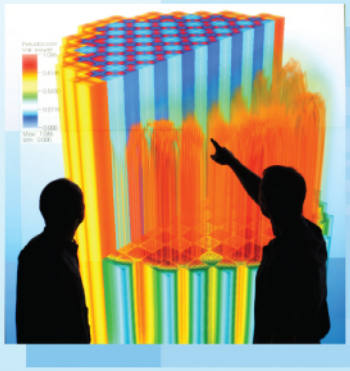
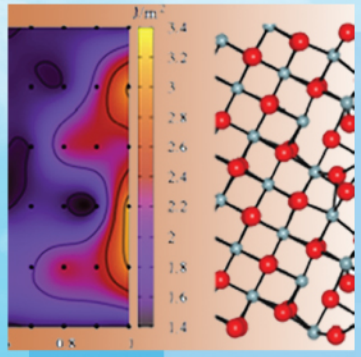
CASL-U-2013-0288-000

Power uprates
and plant life extension



Engineering design
and analysis

Science-enabling
high performance
computing



Fundamental science

Plant operational data

Consortium for Advanced Simulation of LWRs

Milestone L3:THM.CFD.P6.01: Hydra-TH Advanced Capabilities

J. Bakosi, M.A. Christon, L.A. Pritchett-Sheats, LANL

R.R. Nourgaliev, INL

February 1, 2013

U.S. DEPARTMENT OF ENERGY | Nuclear Energy

CASL-U-2013-0288-000

Hydra-TH Advanced Capabilities (L3 Milestone THM.CFD.P6.01)

LA-UR 13-20572

J. Bakosi, M.A. Christon, L.A. Pritchett-Sheats
Computational Physics Group (CCS-2)
Computer, Computational and Statistical Sciences Division
Los Alamos National Laboratory
Los Alamos, NM 87544

R.R. Nourgaliev
Reactor Safety Simulation Group
Thermal Science and Safety Analysis Department
Idaho National Laboratory
Idaho Falls, ID 83415

1 Executive Summary

This report describes the work carried out for completion of the Thermal Hydraulics Methods (THM) Level 3 Milestone THM.CFD.P6.01 for the Consortium for Advanced Simulation of Light Water Reactors (CASL). An initial multi-field flow capability has been implemented in Hydra-TH as a preliminary step towards computing thermal-hydraulics problems with phase change. The new algorithm is based on the existing single-field projection method and integrates N volume fractions, N momentum, and N internal energy equations. The N different-density fluids, at this time, are coupled via a single pressure and volume fractions (i.e., no phase change). The user input, data and output delegate registration, the initialization procedure, and time marching have been extended to multiple transport objects and N fields. A fully coupled, projection-precondtioned, fully implicit solution algorithm has also been implemented for single-phase flows using the Picard iteration technique.

2 Introduction

The primary objectives of this milestone were to report on the implementation of the following capabilities in Hydra-TH:

1. Initial multi-field capability: computing N fluid fields coupled via volume fractions and a single pressure using the modified projection algorithm for incompressible flows.

2. Fully implicit time marching using Picard iterations, providing stability and removing any time-step size restrictions.

The milestone was comprised of the following tasks:

- Implementation of multiple momentum transport objects
- Extension of data and output delegate registration
- Extension of user input to N fields

- Implementation of initialization procedure for velocity and pressure
- Implementation of volume-fraction transport classes
- Extension of momentum solve and time integration for N fields
- Demonstration of bubbly (i.e., N-field) flow
- L3 milestone report
- Stretch-goal: Implementation of a fully coupled, fully implicit, projection-preconditioned Newton-Krylov solution algorithm, demonstrated on single-phase flows.

The above tasks are discussed in §3, while summary and an outline for future work are given in §4.

3 Milestone Accomplishments

This section discusses the milestone tasks in some detail: §3.1 outlines the Hydra-TH multi-field-flow strategy and explains how the current work fits in a series of developmental steps, §3.2 and §3.3 give an overview of two options for computing multi-fluid flows. §3.4 discusses some software implementation details, including the changes to the user input deck. Finally, §3.5 describes proof-of-concept calculations.

3.1 Multi-Field Flow Solution Strategy in Hydra-TH

The implementation of the compute capability in Hydra-TH for thermal-hydraulics problems with phase change is comprised of the following steps, in increasing order of complexity, see also Fig. 1:

1. *Semi-Implicit Algorithm.* ("Option 1" in Fig. 1.) Based on the already implemented projection algorithm for single-phase incompressible flows, this algorithm advances N volume fractions, N momentum, and N energy equations coupled by volume fractions and a single pressure. Phase change in this algorithm is not taken into account. The equations are coupled through the volume fractions and the pressure only. This algorithm is similar to the stability-enhancing two-step (SETS) and nearly-implicit algorithms in existing reactor safety codes, such as RELAP5, TRAC, TRACE, CATHARE-3D, and RELAP5-3D. The time-marching is semi-implicit but not unconditionally stable. This algorithm is completed in Hydra-TH, discussed briefly in §3.2, and will be used as a base for the two algorithms below, augmented by interfacial mass, momentum, and energy transfer models.

2. *Fully-Implicit Picard Algorithm.* ("Option 2" in Fig. 1.) Due to the very different temporal and spatial scales of the various physical processes represented in multi-phase flows, numerical computations with phase change models may require removing the numerical time-step size limitation of the semi-implicit algorithm. A version of the multi-field solver in conjunction with fully-implicit time marching with the Picard iteration technique will be a second step towards a fully-functional multi-field capability in Hydra-TH. Picard iterations for solving the nonlinear system of equations, resulting from a fully-implicit time marching formulation will be combined with the existing

