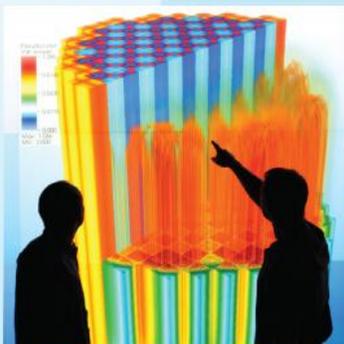


Power updates
and plant life extension

CASL-I-2012-0155-000



Engineering design
and analysis



L3:THM.CFD.P5.01

Mark Christon

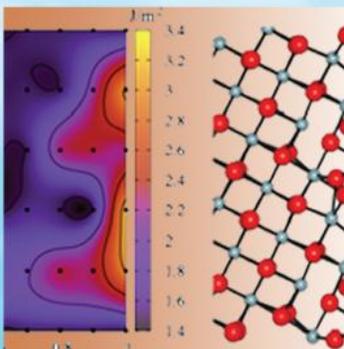
LANL

Completed: 9/29/2012

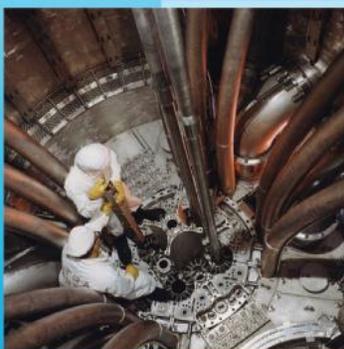
Science-enabling
high performance
computing



Fundamental science



Plant operational data



U.S. DEPARTMENT OF
ENERGY

Nuclear Energy

Notes on Newton-Krylov based Incompressible Flow Projection Solver

R.R. Nourgaliev^a, M.Christon^b, J.Bakosi^c

^a Idaho National Laboratory, Nuclear Science and Technology Division, Thermal Science & Safety Analysis Department, 2525 North Fremont Ave. Idaho Falls, ID 83415-3860

Email: Robert.Nourgaliev@inl.gov

^b Los Alamos National Laboratory, Computer, Computational and Statistical Sciences Division, Computational Physics Group (CCS-2) Los Alamos, NM 87545

Email: christon@lanl.gov

^c Los Alamos National Laboratory, Computer, Computational and Statistical Sciences Division, Computational Physics Group (CCS-2) Los Alamos, NM 87545

Email: jbakosi@lanl.gov

Abstract

The purpose of the present document is to formulate Jacobian-free Newton-Krylov algorithm for approximate projection method used in Hydra-TH code. Hydra-TH is developed by Los Alamos National Laboratory (LANL) under the auspices of the Consortium for Advanced Simulation of Light-Water Reactors (CASL) for thermal-hydraulics applications ranging from grid-to-rod fretting (GTRF) to multiphase flow subcooled boiling. Currently, Hydra-TH is based on the semi-implicit projection method, which provides an excellent platform for simulation of transient single-phase thermalhydraulics problems. This algorithm however is not efficient when applied for very slow or steady-state problems, as well as for highly non-linear multiphase problems relevant to nuclear reactor thermalhydraulics with boiling and condensation. These applications require fully-implicit tightly-coupling algorithms. The major technical contribution of the present report is the formulation of *fully-implicit projection algorithm* which will fulfill this purpose. This includes the definition of non-linear residuals used for GMRES-based linear iterations, as well as physics-based preconditioning techniques.

Key words: Multi-Physics, Multi-Scale problems, Finite Volume, Incompressible Flow, Approximate Projection, Jacobian-Free Newton Krylov (JFNK), Physics-Based Preconditioning

1. Introduction

The solution of the time-dependent incompressible single- and multi-phase flows poses several algorithmic problems due to the div-free constraint, and the concomitant spatial and temporal resolution required to perform time-accurate solutions particularly when complex geometry is involved. The initial deployment of Hydra-TH has focused on projection methods because of their computational efficiency and accuracy for transient flows. However, when applied to slow transients and steady-state problems, the currently existing projection methods are not cost-effective, due to stability restrictions imposed by material Courant limit. For these applications, fully-implicit algorithms are required. Here, we reformulate semi-implicit projection method to fit into the fully-implicit Jacobian-free Newton Krylov solution strategy.

