Computer Physics Communications 253 (2020) 107188

Contents lists available at ScienceDirect

Computer Physics Communications

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

A finite-element toolbox for the simulation of solid–liquid phase-change systems with natural convection☆,☆☆

Aina Rakotondrandisa, Georges Sadaka, Ionut Danaila *

Laboratoire de Mathématiques Raphaël Salem, Université de Rouen Normandie, CNRS UMR 6085, Avenue de l'Université, BP 12, F-76801 Saint-Étienne-du-Rouvray, France

ARTICLE INFO

Article history:
Received 14 May 2019
Received in revised form 31 October 2019
Accepted 13 January 2020
Available online 25 January 2020

Keywords:
Melting
Solidification
PCM
Finite element
Mesh adaptivity
Navier–Stokes
Boussinesq
FreeFem++

ABSTRACT

We present and distribute a new numerical system using classical finite elements with mesh adaptivity for computing two-dimensional liquid–solid phase-change systems involving natural convection. The programs are written as a toolbox for FreeFem++ (www3.freefem.org), a free finite-element software available for all existing operating systems. The code implements a single domain approach. The same set of equations is solved in both liquid and solid phases: the incompressible Navier–Stokes equations with Boussinesq approximation for thermal effects. This model describes naturally the evolution of the liquid flow which is dominated by convection effects. To make it valid also in the solid phase, a Carman–Kozeny-type penalty term is added to the momentum equations. The penalty term brings progressively (through an artificial mushy region) the velocity to zero into the solid. The energy equation is also modified to be valid in both phases using an enthalpy (temperature-transform) model introducing a regularized latent-heat term. Model equations are discretized using Galerkin triangular finite elements. Piecewise quadratic (P2) finite-elements are used for the velocity and piecewise linear (P1) for the pressure. For the temperature both P2 and P1 discretizations are possible. The coupled system of equations is integrated in time using a second-order Gear scheme. Non-linearities are treated implicitly and the resulting discrete equations are solved using a Newton algorithm. An efficient mesh adaptivity algorithm using metrics control is used to adapt the mesh every time step. This allows us to accurately capture multiple solid–liquid interfaces present in the domain, the boundary-layer structure at the walls and the unsteady convection cells in the liquid. We present several validations of the toolbox, by simulating benchmark cases of increasing difficulty: natural convection of air, natural convection of water, melting of a phase-change material, a melting-solidification cycle, and, finally, a water freezing case. Other similar cases could be easily simulated with this toolbox, since the code structure is extremely versatile and the syntax very close to the mathematical formulation of the model.

Programm summary

Program Title: PCM-Toolbox-2D
Program Files doi: http://dx.doi.org/10.17632/phby62rhgv.1
Licensing provisions: Apache License, 2.0
Programming language: FreeFem++(free software, www3.freefem.org)
Nature of problem: The software computes 2D configurations of liquid–solid phase-change problems with convection in the liquid phase. Natural convection, melting and solidification processes are illustrated in the paper. The software can be easily modified to take into account different related physical models.
Solution method: We use a single domain approach, solving the incompressible Navier–Stokes equations with Boussinesq approximation in both liquid and solid phases. A Carman–Kozeny-type penalty term is added to the momentum equations to bring the velocity to zero into the solid phase. An enthalpy model is used in the energy equation to take into account the phase change. Discontinuous variables (latent heat, material properties) are regularized through an intermediate (mushy) region. Space

☆ The review of this paper was arranged by Prof. Hazel Andrew.

☆☆ This paper and its associated computer program are available via the Computer Physics Communication homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).

* Corresponding author.
E-mail addresses: aina.rakotondrandisa@etu.univ-rouen.fr (A. Rakotondrandisa), georges.sadaka@univ-rouen.fr (G. Sadaka), ionut.danaila@univ-rouen.fr (I. Danaila).

https://doi.org/10.1016/j.cpc.2020.107188

discretization is based on Galerkin triangular finite elements. Piecewise quadratic (P2) finite-elements are used for the velocity and piecewise linear (P1) for the pressure. For the temperature both P2 and P1 discretizations are possible. A second order Gear scheme is used for the time integration of the coupled system of equations. Non-linear terms are treated implicitly and the resulting discrete equations are solved using a Newton algorithm. A mesh adaptivity algorithm is implemented to reduce the computational time and increase the local space accuracy when (multiple) interfaces are present.

1. Introduction

Solid–liquid phase-change problems are encountered in numerous practical applications, ranging from metal casting and thermal energy storage (phase-change materials) to food freezing and cryosurgery. Melting and solidification are also fundamental processes in geophysical problems, such as Earth's mantle formation, lava lakes or magma chambers. In many of these problems, temperature gradients induce buoyancy forces in the liquid (melted) phase generating a significant convective flow. Consequently, the appropriate mathematical description of the liquid phase is the usual model for the natural convection flow: the incompressible Navier–Stokes system of equations with Boussinesq approximation for thermal (buoyancy) effects. In this model, the energy conservation equation is written as a convection–diffusion equation for the temperature. In the solid phase, conduction is the main phenomenon and the appropriate model is the classical heat equation. The main modelling difficulty is to link these two models by taking into account the separation of the two phases by a sharp interface, across which thermodynamic properties are discontinuous. We offer below a short description of the two main approaches suggested in the literature to deal with this problem. For a comprehensive review of models for phase-change problems with convection, see [1]. Note that a different category of models was recently suggested in the literature, based on the Lattice Boltzmann Method [2,3]. Such methods based on non-deterministic models are not discussed in this introduction.

A first modelling approach, usually referred to as the multi-domain (or deforming-grid) method, is based on the classical Stefan two-phase model. Solid and liquid domains are separated and the corresponding conservation equations are solved in each domain. Boundary conditions at the interface between domains are obtained by imposing the Stefan condition (balance of heat fluxes at the interface). The position of the solid–liquid interface is tracked and moved explicitly using either *front tracking* or *front fixing* methods. The former method uses deforming grids to reconstruct the interface, while the latter is based on a time-depending coordinate transform, mapping the physical domain into a fixed computational domain. For a detailed description of these methods, see *e.g.* [4–7]. The main drawback of deforming-grid methods is their algorithmic complexity, which makes difficult to accurately capture solid–liquid interfaces of complicated shape or structure (*e.g.* with mushy regions between solid and liquid phases). Configurations with multiple interacting interfaces (see the solidification of a phase-change material presented in this paper) are also difficult to simulate with these methods (see also [8]).

The second modelling approach avoids to impose explicitly the Stefan condition at the solid–liquid interface and therefore uses a single-domain (or fixed-grid) model. The same system of equations is solved in both liquid and solid phases. The energy balance at the interface is implicitly taken into account by the model. Consequently, the position of the interface is obtained a posteriori by post-processing the computed temperature field. Phase-field methods [9] and enthalpy methods [10,11] are the most commonly used single-domain models. In phase-field methods, a supplementary partial differential equation for the evolution of the order parameter (a continuous variable taking the value 0 in the solid and 1 in the liquid) has to be solved, coupled with the conservation laws [12]. This new equation is model dependent and its numerical solution could lead to diffuse solid–liquid interfaces. For recent contributions in this area, see [13–15]. We focus below on enthalpy methods, which are the most widely used single-domain models due to their algorithmic simplicity.

The main idea behind enthalpy models is to formulate the energy conservation law in terms of enthalpy and temperature and include latent heat effects in the definition of the enthalpy. The obtained equation applies to both liquid and solid phases and implicitly takes into account the separation of the phases. Another advantage of enthalpy methods, when compared to previously described models, is to remove the limitation of the phase-change occurring at a fixed temperature. The presence of mushy regions can be easily modelled with these methods. Two types of formulations of enthalpy methods exist in the literature, depending on the main variable used to solve the energy equation: enthalpy or temperature-based methods. In enthalpy-based formulations the main variable is the enthalpy [16–18]. Temperature is computed from the temperature–enthalpy coupling model. An iterative loop is necessary to solve the energy equation, formulated with both enthalpy and temperature variables. For a review of different iterative techniques to solve the energy equation, see [19]. A second variety of enthalpy-based formulations consists in rewriting the energy equation with enthalpy terms only [20,21]. In temperature-based formulations, the energy equation is formulated in terms of temperature only. The latent heat is treated either by deriving an apparent heat capacity coefficient to define the total enthalpy [22–24] or by introducing a source term in the energy equation [19,25]. Advantages and drawbacks of each approach are discussed in detail in [26].

Single-domain methods are very appealing for numerical implementations. The same system of equations is solved in the entire computational domain, making possible algorithmic or computer-architecture optimizations. If enthalpy models offer an elegant solution to deal with the same energy conservation equation in both phases, a last modelling problem has to be solved. It concerns the extension of the Navier–Stokes–Boussinesq equations from the liquid to the solid phase. Different techniques to bring the velocity to zero in the solid region were suggested. The most straightforward is the switch-off technique, which decouples solid and liquid computational points and overwrites the value of the velocity by setting it to zero in the solid region. Different implementations of this technique with finite-volume methods are presented in [27,28]. In variable viscosity techniques [29–31], the fluid viscosity depends on the temperature and is artificially increased to huge values in the solid regions through a regularization or mushy zone. To avoid blow-up or numerical inconsistencies, the large gradients of viscosity must be correctly resolved in the mushy region. This is naturally achieved in finite-element methods with dynamical mesh adaptivity [32], while in finite-volume methods with fixed grids, the time step has to be adapted to the space resolution [27]. Versions of the variable viscosity approach were theoretically studied by Aldbaissy et al. [33]; Woodfield et al. [34]. A new

formulation of the phase-change problem using as variables the pseudo-stress, strain rate and velocity for the Navier–Stokes–Brinkman equations was recently suggested by Alvarez et al. [35].

A third technique used to ensure a zero velocity field in the solid phase is the so-called enthalpy-porosity model [36]. A penalization source term is introduced in the momentum equation to allow the switch from the full Navier–Stokes equations in the liquid phase to a Darcy equation for porous media. The mushy region is thus regarded as a very dense porous medium that sharply brings the velocity to zero in the solid region. The expression of the penalization source term generally follows the Carman–Kozeny model for the permeability of a porous medium [21,37,38], but other mathematically equivalent expressions were suggested [15,39]. Different formulations and implementations of the enthalpy-porosity model are presented in [8,40,41].

Concerning the space discretization of these models, finite difference (FD) or finite volume (FV) methods are generally used in the literature. When single-mesh models are used, the general strategy to capture the solid–liquid interface is to dramatically increase the mesh resolution in the whole domain. This results in a considerable increase of the computational time, even for two dimensional cases. Finite element (FE) methods offer the possibility to dynamically refine the mesh only in the regions of the domain where sharp phenomena take place (*e.g.* solid–liquid interface, recirculation zones). Different FE approaches were suggested, from enthalpy-type methods (*e.g.* [42]) to front-tracking methods (*e.g.* [43]). Recently, adaptive FE methods were suggested for classical two-phase Stefan problem and phase-change systems with convection. In [44] a hierarchical error estimator was used for 2D Stefan problems (without convection). A different mesh adaptivity method, based on the definition of edge length from a solution dependent metric, was used in [38] to deal with the same Stefan problems in 3D and also with 2D phase-change systems with convection. The time-dependent mesh adaptivity strategy based on metrics control suggested by Danaila et al. [32] proved very effective in refining the mesh near walls and interfaces, when simulating melting and solidification phase-change systems (see also [45]). Zimmerman and Kowalski [46] implemented the variable viscosity model suggested by Danaila et al. [32] in a different finite-element framework using FEniCS and an AMR (Adaptive Mesh Refinement) technique based on a dual-weighted residual method. In a very recent contribution, Belhamadia et al. [47] derived a time-dependent adaptive remeshing method for phase-change problem with convection based on an error estimator applicable to second or higher order variables.

However, to the best of our knowledge, no adaptive finite-element programs exist in the CPC Program Library for phase-change problems. The purpose of this paper is therefore to distribute a finite-element toolbox to solve two-dimensional solid–liquid phase-change problems.

The present toolbox is based on a single-domain enthalpy-porosity model for solid–liquid phase change problems with convection. For the energy conservation equation, a temperature-based formulation takes into account the latent heat by introducing a discontinuous source term. For the mass and momentum conservation equations, we solve in the entire domain the incompressible Navier–Stokes equations with Boussinesq approximation for buoyancy effects. To bring the velocity to zero in the solid phase, we introduce in the momentum equation a penalty term following the Carman–Kozeny model. The coupled system of momentum and energy equations is integrated in time using a second-order Gear scheme. All the terms are treated implicitly and the resulting discretized equations are solved using a Newton method [32]. The advantage of this formulation is to permit a straightforward implementation of different types of non-linearities. For the space discretization we use Taylor–Hood triangular finite elements, *i.e.* P_2 for the velocity and temperature and P_1 for the pressure. Temperature is discretized using P_2 or P_1 finite elements. The choice of using Taylor–Hood finite elements for the fluid flow and P_2 for the temperature was also made in [34,47]. It offers second order accuracy and (inf–sup) stability of the space discretization of the fluid flow. Other possible choices for the finite-element discretization could be tested for the implementation of the algorithm presented in this paper. For instance, the mini-element (see [48,49]), offering a global linear convergence, could be an interesting alternative to replace the Taylor–Hood element and thus optimize the computational time. This possibility could be explored with the present toolbox, since the mini-element exists in FreeFem++.

Discontinuous variables (latent heat, thermal diffusivity, etc.) at the solid–liquid interface are regularized through an intermediate artificial mushy region. Single domain methods require a refined mesh near the interface, where large enthalpy gradients have to be correctly resolved. An optimized dynamical mesh adaptivity algorithm allows us to adapt the mesh every time step and thus accurately capture the evolution of the interface. Mesh adaptivity feature of the toolbox offers the possibility to deal with complicated phase-change cases, involving multiple solid–liquid interfaces. There are two main novelties in the present numerical approach, when compared to Danaila et al. [32]: *(i)* we use the Carman–Kozeny model to bring the velocity to zero inside the solid phase, instead of a viscosity penalty method (imposing a large value of the viscosity in the solid); *(ii)* we increase the time accuracy of the algorithm by replacing the first-order Euler scheme with the second-order Gear (BDF2) scheme (see also [38]).

The programs were built and organized as a toolbox for FreeFem++ [50,51], which is a free software (under LGPL license). FreeFem++[1] offers a large variety of triangular finite elements (linear and quadratic Lagrangian elements, discontinuous P_1, Raviart–Thomas elements, etc.) to solve partial differential equations. It is an integrated product with its own high level programming language and a syntax close to mathematical formulations, making the implementation of numerical algorithms very easy. Among the features making FreeFem++ an easy-to-use and highly adaptive software we recall the advanced automatic mesh generator, mesh adaptation, problem description by its variational formulation, automatic interpolation of data, colour display on line, postscript printouts, etc. The FreeFem++ programming framework offers the advantage to hide all technical issues related to the implementation of the finite element method. It becomes then easy to use the present toolbox to code new numerical algorithms for similar problems with phase-change.

The paper is organized as follows. Section 2 introduces the governing equations and the regularization technique used in the enthalpy-porosity model. Section 3 presents the Newton algorithm for the Navier–Stokes–Boussinesq system of equations. The mesh adaptivity technique is also discussed in this section. A description of the programs is given in Section 4. The accuracy of the numerical method is tested in Section 5 using manufactured solutions. Finally, Section 6 is devoted to extensive numerical validations of the method. The robustness of the algorithm is demonstrated by comparing our results with well defined classical benchmarks. The capabilities of the toolbox to deal with complex geometries are also illustrated. The main features of the software and possible extensions are summarized in Section 7.

[1] FreeFem++ for different OS can be downloaded from http://www3.freefem.org/.

