done by surface forces: part of this work $p\boldsymbol{\nabla}\cdot\mathbf{U}$ due to compression is reversible; the viscous component $\boldsymbol{\tau}:\boldsymbol{\nabla}\mathbf{U}$ represents an irreversible mechanical-to-thermal conversion due to the action of shear forces transformed into heat.

2.2. **Specific enthalpy as the primary unknown.** Equation (4) can be rewritten in terms of specific enthalpy h by using the expression $e = h - \dfrac{p}{\rho}$. Additionally, the definition of specific heat at a constant pressure $C_p = \dfrac{\partial h}{\partial T}$ is utilised to substitute temperature with enthalpy in the conduction term

$$\underbrace{\frac{\partial(\rho e)}{\partial t}}_{\frac{\partial(\rho h)}{\partial t}-\frac{\partial p}{\partial t}} + \underbrace{\boldsymbol{\nabla}\cdot(\rho\mathbf{U}e)}_{\boldsymbol{\nabla}\cdot(\rho\mathbf{U}h)-\boldsymbol{\nabla}\cdot(\mathbf{U}p)} + \frac{\partial(\rho K)}{\partial t} + \boldsymbol{\nabla}\cdot(\rho\mathbf{U}K) = \boldsymbol{\nabla}\cdot\left(\frac{k}{C_p}\boldsymbol{\nabla}h\right) + \rho r + \underbrace{\boldsymbol{\nabla}\cdot(\boldsymbol{\sigma}\cdot\mathbf{U})}_{-\boldsymbol{\nabla}\cdot(\mathbf{U}p)+\boldsymbol{\nabla}\cdot(\boldsymbol{\tau}\cdot\mathbf{U})} + \rho\boldsymbol{g}\cdot\mathbf{U} \tag{6}$$

Canceling out the term $-\boldsymbol{\nabla}\cdot(\mathbf{U}p)$ from both sides gives:

$$\frac{\partial(\rho h)}{\partial t} + \boldsymbol{\nabla}\cdot(\rho\mathbf{U}h) + \frac{\partial(\rho K)}{\partial t} + \boldsymbol{\nabla}\cdot(\rho\mathbf{U}K) = \boldsymbol{\nabla}\cdot\left(\frac{k}{C_p}\boldsymbol{\nabla}h\right) + \rho r + \boldsymbol{\nabla}\cdot(\boldsymbol{\tau}\cdot\mathbf{U}) + \rho\boldsymbol{g}\cdot\mathbf{U} + \frac{\partial p}{\partial t} \tag{7}$$

Equation (7) is used in OpenFOAM-v2112 and OpenFOAM-9, where the shear stress term is dropped again. Alternatively, if we replace $e = h - \dfrac{p}{\rho}$ within Eqn (5), we arrive at another form for the specific enthalpy equation:

$$\frac{\partial(\rho h)}{\partial t} + \boldsymbol{\nabla}\cdot(\rho\mathbf{U}h) = \boldsymbol{\nabla}\cdot\left(\frac{k}{C_p}\boldsymbol{\nabla}h\right) + \rho r + \tau:\boldsymbol{\nabla}\mathbf{U} + \rho\boldsymbol{g}\cdot\mathbf{U} + \frac{Dp}{Dt} \tag{8}$$

Equation (8) is used in the `chtMultiRegionFoam` solver in foam-extend-4.0, where surface work (including viscous heating), body force, and heat source terms are dropped. Also, chemical reactions and radiation are not available in this solver version. It should be noted that $\frac{D}{Dt}$ in Eqn. (8) denotes the material

derivative, which is different from the temporal derivative in the Eulerian approach, i.e. $\frac{\partial}{\partial t}$ in Eqn. (7). In foam-extend-4.0, the material derivative is explicitly evaluated via $\frac{Dp}{Dt} = \frac{\partial p}{\partial t} + \mathbf{U} \cdot \boldsymbol{\nabla} p$, where $\mathbf{U}$ is the advection velocity.

After solving for the specific enthalpy, the temperature field is computed based on the fluid type and using a variety of thermophysical libraries in OpenFOAM® .

2.3. **Temperature as the primary unknown.** By substituting $\partial e = C_v \partial T$ in Eqn. (5), the energy equation can be written in terms of temperature as

$$\frac{\partial(\rho C_v T)}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \mathbf{U} C_v T) = \boldsymbol{\nabla} \cdot (k \boldsymbol{\nabla} T) + \rho r - p \boldsymbol{\nabla} \cdot \mathbf{U} + \boldsymbol{\tau} : \boldsymbol{\nabla} \mathbf{U} + \rho \boldsymbol{g} \cdot \mathbf{U} \tag{9}$$

The solution of Eqn. (9) gives the temperature directly. This equation is used in the monolithic solver (`conjugateHeatFoam`) of foam-extend-4.0, where the three last terms on the right-hand side (work done by surface and body forces) are dropped, and the radiation is modelled via a heat source term, i.e. ρr.

As mentioned, Eqns. (4), (7), (8), and (9) are employed in heat transfer solvers in different versions of OpenFOAM® . As detailed above, all OpenFOAM® forks neglect the viscous heating effect (heat dissipation generated by shear stress term). However, this term plays a major role in various applications, including lubrication and tribology [30, 31] and compressible flow [32], and one should be careful when applying these solvers for problems with a considerable contribution of the viscous heating phenomena. Additionally, both `conjugateHeatFoam` and `chtMultiRegionFoam` solvers in foam-extend-4.0 neglect the reversible work ($p\boldsymbol{\nabla} \cdot \mathbf{U}$), which can be considerable for compressible flow with high compression; it should be noted that the solvers in OpenFOAM-v2112 and OpenFOAM-9 accounts for $p\boldsymbol{\nabla} \cdot \mathbf{U}$ term.

2.4. **Energy equation for solid region.** For the energy equation in solid regions, these equations are simplified by neglecting $\mathbf{U}$ and p terms. For instance, in `conjugateHeatFoam`, it becomes

$$\frac{\partial}{\partial t}(\rho_s C_s T_s) = \boldsymbol{\nabla} \cdot (k_s \boldsymbol{\nabla} T_s) + \rho_s r_s \tag{10}$$

where subscript s is used to indicate quantities in the solid domain.

2.5. **Governing equation for Boussinesq flow.** For the rest in this paper and to make a fair comparison between all solution algorithms in OpenFOAM® forks, the fluid is assumed to be incompressible, Newtonian, and laminar with the Boussinesq approximation, and the heat capacity is constant; additionally, the temperature-based formulation is applied for the energy equation in fluid and solid regions. According to first three assumptions, we have $\boldsymbol{\tau} = \mu(\boldsymbol{\nabla}\mathbf{U} + \boldsymbol{\nabla}\mathbf{U}^T)$, where μ is viscosity. Furthermore, the terms for heat source, i.e. ρr, and works done by surface force, i.e. $-p\boldsymbol{\nabla} \cdot \mathbf{U} + \boldsymbol{\tau} : \boldsymbol{\nabla}\mathbf{U}$, and body force, i.e. $\rho \boldsymbol{g} \cdot \mathbf{U}$, are neglected in the energy equation. Dividing the momentum equation, i.e. Eqn. (2), by ρ_0 (as is common for incompressible flow), the final flow equations are:

$$\boldsymbol{\nabla} \cdot \mathbf{U} = 0 \tag{11}$$

$$\frac{\partial \mathbf{U}}{\partial t} + \boldsymbol{\nabla} \cdot (\mathbf{U}\mathbf{U}) = -\boldsymbol{\nabla} \left\{ \frac{p_{rgh}}{\rho_0} + \left[1.0 - \beta \left(T - T_0\right)\right] \boldsymbol{g} \cdot \boldsymbol{z} \right\} + \boldsymbol{\nabla} \cdot (\nu \boldsymbol{\nabla} \mathbf{U}) \tag{12}$$

$$\frac{\partial T}{\partial t} + \boldsymbol{\nabla} \cdot (\mathbf{U} T) = \boldsymbol{\nabla} \cdot (\alpha \boldsymbol{\nabla} T) \tag{13}$$

where $\alpha = \dfrac{k}{\rho C_p}$ is the thermal diffusivity and $\nu = \dfrac{\mu}{\rho_0}$ is the kinematic viscosity. It should be noted for incompressible flow, $\frac{p_{rgh}}{\rho_0}$ in Eqn. (12) is treated as a kinematic pressure field with units $[m^2/s^2]$, which we refer to here as p_r.

According to the interface continuity conditions, the temperature and normal component of the heat flux must be continuous at the shared interface, indicated by i, as follows

$$(T_i)_f = (T_i)_s \tag{14}$$

$$\boldsymbol{n_i} \cdot (k_i \boldsymbol{\nabla} T_i)_f = \boldsymbol{n_i} \cdot (k_i \boldsymbol{\nabla} T_i)_s \tag{15}$$

Where $\boldsymbol{n_i}$ denotes the outward normal vector at the interface. Eqns. (14) and (15), along with Eqns. (10) to (13) state that the pressure, velocity and temperature in the fluid regions and temperature across solid and fluid regions are coupled. By expressing the momentum and energy equations in dimensionless

form, several dimensionless quantities appear, which describe the strength of each type of coupling:

$$\frac{\widetilde{\mathbf{U}}}{d\widetilde{t}} + \widetilde{\mathbf{U}} \cdot \boldsymbol{\nabla}\widetilde{\mathbf{U}} = -\boldsymbol{\nabla}\widetilde{p} + \frac{Gr}{Re^2}\widetilde{T} \tag{16}$$

$$\frac{\widetilde{T}}{d\widetilde{t}} + \widetilde{\mathbf{U}} \cdot \boldsymbol{\nabla} \cdot \widetilde{T} = \frac{1}{RePr}\boldsymbol{\nabla}^2\widetilde{T} \tag{17}$$

According to these dimensionless equations, the coupling degree depends on three dimensionless numbers:

- Grashof number: $Gr = \dfrac{|\boldsymbol{g}|\beta(T_S - T_\infty)}{\nu^2}$;
- Prandtl number: $Pr = \dfrac{\nu}{\alpha}$;
- Reynolds number: $Re = \dfrac{UL}{\nu}$.

where L is a characteristic length; β is the coefficient of thermal expansion of the fluid; U is a characteristic flow speed; T_S is a characteristic temperature; T_∞ is bulk temperature. The larger the value of Gr, the stronger the coupling between momentum and energy equations (pUT). In addition, the dependency of the momentum equation on the temperature (and, hence, the strength of the pressure-velocity-temperature coupling) decreases as the Reynolds number increases.