We start with a short description of governing equations, defined in Section 3. Even though we limit our discussion here to single-phase flows, the basic ideas introduced are extendable to multi-fluid formulation [1].

A detailed review of projection methods is beyond the scope of this document, but a partial list of relevant work is provided for the interested reader. Projection methods, also commonly referred to as fractional-step, pressure correction methods, or Chorin's method [2] have grown in popularity over the past 20 years due to the relative ease of implementation and computational performance. This is reflected by the volume of work published on the development of second-order accurate projection methods, see for example van Kan [3], Bell, et al. [4], Gresho, et al. [5, 6, 7, 8], Almgren, et al. [9, 10, 11], Rider [12, 13, 14, 15], Minion [16], Guermond and Quartapelle [17], Puckett, et al. [18], Sussman, et al. [19], and Knio, et al. [20]. The numerical performance of projection methods has been considered by Brown and Minion [21, 22], Wetton [23], Guermond [24, 25], Guermond and Quartapelle [26, 27], and Almgren et al. [11]. A short introduction to semi-implicit projection method is given in Section 4.

The main technical contribution of this report is described in sections 5 and 6, introducing fully-implicit projection and its physics-based preconditioning.

Concluding remarks are given in the final section 7.

2. HydraTH

Hydra-TH [28] refers to the specific physics module that provides the hybrid finite-volume/finite-element incompressible/low-Mach flow solver. This is built as one of the many physics modules using the Hydra multiphysics toolkit. The toolkit provides a rich suite of components that permits rapid application development, I/O interfaces to permit reading/writing multiple file formats for meshes, plot data, time-history and surface-based output. The toolkit also provides run-time parallel domain decomposition with data-migration for both static and dynamic load-balancing. Linear algebra is handled through an abstract interface that permits use of popular libraries such as PETSC and Trilinos. Hydra's toolkit model for development provides lightweight, high-performance and reusable code components for agile development. Currently the toolkit supports finite-element based solvers for time-dependent heat conduction, time-dependent advection-diffusion, time-dependent incompressible flow, multiple Lagrangian hydrodynamics solvers, rigid-body dynamics, etc. In addition, unstructured-grid finite-volume solvers are available for solving time-dependent advection-diffusion, Burgers' equation, the compressible Euler equations, and incompressible/low-Mach Navier-Stokes equations. There are also interfaces to the FrontTier front-tracking software and to level-set methods.

3. Governing Equations

In the following discussion, we allow variable-density formulation. The **mass conservation** principle in divergence form is

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho v_j)}{\partial x_j} = 0. \quad (1)$$

In the incompressible limit, the velocity field is solenoidal,

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (2)$$

which implies a mass density transport equation,

$$\frac{\partial \rho}{\partial t} + v_j \frac{\partial \rho}{\partial x_j} = 0. \quad (3)$$

For constant density, Eq. (2) is neglected with Eq. (3) remaining as a constraint on the velocity field.

Momentum conservation. The conservation of linear momentum is

$$\rho \frac{\partial v_i}{\partial t} + \rho v_j \frac{\partial v_i}{\partial x_j} = \frac{\partial \sigma_{ij}}{\partial x_j} + \rho f_i \quad (4)$$

where v_i is the velocity, σ_{ij} is the stress tensor, ρ is the mass density, and f_i is the body force. The body force contribution ρf_i typically accounts for buoyancy forces with f_i representing the acceleration due to gravity.