2. Enthalpy-porosity model

We consider a solid–liquid system placed in a two-dimensional domain Ω. In the following, subscripts s and l will refer to the solid and the liquid phase, respectively. We present in this section the single-domain model, using the same system of equations in both liquid and solid phases.

2.1. Energy equation

The phase change process is modelled using an enthalpy method with a temperature-based formulation [11,30,31]. We start with the classical energy equation:

$$\frac{\partial(\rho h)}{\partial t_\varphi} + \nabla\cdot(\rho h\boldsymbol{U}) - \nabla\cdot(k\nabla T) = 0, \tag{1}$$

where, t_φ is the physical time, h the enthalpy, ρ the density, $\boldsymbol{U}$ the velocity vector, T the temperature and k the thermal conductivity. The total enthalpy h is transformed as the sum of the sensible heat and the latent heat:

$$h = h_{sen} + h_{lat} = c(T + s(T)), \tag{2}$$

with c the local specific heat. The function s is introduced to model the jump of the enthalpy during the solid–liquid transition. For pure materials, s is theoretically a Heaviside step function depending on the temperature: it takes the zero value in the solid region and a large value in the liquid, equal to h_{sl}/c_l, with h_{sl} the latent heat of fusion. If the phase-change is assumed to occur within a mushy zone defined by a small temperature interval $T \in [T_f - T_\varepsilon, T_f + T_\varepsilon]$ around the temperature of fusion T_f, a model for $s(T)$ is necessary. Linear [28,30] or more smooth functions [32] can be used to regularize $s(T)$ and thus model the jump of material properties from solid to liquid. In this paper we use a regularization of all step-functions (latent heat source, specific heat, thermal diffusivity or conductivity) by a continuous and differentiable hyperbolic-tangent function suggested by Danaila et al. [32]. We assume moreover that the undercooling phenomenon is negligible during the solidification stage (see also [1,52]).

Equation (1) can be simplified under the following assumptions: (i) the density difference between solid and liquid phases is negligible, *i.e.* $\rho_l = \rho_s = \rho$ is constant; (ii) the mushy region is narrow and the velocity inside this zone is zero. Consequently, the final temperature transforming model is obtained by combining (2) and (1) and neglecting the convection term $\nabla\cdot(cs\boldsymbol{U})$[2]:

$$\frac{\partial(cT)}{\partial t_\varphi} + \nabla\cdot(cT\boldsymbol{U}) - \nabla\cdot\left(\frac{k}{\rho}\nabla T\right) + \frac{\partial(cs)}{\partial t_\varphi} = 0. \tag{3}$$

2.2. Navier–Stokes equations with Boussinesq approximation

The natural convection in the liquid part of the system is modelled using the incompressible Navier–Stokes equations, with Boussinesq approximation for buoyancy effects. To make this model valid in the entire domain, the momentum equation is modified as following:

$$\frac{\partial\boldsymbol{U}}{\partial t_\varphi} + (\boldsymbol{U}\cdot\nabla)\boldsymbol{U} + \frac{1}{\rho}\nabla p - \frac{\mu_l}{\rho}\nabla^2\boldsymbol{U} - f_B(T)\boldsymbol{e}_y = A(T)\boldsymbol{U}, \tag{4}$$

where p denotes the pressure, μ_l the dynamic viscosity of the liquid (assumed to be constant) and $f_B(T)$ the Boussinesq force. The Carman–Kozeny penalty term $A(T)\boldsymbol{u}$ is artificially introduced in (4) to extend this equation into the solid phase, where the velocity, pressure, viscosity and Boussinesq force are meaningless. Consequently, $A(T)$ is modelled to vanish in the liquid, where the Navier–Stokes–Boussinesq momentum equation is recovered. A large value of $A(T)$ is imposed in the solid, reducing the momentum equation (4) to $A(T)\boldsymbol{u} = 0$, equivalent to $\boldsymbol{u} = 0$. Exact expressions for f_B and A will be given in the next section.

Finally, the conservation of mass in the liquid phase is expressed by the continuity equation:

$$\nabla\cdot\boldsymbol{U} = 0. \tag{5}$$

2.3. Final system of equations for the single-domain approach

The present numerical code solves a dimensionless form of the previous equations. After choosing a reference length H (usually the height of the cavity when a rectangular domain is considered) and a reference state (ρ, V_{ref}, T_{ref}), we can define the following scaling for the space, velocity, temperature and time variables:

$$\boldsymbol{x} = \frac{\boldsymbol{X}}{H},\ \boldsymbol{u} = \frac{\boldsymbol{U}}{V_{ref}},\ \theta = \frac{T - T_{ref}}{\delta T},\ t = \frac{V_{ref}}{H}t_\varphi. \tag{6}$$

T_{ref} is the reference temperature and in most cases $T_{ref} = T_f$ (the temperature of fusion), unless otherwise specified. Consequently, the non-dimensional temperature of fusion is set to $\theta_f = 0$. Temperature difference δT defines a temperature scale, that will be set differently for melting and solidification cases.

The dimensionless system of equations to be solved in both liquid and solid regions can be finally written as:

$$\nabla\cdot\boldsymbol{u} = 0, \tag{7}$$

$$\frac{\partial\boldsymbol{u}}{\partial t} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} + \nabla p - \frac{1}{Re}\nabla^2\boldsymbol{u} - f_B(\theta)\,\boldsymbol{e}_y - A(\theta)\boldsymbol{u} = 0, \tag{8}$$

$$\frac{\partial(C\theta)}{\partial t} + \nabla\cdot(C\theta\boldsymbol{u}) - \nabla\cdot\left(\frac{K}{RePr}\nabla\theta\right) + \frac{\partial(CS)}{\partial t} = 0. \tag{9}$$

The linearized Boussinesq buoyancy force (f_B), the Reynolds (Re) and Prandtl (Pr) numbers are defined as:

$$f_B(\theta) = \frac{Ra}{PrRe^2}\theta,\quad Re = \frac{\rho V_{ref}H}{\mu_l} = \frac{V_{ref}H}{\nu_l},\quad Pr = \frac{\nu_l}{\alpha_l}, \tag{10}$$

with ν the kinematic viscosity and $\alpha = k/(\rho c)$ the thermal diffusivity. In the expression of f_B, the Rayleigh number of the flow is defined as:

$$Ra = \frac{g\beta H^3\delta T}{\nu_l\alpha_l}, \tag{11}$$

with β the thermal expansion coefficient and g the gravitational acceleration.

If previous non-dimensional numbers are pertinent only in the liquid phase, the non-dimensional conductivity and specific heat are defined in both phases

$$K(\theta) = \frac{k}{k_l} = \begin{cases} 1, & \theta \geq \theta_f, \\ k_s/k_l, & \theta < \theta_f. \end{cases},\quad C(\theta) = \frac{c}{c_l} = \begin{cases} 1, & \theta \geq \theta_f, \\ c_s/c_l, & \theta < \theta_f. \end{cases} \tag{12}$$

The non-dimensional function $S = s/s_l$ in the energy equation (9) takes a similar non-dimensional form:

$$S(\theta) = \frac{s}{s_l} = \begin{cases} \dfrac{h_{sl}/c_l}{\delta T} = \dfrac{1}{Ste}, & \theta \geq \theta_f, \\ 0, & \theta < \theta_f, \end{cases} \tag{13}$$

with Ste the Stefan number.

Discontinuous step-functions defined in (12) and (13) are replaced by continuous and differentiable hyperbolic-tangent functions, generically defined for all θ by the formula [32]:

$$F(\theta; a_s, \theta_s, R_s) = f_l + \frac{f_s - f_l}{2}\left\{1 + \tanh\left(a_s\left(\frac{\theta_s - \theta}{R_s}\right)\right)\right\}, \tag{14}$$

[2] In the liquid phase, $\nabla\cdot(cs\boldsymbol{U}) = h_{sl}\nabla\cdot\boldsymbol{U} = 0$; in the solid phase, $s = 0$; in the mushy region, $\boldsymbol{U} = 0$.

where f_l, f_s are the imposed values in the liquid and solid phases, a_s a smoothing parameter, θ_s the central value (around which we regularize) and R_s the smoothing radius. For example, we use for the non-dimensional source term in (9) the following regularization over the artificial mushy region $\theta \in [-\varepsilon, \varepsilon]$:

$$S(\theta) = \frac{1}{Ste} - \frac{1}{2\,Ste}\left\{1 + \tanh\left(\frac{\theta_r - \theta}{R_s}\right)\right\}, \tag{15}$$

where θ_r is the central value around which we regularize (typically $\theta_r = \theta_f = 0$) and R_r the smoothing radius (typically $R_s = \varepsilon$).

Finally, the penalty term $A(\theta)\boldsymbol{u}$ in momentum equation (8) follows from the Carman–Kozeny model [30,38,53]:

$$A(\theta) = -\frac{C_{CK}(1 - L_f(\theta))^2}{L_f(\theta)^3 + b}, \tag{16}$$

where $L_f(\theta)$ is the local liquid fraction, which is 1 in the fluid region and 0 in the solid. L_f is regularized inside the artificial mushy-region using a hyperbolic-tangent similar to (15). The Carman–Kozeny constant C_{CK} is set to a large value (as discussed below) and the constant $b = 10^{-6}$ is introduced to avoid divisions by zero.

3. Numerical method

To solve the system of equations (7)–(9) we use a finite-element method that was implemented using the open-source software FreeFem++ [50,51]. Among the numerous numerical tools offered by FreeFem++, the use of the powerful mesh adaptivity function proved mandatory in this study to obtain accurate results within reasonable computational time. The numerical code was optimized to afford the mesh refinement every time step: the mesh density was increased around the artificial mushy region, offering an optimal resolution of the large gradients of all regularized functions (S, K, L_f), while the mesh was coarsened (larger triangles) in the solid part, where a coarser mesh could be used. A simulation using a globally refined mesh would require a prohibitive computational time for an equivalent accuracy of the melting front resolution. Similar algorithms based on FreeFem++ were successfully used for solving different systems of equations with locally sharp variation of the solution, such as Gross–Pitaevskii equation [54,55] or Laplace equations with nonlinear source terms [56].

The space discretization is based on Taylor–Hood finite elements, approximating the velocity with P_2 Lagrange finite elements (piecewise quadratic), and the pressure with P_1 finite elements (piecewise linear). The temperature and the enthalpy are discretized using either P_1 or P_2 finite elements. For the time integration, we use a second-order Gear (BDF2) scheme (see also [38]):

$$\frac{d\phi}{dt} \simeq \frac{3\phi^{n+1} - 4\phi^n + \phi^{n-1}}{2\delta t}, \tag{17}$$

computing the solution ϕ^{n+1} at time $t_{n+1} = (n+1)\delta t$ by using two previous states (ϕ^n, ϕ^{n-1}). We use this scheme to advance in time both velocity $(\phi = \boldsymbol{u})$ and temperature fields $(\phi = \theta)$. The other terms in Eqs. (7)–(9) are treated implicitly (*i.e.* taken at time t_{n+1}). The resulting non-linear equations are solved using a Newton algorithm.

3.1. Finite element algorithm

Finite-element methods for solving Navier–Stokes type systems of equations are generally based on a separate discretization of the temporal derivative (using finite difference, splitting or characteristics methods) and the generalization of the Stokes problem for the resulting system [57–59]. To simplify the presentation, we consider in the following that $C = 1$ and $K = 1$. For phase-change materials like paraffins this is a physically valid assumption. Nevertheless, we keep in the equations below the variable K, since for the water freezing case it is necessary to take into account the dependence $K(\theta)$. The derivation of supplementary terms associated to $K(\theta)$ is straightforward.

We use the second-order finite-difference scheme (17) and obtain the following implicit semi-discretization in time of the single-domain model (7)–(9):

$$\nabla \cdot \boldsymbol{u}^{n+1} = 0, \tag{18}$$

$$\frac{3}{2}\frac{\boldsymbol{u}^{n+1}}{\delta t} + (\boldsymbol{u}^{n+1} \cdot \nabla)\boldsymbol{u}^{n+1} + \nabla p^{n+1} - \frac{1}{Re}\nabla^2 \boldsymbol{u}^{n+1} - A(\theta^{n+1})\boldsymbol{u}^{n+1} - f_B(\theta^{n+1})\,\boldsymbol{e}_y = 2\frac{\boldsymbol{u}^n}{\delta t} - \frac{\boldsymbol{u}^{n-1}}{2\delta t}, \tag{19}$$

$$\frac{3}{2}\frac{\theta^{n+1} + S(\theta^{n+1})}{\delta t} + \nabla \cdot \left(\boldsymbol{u}^{n+1}\theta^{n+1}\right) - \nabla \cdot \left(\frac{K}{RePr}\nabla\theta^{n+1}\right) = 2\frac{\theta^n + S(\theta^n)}{\delta t} - \frac{\theta^{n-1} + S(\theta^{n-1})}{2\delta t}. \tag{20}$$

This system of non-linear equations is solved at time $t_{n+1} = (n+1)\delta t$, using two previous states: t_n and t_{n-1}. We denote by $f_B(\theta)$ the Boussinesq force that can be non-linear in the general case (*e.g.* natural convection or freezing of water).

The space discretization of variables over the domain $\Omega \subset \mathbb{R}^2$ uses a finite-element method based on a weak formulation of the system (18)–(20). We consider homogeneous Dirichlet boundary conditions for the velocity, *i.e.* $\boldsymbol{u} = 0$ on $\partial\Omega$, and set the classical Hilbert spaces for the velocity and pressure:

$$\begin{aligned} &\boldsymbol{V} = V \times V,\ V = H_0^1(\Omega), \\ &Q = L_0^2(\Omega) = \left\{ q \in L^2(\Omega) \,\middle|\, \int_\Omega q = 0 \right\}. \end{aligned} \tag{21}$$

Following the generalization of the Stokes problem [57–59], the variational formulation of the system (18)–(20) can be written as: find $(\boldsymbol{u}^{n+1}, p^{n+1}, \theta^{n+1}) \in \boldsymbol{V} \times Q \times V$, such that:

$$b\left(\boldsymbol{u}^{n+1}, q\right) - \gamma(p^{n+1}, q) = 0, \quad \forall\, q \in Q \tag{22}$$

$$\frac{3}{2\delta t}\left(\boldsymbol{u}^{n+1}, \boldsymbol{v}\right) + c\left(\boldsymbol{u}^{n+1}; \boldsymbol{u}^{n+1}, \boldsymbol{v}\right) + \frac{1}{Re}a\left(\boldsymbol{u}^{n+1}, \boldsymbol{v}\right) - \left(A(\theta^{n+1})\,\boldsymbol{u}^{n+1}, \boldsymbol{v}\right) + b\left(\boldsymbol{v}, p^{n+1}\right) - \left(f_B(\theta^{n+1})\,\boldsymbol{e}_y, \boldsymbol{v}\right) = \frac{2}{\delta t}\left(\boldsymbol{u}^n, \boldsymbol{v}\right) - \frac{1}{2\delta t}\left(\boldsymbol{u}^{n-1}, \boldsymbol{v}\right), \quad \forall\, \boldsymbol{v} \in \boldsymbol{V} \tag{23}$$

$$\frac{3}{2\delta t}\left(\theta^{n+1} + S(\theta^{n+1}), \phi\right) + \left(\boldsymbol{u}^{n+1} \cdot \nabla\theta^{n+1}, \phi\right) + \left(\frac{K}{RePr}\nabla\theta^{n+1}, \nabla\phi\right) = \frac{2}{\delta t}\left(\theta^n + S(\theta^n), \phi\right) - \frac{1}{2\delta t}\left(\theta^{n-1} + S(\theta^{n-1}), \phi\right), \quad \forall\, \phi \in V, \tag{24}$$

where $(u, v) = \int_\Omega u \cdot v$ denotes the scalar product in $L^2(\Omega)$ or $\left(L^2(\Omega)\right)^2$; the bilinear forms a, b and trilinear form c are defined as [58,59]:

$$a : \boldsymbol{V} \times \boldsymbol{V} \to \mathbb{R},\ a(\boldsymbol{u}, \boldsymbol{v}) = \int_\Omega \nabla^t \boldsymbol{u} : \nabla \boldsymbol{v} = \sum_{i,j=1}^{2} \int_\Omega \partial_j u_j \cdot \partial_j v_i,$$

$$b : \boldsymbol{V} \times Q \to \mathbb{R},\ b(\boldsymbol{u}, q) = -\int_\Omega \nabla \cdot \boldsymbol{u}\, q = -\sum_{i=1}^{2} \int_\Omega \partial_i u_i \cdot q,$$

$$c : \boldsymbol{V} \times \boldsymbol{V} \times \boldsymbol{V} \to \mathbb{R},\ c(\boldsymbol{w}; \boldsymbol{z}, \boldsymbol{v}) = \int_\Omega [(\boldsymbol{w} \cdot \nabla)\boldsymbol{z}] \cdot \boldsymbol{v} = \sum_{i,j=1}^{2} \int_\Omega w_j (\partial_j z_i) v_i.$$