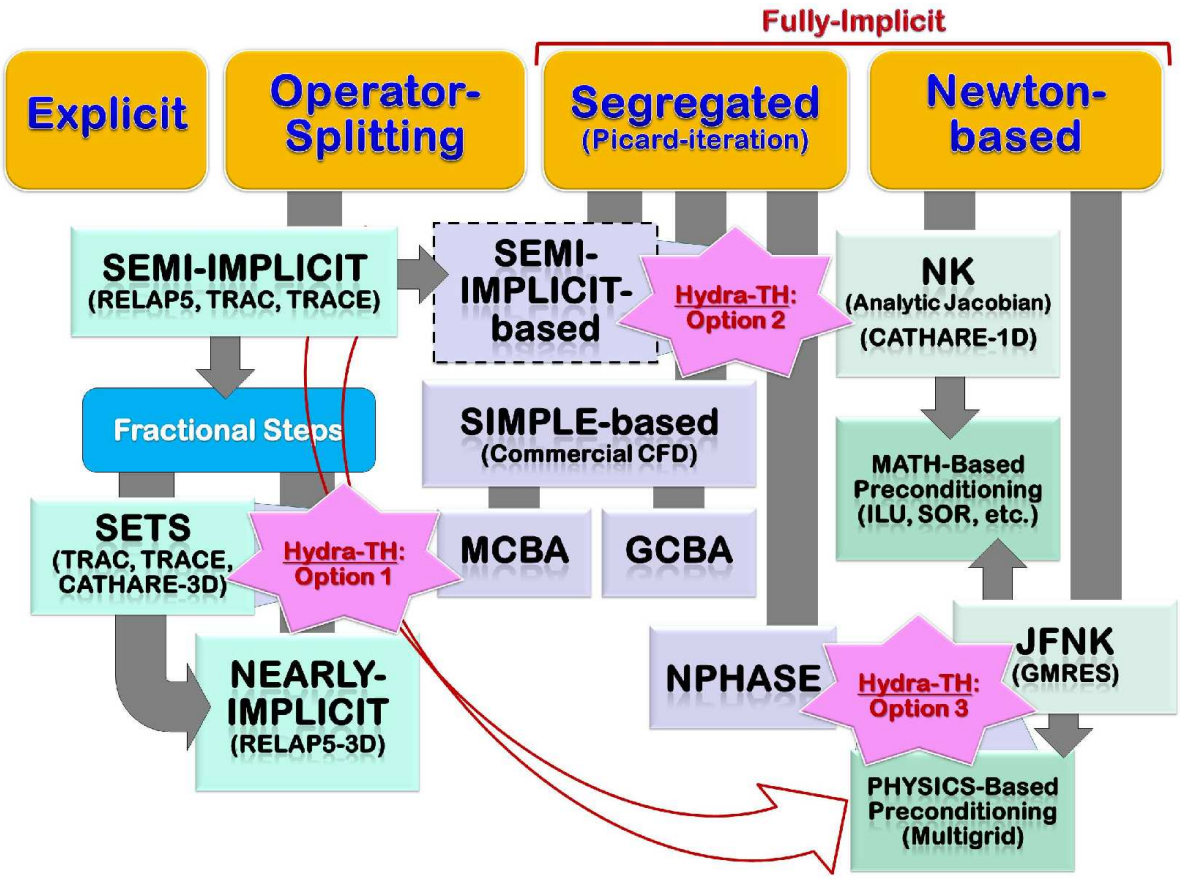


Figure 1: Road-map and options for multi-fluid modeling in Hydra-TH.

solution algorithm, employing the projection method as a preconditioner. At this time, the fully-implicit algorithm, discussed briefly in §3.3, has been implemented for a *single* fluid.

3. *Fully-Implicit Newton Algorithm.* ("Option 3" in Fig. 1.) The final step for the multi-field capability in Hydra-TH will result in a Newton-based iterative solution method with fully-implicit time marching. The goal of the Newton technique is to reduce the number of iterations compared to the Picard method.

3.2 Overview of the Semi-Implicit Multi-Field Projection Algorithm

A brief description of the governing equations of the semi-implicit algorithm, "Option 1" in Fig. 1, is now given. More details on the projection-preconditioned semi-, and fully-implicit solution techniques may be found in [9, 10, 11] and on the multi-field flow solution strategy in Hydra-TH in [8].

The semi-implicit projection method is based on Gresho's second-order "P2" method [6, 7], and is closely related to [2]. A discontinuous-Galerkin/finite-volume (DG/FV) formulation ensures local, i.e., cell-wise, conservation. The velocity, temperature (internal energy or enthalpy), and turbulence variables (e.g., k, ϵ, ω, etc. if a turbulence model is used) are cell-centered, while the pressure is centered at nodes of the computational mesh. The

node-centered pressure precludes checker-board modes, which avoids the use of troublesome pressure-stabilization techniques, e.g., Rhie-Chow filtering. This hybrid DG/FV – Galerkin FEM method forms the basis for the Hydra-TH incompressible solution algorithms.

The semi-implicit projection algorithm for multi-phase flows is based on the single-phase algorithm. The basic governing equations are [4, 12, 3]

$$\frac{\partial \alpha_k}{\partial t} + \nabla \cdot (\alpha_k \overline{\mathbf{v}}_k) = 0 \tag{1}$$

$$\alpha_k \overline{\rho}_k \left(\frac{\partial \overline{\mathbf{v}}_k}{\partial t} + \overline{\mathbf{v}}_k \cdot \nabla \overline{\mathbf{v}}_k \right) = -\alpha_k \nabla \overline{p} + \nabla \cdot (\alpha_k \overline{\boldsymbol{\tau}}_k) \tag{2}$$

$$\alpha_k \overline{\rho}_k C_{pk} \left(\frac{\partial \overline{T}_k}{\partial t} + \overline{\mathbf{v}}_k \cdot \nabla \overline{T}_k \right) = -\nabla \cdot \overline{\mathbf{q}}_k + \overline{q}_k''' \tag{3}$$

$$\nabla \cdot (\alpha_k \overline{\rho}_k \overline{\mathbf{v}}_k) = 0 \tag{4}$$

for $k = 1, \ldots, N$ ensemble-averaged fields, denoted by the overbar. In Eqs. (1–4) α_k, $\overline{\rho}_k$, $\overline{\mathbf{v}}_k$, $\overline{\boldsymbol{\tau}}_k$, $\overline{T}_k$, $\overline{\mathbf{q}}_k$, $\overline{q}_k'''$ denote the ensemble-averaged volume fraction, density, velocity, shear stress, temperature, diffusional heat flux, and volumetric heat source of field k, respectively, while $\overline{p}$ is the ensemble-averaged (single) pressure, and C_{pk} is the specific heat at constant pressure of field k. Eqs. (1–4) contain no inter-field mass, momentum, and energy transfer, and the fluid densities of the fields, $\overline{\rho}_k$ are constants, though they can be different for each field. Keeping the densities constant is intentional for testing and verification.