The stress may be written in terms of the fluid pressure and the deviatoric stress tensor as

$$\sigma_{ij} = -p\delta_{ij} + \tau_{ij} \quad (5)$$

where p is the pressure, δ_{ij} is the Kronecker delta, and τ_{ij} is the deviatoric stress tensor. A constitutive equation relates the deviatoric stress and the strain rate, e.g.,

$$\tau_{ij} = 2\mu S_{ij}. \quad (6)$$

The strain-rate tensor is written in terms of the velocity gradients as

$$S_{ij} = \frac{1}{2} \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right). \quad (7)$$

Energy conservation. The energy equation may be expressed in terms of temperature, T , as

$$\frac{\partial \rho C_p T}{\partial t} + \frac{\partial}{\partial x_j} (\rho v_j C_p T) = -\frac{\partial q_j}{\partial x_j} + \dot{q}''' \quad (8)$$

where C_p is the specific heat at constant pressure, q_i is the diffusional heat flux rate, and \dot{q}''' represents volumetric heat sources and sinks, e.g., due to exothermic/endothermic chemical reactions. Fourier's law relates the heat flux rate to the temperature gradient and thermal conductivity

$$q_i = -\kappa \frac{\partial T}{\partial x_i} \quad (9)$$

where κ is the thermal conductivity.

Alternatively, one can solve in terms of specific internal energy:

$$\frac{\partial \rho u}{\partial t} + \frac{\partial}{\partial x_j} (\rho v_j u) = -\frac{\partial q_j}{\partial x_j} + \dot{q}''' \quad (10)$$

with a given function

$$u = \mathcal{F}(T)$$

For example,

$$u(T) = u_0 + C_v (T - T_0)$$

where u_0 and T_0 are the values of specific internal energy and temperature at some reference point, while C_v is specific heat.

Scalar transport. In addition, we consider a coupled solution for transport of scalars:

$$\frac{\partial \rho \phi_n}{\partial t} + \frac{\partial}{\partial x_j} (\rho v_j \phi_n) = -\frac{\partial \mathcal{J}_{n_j}}{\partial x_j} + \mathcal{J}_n''' \quad (11)$$

where by \mathcal{J}_{n_j} and \mathcal{J}_n''' we denote diffusive flux and volumetric sources for a scalar ϕ_n . Note that ϕ_n could represent turbulence transport quantities (e.g., turbulent kinetic energy k). In this case, momentum and heat diffusion coefficients are considered to be a function of ϕ_n . In the most general case,

$$\mu(T, \phi_n) \quad \text{and} \quad \kappa(T, \phi_n), \quad n = 0, \dots, N-1$$

4. Semi-Implicit Projection

Following the well-established finite-volume procedure, we discretize momentum equation in space, integrate by parts, and apply the divergence theorem. Using a piecewise-constant weight functions yields

$$\rho \frac{d}{dt} \int_{\Omega^e} \mathbf{v} d\Omega^e + \oint_{\Gamma^e} \rho \mathbf{v} (\mathbf{v} \cdot \mathbf{n}) d\Gamma^e - \oint_{\Gamma^e} \boldsymbol{\tau} \cdot \mathbf{n} d\Gamma^e + \int_{\Omega^e} \nabla p d\Omega^e - \int_{\Omega^e} \mathbf{f} d\Omega^e \quad (12)$$

Using definition for the cell-average,

$$\bar{u} = \frac{1}{\Omega^e} \int_{\Omega^e} u^h \quad (13)$$

the spatially-discrete momentum equations become

$$\rho \Omega^e \frac{d\bar{\mathbf{v}}}{dt} + \oint_{\Gamma^e} \rho \mathbf{v} (\mathbf{v} \cdot \mathbf{n}) d\Gamma^e - \oint_{\Gamma^e} \boldsymbol{\tau} \cdot \mathbf{n} d\Gamma^e + \int_{\Omega^e} \nabla p d\Omega^e - \int_{\Omega^e} \mathbf{f} d\Omega^e \quad (14)$$