Note that we introduced in Eq. (22) a penalty term on the pressure. Its role is analysed below at different levels of the implementation of the numerical method. At the discrete level of model equations, we infer from the incompressibility constraint (18) that the pressure is defined up to an additive constant. Requesting that $p^{n+1} \in Q$, *i.e.* the pressure field is of zero average in Ω, removes this uncertainty. For the discretized equations, the zero average constraint is easy to apply in spectral methods by, for instance, cancelling the first Fourier mode, corresponding to the average value [59]. This task becomes more difficult when local approximation numerical methods (as finite differences or finite elements) are used. In these cases, a usual technique is to explicitly prescribe the value of the pressure at a given discrete point. This implies to explicitly modify the matrix of the final linear system. Equation (22) has exactly this role. It is theoretically justified by the fact that the range space of $\nabla \cdot \boldsymbol{u}$ is contained in $Q = L^2_0(\Omega)$, which can be identified isometrically with $L^2(\Omega)/\mathbb{R}$ [58]. At a strictly algebraic level, the penalty term in (22) is introduced to avoid a zero lower diagonal block in the final global matrix. Even though the non-penalized problem is well-posed for finite-elements satisfying the inf–sup condition (as the Taylor–Hood element used here), the zero entries on this (pressure) diagonal block would require pivoting strategies when a direct solver is used for the solution of the linear algebraic system. Using (22), this block is filled with the matrix $(-\gamma M)$, where M corresponds to the mass matrix of the P_1 discretization used for the pressure. Efficient direct methods using LU-type solvers become then available to solve the linear system (UMFPACK solver in our case and [33], SuperLU in [34]).

A last comment on the penalty term on the pressure in (22) concerns its stabilizing effects. Since the Taylor–Hood finite elements used in our simulations satisfy the inf–sup condition, this technique is assimilated to a *stable penalty*, or, in other words, just a computational trick to obtain a good solution (see Remark 4.6, page 86 in [49]) (see Example 6.3.5, page 372 in [60]). The value of the penalty parameter γ could be arbitrarily small, without affecting the overall convergence rate. It is interesting to note that this technique could be used to stabilize finite element formulations that do not satisfy the inf–sup condition (see Remark 4.3, page 67 in [58]). In this case, it is assimilated to a *brute-force penalty* (see also Example 6.3.5 in [60]) and the value of the penalty parameter γ has to be carefully chosen to avoid altering the convergence of the method: it depends on the grid size and the theoretical convergence rate of the used finite elements [60], and also, when applied to Stokes or Navier–Stokes equations, on the viscosity [61].

As a final remark, we should mention that the large body of literature on penalty methods for Stokes or Navier–Stokes equations originates from the *slightly compressible* fluid model introduced by Temam [57]. In this model, Eq. (22) is used to eliminate the pressure, by assuming that $p^{n+1} = (-1/\gamma)\nabla \cdot \boldsymbol{u}^{n+1}$, and finally obtain a momentum equation (23) with a regularization term $(1/\gamma)\left(\nabla \cdot \boldsymbol{v}, \nabla \cdot \boldsymbol{u}^{n+1}\right)$ replacing the pressure term. The equivalence between the two formulations was studied by Bercovier [62] (see also [58,59]). In our calculations, we use in Eq. (22) very low values of the penalty parameter ($\gamma = 10^{-7}$). This ensures very low values of the average on Ω for the pressure and also for the divergence of the velocity field, *e.g.* $\int_\Omega p$ of order of 10^{-11} and $\int_\Omega \nabla \cdot \boldsymbol{u}^{n+1}$ of order of 10^{-18} for the case of the natural convection of air.

The system of non-linear equations (22)–(24) is solved using a Newton method. To advance the solution from time t_n to t_{n+1}, we start from an initial guess $\boldsymbol{w}_0 = (\boldsymbol{u}^n, p^n, \theta^n)$ (which is the solution at t_n), and construct the Newton sequence $\boldsymbol{w}_k = (u_k, p_k, \theta_k)$ by solving for each inner iteration k:

$$b(\boldsymbol{u}_{k+1}, q) - \gamma(p_{k+1}, q) = 0, \tag{25}$$

$$\begin{aligned} &\frac{3}{2\delta t}(\boldsymbol{u}_{k+1}, \boldsymbol{v}) + c(\boldsymbol{u}_{k+1}; \boldsymbol{u}_k, \boldsymbol{v}) + c(\boldsymbol{u}_k; \boldsymbol{u}_{k+1}, \boldsymbol{v}) \\ &+ \frac{1}{Re} a(\boldsymbol{u}_{k+1}, \boldsymbol{v}) - \left(\frac{dA}{d\theta}(\theta_k)\theta_{k+1}\boldsymbol{u}_k, \boldsymbol{v}\right) - (A(\theta_k)\boldsymbol{u}_{k+1}, \boldsymbol{v}) + b(\boldsymbol{v}, p_{k+1}) \\ &- \left(\frac{df_B}{d\theta}(\theta_k)\theta_{k+1}\boldsymbol{e}_y, \boldsymbol{v}\right) = \frac{1}{\delta t}\left(2\boldsymbol{u}^n - \frac{1}{2}\boldsymbol{u}^{n-1}, \boldsymbol{v}\right) \\ &+ c(\boldsymbol{u}_k; \boldsymbol{u}_k, \boldsymbol{v}) - \left(\frac{dA}{d\theta}(\theta_k)\theta_k\boldsymbol{u}_k, \boldsymbol{v}\right) - \left(\left(\frac{df_B}{d\theta}(\theta_k)\theta_k - f_B(\theta_k)\right)\boldsymbol{e}_y, \boldsymbol{v}\right), \end{aligned} \tag{26}$$

$$\begin{aligned} &\frac{3}{2\delta t}\left(\theta_{k+1} + \frac{dS}{d\theta}(\theta_k)\theta_{k+1}, \phi\right) + (\boldsymbol{u}_k \cdot \nabla\theta_{k+1}, \phi) + (\boldsymbol{u}_{k+1} \cdot \nabla\theta_k, \phi) \\ &+ \left(\frac{K}{RePr}\nabla\theta_{k+1}, \nabla\phi\right) = \frac{2}{\delta t}\left(\theta^n + S(\theta^n), \phi\right) \\ &+ (\boldsymbol{u}_k \cdot \nabla\theta_k, \phi) + \frac{3}{2\delta t}\left(\frac{dS}{d\theta}(\theta_k)\theta_k - S(\theta_k), \phi\right) - \frac{1}{2\delta t}\left(\theta^{n-1} + S(\theta^{n-1}), \phi\right). \end{aligned} \tag{27}$$

Note that the last term of Eq. (26) cancels in the case of a linear Boussinesq force f_B (see Eq. (10)); this is not the case when non-linear variations of the density of the liquid are considered (convection or solidification of water). Note also that the previous system of equations (25)–(27) depends only on $\boldsymbol{u}^n$, $\boldsymbol{u}^{n-1}$, θ^n and θ^{n-1} and is independent of p^n, the pressure being in this approach a Lagrange multiplier for the divergence free constraint.

The Newton loop (following k) has to be iterated until convergence for each time step δt following the algorithm:

```
Navier–Stokes time loop following n
set w_0 = (u^n, p^n, θ^n)
    Newton iterations following k
        solve (25)–(27) to get w_{k+1}
    stop when ‖w_{k+1} − w_k‖ < ξ_N
actualize (u^{n+1}, p^{n+1}, θ^{n+1}) = w_{k+1}.
```
(28)

The FreeFem++ syntax to implement the Newton algorithm is very close to the mathematical formulation given above. After defining a vectorial finite-element space `fespace Wh(Th,[P2,P2,P1,P1]);`, associated to the mesh `Th`, we define the velocity, pressure and temperature variables in a compact manner by `Wh [u1,u2,p,T];`. Corresponding test functions are defined similarly. It is then very easy to define a `problem` formulation in FreeFem++ and include all the terms of the algorithm (25)–(27). This makes the reading of the programs very intuitive by comparing each term to its mathematical expression. New terms could be added to the variational formulation expressed in the `problem` structure, without affecting other parts of the program. Consequently, the implementation of new models or numerical methods for this problem is greatly facilitated by this modular structure of programs.

3.2. Mesh adaptivity

We use the standard function (`adaptmesh`) which is a very convenient tool offered by FreeFem++ to efficiently adapt 2D meshes by metrics control [50]. The key idea implemented in this function (see also [63–68]) is to modify the scalar product used in the automatic mesh generator to evaluate distance and volume. Equilateral elements are thus constructed, according to the new metric. The scalar product is based on the evaluation of the Hessian $\mathcal{H}$ of the variables of the problem. For example, for a P_1 discretization of a variable χ, the interpolation error is bounded by:

$$\mathcal{E} = |\chi - \Pi_h\chi|_0 \le c \sup_{T\in\mathcal{T}_h} \sup_{x,y,z\in T} |\mathcal{H}(x)|(y-z, y-z), \tag{29}$$

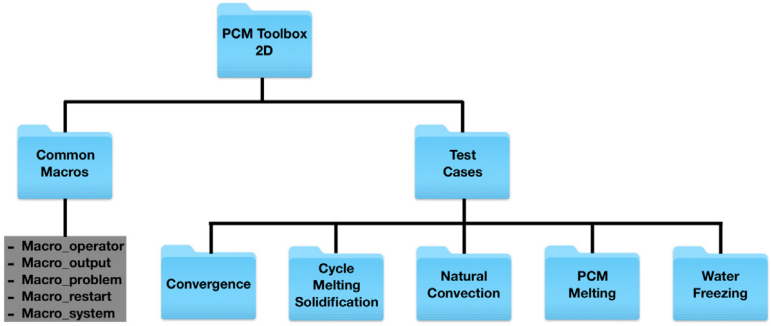

Fig. 1. Folder tree structure of the PCM-Toolbox-2D to solve solid–liquid phase-change problems.

where $\Pi_h\chi$ is the P_1 interpolate of χ, $|\mathcal{H}(x)|$ is the Hessian of χ at point x after being made positive definite. Using the Delaunay algorithm (*e.g.* [66]) to generate a triangular mesh with edges close to the unit length in the metric $\mathcal{M} = \frac{|\mathcal{H}|}{(c\mathcal{E})}$ will result in an equally distributed interpolation error $\mathcal{E}$ over the edges a_i of the mesh. More precisely, we get

$$\frac{1}{c\mathcal{E}} a_i^T \mathcal{M} a_i \leq 1. \tag{30}$$

Note that the FreeFem++ function `buildmesh` efficiently implements the Delaunay algorithm for generating triangular meshes. The mesh generator offered by FreeFem++ is generally sufficient for 2D applications, even when domains with complicated shapes are considered, but meshes may be imported in FreeFem++ from other software, *e.g.* `Gmsh`.

The previous approach could be generalized for a vector variable $\chi = [\chi_1, \chi_2]$. After computing the metrics $\mathcal{M}_1$ and $\mathcal{M}_2$ for each variable, we define a metric intersection $\mathcal{M} = \mathcal{M}_1 \cap \mathcal{M}_2$, such that the unit ball of $\mathcal{M}$ is included in the intersection of the two unit balls of metrics $\mathcal{M}_2$ and $\mathcal{M}_1$ (for details, see the algorithm defined in [67]).

The possibility to use the standard FreeFem++ `adaptmesh` function was one of the main advantages that decided in choosing this software to implement our algorithm to solve phase-change problems. No supplementary script layers or modifications of the standard function were necessary and the adaptivity part of the FreeFem++ scripts contains only a single instruction line. Recently, Zimmerman and Kowalski [46] used the FEniCs finite element software offering a dual-weighted residual method for mesh adaptivity purposes. Using this method with an AMR (Adaptive Mesh Refinement) technique for phase change problems displayed a major drawback: the mesh was refined during the advancement of the liquid–solid front, but never coarsened behind. The metrics control used by the FreeFem++ `adaptmesh` function offers the ability to simultaneously refine the mesh in regions with strong gradients and coarsen the mesh in regions with low gradients. This function proved very efficient and versatile for phase-change problems by offering the possibility to take into account several metrics computed from different variables monitoring the time-evolution of the system. For natural convection systems, the mesh was adapted using the values of the two velocity components and the temperature. For phase-change systems, to accurately track the solid–liquid interface we added the enthalpy source term as a variable in the adaptivity criterion. For water systems (convection or freezing), we also added an extra function tracking the anomalous change of density around 4 °C. To monitor the time change of these variables used to compute the metrics intersection, we considered for each variable the metrics computed at actual (t_{n+1}) and previous (t_n) time instants. This technique also reduces the impact of the interpolation on the global accuracy for time-depending problems (see also [44]). The anisotropy of the mesh is a parameter of the algorithm and it was set to values close to 1. This is an inevitable limitation since we also impose the minimum edge-length of triangles to avoid too large meshes.

The capabilities of the mesh adaptivity algorithm are illustrated in Section 6. It is important to emphasize that the standard mesh adaptivity and interpolation tools offered by FreeFem++ are very efficient. The CPU time needed to adapt the mesh, using up to six variables for the metrics evaluation, do not exceed 4% of the CPU time requested to advance the solution by a time step. The exact values of CPU times for the melting cases presented in this paper are reported in Table 3. Given this computational efficiency of the adaptivity tool, we could afford to refine the mesh every time step during the computation, even when complex domain shapes or physics were considered.

4. Description of the programs

In this section, we first describe the architecture of the programs and the organization of provided files. Then we focus on the list of input parameters and the structure of output files.

4.1. Program architecture

Figure 1 gives a schematic overview of the content of the toolbox. All files are provided in a directory called `PCM-Toolbox-2D`. Many detailed comments are included in the programs, with direct link to the mathematical expressions used in the paper. The syntax was intentionally kept at a low level of technicality and supplemented with detailed comments when specific more technical syntax was used.