A projection algorithm can be derived for Eqs. (1–4) that is analogous to the single-field case, see [5] or [11] for an example. The resulting algorithm advances Eqs. (1–3) using semi-implicit time marching, similar to the single-field case, but enforces the divergence-free constraint in Eq. (4) by solving the following Poisson equation for the time-increment of the pressure

$$\nabla^2 \left(\overline{p}^{n+1} - \overline{p}^n \right) = \nabla \cdot \sum_k \left(\alpha_k \overline{\rho}_k \overline{\mathbf{v}}_k \right)^* / \Delta t \tag{5}$$

where the superscripts n, $n+1$, and $*$ denote time levels n, $n+1$, and the intermediate non-divergence-free state following the momentum update, respectively. Compared to the single-field algorithm, which enforces a divergence-free (single-field) velocity, Eq. (5) advances the pressure while enforcing a divergence-free mixture-momentum, $\nabla \cdot \sum_k \left(\alpha_k \overline{\rho}_k \overline{\mathbf{v}}_k \right)^{n+1} = 0$, satisfying Eq. (4). The velocity field for each field, k, is then projected using

$$\overline{\mathbf{v}}^{n+1} = \overline{\mathbf{v}}^* - \frac{\Delta t}{\alpha_k^{n+1} \rho_k^{n+1}} \left(\alpha_k^{n+1} \nabla p^{n+1} - \alpha_k^n \nabla p^n \right) \tag{6}$$

Note that Eq. (6) yields a divergence-free mixture-momentum, Eq. (4), but the individual field-velocities, $\overline{\mathbf{v}}_k$ are, in general, not divergence-free. This is a consequence of using a single pressure instead of multiple ones.

3.3 Overview of the Fully-Implicit Picard Algorithm

The semi-implicit projection method, discussed in [5] for single and in §3.2 for multiple fluids, has been extended to a fully implicit algorithm using Picard iterations, employing the projection method as a preconditioner. At this time, the Picard algorithm has been implemented and being exercised for a single fluid. A high-level overview of this Picard algorithm is now given; more details can be found in [9, 10, 11]. The governing equations are described first, followed by the algorithm.

Incompressible single-phase flows are computed by solving the conservation equations of mass, momentum, and energy. The conservation of linear momentum is

$$\rho \left\{ \frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} \right\} = \nabla \cdot \boldsymbol{\sigma} + \mathbf{f} \tag{7}$$

where $\mathbf{v} = (v_x, v_y, v_z)$ is the velocity, $\boldsymbol{\sigma}$ is the stress tensor, ρ is the mass density, and $\mathbf{f}$ is the body force. The body force contribution $\mathbf{f}$ typically accounts for buoyancy forces. The stress may be written in terms of the fluid pressure and the deviatoric stress tensor as $\boldsymbol{\sigma} = -p\mathbf{I} + \boldsymbol{\tau}$, where p is the pressure, $\mathbf{I}$ is the identity tensor, and $\boldsymbol{\tau}$ is the deviatoric stress. A constitutive equation relates the deviatoric stress and the strain as $\boldsymbol{\tau} = 2\mu\mathbf{S}$. The strain-rate tensor is written in terms of the velocity gradients as $\mathbf{S} = (\nabla\mathbf{v} + (\nabla\mathbf{v})^T)/2$. In the incompressible limit, the velocity field is solenoidal,

$$\nabla \cdot \mathbf{v} = 0 \tag{8}$$

which acts as a constraint on the momentum equation (7). Conservation of energy is expressed in terms of temperature [1], T, as

$$\rho C_p \left\{ \frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T \right\} = -\nabla \cdot \mathbf{q} + q''' \tag{9}$$

where C_p is the specific heat at constant pressure, $\mathbf{q}$ is the diffusional heat flux, and q''' represents volumetric heat sources and sinks, e.g., due to chemical reactions. Fourier's law relates the heat flux to the temperature gradient and thermal conductivity as $\mathbf{q} = -\kappa\nabla T$, where κ is the thermal conductivity. Naturally, the numerical solution of the constrained system of partial differential equations, Eqs. (7–9), require appropriate boundary and initial conditions; see [5, 11] for details.

For simplicity, only the momentum and pressure updates are discussed in the following, but the transport equations for the internal energy and turbulence variables are also implemented. The governing equations are Eqs. (7) and (8), and the vectors of unknowns are either

$$\mathbf{U} = \begin{bmatrix} p \\ \overline{\mathbf{v}} \end{bmatrix} \quad \text{or} \quad \mathbf{W} = \begin{bmatrix} \lambda \\ \overline{\mathbf{v}}^\star \end{bmatrix} \tag{10}$$

The fully-implicit algorithm seeks the solution at t^{n+1} iteratively, by updating $\mathbf{U}^{\diamond\diamond}$ or $\mathbf{W}^{\diamond\diamond}$ as

$$\begin{aligned} \lambda^{\diamond\diamond} &= \lambda^{\diamond} + \lambda' \\ p^{\diamond\diamond} &= p^{\diamond} + p' \\ \overline{\mathbf{v}}^{\diamond\diamond} &= \overline{\mathbf{v}}^{\diamond} + \mathbf{v}' \\ \overline{\mathbf{v}}^{\star\diamond\diamond} &= \overline{\mathbf{v}}^{\star\diamond} + \mathbf{v}^{\star\prime} = \overline{\mathbf{v}}^{\diamond\diamond} + 1/\rho\nabla\lambda^{\diamond\diamond} \end{aligned} \tag{11}$$

where λ is the Lagrange multiplier, $\lambda = \Delta t(p^{n+1} - p^n)$. We linearize the body force as

$$\mathbf{F}^{\diamond\diamond} = \mathbf{F}^{\diamond} + \mathbb{F}_{\mathbf{v}} \left(\mathbf{U}^{\diamond} \right) \mathbf{v}' + \mathbf{f}_p \left(\mathbf{U}^{\diamond} \right) p' \tag{12}$$

where specific forms of the linearization matrix $\mathbb{F}_{\mathbf{v}}$ and vector $\mathbf{f}_p$ are problem-dependent.

[1] The energy equation can be solved also in terms of specific internal energy u or specific enthalpy h based on user-defined input.

Fully-Implicit Picard Algorithm. Using Eq. (11) in the discretized momentum equation (7) yields

$$
\begin{aligned}
M\frac{\overline{\mathbf{v}}^{*^{\diamond\diamond}}-\overline{\mathbf{v}}^{n}}{\Delta t} &= (1-\theta)\left(K\overline{\mathbf{v}}^{n}-A(\rho,\overline{\mathbf{v}}^{n})\overline{\mathbf{v}}^{n}+\mathbf{F}^{n}\right)-\mathbf{B}p^{n}+ \\
&+\theta\Bigg\{\left[K-A(\rho,\overline{\mathbf{v}}^{\diamond})\right]\overline{\mathbf{v}}^{*^{\diamond\diamond}}+\mathbf{F}^{\diamond}-\underbrace{\left[K-A(\rho,\overline{\mathbf{v}}^{\diamond})\right]\left(\frac{1}{\rho}\nabla\lambda^{\diamond}\right)}_{\boldsymbol{\Xi}_t}\Bigg\} \\
&\underbrace{-\theta\Bigg\{\left[K-A(\rho,\overline{\mathbf{v}}^{n+1})\right]\left(\frac{1}{\rho}\nabla\lambda'\right)+\frac{1}{\theta_p\Delta t}\mathbf{f}_p\lambda'+\mathbb{F}_{\mathbf{v}}\mathbf{v}'\Bigg\}}_{\delta\boldsymbol{\mathcal{J}}_{\mathbf{v}}}
\end{aligned}
\tag{13}
$$