The projection algorithm can be derived a number of ways. Here, we choose to first develop the time-integrator, and identify the terms associated with the projection via a Helmholtz decomposition of the velocity. Before

proceeding we define the following mass, advective, viscous, gradient and body-force operators.

$$M = \rho \Omega^e \quad (15)$$

$$A(\rho, \mathbf{v}) \bar{\mathbf{v}} = \oint_{\Gamma^e} \rho \mathbf{v} (\mathbf{v} \cdot \mathbf{n}) d\Gamma^e \quad (16)$$

$$K \bar{\mathbf{v}} = \oint_{\Gamma^e} \boldsymbol{\tau} \cdot \mathbf{n} d\Gamma^e \quad (17)$$

$$\mathbf{B} \bar{p} = \int_{\Omega^e} \nabla p d\Omega^e \quad (18)$$

$$\mathbf{F} = \int_{\Omega^e} \mathbf{f} d\Omega^e \quad (19)$$

We form the global operators, apply forward-Euler first, then backward-Euler with explicit advection in both cases, and take the sum of the fully-discrete systems results in the following

$$\begin{aligned} M \frac{\bar{\mathbf{v}}^{n+1} - \bar{\mathbf{v}}^n}{\Delta t} &= (1 - \theta) K \bar{\mathbf{v}}^n + \theta K \bar{\mathbf{v}}^{n+1} + (1 - \theta) \mathbf{F}^n + \theta \mathbf{F}^{n+1} - \\ &- (1 - \theta) A(\rho, \mathbf{v}) \bar{\mathbf{v}}^n - \theta A(\rho, \mathbf{v}) \bar{\mathbf{v}}^{n+1} - \mathbf{B} \bar{p}^n - \theta_p \mathbf{B} (\bar{p}^{n+1} - \bar{p}^n) \end{aligned} \quad (20)$$

where $0 \leq \theta \leq 1$, $\theta = 0$ corresponds to a forward-Euler, $\theta = 1/2$ a trapezoidal rule, and $\theta = 1$ backward-Euler treatment of viscous and body-force terms.

Using the *Helmholtz decomposition* as

$$\rho \bar{\mathbf{v}}^* = \rho \bar{\mathbf{v}}^{n+1} + \nabla \lambda \quad (21)$$

we introduce the following definition

$$\lambda = \theta_p \Delta t (\bar{p}^{n+1} - \bar{p}^n) \quad (22)$$

Plugging these into Eq. (20), the momentum equation can be formulated for the approximate (“predictor”) velocity as

$$\begin{aligned} [M - \theta \Delta t (K - A(\rho, \mathbf{v}))] \bar{\mathbf{v}}^* &= [M + (1 - \theta) \Delta t (K - A(\rho, \mathbf{v}))] \bar{\mathbf{v}}^n + \\ &+ \Delta t ((1 - \theta) \mathbf{F}^n + \theta \mathbf{F}^{n+1} - B \bar{p}^n) + \end{aligned} \quad (23)$$

$$+ \theta \Delta t A(\rho, \mathbf{v}) \frac{\nabla \lambda}{\rho} - \theta \Delta t K \frac{\nabla \lambda}{\rho} + \left[M \frac{\nabla \lambda}{\rho} - B \lambda \right]$$

Using the Helmholtz decomposition, and requiring $\nabla \bar{\mathbf{v}}^{n+1} = 0$, yields a pressure-Poisson equation (PPE) that can be solved for the *Lagrange multiplier* λ :

$$\nabla \cdot \frac{1}{\rho} \nabla \lambda = \nabla \cdot \bar{\mathbf{v}}^* \quad (24)$$

Given a velocity and pressure at time-level n , the P2 algorithm proceeds as follows.