The dimensionless quantity for describing the strength of the temperature-temperature coupling in two adjacent regions, 1 and 2, is

$$\sigma = \frac{k_1/\sqrt{\alpha_1}}{k_2/\sqrt{\alpha_2}} \tag{18}$$

For $\sigma << 1$, region 2 undergoes a small temperature variation from the initial condition. Here, we assume the region with larger values of $k_2/\sqrt{\alpha_2}$ is on the denominator; as a result, $0 < \sigma < 1$. Therefore, as σ approaches 1.0, the variation in both adjacent regions becomes considerable, and the coupling strength increases [18].

3. Numerical approaches

This section discusses the numerical solution procedure in OpenFOAM® CHT solvers, including classic partitioned and monolithic solvers and the proposed modifications. This procedure includes discretising the computational domain and the governing equations, treatment of the interface condition in the partitioned and monolithic approaches, the solution algorithms for solving the discrete equations with various couplings, and finally, the convergence criteria for each time step.

3.1. **Discretisation.** All of the OpenFOAM® CHT solvers apply the finite volume method to enforce the conservation of physical quantities in all sub-domains and at the shared interfaces. The computational domain is discretised into a set of cells, as depicted in Fig. 2. Here, P refers to the cell centre, and N is a neighbouring cell centre; $\boldsymbol{d_f}$ is a vector connecting P to N; $\boldsymbol{s}_f$ is the outward-pointing area vector for the face f shared between P and N.

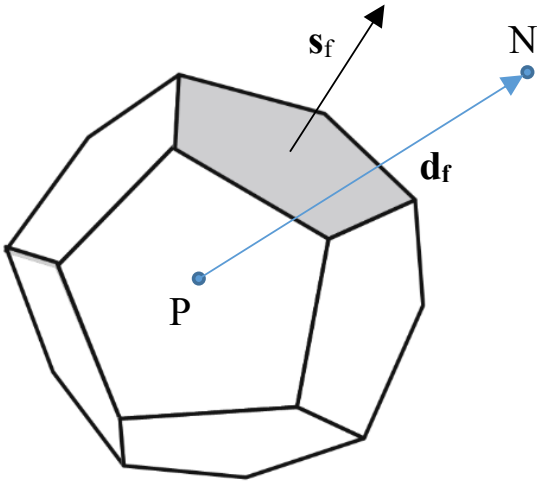


Figure 2. Control volume P, its neighbour N, and the relevant cell-face parameters

By following the approach in [12] and incorporating the buoyant force as a source term, the semi-discretised form of the momentum equation for the fluid regions can be expressed as

$$\mathbf{U}_P = \frac{\boldsymbol{H}(\mathbf{U})}{a_P} - \frac{1}{a_P}\boldsymbol{\nabla}\left[1.0 - \beta\left(T - T_0\right)\right]\boldsymbol{g}\cdot\boldsymbol{z} - \frac{1}{a_P}\boldsymbol{\nabla}p_r \tag{19}$$

where a_P represents the diagonal coefficient for cell P, and operator $\boldsymbol{H}(\mathbf{U})$ is evaluated from the transient term and the coefficients for neighbour cells multiplied by their newly calculated velocities:

$$\boldsymbol{H}(\mathbf{U}) = \sum_{N=0}^{N_b} a_N \mathbf{U}_N + \frac{\mathbf{U}^0}{\Delta t} \tag{20}$$

Here, N_b refers to the number of neighbours around cell P; Δt indicates the time step size, and $\mathbf{U}^0$ denotes the velocity at cell P in the previous (old) time step. The Euler scheme has been used here to discretise the temporal term; however, higher-order schemes could also be used. Similarly, coefficients a_P and a_N depend on the temporal and spatial discretisation schemes. In this study, the Gauss linear scheme [33] is used to discretise the gradient of the pressure and velocity fields; the Gauss linear corrected scheme [34] is used for discretising the diffusion term; and the Gauss upwind scheme is used for discretising the convection terms. Combining Eqns. (19) and (11) leads to pressure equation:

$$\boldsymbol{\nabla}\cdot\left(\frac{1}{a_P}\boldsymbol{\nabla}p_r\right) = \boldsymbol{\nabla}\cdot\left(\frac{\boldsymbol{H}(\mathbf{U})}{a_P} - \frac{1}{a_P}\boldsymbol{\nabla}\left[1.0 - \beta\left(T - T_0\right)\right]\boldsymbol{g}\cdot\boldsymbol{z}\right) \tag{21}$$

In the PIMPLE (combination of PISO and SIMPLE) algorithm, solving the pressure equation is followed by correcting $\mathbf{U}_P$ via Eqn. (19) and then evaluating the mass flux at the cell faces, denoted by F, via the below equation:

$$F = \rho_f \boldsymbol{s_f}\cdot\mathbf{U}_f = \rho_f \boldsymbol{s_f}\cdot\left(\frac{\boldsymbol{H}(\mathbf{U})}{a_P} - \frac{1}{a_P}\boldsymbol{\nabla}\left[1.0 - \beta\left(T - T_0\right)\right]\boldsymbol{g}\cdot\boldsymbol{z} - \frac{1}{a_P}\boldsymbol{\nabla}p\right)_f \tag{22}$$

It should be noted that Eqn. (22) applies Rhie and Chow interpolation, as explained in [12], in the context of OpenFOAM [35] to ensure effective pressure-velocity coupling. This interpolation method is employed to compute the velocity at the cell face and, consequently, the corresponding mass flux.

For the energy equation in both fluid and solid regions, the discretisation process leads to a system of algebraic equations:

$$A_P T_P + \sum_{N=0}^{N_b} A_N T_N = B_P \tag{23}$$

where A_P and A_N are coefficients of the unknown temperature at cell P and its neighbours, and B_P represents the source terms, the effect of boundary conditions on discrete equations in the adjacent cells, and other terms related to the skewness of the mesh.

3.2. **Enforcement of interface conditions.** The distinguishing part of the partitioned and monolithic algorithms is how they implement the interface conditions at shared boundaries, i.e. Eqns. (14) and (15). The partitioned methods adopt an explicit strategy for exchanging the information associated with the interface condition, meaning that the boundary conditions are updated after solving the energy equation in each region. In the monolithic approach, the simultaneous solution of the energy equation in all regions implies that the information is exchanged implicitly; this requires special care when constructing the equations associated with cells adjacent to the interface. In this section, we provide details on the standard implementation of the interface conditions used for the partitioned and monolithic solvers in the OpenFOAM® packages.

A schematic of an interface face and its adjacent cells are shown in Fig. 3. Here, subscripts P and N refer to the cells at the owner and neighbour sides of the interface face, respectively. In this context, the owner is the region for which the solution is currently being performed and the neighbour refers to the opposite side of the owner.

According to Patanakar [36], the discretised form of Eqn. (15) can be used to evaluate effective conductivity at the interface in such a way as to ensure energy conservation at the interface. The discrete form is expressed as below:

$$k_i\frac{T_P - T_N}{\Delta} = k_N\frac{T_i - T_N}{\Delta_N} = k_P\frac{T_P - T_i}{\Delta_P} \tag{24}$$

where subscript i refers to the interface face; k_i is the effective conductivity at the interface face; T_P (T_N) is the temperature at the adjacent owner (neighbour) cell; T_i is the temperature at the interface face; Δ_P (Δ_N) is the perpendicular distance between the interface face and the adjacent cell in the owner

(neighbour) region, and $\Delta = \Delta_P + \Delta_N$ is the distance between the centres of cells at both sides of the interface projected on the normal vector of their shared face. It should be noted that Eqn. (24) is derived

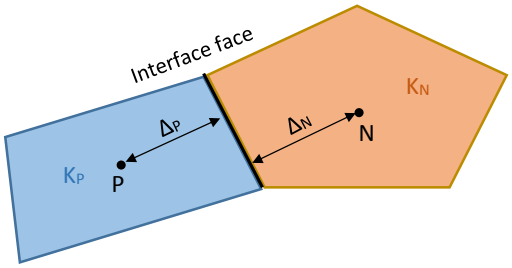


Figure 3. Owner (blue) and neighbour (orange) cells at both sides of the shared interface faces

in the absence of radiation and other source terms at the interface. The temperature continuity condition also implies that T_i must be identical on both interface sides. From Eqn. (24), k_i and T_i can be computed by

$$T_i = \frac{\frac{k_P}{\Delta_P} T_P + \frac{k_N}{\Delta_N} T_N}{\frac{k_P}{\Delta_P} + \frac{k_N}{\Delta_N}} \tag{25}$$

$$k_i = \frac{k_P k_N}{\frac{\Delta_P}{\Delta} k_N + \frac{\Delta_N}{\Delta} k_P} \tag{26}$$

For the partitioned methods in this study, Eqn. (25) is enforced on both sides of the interface using a mixed Robin-type boundary condition. Robin conditions use a linear combination of the Dirichlet (`fixedValue`) and Neumann (`fixedGradient`) boundary types. As a result, the temperature at the boundary of the owner's side $(T_i)_P$ is assigned as

$$(T_i)_P = \underbrace{\omega T_R}_{\text{Dirichlet contribution}} + \underbrace{(1-\omega)\left[T_P + (\boldsymbol{n}\cdot\boldsymbol{\nabla}T)_R\,\Delta_P\right]}_{\text{Neumann contribution}} \tag{27}$$

where ω is the weight of the Dirichlet's contribution to the Robin boundary condition, and T_R and $(n\cdot\boldsymbol{\nabla}T)_R$ are the reference temperature and the reference temperature gradient associated with the Dirichlet and Neumann contributions. In OpenFOAM® , this type of Robin condition is implemented using a combination of `fixedValue` and `fixedGradient` conditions, whose weights, ω and $1-\omega$, are defined in such a way to give the interface temperature according to Eqn. (25). Choosing $T_R = T_N$ and combining Eqns. (25) and (27), the weight ω is computed as

$$\omega = \frac{\frac{k_N}{\Delta_N}}{\frac{k_P}{\Delta_P} + \frac{k_N}{\Delta_N}} \tag{28}$$

Equations (27) and (28) are applied at both sides of the interface, where subscript P refers to the owner region that the equations are currently being solved for, and subscript N refers to the neighbour side. Considering a fluid-solid interface, when these equations are applied for the fluid side: $T_P = T_f$, $T_R = T_s$, $\Delta_P = \Delta_f$; and if they applied for the solid side: $T_P = T_s$, $T_R = T_f$, $\Delta_P = \Delta_s$. Applying Eqn. (27) with the weights computed from Eqn. (28) ensures the interface conditions are satisfied in a deferred correction manner in the partitioned methods.