which, using $\mathbf{m}'=-\theta\left(\boldsymbol{\Xi}_t+\delta\boldsymbol{\mathcal{J}}_{\mathbf{v}}\right)$, is written as

$$
\begin{aligned}
\left[M-\theta\Delta t\Big(K-A\big(\rho,\overline{\mathbf{v}}^{\diamond}\big)\Big)\right]\overline{\mathbf{v}}^{*^{\diamond\diamond}} &= \left[M+(1-\theta)\Delta t\left(K-A\left(\rho,\overline{\mathbf{v}}^{n}\right)\right)\right]\overline{\mathbf{v}}^{n}+ \\
&+\Delta t\left[(1-\theta)\mathbf{F}^{n}+\theta\mathbf{F}^{\diamond}-\mathbf{B}p^{n}-\theta\,\delta\tilde{A}\overline{\mathbf{v}}^{\diamond}+\mathbf{m}'\right]
\end{aligned}
\tag{14}
$$

where θ is the implicitness parameter and $\delta\tilde{A}\overline{\mathbf{v}}^{\diamond}$ denotes the linearization for the quadratic advection terms, whose details are omitted for brevity. (Note that the overbar here indicates the finite-volume cell-averages; accordingly the pressure is not an average in the single-field algorithm as it is obtained from a finite-element procedure.) Now the Helmholtz decomposition is

$$
\overline{\mathbf{v}}^{*^{\diamond\diamond}}=\underbrace{\overline{\mathbf{v}}^{\diamond}+\overline{\mathbf{v}}'}_{\overline{\mathbf{v}}^{\diamond\diamond}}+\frac{1}{\rho}\nabla\left(\lambda^{\diamond}+\lambda'\right)
\tag{15}
$$

which yields the Poisson equation

$$
\nabla\cdot\frac{1}{\rho}\nabla\lambda^{\diamond\diamond}=\nabla\cdot\overline{\mathbf{v}}^{*^{\diamond\diamond}}
\tag{16}
$$

The fixed-point iteration algorithm is based on Eqs. (14–16):

Algorithm 1 *Picard Iteration*

1. *Set initial guess for $m=0$:*

$$
\begin{aligned}
\overline{\mathbf{v}}^{\diamond} &= \overline{\mathbf{v}}^{n} & \qquad \mathbf{v}' &= 0 \\
p^{\diamond} &= p^{n} & \qquad p' &= 0 \\
\lambda^{\diamond} &= 0 & \qquad \lambda' &= 0
\end{aligned}
$$

2. *<u>Start the mth iteration</u>.*

3. *Solve for $\overline{\mathbf{v}}^{*^{\diamond\diamond}}$, Eq.(14), with $\delta\boldsymbol{\mathcal{J}}_{\mathbf{v}}=0$.*

4. *Compute face-centered velocities, $v_f^{\star}$.*

5. *Compute the right-hand-side of Eq. (16) and solve for* $\lambda^{\diamond\diamond}$,

$$K_p \lambda^{\diamond\diamond} = D^{\diamond\diamond} \tag{17}$$

6. *Update the pressure*

$$p^{\diamond\diamond} = p^n + \frac{1}{\theta_p \Delta t} \lambda^{\diamond\diamond} \tag{18}$$

7. *Project the cell-centered velocities*

$$\overline{\mathbf{v}}^{\diamond\diamond} = \overline{\mathbf{v}}^{*} - \frac{1}{\rho} \mathbf{B} \lambda^{\diamond\diamond} \tag{19}$$

8. *Compute face gradients and project the face-centered velocities*

$$v_f = v_f^{*} - \frac{1}{\rho_f} ((B)\lambda^{\diamond\diamond})_f \cdot \mathbf{n} \tag{20}$$

9. *Compute velocity and pressure corrections,*

$$\begin{aligned} \mathbf{v}' &= \overline{\mathbf{v}}^{\diamond\diamond} - \overline{\mathbf{v}}^{\diamond} \\ \lambda' &= \lambda^{\diamond\diamond} - \lambda^{\diamond} \end{aligned} \tag{21}$$

and compute errors

$$\begin{aligned} \mathcal{E}_{\mathbf{V}}^{(m)} &= \mathcal{L}_2 (\mathbf{v}') \\ \mathcal{E}_{\lambda}^{(m)} &= \mathcal{L}_2 (\lambda') \end{aligned} \tag{22}$$

10. *Check convergence:* ***If*** *<u>any</u> of*

$$\begin{aligned} \mathcal{E}_{\mathbf{V}}^{(m)} &< \mathrm{tol}_a \\ \mathcal{E}_{\lambda}^{(m)} &< \mathrm{tol}_a \\ \mathcal{E}_{\mathbf{V}}^{(m)} &< \mathrm{tol}_r \ \mathcal{E}_{\mathbf{V}}^{(0)} \\ \mathcal{E}_{\lambda}^{(m)} &< \mathrm{tol}_r \ \mathcal{E}_{\lambda}^{(0)} \end{aligned} \tag{23}$$

are <u>not</u> satisfied, start a new Picard iteration: increment m, $\overline{\mathbf{v}}^{\diamond} = \overline{\mathbf{v}}^{\diamond\diamond}$, $p^{\diamond} = p^{\diamond\diamond}$, $\lambda^{\diamond} = \lambda^{\diamond\diamond}$, *and repeat from Step 2,* ***otherwise****, finish the time step:*

$$\begin{aligned} \overline{\mathbf{v}}^{n+1} &= \overline{\mathbf{v}}^{\diamond\diamond} \\ p^{n+1} &= p^{\diamond\diamond} \end{aligned} \tag{24}$$

3.4 Some Implementation Details for Multi-Field Flow

This section delves into some details on the software implementation for multiple fluids.