This directory is organized as follows:

1. The directory `Common-Macros` contains five files:
   - *Macro_operator.idp* includes macros and functions defining mathematical operators,
   - *Macro_problem.idp*: macros defining the variational formulation of the problem,
   - *Macro_restart.idp*: macros used to start a new simulation from a saved field,
   - *Macro_output.idp*: macros used to save the solution with different formats,
   - *Macro_system.idp*: macros identifying the OS and defining specific OS-commands.
2. The directory `Test-Cases` contains four subdirectories, each of them defining one of the following applications:
   - test of the convergence of the method using manufactured solutions,
   - natural convection of air or water in a differentially heated square cavity,
   - melting of a PCM stored in containers of different shapes,
   - melting followed by solidification of a rectangular PCM,

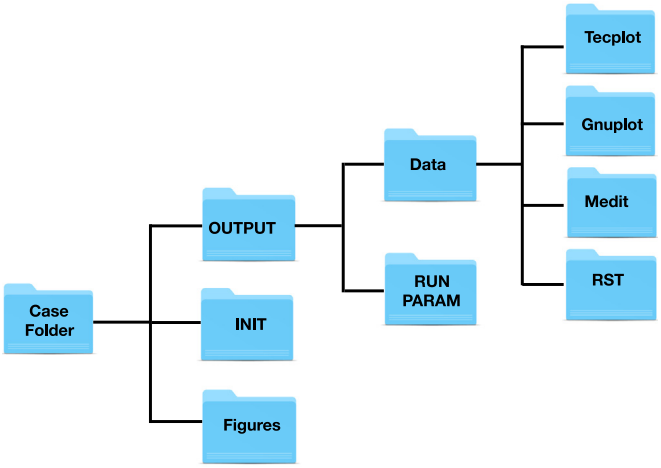

Fig. 2. Structure of each Test-case folder.

• freezing of pure water in a square cavity.

Each subdirectory contains three files: *NEWTON_$case.edp* is the main FreeFem++ script file, *param_phys.inc* defines the physical parameters and *param_num.inc* the numerical parameters. For example, to run the stationary natural convection case of air in a square cavity, one can use the following command in a terminal window: `FreeFem++ NEWTON_stat_natconv.edp`.

The folder structure of each test case is illustrated in Fig. 2. The obtained solutions are saved in the folder `OUTPUT/Data`. Depending on the output format selected by the user, data files are generated in specific folders for being visualized with: Tecplot, Paraview, Gnuplot or Medit. We also provide in the folder `Figures` ready-made layouts for these visualization softwares. The user can thus obtain the figures from this paper using newly generated data. More details about the output structure are given below.

4.2. Input parameters

Physical parameters and parameters related to the run are separated into two files.

(1) The file *param_phys.inc* contains the physical descriptions of the problem:

- **typeT**: is the finite-element type for the temperature, with possible values `P2` or `P1`,
- **Torder**: is the accuracy order of the time integration scheme, with possible values 1 (Euler scheme) or 2 (Gear scheme),
- **scalAdim**: defines the characteristic scales of the problem, see (6). Possible values 1, 2 or 3 correspond to the following choice of the characteristic scales [32]:

$$(1): V_{ref}^{(1)} = \frac{\nu_l}{H} \Longrightarrow t_{ref}^{(1)} = \frac{H^2}{\nu_l} \Longrightarrow \mathcal{R}e = 1, \tag{31}$$

$$(2): V_{ref}^{(2)} = \frac{\alpha}{H} \Longrightarrow t_{ref}^{(2)} = t_{ref}^{(1)}\mathcal{P}r \Longrightarrow \mathcal{R}e = 1/\mathcal{P}r, \tag{32}$$

$$(3): V_{ref}^{(3)} = \frac{\nu_l}{H}\sqrt{\frac{\mathcal{R}a}{\mathcal{P}r}} \Longrightarrow t_{ref}^{(3)} = t_{ref}^{(1)}\sqrt{\frac{\mathcal{P}r}{\mathcal{R}a}} \Longrightarrow \mathcal{R}e = \sqrt{\frac{\mathcal{R}a}{\mathcal{P}r}}, \tag{33}$$

- $\mathbf{x}_l$, $\mathbf{x}_r$, $\mathbf{y}_l$, $\mathbf{y}_r$: are the values defining the dimensions of the cavity $[x_l, x_r] \times [y_l, y_r]$,
- **Pr, Ra, Ste**: are the Prandtl, Rayleigh and Stefan numbers, see (11) and (10),
- $\mathbf{T}_{hot}$, $\mathbf{T}_{cold}$: are dimensionless temperatures according to (6),
- $\mathbf{bcu}_1$, $\mathbf{bcu}_2$, **bcT**: are macros defining the velocity (u) and the temperature (T) boundary conditions.
- **epsi**: is the half width ε of the mushy region. Default value $= 0.01$,
- **dt**: is the dimensionless time step,
- $\mathbf{t}_{max}$: is the dimensionless final time,
- **Parameters for regularization functions**:
  The parameters of the hyperbolic-tangent function (14) used to regularize discontinuous functions are set by default as follows:

	f_s	f_l	a_s	θ_s	R_s	C_{CK}	b
Enthalpy	0	1/Ste	1	0.01	0.01	–	–
Carman–Kozeny	0	1	1	0.01	0.01	10^6	10^{-7}
Conductivity (water)	1	2.26/0.578	1	θ_f	0.015	–	–

- **rho(T) and Drho(T)**: (water cases only) define the density and its derivative as functions of the temperature, following the model [69]:

$\rho(T) = \rho_m(1 - \omega\lvert T - T_m\rvert^q)$,			
ρ_m [kg/m^3]	ω [°C^{-q}]	q	T_m [°C]
999.972	$9.2793 \cdot 10^{-6}$	1.894816	4.0293

- $\mathbf{f}_B(\mathbf{T})$, $\mathbf{df}_B(\mathbf{T})$: define the buoyancy force and its derivative.

(2) The file *param_num.inc* contains the parameters controlling the run.

Restart parameters:

- **Nsave**: the solution is saved every *Nsave* time steps in the `Data` folder (see Fig. 2). The temperature and the velocity fields are saved in `Tecplot` and `Medit` folders, while the liquid fraction, the Nusselt number, and the accumulated heat input are saved in the `Gnuplot` folder.
- **Nrestart**: restart files (mesh and solution) are saved every *Nrestart* time steps. Solutions at current and previous iterations, the CPU time, the accumulated heat input Q_0, and the time step *dt* are saved in the folder `RST`.
- **Ncondt**: allows the user to stop the run and save the solution properly. The file `OUTPUT/zz.condt` is read every *Ncondt* time steps: if the user replaces the value "0" in this file by "1" the run is stopped. This is a simple solution for a clean stop of the job by the user. Default value $= 20$.
- **Nremesh**: the mesh is adapted every *Nremesh* iterations. If this parameter is set to "1" the mesh is adapted every time step.
- **IFrestart**: is a Boolean controlling the set up of the initial field.
  IFrestart $= 0$, the initial condition is built in the code for each test case. For the PCM melting cases, the PCM is initially motionless at isothermal temperature. To set-up a smooth initial field, a few time steps (with very small δt) are computed by increasing progressively the boundary temperature at the hot wall (by continuation). Outputs are saved in `OUTPUT/Data-RST-0`.
  IFrestart > 0, (positive integer values) the solution field previously computed at iteration *IFrestart* is loaded from the folder `OUTPUT/Data-RST-filenameRST/RST`, with `filenameRST` a variable selecting the restart folder.
  IFrestart < 0, (negative integer values), the same principle for loading a solution is used, but from the folder `INIT`

(see Fig. 2). The solution fields stored in this folder could come from different previous calculations (*e.g.* a steady state solution or, for the water, the natural convection field before freezing).

Newton parameters:

- **epsconv**: is the value of the stopping criterion for steady cases,
- **gamma**: is the penalty parameter in (18). Default value $= 10^{-7}$,
- **tolNewton**: is the Newton tolerance ξ_N (see (28)). Default value $= 10^{-6}$,
- **newtonMax**: limits the maximum number of iterations in the Newton algorithm (28). Default value $= 50$,

Mesh parameters:

- **nbseg**: is the number of segments for the discretization along the x and y directions,
- **errh**: is the interpolation error level. Default value $= 0.02$,
- **hmin, hmax**: are the minimum and maximum edge size, respectively,
- **adaptratio**: is the ratio for a prescribed smoothing of the metric. For a value less than 1.1 no smoothing is done. Default value $= 1.5$,
- **nbvx**: is the maximum number of vertices allowed in the mesh generator. Default value $= 50000$.

Output parameters:

- **dircase**: is the name of the output folder,
- **fcase**: is the prefix-name for output files.
- **Tecplot, Medit, Gnu**: correspond to the name of the visualization software to be used; the format of the outputs written in `OUTPUT/Data` (see Fig. 2) is accordingly set. The files from the Tecplot folder can be easily read also with Paraview.

4.3. Outputs

When a computation starts, the `OUTPUT` directory is created (see Fig. 2). It contains two folders storing the output data and the echo of the run parameters. The folder `Data` contains four subdirectories with different output format files of the computed solution. File names are created using the prefix defined by the parameter **fcase**, the current iteration and the current dimensionless time t. Solution files can be visualized using either Tecplot or any other CFD Visualization tools (Paraview, Visit, etc.). Moreover, *.gmsh* (mesh) and *.rst* (fields) files are generated in the folder `RST` to enable restarts of the computation. Note that the folder `FFglut` contains FreeFem++ scripts that re-read and visualize the RST-files to facilitate the selection of a restart field. An *.echo* file with a summary of the main parameters, information on the run and the names of the output files is saved in the folder `RUNPARAM`. This directory additionally contains a copy of the *.inc* parameter files, allowing an easy identification of each case and preparing an eventual rerun of the same case.

5. Numerical tests of the accuracy of the numerical method

We start by presenting tests of the accuracy of our numerical method. We used the technique of manufactured solutions (*e.g.* [70]) which has the advantage of providing an exact solution to a modified problem, related to the initial one. The general idea is to modify the original system of equations by introducing an extra source term, such that the new system admits an exact solution given by a convenient analytic expression. Even though in most cases exact solutions constructed in this way are not physically realistic, this approach allows one to rigorously verify computations.

We tested the space and time accuracy using manufactured solutions for the system of equations (7)–(9) for a stationary case (Burggraf flow) and a time-dependent one [71]. For both cases, we computed the global error ε for different norms in space:

$$\varepsilon = \|\Phi_e - \phi_h\|, \tag{34}$$

with Φ_e the exact solution and ϕ_h the numerical solution. Computations were performed for the convection of air ($C = K = 1$, $A(\theta) = S(\theta) = 0$), with a Rayleigh number $Ra = 10^4$ and a Prandtl number $Pr = 0.71$.

5.1. Space accuracy: Burggraf stationary flow with thermal effects

The Burggraf manufactured solution is a time-independent recirculating flow inside a square cavity $[0, 1]\times[0, 1]$. It is similar to the well-known entrained cavity flow, with the difference that the velocity singularity at the top corners of the cavity is avoided. We added to the classical Burggraf flow [72,73] a manufactured solution for the temperature, with constant temperature imposed at the top and the bottom walls. Vertical walls are assumed to be adiabatic. The exact solution of the new flow with thermal effects is:

$$\begin{aligned} u_1(x,y) &= \sigma g'(x)h'(y), \\ u_2(x,y) &= -\sigma g''(x)h(y), \\ p(x,y) &= \frac{\sigma}{Re}\left(h^{(3)}(y)g(x) + g''(x)h'(y)\right) \\ &\quad + \frac{\sigma^2}{2}g'(x)^2\left(h(y)h''(y) - h'(y)^2\right), \\ T(x,y) &= T_c + (T_h - T_c)y + a(x)b(y), \end{aligned} \tag{35}$$

with $\sigma > 0$ a scaling parameter and functions

$$\begin{aligned} g(x) &= \frac{x^5}{5} - \frac{x^4}{2} + \frac{x^3}{3}, \\ h(y) &= y^4 - y^2, \\ a(x) &= \cos(\pi x), \\ b(y) &= y(1-y). \end{aligned} \tag{36}$$

Note that the velocity at the top border of the cavity is:

$$u_1(0,1) = 2\sigma(x^4 - 2x^3 + x^2), \quad u_2(x,1) = 0, \tag{37}$$

which ensures the continuity of the velocity at corners ($\boldsymbol{u}(0,1) = \boldsymbol{u}(1,1) = 0$), since non-slip walls are imposed for the other borders: $\boldsymbol{u}(x,0) = \boldsymbol{u}(0,y) = \boldsymbol{u}(1,y) = 0$.

The forcing terms that have to be added to the momentum and energy (temperature) equation are derived by injecting the exact solution (35) into the system (7)–(9):

$$\begin{aligned} f_{u_1} &= 0, \\ f_{u_2} &= \sigma^2 h(y)h'(y)\left(g''(x)^2 - g'(x)g^{(3)}(x)\right) \\ &\quad + \frac{\sigma}{Re}\left(g^{(4)}(x)h(y) + 2g''(x)h''(y) + g(x)h^{(4)}(y)\right) \\ &\quad + \frac{\sigma^2}{2}g'(x)^2\left(h(y)h^{(3)}(y) - h'(y)h''(y)\right) - \frac{Ra}{PrRe^2}T(x,y), \\ f_T &= u_1(x,y)a'(x)b(y) + u_2(x,y)\left(T_h - T_c + a(x)b'(y)\right) \\ &\quad - \frac{K}{RePr}\left(a''(x)b(y) + a(x)b''(y)\right). \end{aligned} \tag{38}$$

We used the Taylor–Hood finite element (P_2 for the velocity and P_1 for the pressure) and tested P_1 or P_2 finite elements for the temperature. Figure 3 illustrates the streamlines and the

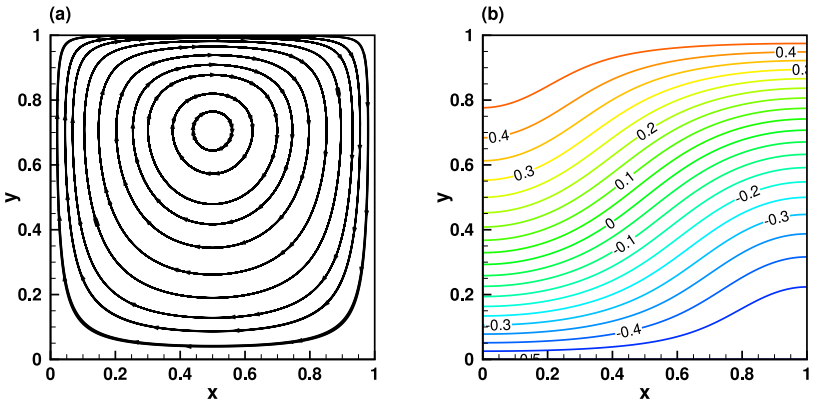

Fig. 3. Burggraf stationary flow with thermal effects used to test the space accuracy of the numerical scheme. Streamlines (a) and temperature contours (b) of the flow field.