For the monolithic solver, the interface is considered an internal boundary whose material properties are defined by Eqn. (26) to guarantee the satisfaction of the interface conditions, i.e. Eqns. (14) and (15). This definition of k_i ensures enforcement of energy conservation and temperature continuity at the interface. It should be noted that for the monolithic approach, the normal temperature gradient at the interface is evaluated using temperature values on both sides. For an orthogonal mesh, for example, $(\boldsymbol{n_i}\cdot\boldsymbol{\nabla}T_i)_P = (\boldsymbol{n_i}\cdot\boldsymbol{\nabla}T_i)_N = \frac{T_P - T_N}{\Delta}$. While the equation shown is simplified for clarity, OpenFOAM applies

non-orthogonal corrections to the interface conditions, treating interface faces as internal faces. The non-orthogonality contribution depends on the chosen numerical scheme. In contrast, for the partitioned approaches, the normal temperature gradient is evaluated using the interface value, i.e. $(\boldsymbol{n_i} \cdot \boldsymbol{\nabla} T_i)_P = \frac{T_P - T_i}{\Delta_P}$ and $(\boldsymbol{n_i} \cdot \boldsymbol{\nabla} T_i)_N = \frac{T_i - T_N}{\Delta_N}$. For the computation of the Gauss gradient at the adjacent cells, both monolithic and partitioned approaches use the values at their region and the interface.

3.3. **Solution algorithms.** Enforcing the pressure-velocity, pressure-velocity-temperature, and temperature-temperature couplings can be carried out in various ways. For each time step within the current OpenFOAM® monolithic solver (`conjugateHeatFoam`), the pressure-velocity coupling is implemented within the PISO loop and temperature-temperature is enforced by the simultaneous solution of the energy equation for all fluid and solid regions. The pressure-velocity-temperature coupling is also enforced within an additional outer iterative loop in the proposed monolithic solver. For the partitioned methods, OpenFOAM® solvers adopt an integrated loop strategy, i.e. algorithms SIPM-IP in Section 1, whereas the proposed version applies a separated loop strategy, i.e. algorithms SIPM-SP in Section 1. Fig. 4 illustrates the solution procedure for each time step within the current partitioned, proposed partitioned, and proposed monolithic solvers.

As seen in Fig. 4, all three algorithms employ the same PISO loop to apply pressure-velocity coupling for fluid regions. The PISO algorithm is defined to achieve a specific initial residual for pressure equations in all regions (here 10^{-1}) while enforcing a certain number of iterations (here 2). The proposed partitioned algorithm, i.e. semi-implicit projection method with separated loop (SIPM-SP), enforces the pressure-velocity-temperature coupling within an outer predictor-corrector loop and applies an additional internal (interface) loop to ensure interface conditions at the fluid-solid interface for each outer iteration. The interface loop terminates once either the maximum relative temperature difference at both sides of all interfaces, denoted as $max[\overline{\Delta T_{int}}]$, or the average residual for the energy equation in all regions $mean[Res_T]_l$ reduces to a certain level (here 10^{-6}). Here, subscript l indicates the last iteration within the interface loop; $\overline{\Delta T_{int}} = \frac{(T_i)_P - (T_i)_N}{\Delta T_{max}}$, where ΔT_{max} is the maximum temperature variation across all regions. The classic partitioned algorithm, i.e. semi-implicit projection method with integrated loop (SIPM-IP), does not employ a separate loop for fluid-solid interface and, hence, enforces the pressure-velocity-temperature and temperature-temperature couplings within the outer predictor-corrector loop. In other words, the classic partitioned solver is the same as that proposed one with a single iteration for the interface loop. For the developed monolithic algorithm, semi-implicit projection method with monolithic solution (SIPM-M), the interface loop in the proposed partitioned algorithm is replaced with building and simultaneously solving the overall energy equation across all regions. Building the energy equation in the monolithic algorithm gives the benefit of directly evaluating the overall energy residual by substituting the latest computed temperature fields. Therefore, besides implicit information exchange, this algorithm facilitates the definition of the overall initial energy residual for the convergence criteria.

The classic partitioned and monolithic algorithms are available in the foam-extend-4.0 package (namely `chtMultiRegionFoam` and `conjugateHeatFoam`, respectively). For the proposed monolithic algorithm, `conjugateHeatFoam` was modified to ensure pressure-velocity-temperature coupling by adding an outer iterative loop. Moreover, the modified version can deal with multiple regions (more than two regions, to which the standard version is limited). The proposed partitioned algorithm is implemented via a few modifications (e.g. adding a predictor-corrector interface loop) within the `chtMultiRegionFoam` solver. Section 4 discusses the effects of these modifications on the solvers' accuracy and performance.

3.4. **Convergence criteria.** The current OpenFOAM® CHT solvers apply a predetermined number of iterations for their outer loop; hence, the user should monitor the residuals to ensure the desired convergence. For a fair comparison, the algorithms for the current solvers in OpenFOAM® are modified to apply similar convergence criteria. The proposed partitioned and monolithic algorithms enforce convergence criteria for each time step to ensure an accurate solution of the mass, momentum and energy equations. These criteria ensure that the initial residual of the mass (or pressure) and momentum (or velocity) equations in all fluid regions and that of the energy equations in all fluid and solid regions reach a certain level (here 10^{-3} for the pressure and velocity equations and 10^{-6} for the energy equations). Despite these similarities in the convergence criteria, the partitioned methods, i.e. algorithms SIPM-IP and SIPM-SP, and monolithic method, i.e. algorithm SIPM-M, apply different definitions for the overall initial residual of the energy equations. Specifically, the partitioned methods use an average of the initial residual in all regions, whereas the monolithic approach uses the initial residual of the overall system of energy equations. This difference in evaluating the initial residual is due to the nature of partitioned and monolithic approaches. It underlies different convergence behaviour (and sometimes unnecessary

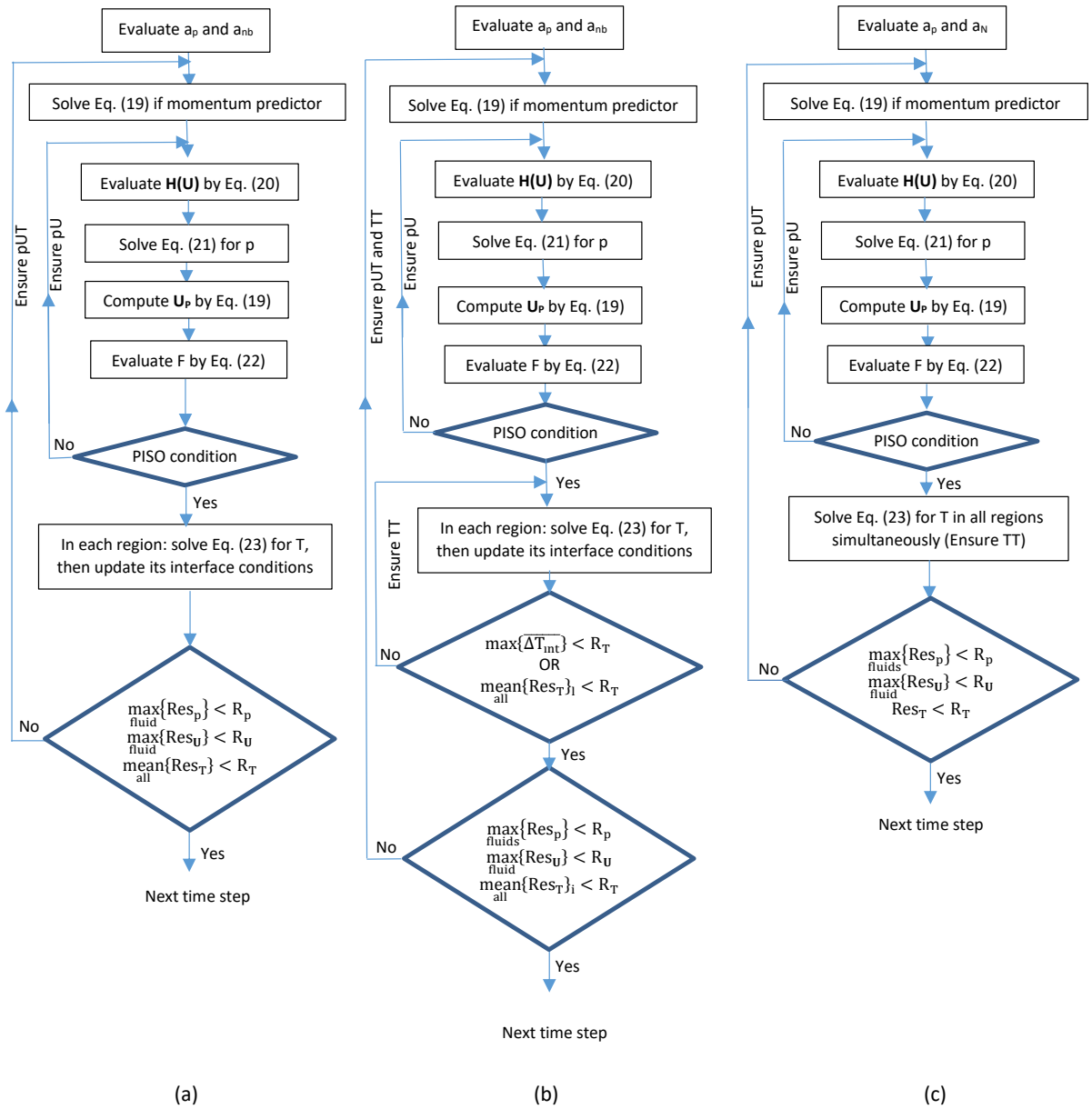


Figure 4. Flowchart of the algorithms for each time step (a) classic partitioned algorithm, SIPM-IP (b) proposed partitioned, SIPM-SP (c) proposed monolithic algorithm, SIPM-M

additional outer iterations for the partitioned methods), as demonstrated later (Section 4). To alleviate the issue of unnecessary outer iterations, we define an alternative energy convergence criteria based on which the relative temperature changes in two sequential outer iterations. It should be noted that the proposed partitioned algorithm uses the overall initial energy residual evaluated in the first (and not final) iteration of the *interface loop* to check the time step convergence criteria.