Fig. 2 depicts the current physics class inheritance tree in Hydra. The Hydra-TH subset consists of CCINSFlow (Cell-Centered Incompressible Navier-Stokes Flow) and its descendants. The three different stages (options) of multi-field functionality, discussed in §3.1, are planned to be implemented in classes CCSemiImplicit, CCPicard, and CCNewton, respectively,

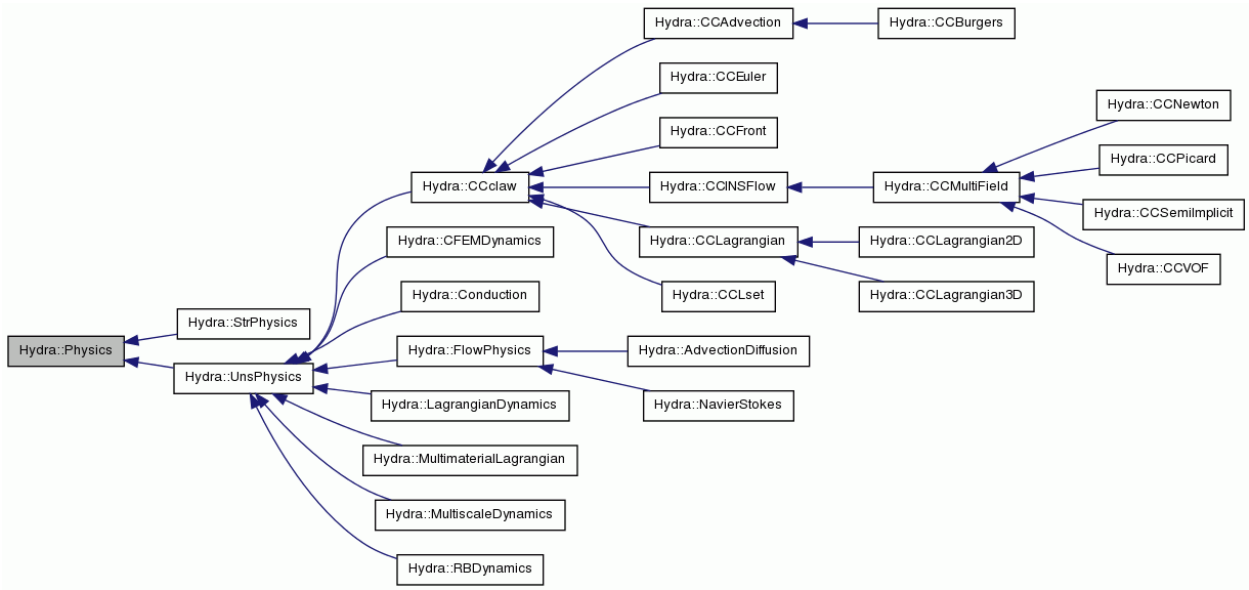


Figure 2: Physics objects inheritance tree in Hydra.

however, the design may undergo changes. At this time, the distinction between the functionalities of CCMultiField and CCSemiImplicit are somewhat blurred and for simplicity, all functionality is currently being put in the base CCMultiField. The evolving class design may also necessitate renaming the above classes.

The multi-field solver exercises significant code re-use by instantiating multiple volume fraction, momentum, and energy transport objects, containing single-field functionality. Data and output delegate registration has been extended to multiple fields, several of which the user can request. The statistics output requires no changes for multi-field flows, only new output delegates and some specialized accumulators will need to be added for statistics in the future. Examples are pressure-kth-volume-fraction covariance $\langle p'\alpha_k' \rangle$ or the kth-volume-fraction-velocity covariance $\langle \alpha_k' \mathbf{v}_k' \rangle$, where the comma denotes the fluctuation about the mean, e.g. $p' = p - \langle p \rangle$. The initialization procedure, yielding velocity and pressure fields that are mathematically and physically consistent with incompressible flows [1, 5], has been extended to multi-field flows.

The user input has also been extended to handle multiple fields. The design goals of the input deck's extension to multi-field flows were:

- User-friendly
- Clean
- Least intrusion to current
- Backward compatible
- Phase ID appears once per quantity (little burden on user)
- No significant change to existing (single-field) physics keywords parser
- No interface change to any analysis-keyword call-back functions

Listing (1) shows an example of the user input deck differences between the single-field incompressible solver (`CCINSFlow`) and its multi-field counterpart (`CCMultiField`). The

keyword `cc_multifield` is used to activate the (current and future) multi-field solver(s). The `plotvar - end` block still identifies the user-requested output variables, but the parser now accepts field ID augmented by an underscore for those variables where multiple fields are computed. Typos and wrong field IDs trigger warnings. The `initial - end` block is still used to specify initial conditions, which now takes an optional field ID or field ID range. In the absence of a field ID, the specifications apply to all fields, in the example in Listing (1), for all 4 fields. New keywords, such as `volfrac`, i.e., volume fraction, have also been introduced. Specifying boundary conditions is similar to the existing single-field syntax, but now also take optional field IDs or field ID ranges. The individual boundary and initial condition specifications have not changed. These relatively minor changes in syntax ensures backward compatibility with existing (single-field) input decks and allow for an easy porting of single-field problems to multi-phase descriptions. Statistics output (no example shown) will also be similar to the existing `plotstatvar - end` block and will take statistics with field ID, e.g. ⟨`pressure',volfrac2'`⟩ denoting the pressure-2nd-field-volume-fraction covariance. Run-time statistics for multi-field flows is a subject of future work. Various designs were entertained for the multi-field input deck syntax described above; more details may be found in the design slides in the Hydra repository at `<$HYDRA_SOURCE>/doc/design/multiphase_design.odp`.

3.5 Proof-of-Concept Calculations

This section gives proof-of-concept examples utilizing the new functionality discussed above.

Multi-field flow example. Fig. 3 shows a snapshot of a simple exercise calculation using the multi-field algorithm, Option 1, discussed in §3.2. Two fields with volume fractions, α_1, α_2, are simulated in a vortex street of a circular cylinder using the semi-implicit projection algorithm in which the fields are coupled only via the single pressure. Initially $\alpha_1 = 0$, $\alpha_2 = 1$, and a mass of $\alpha_1 = 0.1$ is injected at the cylinder wall into the carrier fluid. It is interesting to note that the extreme cases of $\alpha_k = 0$ and 1, i.e. phase appearance and disappearance, are properly handled by the algorithm, without requiring any special treatment, required in most commercial multi-phase flow algorithms.