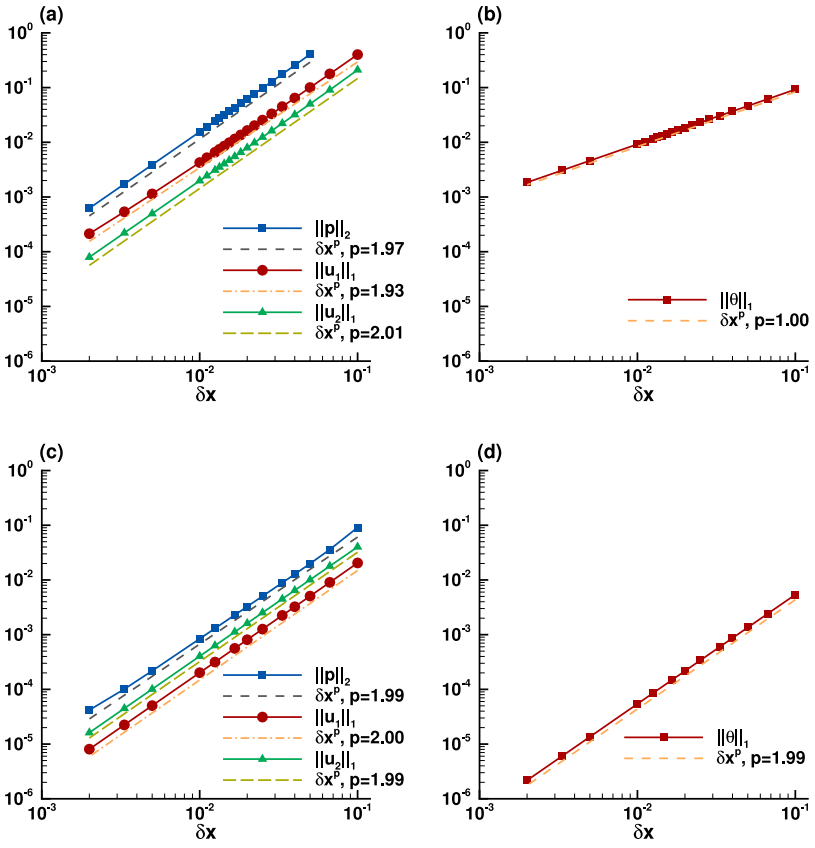

Fig. 4. Space accuracy of the numerical scheme tested using the Burggraf manufactured solution. The global error ε (34) was computed using natural norms: L^2-norm for the pressure p and H^1-norm for velocity components u_1, u_2 and temperature θ. Taylor–Hood P_2/P_1 finite elements are used for the velocity and pressure. Discretization of the temperature using P_1 (plots a and b) and P_2 (plots c and d) finite elements.

temperature field. Figure 4 plots the discretization error ε as a function of the grid size $h = \delta x = \delta y$. The grid was not adapted for these computations. Errors are measured in natural norms: H^1-norm for the velocity components u_1, u_2 and temperature θ, and L^2-norm for the pressure p. Figures 4a and b correspond to the simulation using P_1 finite elements for the temperature, while in Figs. 4c and d P_2 finite elements were used for the temperature. Convergence rates computed by a least-squares fit of the slopes of the curves are displayed for reference. The optimal order of accuracy ($\mathcal{O}(h^2)$) corresponding to the Taylor–Hood discretization of the flow field is obtained (Figs. 4a and c), independently of the temperature discretization. Expected orders of accuracy for the temperature are confirmed in Figs. 4b (first order) and 4d (second order) for the two implementations. In the subsequent simulations, we use the P_2 discretization of the temperature, offering an optimal coupling between momentum and energy equations (note in Fig. 4 the lower values of the errors obtained with this discretization). Similar discretizations using Taylor–Hood finite

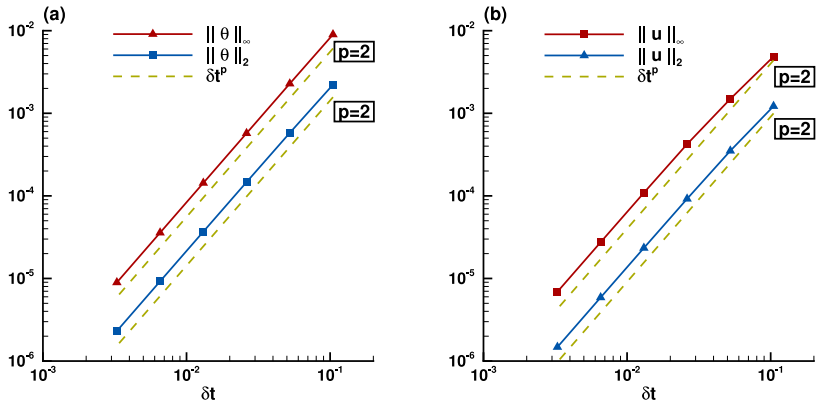

Fig. 5. Time accuracy of the numerical scheme tested using the time-dependent manufactured solution of Nourgaliev et al. [71]. Global error ε (34) computed at $t_{max} = \pi$ for the temperature θ (a) and total velocity $u = \sqrt{u_1^2 + u_2^2}$ (b) using L^2 and infinity norms. Discretization using P_2 finite elements for the temperature.

Table 1
Parameter for the time-dependent manufactured solution (39).

γ_1	γ_2	$\bar{P}$	$\bar{T}$	δP_0	δT_0	δU_0	α_p	α_u	α_t
1	1	0	1.0	0.1	1.0	1.0	0.05	0.4	0.1

elements for the fluid flow and P_2 for the temperature were used in recent contributions by Woodfield et al. [34]; Belhamadia et al. [47].

5.2. Time accuracy: manufactured unsteady solution

To test the time accuracy of the Gear (BDF2) scheme, we used the manufactured time-dependent solution suggested in [71]:

$$
\begin{aligned}
u_1(x,y,t) = &\quad (\delta U_0 + \alpha_u \sin(t)) \cos(x+\gamma_1 t) \sin(y+\gamma_2 t), \\
u_2(x,y,t) = &\quad -(\delta U_0 + \alpha_u \sin(t)) \sin(x+\gamma_1 t) \cos(y+\gamma_2 t), \\
T(x,y,t) = &\quad \bar{T} + (\delta T_0 + \alpha_t \sin(t)) \cos(x+\gamma_1 t) \sin(y+\gamma_2 t), \\
p(x,y,t) = &\quad \bar{P} + \big(\delta P_0 + \alpha_p \sin(t)\big) \sin(x+\gamma_1 t) \cos(y+\gamma_2 t),
\end{aligned}
\tag{39}
$$

The values of the constants are reported in Table 1. The corresponding forcing source terms are:

$$
\begin{aligned}
f_{u_1} = &\ \alpha_u \cos(t) \cos(a) \sin(b) - U_c\, \gamma_1 \sin(a) \sin(b) \\
&+ U_c\, \gamma_2 \cos(a) \cos(b) \\
&- U_c\, u_1(x,y,t) \sin(a) \sin(b) + U_c\, u_2(x,y,t) \cos(a) \cos(b) \\
&+ P_c \cos(a) \cos(b) \\
&+ \frac{2}{Re}\, u_1(x,y,t), \\
f_{u_2} = &\ -\alpha_u \cos(t) \sin(a) \cos(b) - U_c\, \gamma_1 \cos(a) \cos(b) \\
&+ U_c\, \gamma_2 \sin(a) \sin(b) \\
&- U_c\, u_1(x,y,t) \cos(a) \cos(b) + U_c\, u_2(x,y,t) \sin(a) \sin(b) \\
&- P_c \sin(a) \sin(b) \\
&+ \frac{2}{Re}\, u_2(x,y,t) - \frac{Ra}{PrRe^2}\, T(x,y,t), \\
f_T = &\ \alpha_t \cos(t) \cos(a) \sin(b) - T_c\, \gamma_1 \sin(a) \sin(b) \\
&+ T_c\, \gamma_2 \cos(a) \cos(b) \\
&- T_c\, u_1(x,y,t) \sin(a) \sin(b) + T_c\, u_2(x,y,t) \cos(a) \cos(b) \\
&+ \frac{2K}{RePr}\, T_c \cos(a) \sin(b),
\end{aligned}
\tag{40}
$$

where $a = (x+\gamma_1 t)$, $b = (y+\gamma_2 t)$ and $U_c = (\delta U_0 + \alpha_u \sin(t))$, $T_c = (\delta T_0 + \alpha_u \sin(t))$, $P_c = (\delta P_0 + \alpha_u \sin(t))$.

Guided by the results obtained in Section 5.1 for the space accuracy, we fixed the grid size to $h = \delta x = 0.01$ to ensure small spatial discretization errors and used P_2 finite elements for the temperature. For diminishing values of the time step δt, the solution was evolved in time up to the time instant $t_{max} = \pi$ at which the error (34) was computed. The time convergence is displayed in Fig. 5a for the temperature and in Fig. 5b for the total velocity $u = \sqrt{u_1^2 + u_2^2}$. The expected second order convergence in time is obtained for all variables of the problem.

6. Numerical simulations of natural convection and phase-change problems

In this section we test the robustness of the distributed toolbox. We consider well defined benchmarks used to validate numerical codes for natural convection and phase-change problems. The difficulty of the computed cases is increased progressively by considering the following physical systems: (i) natural convection of air (Section 6.1), (ii) melting of a phase-change material (Section 6.2), (iii) alternate melting and solidification of a phase-change system (Section 6.3) and (iv) the convection and the freezing of pure water (Section 6.4). This approach allows us to test the programs by adding progressively non-linearities in the Newton algorithm. For each test case, we compare our results with experimental and previously published numerical data.

6.1. Natural convection of air

We start by testing the Newton algorithm (25)–(27) for the case of natural convection, *i.e.* $C = K = 1$, $A(\theta) = S(\theta) = 0$. We consider the classical problem of the thermally driven square cavity $[0, 1] \times [0, 1]$, filled with air. The Boussinesq term $f_B(\theta)$ is then linear and takes the form (10). Top and bottom walls are adiabatic, while the temperature is fixed on the left (hot) wall and the right (cold) wall. Natural convection flows are computed for three values of the Rayleigh number: $Ra = 10^4, 10^5, 10^6$. The Prandtl number is set to $Pr = 0.71$. It was shown in [74] that the flow in this configuration becomes unsteady for $Ra = 10^{8.5}$. Therefore, steady states could be computed for the chosen values of the Rayleigh number.

We provide programs for both steady (time-independent) and time-dependent cases. The steady case is performed using a continuation following the boundary value of the temperature: a small value for θ_c (or θ_h) is set initially and is then smoothly increased until reaching the wanted value. The time-dependent case is computed until a steady state with a single convection cell is reached. To make the simulation more complicated, we present

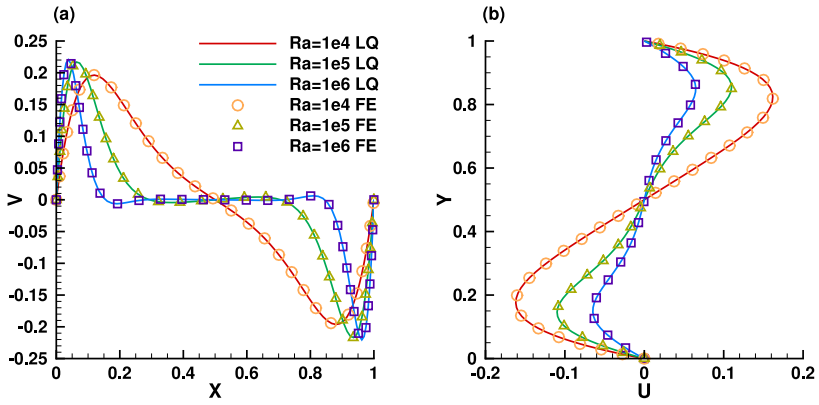

Fig. 6. Natural convection of air in a differentially heated cavity for values of Ra ranging from 10^4 to 10^6. (a) Vertical velocity profile $v(x)$ along the horizontal symmetry line ($y = 0.5$). (b) Longitudinal velocity profile $u(y)$ along the vertical symmetry line ($x = 0.5$). Results obtained using the present Newton method (symbols), with an initial mesh resolution $nbseg = 80$. Comparison with the spectral simulations by Le Quéré [74] (solid lines).

Table 2
Natural convection of air in a differentially heated cavity. Maximum value u_{max} of the horizontal velocity profile at mid-domain ($x = 0.5$) and location Y of this maximum. Comparison to reference values by Le Quéré [74].

Run		u_{max} at x = 0.5 (error)	Y (error)
Reference values	Spectral	0.0648344	0.850
Newton (Steady)	$nbseg = 80$	0.0648297 (0.007%)	0.850394 (0.05%)
Newton (Unsteady)	$nbseg = 80$	0.0648296 (0.007%)	0.850532 (0.06%)

two configurations: (*i*) the classical differentially heated square cavity and (*ii*) a differentially heated cavity with an inner heated obstacle.

All computations for the natural convection cases are performed using the stationary solver and mesh adaptivity. The initial mesh was generated using $nbseg = 80$ segments on each side, *i.e.* $h_{min} = 1/80 = 0.0125$.

6.1.1. Classical differentially heated square cavity

The temperature is imposed at the left (hot) wall as $\theta_h = 0.5$ and at the right (cold) wall as $\theta_c = -0.5$. Top and bottom walls are adiabatic. The initial condition models a cavity filled with motionless air ($\boldsymbol{u} = 0$), with a linear distribution of the temperature. Both steady and time-dependent codes converge to the same flow state with a single convection cell. For this final state, horizontal $u(y)$ and vertical $v(x)$ velocity profiles were extracted at mid-domain ($y = 0.5$ and $x = 0.5$, respectively) and plotted in Fig. 6. Our results are in very good agreement with reference numerical results obtained by Le Quéré [74] with a spectral code.

Table 2 offers a quantitative assessment of the accuracy of the present Newton method. The values of u_{max} and its location Y are compared to reference values from [74]. The Newton method gives results identical to reference values, with a difference less than 0.01%.

6.1.2. Differentially heated cavity with an inner heated obstacle

We consider the same differentially heated cavity as previously and add a centred square obstacle. The boundaries of the inner square are non-slip isothermal walls, maintained at a dimensionless hot temperature $\theta_h = 0.8$. The solution computed for $Ra = 10^6$ and $Pr = 0.71$ is compared with the results obtained by Moglan [75], who used a 6th order finite-difference method with an immersed boundary method to model the obstacle. The temperature distribution in the cavity is shown in Fig. 7a. The vertical velocity profile $v(x)$ along the horizontal symmetry line ($y = 0.5$) is displayed in Fig. 7b and shows very good agreement with the numerical results reported by Moglan [75].

6.2. Melting of a phase-change material (PCM)

We continue our validation tests by considering the full system (25)–(27) for the case of the melting of a phase-change material. Two new non-linearities are now present in the system: the Carman–Kozeny penalty term $A(\theta)$ and the enthalpy source term $S(\theta)$. The function S is regularized using (15). We also consider that the material properties in the liquid and solid are the same, *i.e.* $C = K = 1$. This is a frequent assumption [27,28]. Five cases were computed (the exact values of the defining parameters are summarized in Table 3):

- PCM-Case #1 simulates the experimental study of Okada [76]. It consists of a differentially heated square cavity, filled with octadecane paraffin.
- PCM-Case #2 is extracted from the collective publication by Bertrand et al. [77], in which the results of different numerical approaches were compared for the simulation of the melting of a PCM.
- PCM-Case #3 simulates the melting of a cylindrical PCM with heated inner tubes, as in [2].
- PCM-Case #4 simulates the melting of Gallium in a rectangular cavity heated by the side-wall, as in [21].
- PCM-Case #5 reproduces the simulation of Nourgaliev et al. [71] using highly distorted meshes to compute a natural convection flow with solid crust formation.

To guide the user of the toolbox, we also provide in Table 3 the values of the CPU time necessary to run each case (with default parameters) on a personal computer and the typical number of triangles of the generated adaptive mesh.

Supplemental Material for animations depicting the dynamics of the cases presented in this section are provided at http://lmrs-num.math.cnrs.fr/2019CPCP1.html.

6.2.1. PCM-Case #1: Melting of an octadecane PCM in a square cavity

Okada [76] studied experimentally the melting of an octadecane PCM in a square cavity of height $H = 1.5$ cm. His results

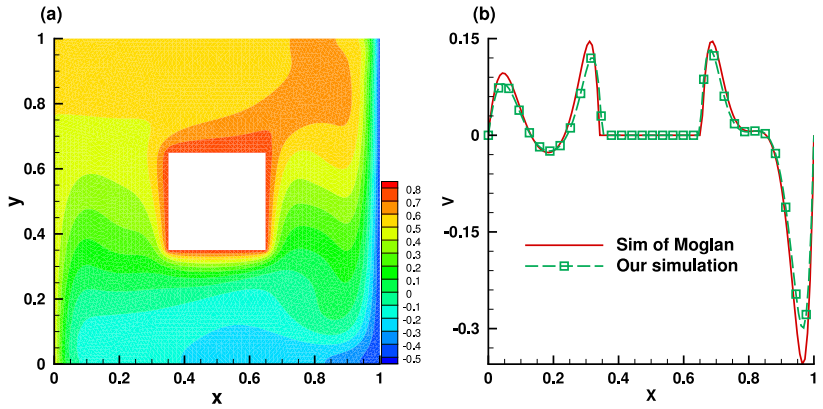

Fig. 7. Natural convection of air in a differentially heated cavity with an inner heated square for $Ra = 10^6$. Temperature field (a) and vertical velocity profile along the horizontal symmetry line (b). Results obtained using the present Newton method (red solid line), with an initial mesh resolution $nbseg = 80$. Comparison with the results obtained using a 6th order finite-difference method by Moglan [75].