4. **Test cases**

The performance of the classic partitioned, the proposed partitioned, and monolithic algorithms are compared by solving an unsteady conjugate natural convection problem in a square enclosure. Many researchers have simulated this test case due to its simplicity and capability to investigate the effect of two types of coupling (pressure-velocity-temperature and temperature-temperature) via changing the corresponding dimensionless parameters. Kazemi-Kamyab et al. [37] solved this case using their proposed loosely-coupled partitioned algorithm to demonstrate their solver's applicability for two different assumptions: constant material properties and temperature-dependent thermal conductivities in the fluid and solid sub-domains. In another study [18], the authors examined the stability and convergence rate

of their proposed strongly-coupled partitioned solver through a numerical solution of this case for different coupling strengths via changing σ and Gr. Pan et al. [38] validated the accuracy of their proposed monolithic solver through this case for various Gr and σ.

In the current study, the accuracy of three algorithms is validated against available results in the literature. Then, the benefits of the proposed monolithic solver against the standard solver in the foam-extend package (the `conjugateHeatFoam` solver) are examined. Four new coupling strengths are suggested and solved by classic partitioned, proposed partitioned, and monolithic algorithms to cover various coupling conditions for a fair comparison. Finally, a new configuration of this test case with additional fluid/solid regions is presented to examine the possible effects of additional regions on the performance. Adding more regions to the computational domain imposes a challenge, especially for the partitioned methods. The partitioned strategy for solving (or explicit exchange of the information between) multiple systems of energy equations might encounter issues such as instability or slow convergence. Therefore, such a new test case is expected to provide an opportunity to compare monolithic and partitioned approaches more deeply. For all test cases in this study, the same discretisation setup is applied unless otherwise stated: the time step size is $\Delta t = 0.00025$ s, and the dimensions of the grids are 80×80 cells for the fluid regions and 80×20 cells for the solid regions.

4.1. **Classical case of unsteady conjugate natural convection in a single square enclosure.** The standard case of unsteady conjugate natural convection in a single square enclosure is depicted in Fig. 5. For verification purposes, the material properties are taken from the study by Pan et al. [38]. Tab.

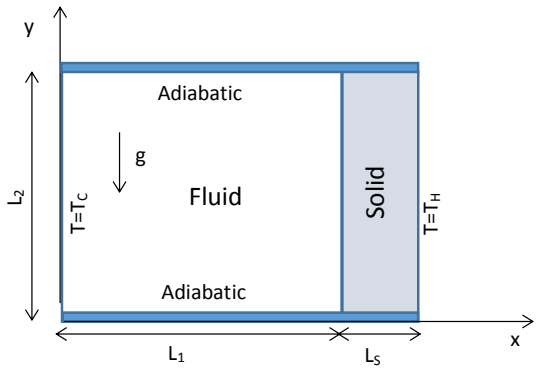


Figure 5. Conjugate natural convection in a single square enclosure

2 presents dimensionless quantities for these states. As the parameters Grashof and σ in this table show, the pUT-TT couplings in states 1, 2, and 3 are strong-weak, weak-weak, and weak-strong, respectively. It should be noted that the material properties in this simulation are not physical but were set to reproduce the ratio between the material properties of the fluid-solid pair and the Grashof number in the fluid.

Table 2. Dimensionless properties for three coupling conditions in the test case of unsteady conjugate natural convection in a single square enclosure

State	Fluid	Solid	Pr	k_s/k_f	Gr	σ
1	Air	Steel	0.7	1600	1.43×10^5	3.95×10^{-4}
2	Water	Steel	7.0	80	10^4	0.12
3	Water	Concrete	7.0	2.7	10^4	0.93

In these simulations, we set $L_1 = L_2 = 1$ m, and the wall thickness $L_S = 0.2$ m. The fluid is initially at rest with $p = 0$ Pa and $T = 0^o$ C. Then it starts to be heated by the thermal energy transferred from the outer surface of the right thick wall with an initial temperature of 0^o C. The temperature on the right side of the wall is fixed at $T_H = 1^o$ C, and the temperature on the left side of the enclosure is fixed at $T_C = 0^o$ C. The no-slip boundary condition is applied to all walls of the fluid regions.

Figure 6 presents streamlines and contours of temperature in the fluid and solid regions at $t = 0.07$ s.

Figure 7 compares the temperature profile at the horizontal mid-line for $t = 0.07$ s obtained by the available and proposed solvers with the results in the literature [38]. As seen in this figure, all three solvers give results with acceptable accuracy for states 1, 2 and 3.

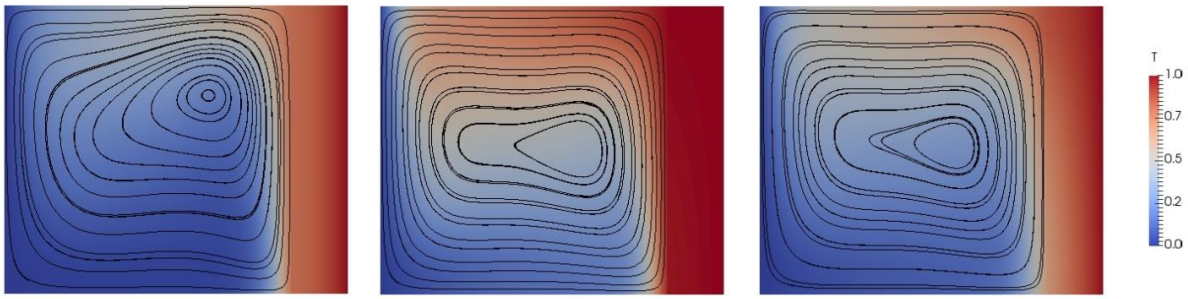
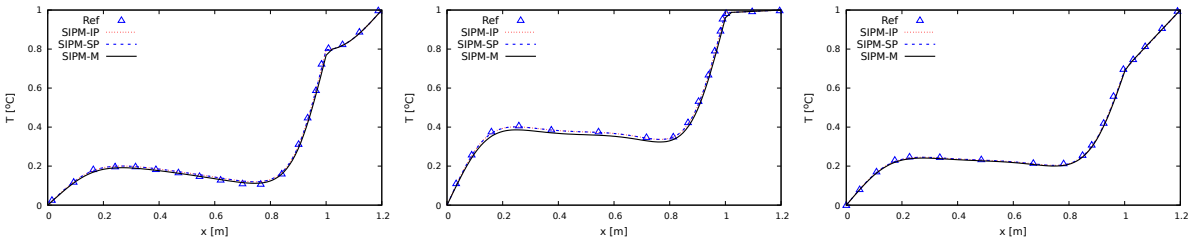


Figure 6. Contour of temperature and streamline for three states of the unsteady conjugate natural convection in a single square enclosure at $t = 0.07$ s



(a) State 1 **(b)** State 2 **(c)** State 3

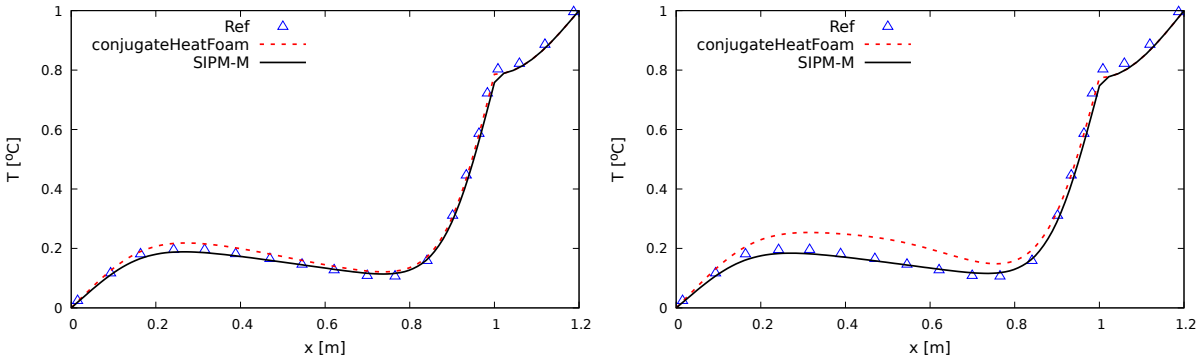
Figure 7. Temperature profile for three states of the unsteady conjugate natural convection in a single square enclosure at $y = 0.5$ m and $t = 0.07$ s given by three solvers (SIPM-IP or the classic partitioned, SIPM-SP or the proposed partitioned, SIPM-M or the proposed monolithic) and literature [38].

As mentioned earlier, the `conjugateHeatFoam` solver in the foam-extend-4.0 package applies a monolithic approach similar to the proposed monolithic algorithm, except it lacks the outer loop for ensuring the pressure-velocity-temperature coupling in the fluid regions. To demonstrate the effect of this loop on the accuracy of the results, state 1, whose pressure-velocity-temperature coupling is strong, is solved by both the `conjugateHeatFoam` and the proposed monolithic solvers with two different time step sizes: $\Delta t = 0.0025$ s, $\Delta t = 0.005$ s. Figure 8 compares the temperature profile at the horizontal midline for $t = 0.07$ s obtained by `conjugateHeatFoam` and proposed monolithic solvers with results in the literature [38]. As is clear, the accuracy of the `conjugateHeatFoam` solver is lower than the proposed monolithic solver, and it reduces for the larger time steps due to not ensuring pressure-velocity-temperature coupling within the fluid region. On the other hand, the proposed monolithic solver gives results that agree with the reference. In the authors' view, the slight difference for the larger time step size is only due to the temporal discretisation error.