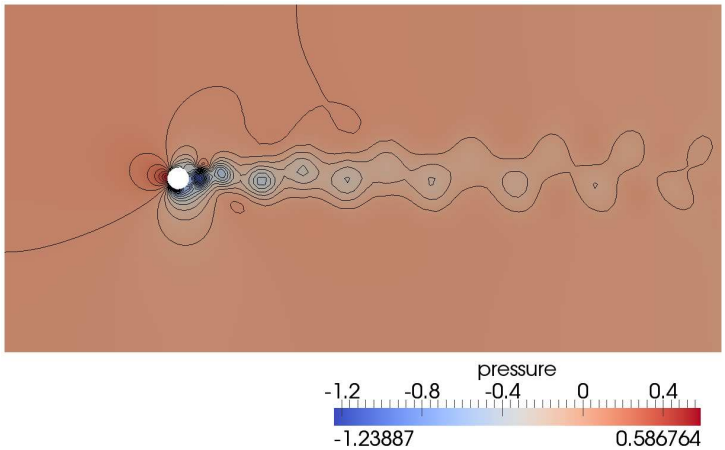
Fully implicit algorithm example. The fully implicit Picard algorithm, Option 2, discussed in §3.3, is free of operator-splitting errors and time-centering ensures that both velocity and pressure are second-order accurate in time Fig. 6. The method is unconditionally stable for arbitrarily large CFL numbers. This is demonstrated in Figs. 4 and 5, where instantaneous velocity fields of $Re = 100$ vortex streets are compared as computed by the semi-implicit and the fully implicit algorithms. The figures show that increasing the CFL number from 0.88 to 56 drives the semi-implicit algorithm unstable, while the fully implicit method remains stable, even as the Hopf bifurcation is stepped over by ever larger time steps, see Fig. 5. Naturally, the fully implicit algorithm is intended for rapid convergence to steady state and not for time-accurate transients.

The Picard algorithm may be suitable for a number of problems. However, it may take a large number of iterations for complex geometries and high Reynolds numbers, especially with stiff source terms, prevalent in multi-phase flow models. To further reduce the number of iterations required, a Newton-based algorithm will also be implemented in the future, discussed in §3.1.

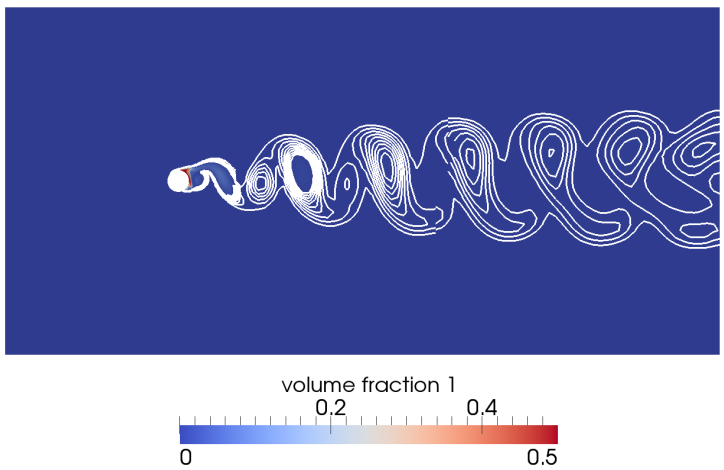
```
cc_navierstokes



  plotvar
    elem div
    elem procid
    elem vel
    node temp
  end



  initial

    temperature 100.0
  end



  velocity
    velx sideset 1 -1 0.0
    vely sideset 1 -1 0.1
    velz sideset 1 -1 0.1
  end

end
```

```
cc_multifield

  nfields 4

  plotvar
    elem div
    elem procid
    elem vel_1
    elem vel_2
    node temp_1
    node temp_2
  end

  initial 1
    volfrac 0.1
    temperature 100.0
  end

  initial 2
    volfrac 0.9
    temperature 100.0
  end

  volfrac 2:4
    sideset 1 -1 0.1
  end

  velocity 1-2
    velx sideset 1 -1 0.0
    vely sideset 1 -1 0.1
    velz sideset 1 -1 0.1
  end

end
```

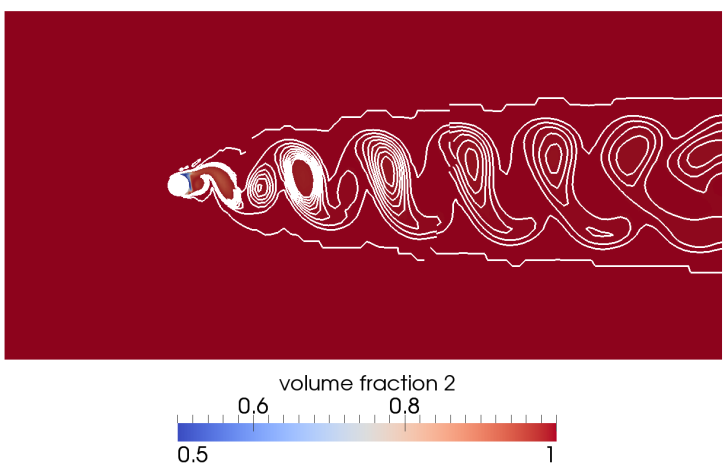
Listing 1: Comparative examples of single-, and multi-field input decks.



(a) Pressure.



(b) Volume fraction 1. The color legend is scaled to the extremes, while the contour lines divide the range 0.001-0.01 into 10 equal portions.



(c) Volume fraction 2. The color legend is scaled to the extremes, while the contour lines divide the range 0.99-1.0 into 10 equal portions.

Figure 3: Proof-of-concept demonstration of the "Option 1" (see Fig. 1) multi-field solver in Hydra-TH using vortex shedding with two fields of equal densities. The volume fraction initial conditions are $\alpha_1 = 0$ and $\alpha_2 = 1$. An influx of $\alpha_1 = 0.1$ is prescribed at the cylinder surface and advected downstream, from left to right.