Table 3
Parameters for the cases simulating the melting of a phase-change material.

	Case #1	Case #2	Case #3	Case #4	Case #5
Ra	$3.27 \cdot 10^5$	10^8	$5 \cdot 10^4$	$7 \cdot 10^5$	10^6
Pr	56.2	50	0.2	0.0216	0.1
Ste	0.045	0.1	0.02	0.046	4.854
δt	0.1	10^{-3}	10^{-3}	10^{-5}	10^{-3}
V_{ref}	$\frac{\nu_l}{H}$	$\frac{\nu_l}{H}$	$\frac{\nu_l}{H}$	$\frac{\nu_l}{H}$	$\frac{\nu_l}{H}\sqrt{\frac{Ra}{Pr}}$
CPU time	01:09:05	18:17:56	00:49:21	08:32:58	02:49:20
Number of triangles	2900	7000	2076	3600	2769
Number of iterations	798	5000	491	2001	3027
ratio CPU-time adapt/iteration	2.5%	2.5%	2%	2%	4%

were often used to validate numerical methods [27,28,32,76]. The material is initially solid ($\theta_0 = -0.01$) and melts progressively starting from the left boundary, maintained at a hot temperature $\theta_h = 1$. The right boundary is also isothermal, with cold temperature $\theta_c = -0.01$. Horizontal boundaries are adiabatic. The other parameters of this case are reported in Table 3.

The computation starts from a refined mesh near the hot boundary. Mesh adaptivity is applied at each time step using metrics computed from three variables: the two fluid velocities and the enthalpy source term S. To reduce the impact of the interpolation on the global accuracy, we use two successive fields (S^n) and (S^{n+1}) in the adaptivity procedure. This allows us to refine the mesh in the fluid part of the domain and inside the artificial mushy. Figure 8a gives an illustration of the adapted mesh at dimensionless time $t = 78.7$. In Fig. 8b we compare the position of the solid–liquid interface at $t = 39.9$ and $t = 78.7$ with the experimental data of Okada [76] and previously published numerical results [28,32]. The obtained shape and position of the liquid–solid interface is closer to experimental results than numerical results reported in [28]. This is a direct consequence of the mesh adaptivity capabilities of our method.

This comparison also allowed us to finely tune the value of the constants used in the model (16). Even though it is generally assumed that a large value for C_{CK} must be set, the exact value of this constant could influence the accuracy of the results [53,78]. This choice of the value of this constant is a still open problem. Very good agreement with the experimental result of Okada [76] is obtained for C_{CK} varying in the range $[10^6, 10^8]$. Imposing a too large value ($C_{CK} = 10^{10}$) results in artificially slowing the propagation of the melting front. Consequently, we set for all subsequent simulations $C_{CK} = 10^6$.

6.2.2. PCM-Case #2: Melting of an octadecane PCM with high Rayleigh number

This case considers the same problem of the melting of a PCM, but with a very high value of the Rayleigh number $Ra = 10^8$ (see Table 3). This case is very challenging since the natural convection becomes important in the fluid flow, and enhances considerably the heat transfer.

Bertrand et al. [77] compiled results provided by five different authors (Lacroix, Le Quéré, Gobin-Vieira, Delannoy and Binet-Lacroix). These results will be hereafter referred to as (say) 'Lacroix, from [77]'. They have attempted a first comparison by taking several numerical methods to compute the basic configuration presented in this section. Two investigators among the five failed to predict the process and showed unrealistic behaviours in Fig. 9a and b: Lacroix and Delannoy seem to be insufficiently converged as shown by Fig. 9a, and Binet-Lacroix overestimates the average Nusselt number by more than 30% (Fig. 9b). Hence, this collection of results allows us to validate our numerical method and check if realistic results are obtained for complex physical configurations. For comparison purpose, we extract from simulations the position of the melting front and the Nusselt number Nu at the left wall ($x = 0$) for each of the five methods presented by Bertrand et al. [77]. The Nusselt number Nu is defined as follows:

$$Nu = \int_0^1 \left(\frac{\partial \theta}{\partial x}\right)_{x=0} dy. \tag{41}$$

The position of the melting front for three time instants, $t = 1$, 3 and 5 is reported in Fig. 9a. Our results are for each case in fairly good agreement with those of Gobin and Le Quéré. Details

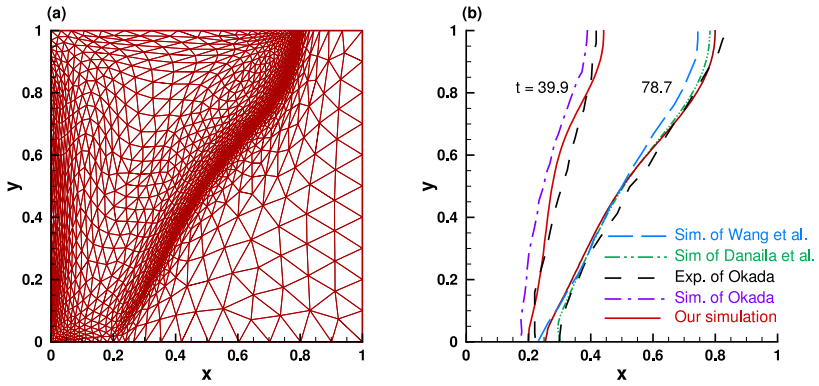

Fig. 8. PCM-Case #1. Melting of the PCM. (a) Adapted mesh at time instant $t = 78.7$. (b) Position of the solid–liquid interface. Comparison with experimental data of Okada [76] and numerical results of Danaila et al. [32] and Wang et al. [28] for two time instants, $t = 39.9$ and $t = 78.7$.

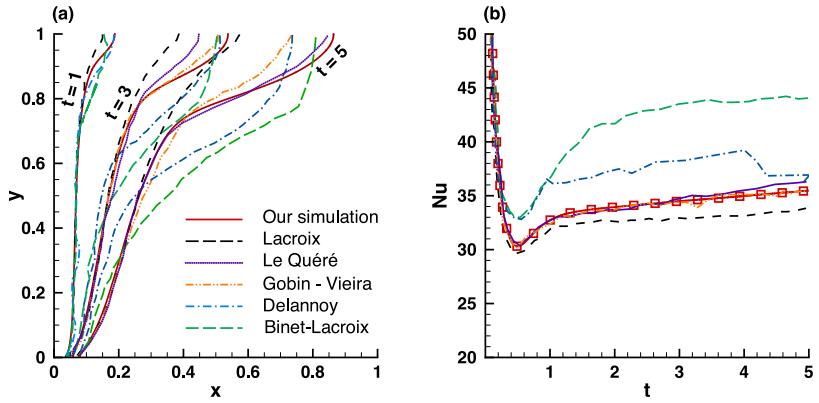

Fig. 9. PCM-Case #2. Melting of the PCM with high Rayleigh number ($Ra = 10^8$). Comparison with five sets of results presented in [77]. (a) Location of the solid–liquid interface at dimensionless time instants $t = 1$, 3 and 5. (b) Temporal evolution of the Nusselt number.

of their numerical method are presented in [79]. Gobin used a front-tracking method based on a coordinate transformation with a finite volume method in a 62×42 grid and Le Quéré solved a single domain method using a second-order scheme with a finite volume method and a 192×192 grid. The time evolution of the Nusselt number is presented in Fig. 9b. Very good agreement is obtained with the results of Gobin and Le Quéré. A relative difference, less than 2%, is observed for the Nusselt number.

The high Rayleigh number $Ra = 10^8$ is a very demanding numerical test. The high velocity, inducing a very narrow thermal boundary layer, can lead to unrealistic results if under-resolved. This explains why some numerical methods failed in [77]. The interest of the mesh adaptation is clearly demonstrated for this case, since we initially used a coarse 40×40 grid.

6.2.3. PCM-Case #3: Melting of cylindrical PCM with inner heated tubes

Previous cases #1 and #2 considered phase change problems evolving in a simple geometry, a square cavity. A more complex geometry, suggested by Luo et al. [2], is simulated in this section. It consists of a cylindrical PCM of radius $R = 1$ with tube inclusions of different arrangements. The interest in studying this case is not solely the challenge of the complex configuration, but also the possibility to compare our results with those of Luo et al. [2], obtained using a completely different model based on the Lattice Boltzmann Method. This configuration is also interesting from a practical point of view. Agyenim et al. [80] pointed out that more than 70% of the PCM containers used for heat storage are using shell-tube systems.

We simulate three configurations with one, four and nine heated tubes. The size of the tubes is adjusted to have the same total tube area for all configurations. The radius R_i of the inner tube is $R_i = R/4$ for the case with one tube, $R_i = R/8$ for the four heated tubes case and $R_i = R/12$ for the case with nine tubes. A Dirichlet boundary condition ($\theta = \theta_h$) is applied to the boundary of inner tubes. A Neumann boundary condition ($\partial\theta/\partial n = 0$) is used for the outer boundary. For the velocity, all boundaries are considered as non-slip walls ($\boldsymbol{u} = 0$). Only half of the domain is simulated since all configurations are symmetric with respect to the vertical axis (see Fig. 10). The mesh is refined initially around the inner tubes, and is dynamically adapted at each time step around the melting front and the thermal boundary layer area. The same metrics presented in Section 6.2.1 are used for the mesh adaptivity.

Figure 10 shows the temperature field and the position of the solid–liquid interface (black line) for the three configurations for time instants corresponding to the same liquid fraction $L_f = 80\%$. The distribution of the inner tubes in the liquid phase influences directly the fluid motion and the shape of the melting front. The more the number of inner tubes, the stronger the natural convection is in the melted PCM. The shape of the solid–liquid interface displays complex patterns, depending on the space arrangement of the inner tubes. The mesh is nicely adapted following the evolution of the melting interface, even after its separation in several distinct fronts touching the outer boundary (see Figs. 10b, c).

To estimate the efficiency of each configuration, we plot in Fig. 11 the time evolution of the liquid fraction L_f. By including

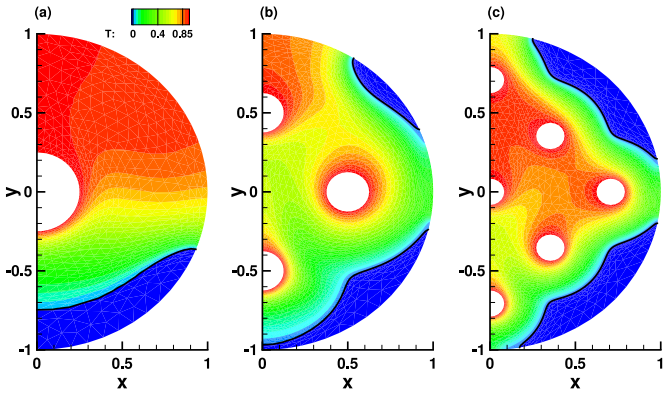

Fig. 10. PCM-Case #3. Temperature fields for the melting of a cylindrical PCM with inner heated tubes. Time instants corresponding to the same liquid fraction $L_f = 80\%$. Configurations with (a) one tube ($t = 2.5$), (b) four tubes ($t = 0.99$) and (c) nine tubes ($t = 0.4$). Melting fronts are localized with black lines (only half of the domain is simulated).

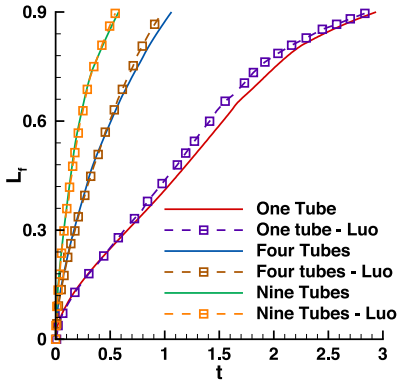

Fig. 11. PCM-Case #3. Time evolution of the liquid fraction for configurations with one, four, and nine heated tubes. Comparison with numerical results of Luo et al. [2].

more heated tubes the heat transfer is enhanced, inducing a faster melting time. The nine-tube configurations melts 5 times faster than the reference configuration with one tube. Note also from Fig. 11 a good agreement between our results and those reported by Luo et al. [2] for the evolution of the liquid fraction.

6.2.4. PCM-Case #4: Melting of Gallium in a rectangular cavity

The melting of the Gallium in a rectangular cavity was a controversial case since [81] raised the question whether the convection in the fluid is mono-cellular or multi-cellular. Experimental results exhibited indeed a mono-cellular structure, while many researchers claimed that this observation was incorrect. Prior to Dantzig [81], both experimental and numerical studies reported a single convection cell in the fluid phase. Later, simulations provided solutions with multi-cellular flow. Hannoun et al. [21] concluded that the mono-cellular observation was caused by a problem of convergence of the numerical solution, due to coarse grids or inconsistencies in the mathematical model.

Therefore, this test case simulating the melting of the Gallium is a relevant exercise to test the accuracy of our method. The parameters of this case are reported in Table 3. To capture the very small convection cells during the first step of the melting, Hannoun et al. [21] used a $800 \times 1,120$ fixed grid in a rectangular domain of dimensions 6.35 cm $\times$ 8.89 cm. With our adaptive method, a maximum number of 4820 triangles are necessary to reproduce the numerical result of Hannoun et al. [21]. The grid size is thus reduced with our method by a factor of 100.

The time evolution of the flow is presented in Fig. 12. Temperature field, streamlines and position of the melting front are plotted for several time instants: $t = 0.0015$, 0.006, 0.01, and 0.019. These values were chosen to visualize the merging of convection cells in the fluid flow and correspond to physical times 20 s, 85 s, 155 s, 280 s in [21]. The number of rolls was considered as a validation criterion by several authors [21,82,83]. Three cells are observed at $t = 0.006$ (Fig. 12). The number of cells decreases later through a process of roll merging, as it was also reported by Hannoun et al. [21]. Our numerical results are in good agreement with the observations of Hannoun et al. [21], Cerimele et al. [82] and Giangi and Stella [83].

6.2.5. PCM-Case #5: Solid crust formation in a highly distorted mesh

Nourgaliev et al. [71] used a discontinuous Galerkin finite element method to simulate the solid crust formation inside a highly distorted domain (Fig. 13). The fluid is initially motionless with an initial dimensionless temperature $\theta_0 = 2$. The temperature of fusion is set to $\theta_f = 1.4$, according to Nourgaliev et al. [71] parameters (see Table 3 for the values of all parameters). The left side boundary is maintained at a cold temperature $\theta_c = 1.39$ in the initial stage. The right wall is isothermal, with hot temperature $\theta_h = 2$. A nearly steady-state natural circulation is induced in the early time evolution of the flow. Then, the cold temperature at the left wall is decreased to $\theta_c = 1$, below the temperature of solidification. At this point, the formation of a solid crust layer starts at the left boundary. Figure 13 shows the temperature field and the streamlines for the time instant $t = 30$. Our results are in very good agreement with those of Nourgaliev et al. [71].