For the `conjugateHeatFoam` solver, the default number of PISO iterations in the standard OpenFOAM® tutorials, two iterations, will lead to solution instability. So, it is crucial to ensure a partial convergence of the PISO loop before solving temperature equations to avoid instability for the proposed monolithic algorithm. For the stability of the conjugateHeatFoam, we have increased the number of PISO iterations by trial and error, as reported below.

For comparing the computational cost of three CHT solvers, we suggest using new material properties, given in Tab. 3, to cover different levels of pUT-TT coupling for this case: strong-strong, weak-strong, strong-weak, and weak-weak. Note that the first refers to pressure-velocity-temperature coupling in the fluid regions, and the second represents the temperature-temperature coupling between two adjacent regions.

Computational costs of the proposed algorithms for solving states A to D are compared in Fig. 9. This figure shows that classic partitioned (SIPM-IP) produces the slowest convergence rate, especially in states A and B, where strong thermal interaction exists between fluid and solid regions. Comparing the results for classic partitioned (SIPM-IP) and proposed partitioned (SIPM-SP) reveals the latter's superiority in all coupling conditions up to about two times speedup. In addition, for all coupling conditions, especially A and B, the proposed monolithic (SIPM-M) results in less computational time than other algorithms. However, as the results suggest, the proposed partitioned algorithm presents



(a) $\Delta t = 0.0025$ s **(b)** $\Delta t = 0.005$ s

Figure 8. Temperature profile for state 1 of the unsteady conjugate natural convection in a single square enclosure at $y = 0.5$ m and $t = 0.07$ s given by proposed monolithic (SIPM-M), OpenFOAM® (conjugateHeatFoam), and literature

Table 3. Dimensionless properties for four of the proposed states

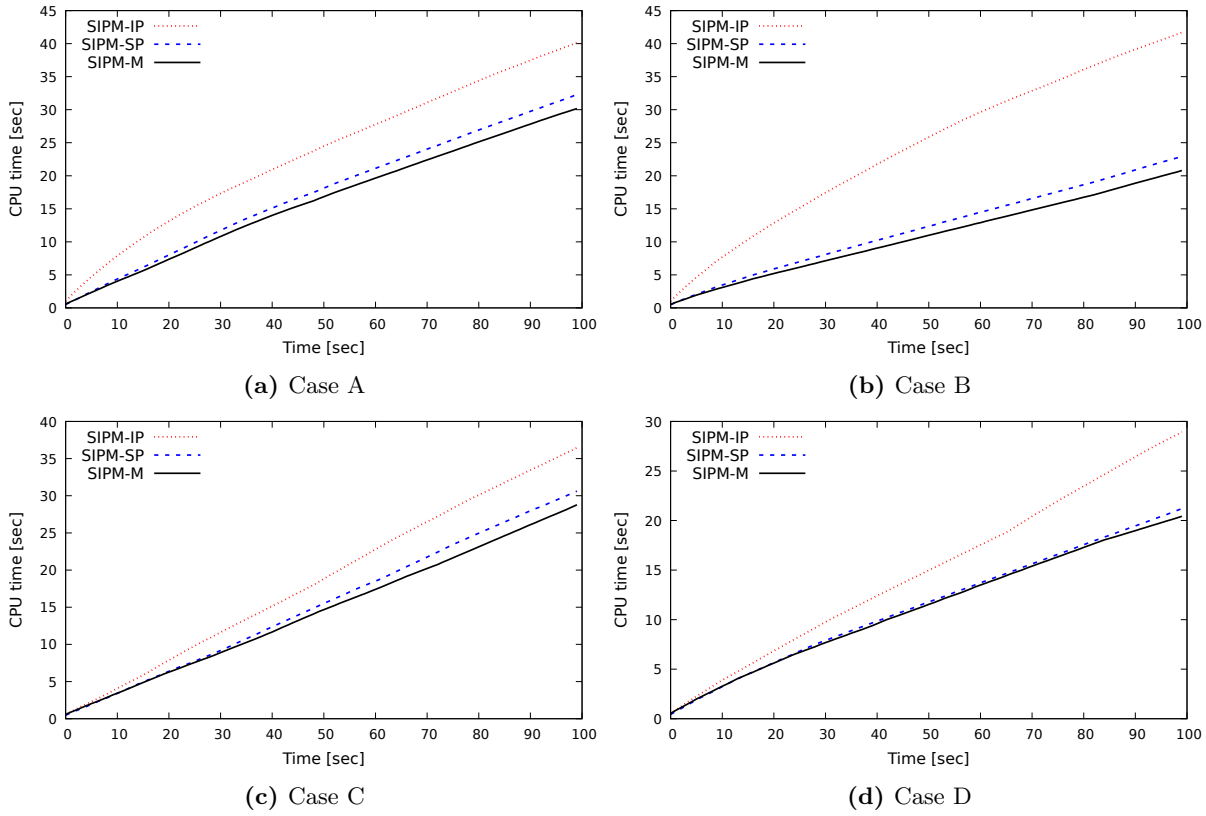
State	Pr	k_s/k_f	Gr	σ
A	0.7	1.6	1.43×10^5	0.93
B	0.7	1.6	10^4	0.93
C	0.7	1600	1.43×10^5	4×10^{-4}
D	0.7	1600	10^4	4×10^{-4}

approximately the same convergence rate as the proposed monolithic solver. Such a similarity in the convergence behaviour reflects that for this test case, ensuring temperature-temperature coupling via either simultaneous solution or a predictor-corrector loop within each outer iteration can significantly reduce the overall solution time. The results also suggest that the monolithic approach has slight benefits over the partitioned one for this test case.

The PISO algorithm can be applied with or without solving the momentum equation, i.e. momentum predictor. In the case of enabling momentum predictor in the PIMPLE algorithm, operator $\boldsymbol{H}(\mathbf{U})$ is evaluated by velocities obtained by solution of Eqn. (19); otherwise, the velocity field from the previous iteration is used to compute this operator. In all previous solutions, the momentum predictor was applied. To investigate more, Tab. 4 illustrates the impact of the momentum predictor on the number of outer iterations and solution time across different coupling conditions for these three algorithms. As shown, enabling the momentum predictor in the PIMPLE algorithm decreases the necessary outer iterations for all states by up to three times less in both the proposed partitioned and monolithic solvers. Consequently, this enhancement results in a doubling of solution speed.

Table 4. Effect of the momentum predictor on the CPU time and average number of outer iterations in the solution of the case of conjugate natural convection in a single square enclosure with three algorithms in various coupling conditions (F and T denote False and True for applying the momentum predictor; SIPM-IP indicates classic partitioned solver in OpenFOAM® forks; SIPM-SP indicates the proposed partitioned solver; SIPM-M indicate proposed monolithic solver)

State	**CPU time (in s)**						**Ave. of out. iter.**					
	SIPM-IP		SIPM-SP		SIPM-M		SIPM-IP		SIPM-SP		SIPM-M	
	F	T	F	T	F	T	F	T	F	T	F	T
A	59.1	40.1	64.1	32.3	43.9	30.17	28.4	12.2	28.2	9.3	28.2	9.3
B	43.1	41.7	42.46	22.9	41.4	20.79	17.6	12.7	16.6	5.6	17.1	5.6
C	60.0	36.5	60.2	30.6	57.8	28.8	28.4	10.3	28.4	9.3	28.3	8.7
D	39.4	29.0	39.6	21.2	39.2	20.43	16.5	7.5	16.5	5.5	16.3	5.4



(a) Case A **(b)** Case B

(c) Case C **(d)** Case D

Figure 9. Cumulative CPU time as a function of simulation time for unsteady conjugate natural convection in a single square enclosure with a single solid (state A to D) solved by classic partitioned (SIPM-IP), proposed partitioned (SIPM-SP), and proposed monolithic solvers (SIPM-M)

The different behaviour of the partitioned and monolithic approaches enabling the momentum predictor originates from the various definitions of the overall energy residual as the convergence criteria. According to Tab. 4, with the momentum predictor, the average number of outer iterations in the monolithic methods for states C and D increases, whereas for the partitioned methods it decreases. Delving into the solution histories reveals that the partitioned methods, compared to the monolithic approach, demand more outer iterations due to the satisfaction of the overall energy residual condition. Additionally, the results in this table suggest that without the momentum predictor, there is no considerable benefit for the proposed partitioned approach over the classic partitioned approach, meaning that adding an interface loop in the partitioned methods will be advantageous only when the momentum equation is solved.

4.2. **Unsteady conjugate natural convection in three square enclosures with three walls.** This section introduces a new case, similar to the previous one, but with multiple fluid and solid regions. The motivation for studying this case is to assess the scalability of the solvers in multi-region scenarios. The schematic of this case is shown in Fig. 10. The dimensions of the additional fluid and solid regions are the same as the case with a single fluid, i.e. $L_1 = 1$ m, $L_2 = 1$ m, and the wall thickness $L_S = 0.2$ m. The initial conditions for the fluid and solid regions are the same as the previous standard case, i.e. all fluids are at rest and $T_{s1} = T_{s2} = T_{s3} = 0^o$C, $p_{f1} = p_{f2} = p_{f3} = 0$ Pa, $T_{f1} = T_{f2} = T_{f3} = 0^o$C. The temperature boundary conditions are shown in the schematic, where $T_H = 1^o$C and $T_C = 0^o$C. Therefore, the left wall of the right solid (Solid1), the two walls of the middle solid (Solid2) and the two walls of the top solid (Solid3) are treated as shared interfaces.

Figure 11 shows the temperature contours in all regions and streamlines in the fluid regions for state A at $t = 0.07$ s.

Figure 12 presents the temperature profile along the horizontal line at $y = 0.5$ m and the vertical line at $x = 0.5$ m. Clearly, the three solvers give nearly the same temperature profiles.