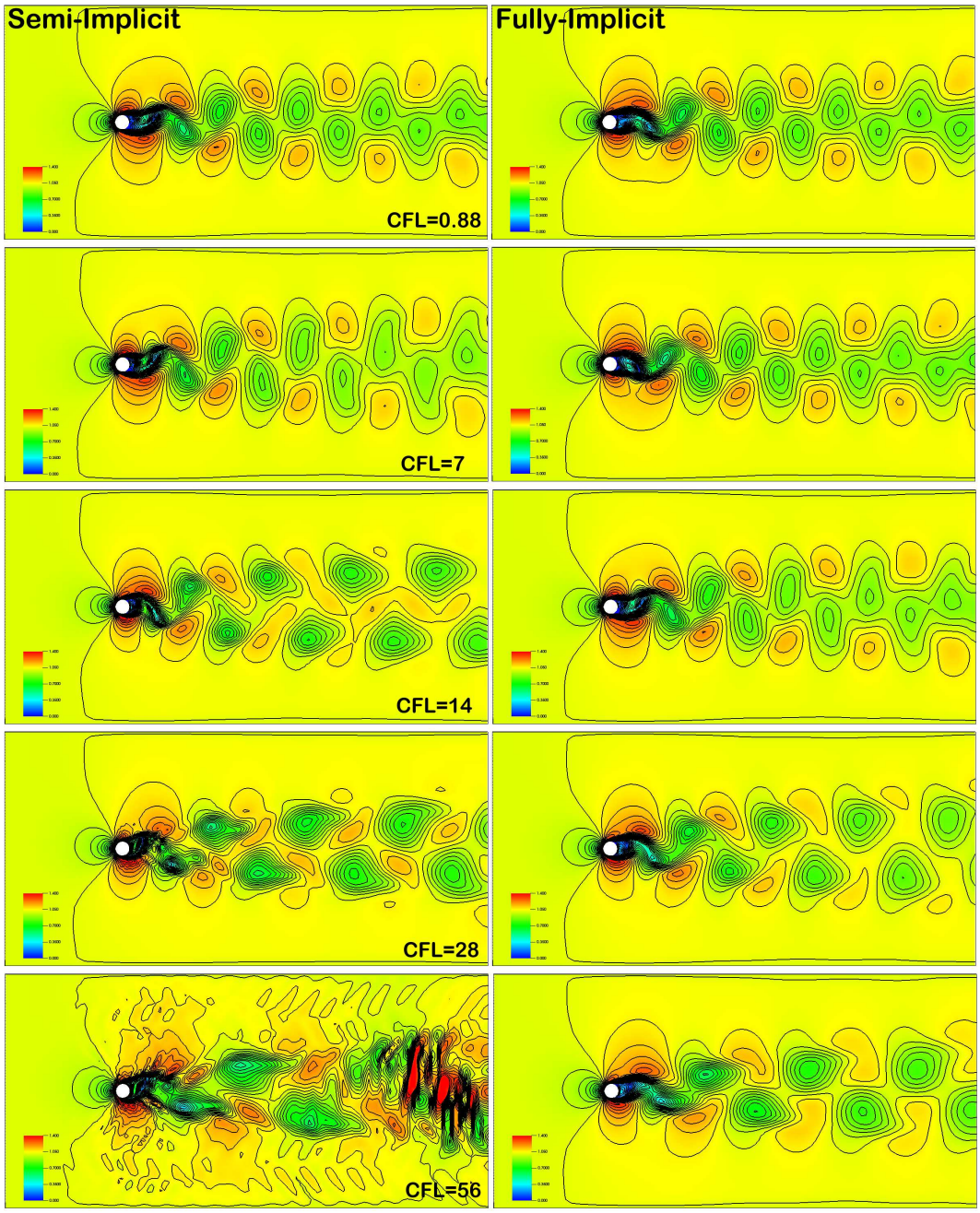


Figure 4: Vortex shedding behind a circular cylinder at $Re = 100$, computed by the semi-, and fully implicit projection algorithms, discussed in §3.2 and §3.3, respectively.

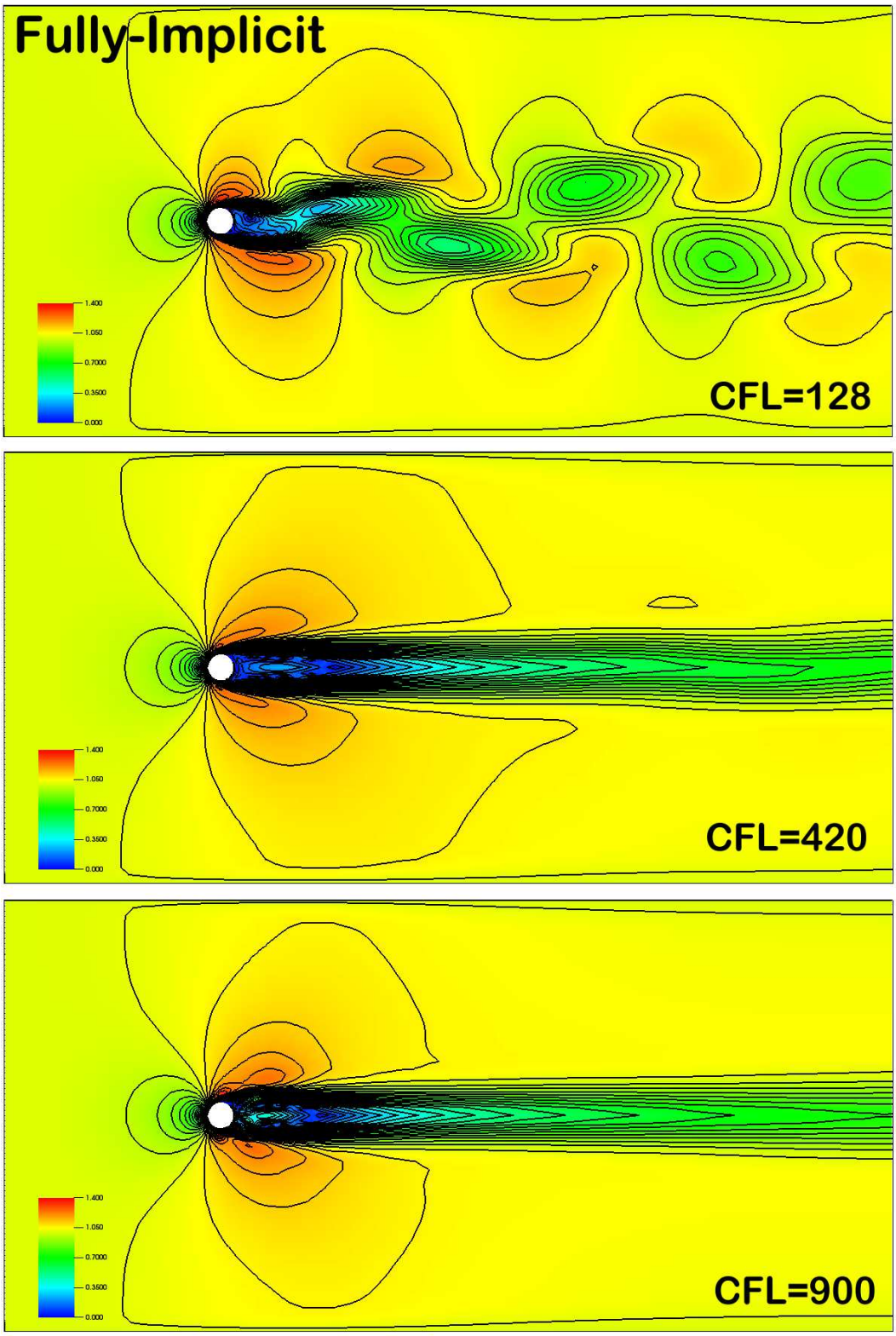


Figure 5: Vortex shedding behind a circular cylinder, at $Re = 100$, computed by the fully implicit projection algorithm, discussed in §3.3, using large time steps.