6.3. Melting-solidification cycle of a PCM

We address in this section the challenging problem of simulating a complete melting-solidification cycle of a PCM. The simulation starts from the final state obtained in Section 6.2.1. At $t = 78.7$, the PCM is partially melted, with the liquid fraction $L_f = 0.5$ (see Fig. 14a). The temperature of the left boundary is then dropped to a cold temperature $\theta_c = -0.01 < \theta_f$, identical to that of the right wall. The solidification starts and the solid phase propagates into the cavity from both left and right sides. Panels (a) to (e) in Fig. 14 depict the time evolution of the solidification process. The solid phase is represented in blue and corresponds to the region of temperature $\theta \leq \theta_f = 0$, while the solid–liquid interfaces (*i.e.* contour lines of $\theta = 0$) are represented by bold solid lines. The solidification is completely achieved at $t = 4600$. Note that the mesh adaptivity of our toolbox is able to accurately

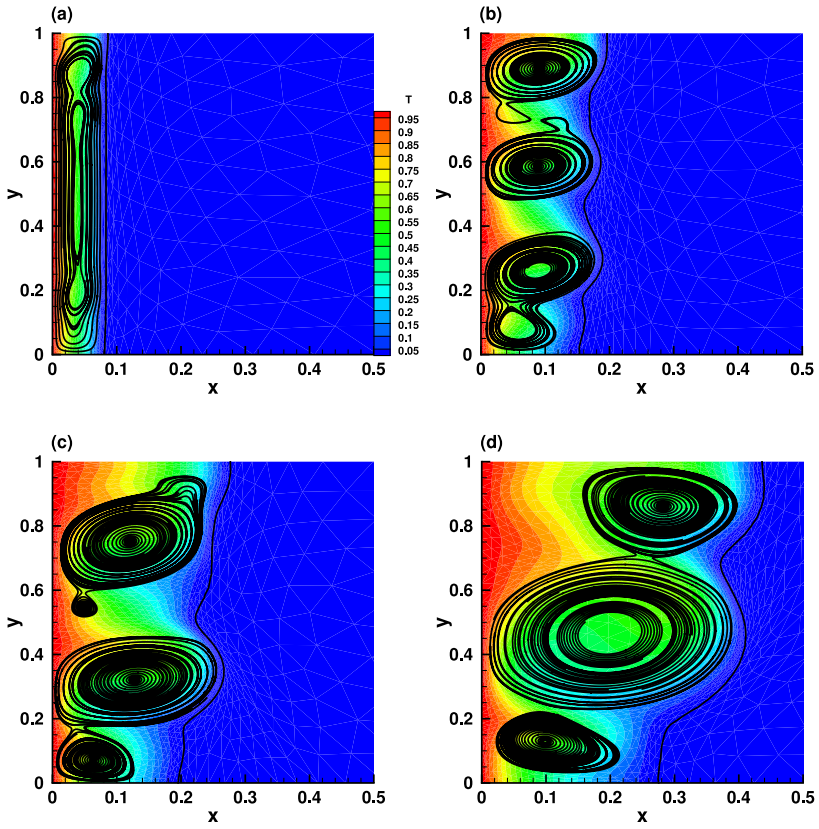

Fig. 12. PCM-Case #4. Melting of Gallium: temperature field, streamlines, and melting front for dimensionless time instants (panels a to d): 0.0015, 0.006, 0.01, and 0.019. For a better view of the convection cells, a ratio 2:1 was used for the axis dimensions.

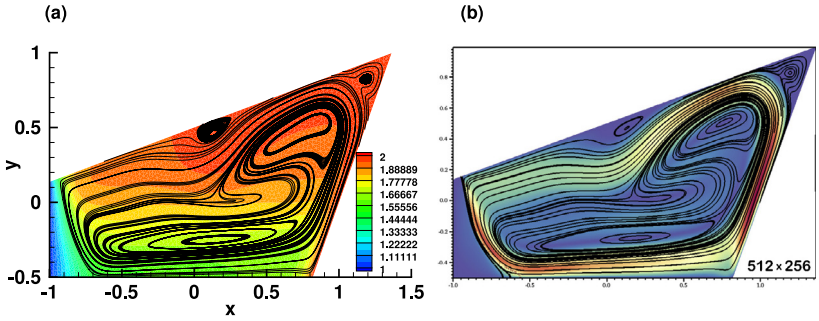

Fig. 13. PCM-Case #5. Solid crust formation in a distorted mesh. Temperature field and streamlines at dimensionless time $t = 30$: our simulation (a) and [71] (b).

track the two solidification fronts. The adapted mesh shown in Fig. 14f illustrates that a finer mesh is well generated along the interface ($\theta = 0$), while a coarser mesh is obtained in the regions of low gradients.

A comprehensive physical description of this case is given in our recent paper [45]. The case of the solidification process starting after a complete melting of the PCM is also presented in this paper. We can conclude from this section that the Newton method is able to deal with either melting or solidification process. The simulations are in good agreement with existing experimental and numerical results for the melting and show consistent behaviour for the solidification.

6.4. Natural convection of water and water freezing

We consider in this section the natural convection and the solidification of water. With these cases, we test the ability of our numerical system to deal with additional non-linear terms. Since pure water exhibits non-linear density variation for $T < 10.2\,^{\circ}\mathrm{C}$, with a maximum at $T_m = 4.0293\,^{\circ}\mathrm{C}$, the Boussinesq force becomes non-linear. We used the following density–temperature relationship suggested by Gebhart and Mollendorf [69]:

$$\rho(T) = \rho_m \left(1 - w\,|T - T_m|^q\right), \tag{42}$$

with $\rho_m = 999.972$ [kg/m^3], $w = 9.2793 \cdot 10^{-6}$ [(°C)$^{-q}$], and $q = 1.894816$.

Hence, the buoyancy term $f_B = g(\rho_{ref} - \rho)/\rho_{ref}$ in (7) is not any more linear and becomes after scaling:

$$f_B(\theta) = \frac{Ra}{Pr\,Re^2}\,\frac{1}{\beta\delta T}\,\frac{\rho(\theta_f) - \rho(\theta)}{\rho(\theta_f)}, \tag{43}$$

where $\beta = (1/\rho_m)\,(d\rho/dT)$ is the thermal expansion coefficient taking the value $\beta = 6.91 \cdot 10^{-5}$ [(K)$^{-1}$] [84].

We simulate a differentially heated square cavity filled with liquid pure distilled water. This problem was investigated experimentally and numerically by Giangi et al. [41]; Kowalewski and

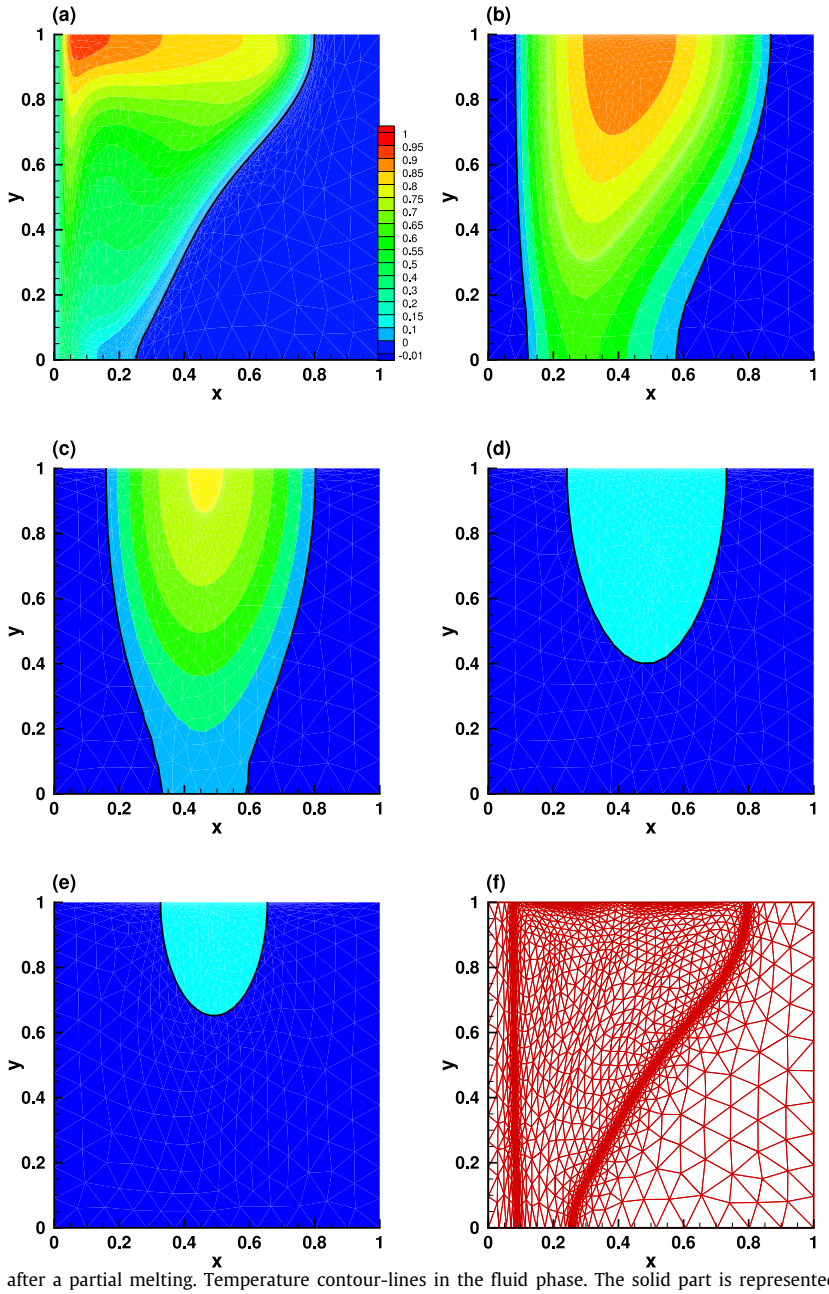

Fig. 14. Solidification of the PCM after a partial melting. Temperature contour-lines in the fluid phase. The solid part is represented in blue and corresponds to the region of temperature $\theta \leq \theta_f = 0$. Time instants (panels a to e): $t = 78.92$, $t = 1072$, $t = 2702$, $t = 3902$ and $t = 4501$. The adapted mesh corresponding to panel (b) is shown in panel (f).

Rebow [40]; Michalek and Kowalewski [85]. The non-dimensional parameters describing the problem are (see [85] for physical details): $\mathcal{R}a = 2.518084 \cdot 10^6$, $\mathcal{P}r = 6.99$ and $\mathcal{S}te = 0.125$.

6.4.1. Natural convection of water

The initial temperature is linearly distributed in the square cavity, with a hot temperature $T_h = 10$ °C at the left wall and a cold temperature $T_c = T_f = 0$ °C at the right wall. The temperature field and the streamlines of the steady state are presented in Fig. 15a. The isoline $\theta = \theta_m$, corresponding to the line of maximum density is represented by a dashed line. Due to the anomalous thermal variation of water density, two recirculating zones are formed in the flow: a lower (abnormal) recirculation in the vicinity of the cold wall where $\theta < \theta_m$ and an upper (normal) one where the density decreases with temperature ($\theta > \theta_m$).

A more precise comparison with previously published results is shown in Fig. 15b. The obtained temperature profile $\theta(x)$ along the horizontal symmetry line of the cavity ($y = 0.5$) is in good agreement with the numerical results of Michalek and Kowalewski [85]. Their results were obtained with finite-volume and finite-difference codes (FLUENT and FRECONV3V). Differences are visible in the vicinity of the maximum density line, region where our mesh is well refined to capture the separation line between the two recirculation zones. It should be noted that the FLUENT simulations in [85] are performed with a fixed uniform grid with 380 × 380 nodes, while our adapted grid has only 1807 vertices (3430 triangles).

6.4.2. Water freezing

We finally consider the difficult case of water freezing in a square cavity. The initial state for this computation is the convection steady pattern in the cavity presented in Fig. 15. The freezing

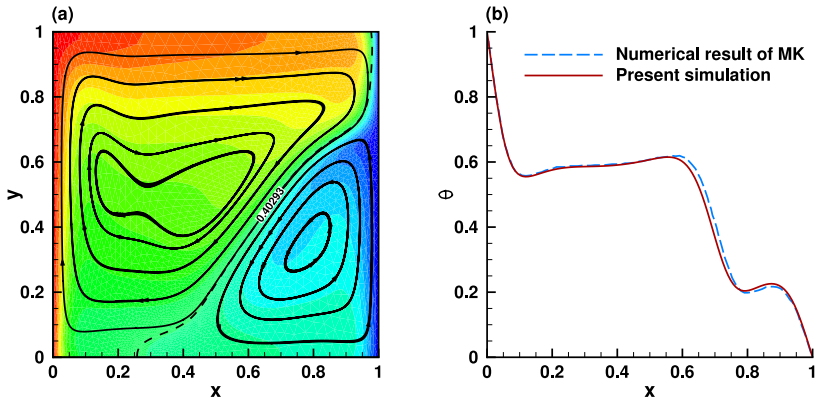

Fig. 15. Natural convection of water in a differentially heated cavity. Non-dimensional temperature θ at steady state. (a) Two-dimensional temperature field and streamlines showing the two recirculating zones. (b) Temperature profile along the horizontal symmetry line. Comparison with the numerical results of Michalek and Kowalewski [85].

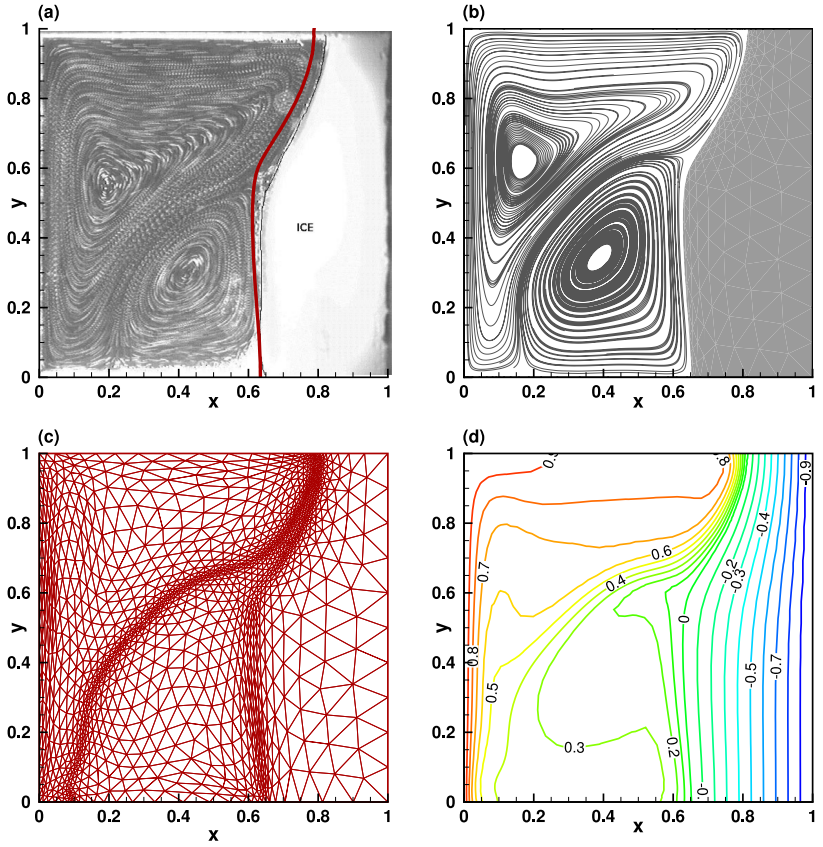

Fig. 16. Freezing of pure water. Configuration at (physical time) $t_\varphi = 2340$ [s]: (a) experimental image from [40]; the thick red line represents the solid–liquid interface computed with the present method (b) computed streamlines showing the two recirculating zones in the fluid phase (c) finite-element mesh refined along the solid–liquid interface ($T = 0$ °C) and also along the line of maximum water density ($T = 4$ °C) (d) temperature iso-lines.

starts by dropping smoothly the temperature of the cold right wall from $T_c = 0$ °C to $T_c = -10$ °C.