Figure 13 compares CPU time used by the three solvers in four coupling states. Again, the comparison for this case reveals the superiority of the proposed monolithic over the partitioned methods (classic partitioned and proposed partitioned). Diving into convergence histories shows that the slow convergence

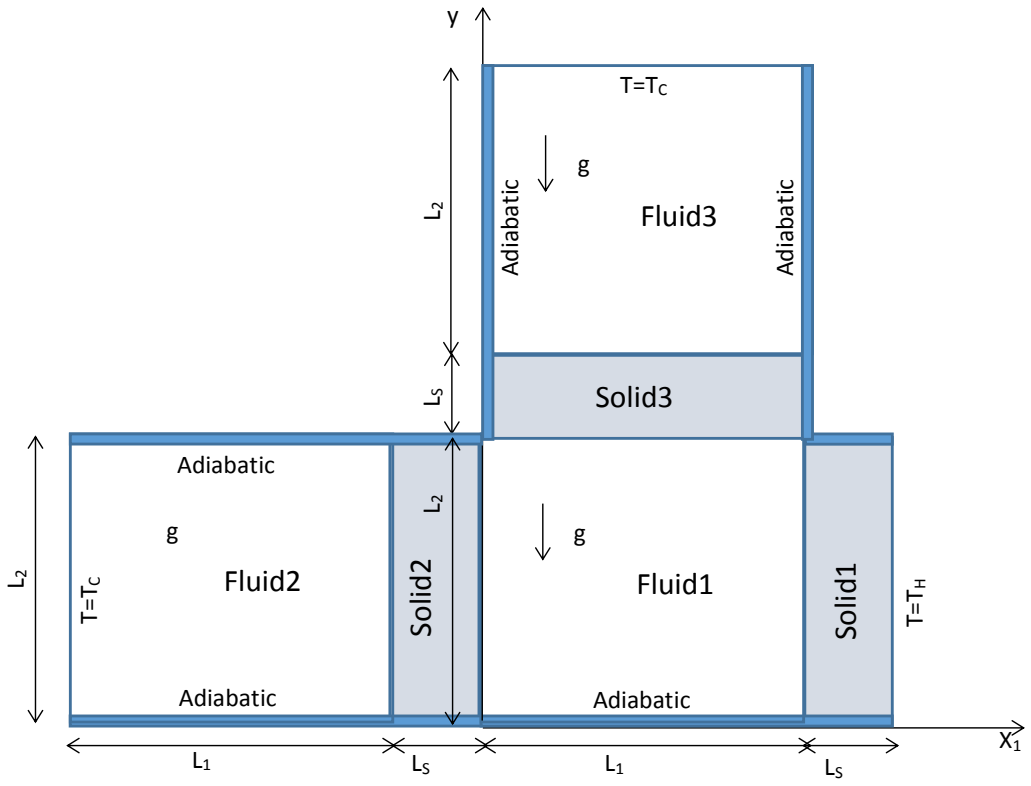


Figure 10. Schematic of unsteady conjugate natural convection in three square enclosures with three solid walls

of the partitioned methods for state C is rooted in the convergence criteria for the overall energy residual. More specifically, the energy residual in the Solid3 region oscillates for the early time steps, which demands more outer iterations at these time steps. As the time step progresses, the required outer iteration reduces due to less oscillatory behaviour in this residual.

Table 5 details the effect of the momentum predictor on the solution of this case via the three algorithms. According to the data in the table, deactivating the momentum predictor leads to a higher requirement for outer iterations across all three solvers. Regarding solution efficiency, integrating the momentum predictor into the proposed partitioned and monolithic solvers reduces CPU time. On the other hand, when considering the classic partitioned solver for scenario (B) of the test case, its inclusion decelerates the solution process.

Table 5. Effect of the momentum predictor on the CPU time and average number of outer iterations in the solution of the case of conjugate natural convection in three square enclosures with three algorithms in various coupling conditions (F and T denote False and True for applying the momentum predictor; SIPM-IP indicates integrated loop algorithm in OpenFOAM® solvers; SIPM-SP indicates the proposed separate loop algorithm; SIPM-M indicate proposed monolithic algorithm)

State	CPU time (sec)						Ave. of out. iter.					
	SIPM-IP		SIPM-SP		SIPM-M		SIPM-IP		SIPM-SP		SIPM-M	
	F	T	F	T	F	T	F	T	F	T	F	T
A	215.9	159.6	227.5	137.4	201.7	127.3	42.9	18.2	42.9	14.8	38.5	14.0
B	125.2	137.2	109.7	78.6	88.5	69.3	21.6	15.6	18.3	7.1	14.6	6.6
C	158.1	123.6	158.8	120.5	144.4	100.0	34.6	13.9	34.5	13.8	30.8	10.3
D	102.2	81.6	109.7	71.48	88.5	68.6	19.7	8.0	19.7	7.0	16.0	6.6

To investigate the impact of the momentum predictor, Figs. 14 and 15 depict the convergence history for continuity, momentum, and energy equations over five consecutive time steps while being solved using

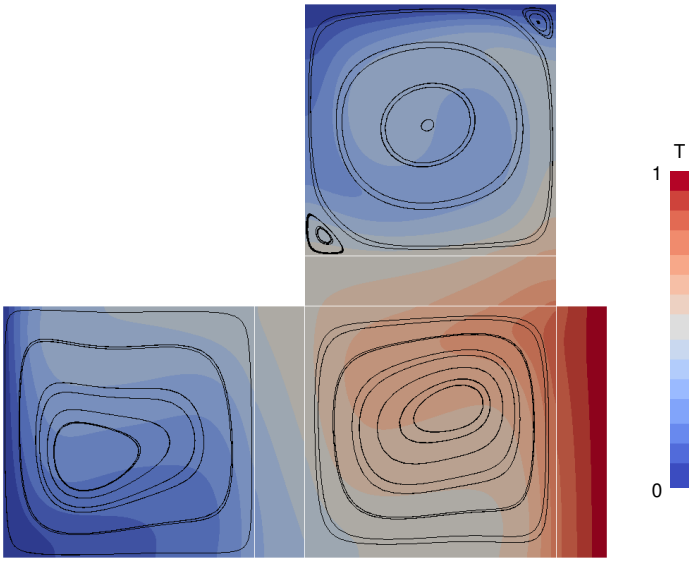
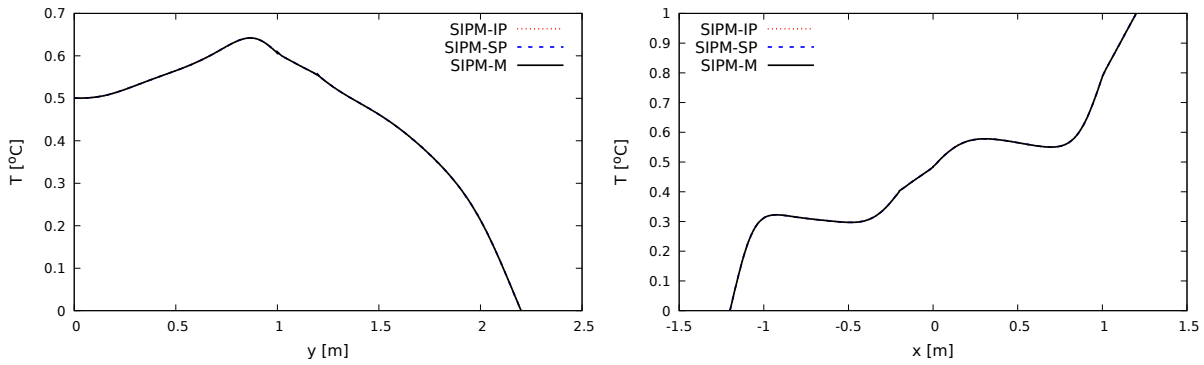


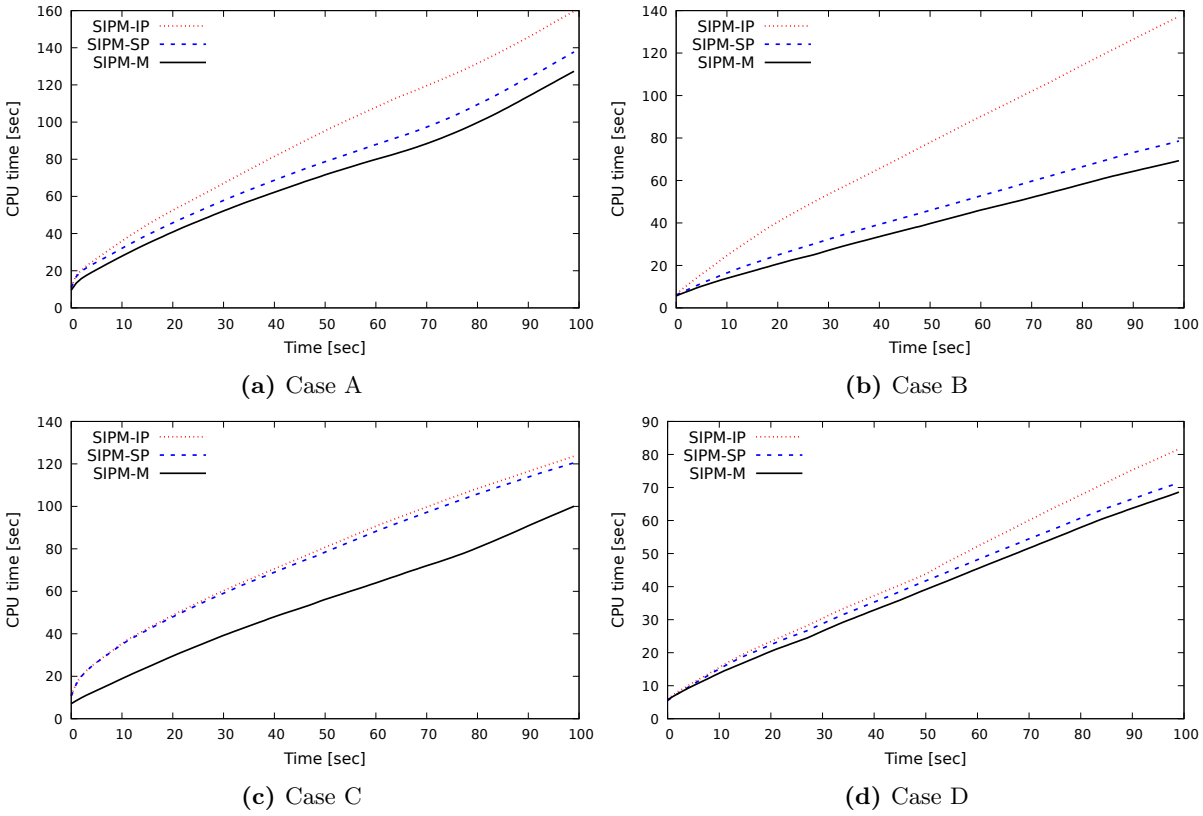
Figure 11. Contour of temperature and streamline for unsteady conjugate natural convection in three square enclosures with three walls (state A) at $t = 0.07$ s



(a) Horizontal line: $y = 0.5$ m **(b)** Vertical line: $x = 0.5$ m

Figure 12. Temperature profile at lines $y = 0.5m$ and $x = 0.5m$ given by three proposed solvers for unsteady conjugate natural convection in three square enclosures with three walls (state A) at $t = 0.07$ s

the SIMP-SP solver, *with* and *without* the application of the momentum predictor. As evident from Fig. 14, the convergence of the momentum equation serves as the bottleneck for overall time step convergence. Conversely, in Fig. 15, employing the momentum predictor facilitates simultaneous convergence of momentum and continuity equations at a similar pace. Additionally, it can be observed from both figures that for Cases (A) and (C), characterized by strong pUT coupling (indicated by high Grashof numbers), more iterations are necessary for time step convergence. This underscores the importance of employing efficient strategies for pUT coupling, such as applying the momentum predictor, to expedite simulations.