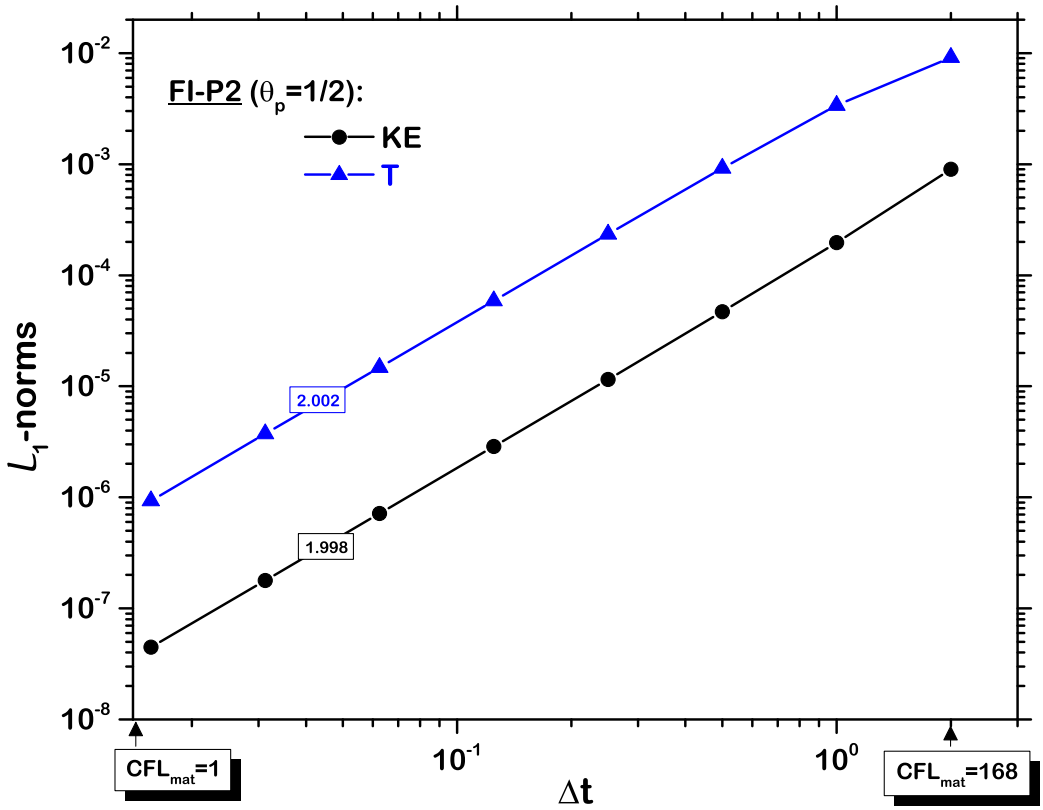


Figure 6: Convergence of L_1-norm errors of domain-average kinetic energy and temperature for natural convection in a square cavity with oscillatory temperature boundary conditions computed by the semi-implicit projection method and the fully implicit Picard algorithm. See [11] for more details.

4 Summary and Future Work

Some initial steps have been taken to prepare Hydra-TH for computing multi-phase thermal-hydraulics problems. Progress has been made on two fronts: deriving and implementing (1) a semi-implicit projection algorithm for N coupled fluids, and (2) a fully implicit projection-based Picard technique for a single fluid. Several software design and implementation issues have been explored and resolved. Work will continue on both fronts, ultimately resulting in several options for computing multi-phase flows, including a fully implicit Newton-Krylov algorithm with various options for inter-phase mass, momentum, and energy transfer, documented in L3:THM.CFD.P7.01, *Demonstration of Multiphase Flow with Hydra-TH*.

References

[1] J. Bakosi, M. Christon, R. Lowrie, L. Pritchett-Sheats, and R. Nourgaliev, *Large-Eddy Simulations of Turbulent Flow for Grid-to-Rod Fretting in Nuclear Reactors*, Nuclear Engineering and Design, (2013).

[2] J. B. Bell, P. Colella, and H. M. Glaz, *A second-order projection method for the incompressible navier-stokes equations*, Journal of Computational Physics, 85 (1989), pp. 257–283.

[3] C. Brennen, *Fundamentals of Multiphase Flow*, Cambridge University Press, 2005.

[4] G. Černe, S. Petelin, and I. Tiselj, *Coupling of the interface tracking and the two-fluid models for the simulation of incompressible two-phase flow*, Journal of Computational Physics, 171 (2001), pp. 776–804.

[5] M. A. Christon, *Hydra-TH Theory Manual*, Tech. Rep. LA-UR 11-05387, Los Alamos National Laboratory, September 2011.

[6] P. M. Gresho, *On the theory of semi-implicit projection methods for viscous incompressible flow and its implementation via a finite element method that also introduces a nearly consistent mass matrix. part 1: Theory*, International Journal for Numerical Methods in Fluids, 11 (1990), pp. 587–620.

[7] P. M. Gresho and S. T. Chan, *On the theory of semi-implicit projection methods for viscous incompressible flow and its implementation via a finite element method that also introduces a nearly consistent mass matrix. part 2: Implementation*, International Journal for Numerical Methods in Fluids, 11 (1990), pp. 621–659.

[8] R. Nourgaliev and M. Christon, *Solution Algorithms for Multi-Fluid-Flow Averaged Equations*, Tech. Rep. INL/EXT-12-27187, Idaho National Laboratory, December 2012.

[9] R. Nourgaliev, M. Christon, and J. Bakosi, *Notes on Newton-Krylov based Incompressible Flow Projection Solver*, Tech. Rep. INL/EXT-12-27197, Idaho National Laboratory, December 2012.

[10] ——, *Newton-Krylov based P2 Projection Solver for Fluid Flows*, Tech. Rep. INL/EXT-13-28278, Idaho National Laboratory, January 2013.

[11] R. Nourgaliev, M. Christon, J. Bakosi, R. Lowrie, and L. Pritchett-Sheats, *Hydra-TH: A Termal-Hydraulics Code for Nuclear Reactor Applications*, in NURETH-15 – Fifteenth International Topical Meeting on Nuclear Reactor Thermal Hydraulics, Pisa, Italy, May 2013.

[12] M. Wörner, *A compact introduction to the numerical modeling of multiphase flows*, tech. rep., Institute for nuclear reactor safety, Karlsruhe Research Center in the Helmholtz Association, 2003.