Figure 16a superimposes the experimental image from [40] with our numerical results for the same physical time $t_\varphi = 2340$ [s]. The flow pattern in the liquid phase, shown in Fig. 16b also corresponds very well qualitatively to the experimental image. The simulation was performed with small time steps ($\delta t = 10^{-2} \approx 15$ [s]) and reasonable-size grids (less than 3000 nodes) due to the efficiency of the adaptivity algorithm.

The two recirculating zones being separated by the line $T = T_m$ we used the mesh adaptivity capability of the method to refine the grid along this line. The metrics used for adaptivity were computed from the two components of the velocity and a P_1 function $\phi(T)$ "tracking" the value T_m, defined by the general regularization expression:

$$\phi(T) = \frac{1}{2}\left\{1 + \tanh\left(\frac{T_m - T}{R_\phi}\right)\right\}, \tag{44}$$

with $R_\phi = 0.02$. To reduce the impact of the interpolation on the global accuracy (see also [44]), we used both $\phi(T^n)$ and $\phi(T^{n+1})$ in the adaptivity procedure. The final mesh is displayed in Fig. 16c, clearly showing that the mesh is refined along the

lines $T = T_m$ and $T = 0$ °C. This allows to accurately capture the structure and the extent of the two recirculating zones, features that are difficult to obtain with fixed meshes (discrepancies in numerical results are described in [40,41,85]). As a consequence, the temperature contours lines in Fig. 16d are smooth and clearly define the two interfaces in the system: the liquid–solid interface ($\theta = 0$) and the density inversion interface ($\theta = 0.4$) separating the two recirculating liquid regions.

7. Summary and conclusions

We provide with this paper an adaptive finite-element toolbox for solving two-dimensional phase-change problems with convection. The programs were written using FreeFem++, a free software offering a programming syntax close to the mathematical formulation. A single domain numerical approach was first derived. The details of the finite-element formulation were then presented. The key ingredients of the implemented method are: (i) a second order accuracy in space and time; (ii) the use of an adaptive finite element method with a well chosen regularization of the functions representing the variation of thermodynamic properties at the solid–liquid interface, and (iii) a fully implicit discretization with a Newton algorithm for solving the non-linear system of equations.

Four test cases were presented, by adding progressively non-linearities in the system of equations:

(i) natural convection of air in a differentially heated cavity,
(ii) melting of a PCM,
(iii) melting-solidification cycle of a PCM,
(iv) natural convection and freezing of water.

The computations for case *(ii)* were rendered more challenging by considering complex geometries (highly distorted mesh, cylindrical PCM with inner heated tubes) and computationally demanding cases (high Rayleigh numbers). The efficiency of the adaptivity method by metric control was investigated by tracking simultaneously several interfaces (two melting fronts during the solidification cycle and density inversion interface for water flows).

For each test case, we provided a separate folder containing all the necessary files (parameters, restart files) necessary to run them directly. We described in the text body of the paper the expected results and their validation. Very good agreement with experimental data or numerical results was obtained for all considered test cases, proving the capability of our method to tackle a large range of problems. Ready-made scripts and layouts are provided with the toolbox to allow the user to generate the figures presented in this paper with newly generated data after running the programs. Validation data sets from experiments or previous publications are included in these layouts. Movies depicting the dynamics of some cases simulated in this paper are provided as Supplemental Material at http://lmrs-num.math.cnrs.fr/2019CPCP1.html.

Since FreeFem++ is a free software, the method could be easily implemented and tested by anyone interested in simulating phase-change problems. All technical issues related to the implementation of the finite element method are hidden, allowing to focus on numerical algorithms and their performance. This offers the possibility to address other computational challenges related to different physical or mathematical models in this field.

The extension of the method for 2D and 3D cases, using domain decomposition methods adapted to parallel computing, will be reported in a forthcoming contribution.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This project was co-financed by the European Union with the European regional development funds and by the Normandy Regional Council, France via the M2NUM (ERDF, HN0002081) and M2SiNUM (ERDF, 18P03390/18E01750/18P02733) projects. Part of this work was performed using computing resources of CRIANN (Centre Régional Informatique et d'Applications Numériques de Normandie, Normandy, France).

References

[1] A. Kowalewski, D. Gobin, Phase Change with Convection: Modelling and Validation, Springer, 2004.
[2] K. Luo, F.-J. Yao, H.-L. Yi, H.-P. Tan, Appl. Therm. Eng. 86 (2015) 238–250.
[3] W. Gong, K. Johannes, F. Kuznik, Commun. Comput. Phys. 17 (5) (2015) 1201–1224.
[4] E.M. Sparrow, P.V. Patankar, S. Ramadhyani, J. Heat Transfer 99 (4) (1977) 520–526.
[5] S.O. Unverdi, G. Tryggvason, J. Comput. Phys. 100 (1992) 25–37.
[6] S.C. Gupta, Comput. Methods Appl. Mech. Engrg. 189 (2) (2000) 525–544.
[7] R.T. Tenchev, J.A. Mackenzie, T.J. Scanlon, M.T. Stickland, Int. J. Heat Fluid Flow 26 (4) (2005) 597–612.
[8] F. Stella, M. Giangi, in: A. Kowalewski, D. Gobin (Eds.), Phase Change with Convection: Modelling and Validation, Springer, 2004, pp. 219–272.
[9] M. Fabbri, V.R. Voller, J. Comput. Phys. 130 (1997) 256–265.
[10] V.R. Voller, C. Prakash, Int. J. Heat Mass Transfer 30 (1987) 1709–1719.
[11] Y. Cao, A. Faghri, W.S. Chang, Int. J. Heat Mass Transfer 32 (7) (1989) 1289–1298.
[12] W. Shyy, H. Udaykurnar, M.M. Rao, R. Smith, Computational Fluid Dynamics with Moving Boundaries, Taylor & Francis, 1996.
[13] W.J. Boettinger, J.A. Warren, C. Beckermann, A. Karma, Annu. Rev. Mater. Res. 32 (1) (2002) 163–194.
[14] I. Singer-Loginova, H.M. Singer, Rep. Progr. Phys. 71 (10) (2008) 106501.
[15] B. Favier, J. Purseed, L. Duchemin, J. Fluid Mech. 858 (2019) 437–473.
[16] N. Eyres, D.R. Hartree, J. Ingham, R.J. Sarjant, J.B. Wagstaff, et al., Philos. Trans. R. Soc. Lond. A Math. Phys. Eng. Sci. 240 (813) (1946) 1–57.
[17] M.E. Rose, Math. Comp. 24 (1960) 9–256.
[18] A. Bhattacharya, A. Kiran, S. Karagadde, P. Dutta, J. Comput. Phys. 262 (2014) 217–230.
[19] V.R. Voller, in: W.J. Minkowycz, E.M. Sparrow (Eds.), Advances in Numerical Heat Transfer, Taylor & Francis, 1996, pp. 341–375.
[20] M.A. Rady, A.K. Mohanty, Numer. Heat Transfer A 29 (1) (1996) 49–63.
[21] N. Hannoun, V. Alexiades, T.Z. Mai, Numer. Heat Transfer B 44 (3) (2003) 253–276.
[22] K. Morgan, R. Lewis, O. Zienkiewicz, Internat. J. Numer. Methods Engrg. 12 (1978) 1191–1195.
[23] F.M. Chiesa, R.I.L. Guthrie, J. Heat Transfer 96 (3) (1974) 377–384.
[24] C. Gau, R. Viskanta, Int. J. Heat Mass Transfer 27 (1) (1984) 113–123.
[25] C.R. Swaminathan, V.R. Voller, Int. J. Heat Mass Transfer 40 (12) (1997) 2859–2868.
[26] A. König-Haagen, E. Franquet, E. Pernot, D. Brüggemann, Int. J. Therm. Sci. 118 (2017) 69–103.
[27] Z. Ma, Y. Zhang, Int. J. Numer. Methods Heat Fluid Flow 16 (11) (2006) 204–225.
[28] S. Wang, A. Faghri, T.L. Bergman, Int. J. Heat Mass Transfer 53 (2010a) 1986–2000.
[29] D. Gartling, in: K. Morgan, C. Taylor, C. Brebbia (Eds.), Computer Methods in Fluids, Pentech, London, 1980, pp. 257–284.
[30] V.R. Voller, M. Cross, N.C. Markatos, Internat. J. Numer. Methods Engrg. 24 (1987) 271–284.
[31] Y. Cao, A. Faghri, ASME J. Heat Transfer 112 (1990) 812–815.
[32] I. Danaila, R. Moglan, F. Hecht, S. Le Masson, J. Comput. Phys. 274 (2014) 826–840.
[33] R. Aldbaissy, F. Hecht, G. Mansour, T. Sayah, Calcolo 55 (4) (2018) 44.
[34] J. Woodfield, M. Alvarez, B. Gamez-Vargas, R. Ruiz-Baier, J. Comput. Appl. Math. 360 (2019) 117–137.
[35] M. Alvarez, G.N. Gatica, B. Gomez-Vargas, R. Ruiz-Baier, J. Sci. Comput. 80 (2019) 141–174.
[36] A.D. Brent, V.R. Voller, K.J. Reid, Numer. Heat Transfer 13 (1988) 297–318.

[37] N. Hannoun, V. Alexiades, T.Z. Mai, Internat. J. Numer. Methods Fluids 48 (11) (2005) 1283–1308.
[38] Y. Belhamadia, A.S. Kane, A. Fortin, Int. J. Numer. Anal. Model. 3 (2) (2012) 192–206.
[39] P. Angot, C.-H. Bruneau, P. Fabrie, Numer. Math. 81 (4) (1999) 497–520.
[40] T.A. Kowalewski, M. Rebow, Int. J. Comput. Fluid Dyn. 11 (1999) 193–210.
[41] M. Giangi, T.A. Kowalewski, F. Stella, E. Leonardi, Comput. Assist. Mech. Eng. Sci. 7 (2000) 321–342.
[42] C.M. Elliott, IMA J. Numer. Anal. 7 (1) (1987) 61–71.
[43] C. Li, IMA J. Numer. Anal. 3 (1983) 87–107.
[44] Y. Belhamadia, A. Fortin, E. Chamberland, J. Comput. Phys. 194 (1) (2004) 233–255.
[45] A. Rakotondrandisa, I. Danaila, L. Danaila, Int. J. Heat Fluid Flow 76 (2019) 57–71.
[46] A. Zimmerman, J. Kowalski, in: M. Schäfer, M. Behr, M. Mehl, B. Wohlmuth (Eds.), Recent Advances in Computational Engineering. ICCE 2017, in: Lecture Notes in Computational Science and Engineering, vol. 124, Springer, 2018, pp. 177–197.
[47] Y. Belhamadia, A. Fortin, T. Briffard, Numer. Heat Transfer A 76 (4) (2019) 179–197.
[48] D.N. Arnold, F. Brezzi, M. Fortin, Calcolo 21 (1984) 337–344.
[49] F. Brezzi, M. Fortin, Mixed and Hybrid Finite Element Methods, Springer Verlag., 1991.
[50] F. Hecht, J. Numer. Math. 20 (2012) 251–266.
[51] F. Hecht, O. Pironneau, A.L. Hyaric, K. Ohtsuke, Freefem++ (manual), 2007, www.freefem.org.
[52] S. Wang, A. Faghri, T.L. Bergman, Numer. Heat Transfer B 58 (6) (2010b) 393–418.
[53] A.C. Kheirabadi, D. Groulx, Proceedings of CHT-15, ICHMT International Symposium on Advances in Computational Heat Transfer, Ichmt Digital Library Online. Begel House Inc, 2015.
[54] I. Danaila, F. Hecht, J. Comput. Phys. 229 (2010) 6946–6960.
[55] G. Vergez, I. Danaila, S. Auliac, F. Hecht, Comput. Phys. Comm. 209 (2016) 144–162.
[56] Y. Zhang, I. Danaila, Appl. Math. Model. 37 (2013) 4809–4824.
[57] R. Temam, Theory and Numerical Analysis, North Holland, Amsterdam, 1977.
[58] V. Girault, P.-A. Raviart, Finite Element Methods for Navier-Stokes Equations, Springer Verlag, Berlin, 1986.
[59] A. Quarteroni, A. Valli, Numerical Approximation of Partial Differential Equations, Springer-Verlag, Berlin and Heidelberg, 1994.
[60] D. Boffi, F. Brezzi, M. Fortin, Mixed Finite Element Methods and Applications, Springer Verlag, 2013.
[61] J. Oden, N. Kikuchi, Y.J. Song, Comput. Methods Appl. Mech. Engrg. 31 (3) (1982) 297–329.
[62] M. Bercovier, RAIRO Anal. Numer. 12 (1978) 211–236.
[63] H. Borouchaki, M.J. Castro-Diaz, P.L. George, F. Hecht, B. Mohammadi, 5th Inter. Conf. on Numerical Grid Generation in Computational Field Simulations, Mississipi State Univ, 1996.
[64] M. Castro-Diaz, F. Hecht, B. Mohammadi, Internat. J. Numer. Methods Fluids 25 (2000) 475–491.
[65] F. Hecht, B. Mohammadi, AIAA Pap. 97 (1997) 0859.
[66] P.L. George, H. Borouchaki, Delaunay Triangulation and Meshing, Hermès, Paris, 1998.
[67] P.J. Frey, P.L. George, Maillages, Hermès, Paris, 1999.
[68] B. Mohammadi, O. Pironneau, Applied Shape Design for Fluids, Oxford Univ. Press, 2000.
[69] B. Gebhart, J. Mollendorf, Deep Sea Res. 24 (1977) 831–848.
[70] P.J. Roache, Verification and Validation in Computational Science and Engineering, Hermosa Publishers, 1998.
[71] R. Nourgaliev, H. Luo, B. Weston, A. Anderson, S. Schofield, T. Dunn, J.-P. Delplanque, J. Comput. Phys. 305 (2016) 964–996.
[72] T.M. Shih, C.H. Tan, B.C. Hwang, Internat. J. Numer. Methods Fluids 9 (2) (1989) 193–212.
[73] S. Laizet, E. Lamballais, J. Comput. Phys. 228 (16) (2009) 5989–6015.
[74] P. Le Quéré, Comput. Fluids 20 (1991) 24–41.
[75] R. Moglan, Modeling and Numerical Simulation of Flow and Heat Phenomena in a Telecommunication Outdoor Cabinet (Ph.D. thesis), Université de Rouen Normandy, 2013.
[76] M. Okada, Int. J. Heat Mass Transfer 27 (1984) 2057–2066.
[77] O. Bertrand, B. Binet, H. Combeau, S. Couturier, Y. Delannoy, D. Gobin, M. Lacroix, P. Le Quéré, M. Médale, J. Mencinger, et al., Int. J. Therm. Sci. 38 (1) (1999) 5–26.
[78] K. Mathura, D.J. Krishna, Energy Procedia 109 (2017) 314–321, international Conference on Recent Advancement in Air Conditioning and Refrigeration, RAAR 2016, 10-12 2016, Bhubaneswar, India.
[79] D. Gobin, P. Le Quéré, Comput. Assist. Mech. Eng. Sci. 7 (3) (2000) 289–306.
[80] F. Agyenim, N. Hewitt, P. Eames, M. Smyth, Renew. Sustain. Energy Rev. 14 (2) (2010) 615–628.
[81] J.A. Dantzig, Internat. J. Numer. Methods Engrg. 28 (8) (1989) 1769–1785.
[82] M.M. Cerimele, D. Mansutti, F. Pistella, Comput. & Fluids 31 (4) (2002) 437–451.
[83] F. Giangi, M. Stella, Numer. Heat Transfer A 38 (2) (2000) 193–208.
[84] T.J. Scanlon, M.T. Stickland, Int. J. Heat Mass Transfer 47 (2004) 429–436.
[85] T. Michalek, T.A. Kowalewski, Task Q. 7 (3) (2003) 389–408.