(a) Case A **(b)** Case B

(c) Case C **(d)** Case D

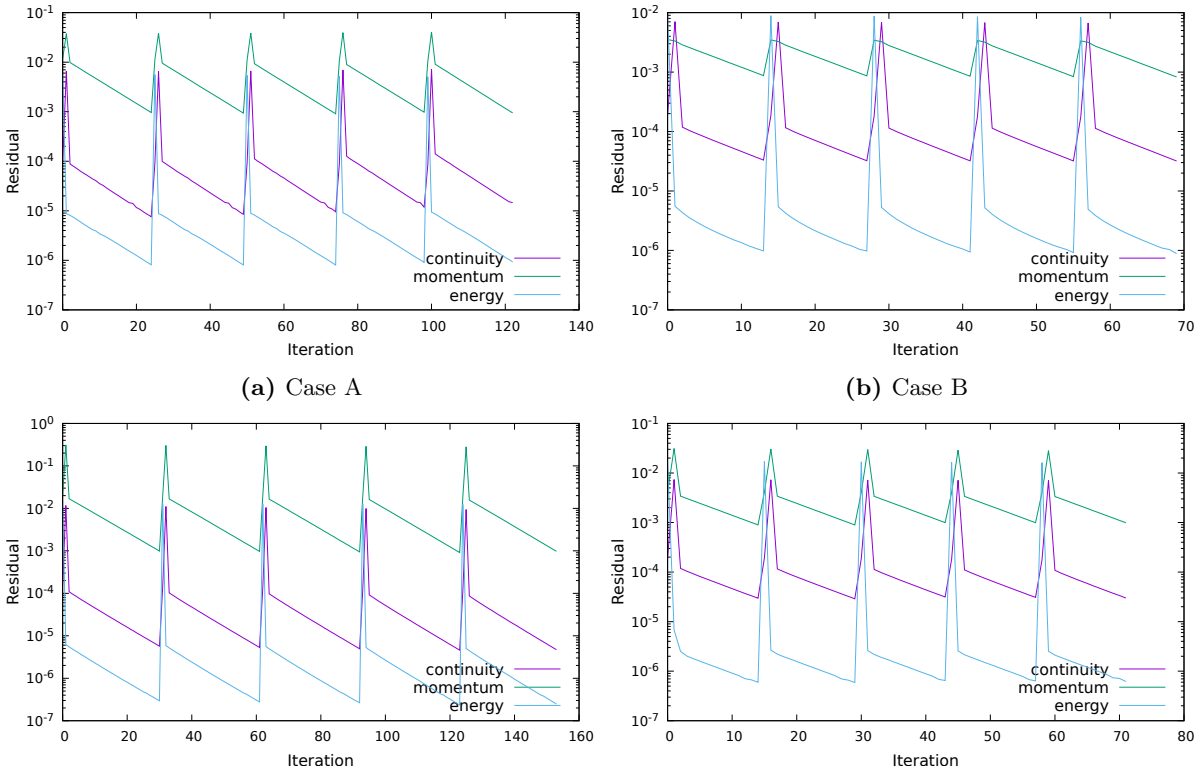
Figure 13. CPU time as a function of simulation time for unsteady conjugate natural convection in three square enclosures with three solids (state A to D) solved by classic partitioned, proposed partitioned, and proposed monolithic solvers

5. **Conclusion**

The numerical simulations in the previous section reveal that enforcing the interface conditions via either an internal predictor-corrector loop or a simultaneous solution of the energy equations can accelerate the solvers up to two times. This conclusion implies that the `chtMultiRegionFoam` solver in OpenFOAM® forks, which typically uses an algorithm similar to the classic partitioned algorithm, is not computationally efficient; the results suggest applying a separate loop to improve its computational efficiency. This solver in OpenFOAM-v2112 allows for using a separate loop to solve the energy equations. In the authors' view, the superiority of the proposed partitioned algorithm over the classic partitioned one is that compared to the temperature-temperature coupling, the pressure-velocity coupling in the fluid part is computationally more expensive to enforce. On the other hand, the pressure and velocity in the fluid regions are coupled to the fluid temperature through pressure-velocity-temperature coupling. Thus, while tight satisfaction of the interface conditions via an additional internal loop demands extra computational effort, it also reduces the number of required iterations in the outer loop and, consequently, the number of pressure-velocity solutions. Therefore, the overall computational time is reduced due to the additional internal loop. Moreover, the superiority of the proposed monolithic algorithm over the proposed partitioned one is rooted in the computational efficiency of the simultaneous solution over the sequential solution within a predictor-corrector loop. Another benefit of the monolithic solver is that it provides a better formulation for the overall energy residual across regions and, as a result, avoids extra iterations for the outer predictor-corrector loop. The results also demonstrate the effect of applying the momentum predictor. In particular, solving the momentum equation in the PIMPLE algorithm increases the benefits of adding an interface loop.

For the analyses presented in this article, we can draw several conclusions:

- The numerical solution of the CHT problems requires special care for coupling in the equations, especially when the equations are strongly linked.
- According to our test results, employing a strategy where the energy equation is solved simultaneously throughout the domain proves to be the most efficient approach. However, it's essential



(a) Case A **(b)** Case B

(c) Case C **(d)** Case D

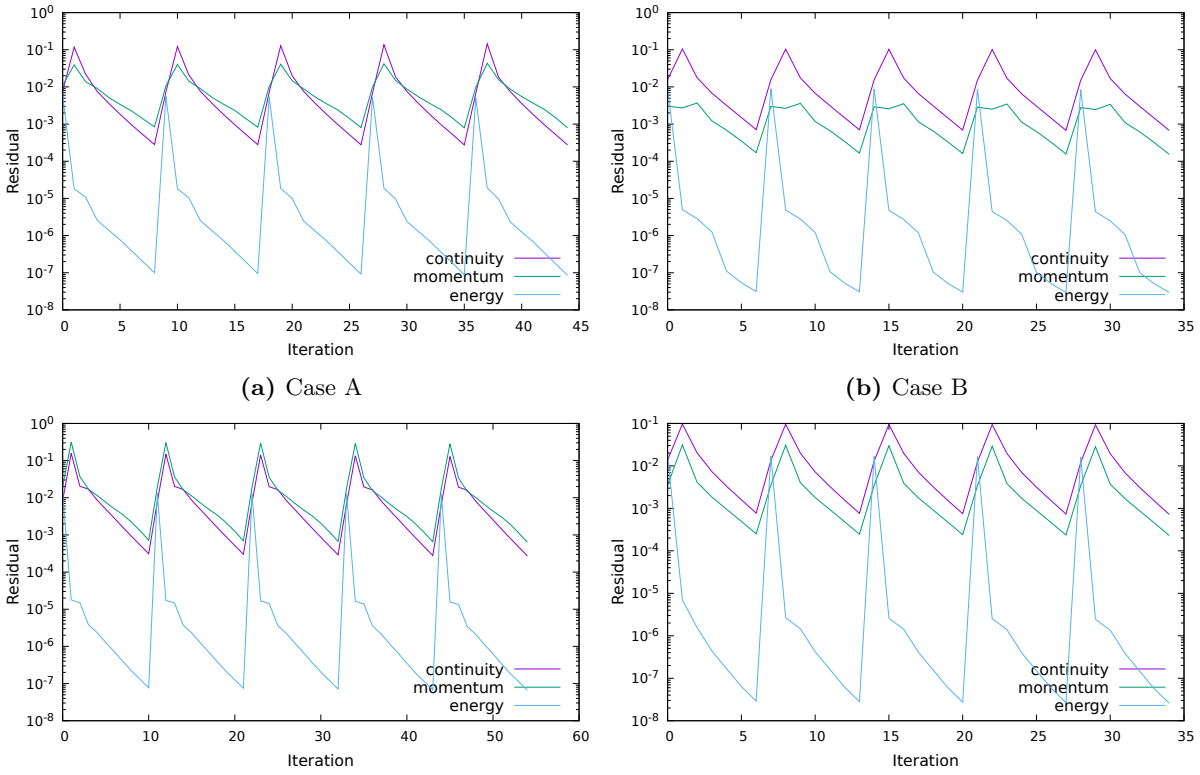
Figure 14. Residual convergence history over five consecutive time steps for unsteady conjugate natural convection in three square enclosures with three solids (states A to D), solved by the proposed partitioned solver `without` utilizing the momentum predictor

to recognize that the current solver within foam-extend-4.0 is constrained to handling only the temperature-based energy equations.

- Applying the momentum predictor significantly improves solution efficiency for the proposed partitioned and monolithic solvers. This enhancement leads to a significant increase in speed (up to twice as fast) across various coupling conditions, as demonstrated by the results of two test cases. On the other hand, while implementing the momentum predictor reduced the outer iterations for all cases, its impact on the speedup was less consistent with the classic partitioned method. In one instance, it even resulted in a slowdown of the solution.
- An analysis of results derived from partitioned methods indicates that integrating an additional iterative loop for resolving energy-energy coupling can enhance the solution speed by up to twofold, particularly in scenarios with strong coupling between energy equations across subdomains.
- While this study offers valuable insights into the performance of partitioned and monolithic solvers, certain scenarios remain unexplored. Specifically, solid-solid interactions, steady-state problems, the influence of different boundary conditions, such as fixed heat flux on solid boundaries instead of fixed temperature, and the effects of larger time step sizes have not been fully investigated. Further research addressing these factors could provide a more complete understanding of solver stability and accuracy.

Acknowledgements

Emad Tandis acknowledges using TAURUS HPC at Aston. Philip Cardiff acknowledges that his contribution to this work has emanated from research conducted with the financial support of/supported in part by a grant from Science Foundation Ireland (SFI) under Grant number RC2302-2. In addition, Philip Cardiff gratefully acknowledges financial support from the Irish Research Council through the Laureate program, grant number IRCLA/2017/45, and the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 101088740).



(a) Case A **(b)** Case B

(c) Case C **(d)** Case D

Figure 15. Residual convergence history over five consecutive time steps for unsteady conjugate natural convection in three square enclosures with three solids (states A to D), solved by the proposed partitioned solver `with` utilizing the momentum predictor

Author Contributions: Conceptualisation, E.T.; methodology, E.T.; software, E.T.; validation, E.T.; formal analysis, E.T., P.C., and A.A.; investigation, E.T., A.A., and P.C.; resources, E.T., A.A., and P.C.; data curation, E.T.; writing—original draft preparation, E.T.; writing—review and editing, E.T., A.A., and P.C.; visualisation, E.T.; supervision, A.A. and P.C.; project administration, E.T.; funding acquisition, E.T., A.A., and P.C. All authors have read and agreed to the published version of the manuscript.

References

[1] V. Gravemeier, S. M. Civaner, and W. A. Wall, "A partitioned-monolithic finite element method for thermo-fluid–structure interaction," *Computer Methods in Applied Mechanics and Engineering*, vol. 401, p. 115596, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S004578252200562X

[2] R. Löhner, C. Yang, J. Cerbal, J. Baum, H. Luo, D. Pelessone, and C. Charman, "Fluid-structure-thermal interaction using a loose coupling algorithm and adaptive unstructured grids," in *29th AIAA, Fluid Dynamics Conference*, 1998, p. 2419.

[3] P. Birken, K. J. Quint, S. Hartmann, and A. Meister, "A time-adaptive fluid-structure interaction method for thermal coupling," *Computing and visualization in science*, vol. 13, pp. 331–340, 2010.

[4] M. Grilli, S. Hickel, N. Adams, G. Hammerl, C. Danowski, and W. Wall, "An innovative approach to thermo-fluid-structure interaction based on an immersed interface method and a monolithic thermo-structure interaction algorithm," in *42nd AIAA Fluid Dynamics Conference and Exhibit*, 2012, p. 3267.

[5] P. Birken, T. Gleim, D. Kuhl, and A. Meister, "Fast solvers for unsteady thermal fluid structure interaction," *International Journal for Numerical Methods in Fluids*, vol. 79, no. 1, pp. 16–29, 2015.

[6] R. I. Issa, "Solution of the implicitly discretised fluid flow equations by operator-splitting," *Journal of computational physics*, vol. 62, no. 1, pp. 40–65, 1986.

[7] D. Jang, R. Jetli, and S. Acharya, "Comparison of the piso, simpler, and simplec algorithms for the treatment of the pressure-velocity coupling in steady flow problems," *Numerical Heat Transfer, Part A: Applications*, vol. 10, no. 3, pp. 209–228, 1986.

[8] B. Latimer and A. Pollard, "Comparison of pressure-velocity coupling solution algorithms," *Numerical Heat Transfer*, vol. 8, no. 6, pp. 635–652, 1985.

[9] M. Perić, "Analysis of pressure-velocity coupling on nonorthogonal grids," *Numerical Heat Transfer*, vol. 17, no. 1, pp. 63–82, 1990.

[10] N. Serra and V. Semiao, "Esimple, a new pressure–velocity coupling algorithm for built-environment cfd simulations," *Building and Environment*, vol. 204, p. 108170, 2021.

[11] J. Martínez, F. Piscaglia, A. Montorfano, A. Onorati, and S. M. Aithal, "Influence of momentum interpolation methods on the accuracy and convergence of pressure–velocity coupling algorithms in openfoam®," *Journal of Computational and Applied Mathematics*, vol. 309, pp. 654–673, 2017.
[12] H. Jasak, "Error analysis and estimation for the finite volume method with applications to fluid flows," *PhD Thesis, Imperial College London (University of London)*, 1996.
[13] S. V. Patankar and D. B. Spalding, "A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows," *International Journal of Heat and Mass Transfer*, vol. 15, pp. 1787–1806, 1972.
[14] R. Moretti, M.-P. Errera, V. Couaillier, and F. Feyel, "Stability, convergence and optimization of interface treatments in weak and strong thermal fluid-structure interaction," *International Journal of Thermal Sciences*, vol. 126, pp. 23–37, 2018.
[15] C. A. Felippa and K.-C. Park, "Staggered transient analysis procedures for coupled mechanical systems: formulation," *Computer Methods in Applied Mechanics and Engineering*, vol. 24, no. 1, pp. 61–111, 1980.
[16] F. Meng, J. W. Banks, W. D. Henshaw, and D. W. Schwendeman, "A stable and accurate partitioned algorithm for conjugate heat transfer," *Journal of Computational Physics*, vol. 344, pp. 51–85, 2017.
[17] M.-P. Errera, R. Moretti, J. Mayeur, M. Gelain, L. Tessé, J.-M. Lamet, and E. Laroche, "A numerical predictive model for conjugate heat transfer with radiation," *International Journal of Heat and Mass Transfer*, vol. 160, p. 120155, 2020.
[18] V. Kazemi-Kamyab, A. Van Zuijlen, and H. Bijl, "Analysis and application of high order implicit runge–kutta schemes for unsteady conjugate heat transfer: A strongly-coupled approach," *Journal of Computational Physics*, vol. 272, pp. 471–486, 2014.
[19] X. Pan, K. Kim, C. Lee, and J.-I. Choi, "A decoupled monolithic projection method for natural convection problems," *Journal of Computational Physics*, vol. 314, pp. 160–166, 2016.
[20] P. J. Oliveira and R. I. Issa, "An improved piso algorithm for the computation of buoyancy-driven flows," *Numerical Heat Transfer, Part B: Fundamentals*, vol. 40, no. 6, pp. 473–493, 2001.
[21] N. O. Moraga, S. del C. Ramírez, M. J. Godoy, and P. D. Ticchione, "Study of convective non-newtonian alloy solidification in molds by the psimpler/finite-volume method," *Numerical Heat Transfer, Part A: Applications*, vol. 57, no. 12, pp. 936–953, 2010. [Online]. Available: https://doi.org/10.1080/10407782.2010.490171
[22] E. Radenac, J. Gressier, and P. Millan, "Methodology of numerical coupling for transient conjugate heat transfer," *Computers & Fluids*, vol. 100, pp. 95–107, 2014.
[23] M.-P. Errera and F. Duchaine, "Comparative study of coupling coefficients in dirichlet–robin procedure for fluid–structure aerothermal simulations," *Journal of Computational Physics*, vol. 312, pp. 218–234, 2016.
[24] S. Scholl, B. Janssens, and T. Verstraete, "Stability of static conjugate heat transfer coupling approaches using robin interface conditions," *Computers & Fluids*, vol. 172, pp. 209–225, 2018.
[25] L. He and M. Oldfield, "Unsteady conjugate heat transfer modeling," *Journal of turbomachinery*, vol. 133(3), p. 031022, 2011.
[26] M. B. Giles, "Stability analysis of numerical interface conditions in fluid–structure thermal analysis," *International journal for numerical methods in fluids*, vol. 25, no. 4, pp. 421–436, 1997.
[27] M.-P. Errera, R. Moretti, R. Salem, Y. Bachelier, T. Arrivé, and M. Nguyen, "A single stable scheme for steady conjugate heat transfer problems," *Journal of Computational Physics*, vol. 394, pp. 491–502, 2019.
[28] E. Tandis and A. Ashrafizadeh, "Comparison of monolithic and partitioned approaches for the solution of fluid-solid thermal interaction problems," *Journal of Applied and Computational Sciences in Mechanics*, vol. 31, no. 1, 2020.
[29] A. Demuren and H. Grotjans, "Buoyancy-driven flows—beyond the boussinesq approximation," *Numerical Heat Transfer, Part B: Fundamentals*, vol. 56, no. 1, pp. 1–22, 2009. [Online]. Available: https://doi.org/10.1080/10407790902970080
[30] T. Almqvist and R. Larsson, "The navier–stokes approach for thermal ehl line contact solutions," *Tribology International*, vol. 35, no. 3, pp. 163–170, 2002.
[31] X. Liu, M. Jiang, P. Yang, and M. Kaneta, "Non-newtonian thermal analyses of point ehl contacts using the eyring model," *J. Trib.*, vol. 127, no. 1, pp. 70–81, 2005.
[32] M. Cavazzuti, "Viscous heating effects on heat transfer characteristics of laminar compressible channel flow," *International Journal of Heat and Mass Transfer*, vol. 153, p. 119608, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0017931019366633
[33] F. Moukalled, L. Mangani, and M. Darwish, *The finite volume method*. Springer, 2016, pp. 103–135.
[34] A. Moraes, P. Lage, G. Cunha, and L. da Silva, "Analysis of the non-orthogonality correction of finite volume discretization on unstructured meshes," in *Proceedings of the 22nd International Congress of Mechanical Engineering, Ribeirão Preto, Brazil*, 2013, Conference Proceedings, pp. 3–7.
[35] F. P. Kärrholm, *Numerical modelling of diesel spray injection, turbulence interaction and combustion*. Chalmers University of Technology Gothenburg, Sweden, 2008.
[36] S. V. Patankar, *Numerical heat transfer and fluid flow*. CRC press, 2018.
[37] V. Kazemi-Kamyab, A. van Zuijlen, and H. Bijl, "Accuracy and stability analysis of a second-order time-accurate loosely coupled partitioned algorithm for transient conjugate heat transfer problems," *International Journal for Numerical Methods in Fluids*, vol. 74, no. 2, pp. 113–133, 2014.
[38] X. Pan, C. Lee, and J.-I. Choi, "Efficient monolithic projection method for time-dependent conjugate heat transfer problems," *Journal of Computational Physics*, vol. 369, pp. 191–208, 2018.