Algorithm 1. *Basic P2 Algorithm*

1. Solve for $\bar{\mathbf{v}}^*$

$$\begin{aligned} [M - \theta \Delta t (K - A(\rho, \mathbf{v}))] \bar{\mathbf{v}}^* &= [M + (1 - \theta) \Delta t (K - A(\rho, \mathbf{v}))] \bar{\mathbf{v}}^n + \\ &+ \Delta t ((1 - \theta) \mathbf{F}^n + \theta \mathbf{F}^{n+1} - B \bar{p}^n) \end{aligned} \quad (25)$$

2. Form the right-hand-side of the PPE, solve for λ ,

$$K_p \lambda = D \quad (26)$$

3. Update the pressure

$$\bar{p}^{n+1} = \bar{p}^n + \frac{1}{\theta_p \Delta t} \lambda \quad (27)$$

Note that testing over the last 20 years or so has indicated that using $\theta_p = 1/2$ to update the pressure can lead to temporal oscillations in the pressure. For this reason, we use $\theta_p = 1$ in the implementation.

4. Project the cell-centered velocities

$$\bar{\mathbf{v}}^{n+1} = \bar{\mathbf{v}}^* - \frac{1}{\rho} \mathbf{B} \lambda \quad (28)$$

5. Compute face gradients and project the face-centered velocities

$$v_f = v_f^* - \frac{1}{\rho_f} ((B)\lambda)_f \cdot \mathbf{n} \quad (29)$$

6. Repeat steps 1 - 5 until the termination time is reached

5. Fully-Implicit Projection

For the sake of simplicity, consider isothermal constant-density flow. Thus, the vector of unknowns is

$$\mathbf{U} = \begin{bmatrix} \bar{p} \\ \bar{\mathbf{v}} \end{bmatrix}$$

or, in terms of Lagrange multiplier:

$$\mathbf{V} = \begin{bmatrix} \lambda \\ \bar{\mathbf{v}} \end{bmatrix}$$

Let us search a new-time solution iteratively, defining new-iteration values $\mathbf{U}^{\diamond\diamond}$ or $\mathbf{V}^{\diamond\diamond}$ in the following incremental form:

$$\bar{p}^{\diamond\diamond} = \bar{p}^{\diamond} + p' \quad (30)$$

$$\lambda^{\diamond\diamond} = \lambda^{\diamond} + \lambda' \quad (31)$$

$$\bar{\mathbf{v}}^{\diamond\diamond} = \bar{\mathbf{v}}^{\diamond} + \mathbf{v}' \quad (32)$$

and assume the following linearization of body force:

$$\mathbf{F}^{\diamond\diamond} = \mathbf{F}^{\diamond} + \mathbb{F}_{\mathbf{v}}(\mathbf{F}^{\diamond}) \mathbf{v}' + \mathbf{f}_p(\mathbf{F}^{\diamond}) p' \quad (33)$$

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

Plug these into eqs.(21), (22) and (20):

$$\bar{\mathbf{v}}^* = \underbrace{\bar{\mathbf{v}}^{\diamond} + \mathbf{v}'}_{\text{Divergence-free part}} + \frac{1}{\rho} \nabla (\lambda^{\diamond} + \lambda') \quad (34)$$

$$\lambda^{\diamond} + \lambda' = \underbrace{\theta_p \Delta t (\bar{p}^{\diamond} - \bar{p}^n)}_{\lambda^{\diamond}} + \underbrace{\theta_p \Delta t p'}_{\lambda'} \quad (35)$$

$$\begin{aligned} M(\bar{\mathbf{v}}^{\diamond} + \mathbf{v}' - \bar{\mathbf{v}}^n) &= \Delta t(1 - \theta)K\bar{\mathbf{v}}^n + \Delta t\theta K(\bar{\mathbf{v}}^{\diamond} + \mathbf{v}') - \\ &- \Delta t(1 - \theta)A(\rho, \mathbf{v})\bar{\mathbf{v}}^n - \Delta t\theta A(\rho, \mathbf{v})(\bar{\mathbf{v}}^{\diamond} + \mathbf{v}') + \\ &+ \Delta t(1 - \theta)\mathbf{F}^n + \Delta t\theta(\mathbf{F}^{\diamond} + \mathbb{F}_{\mathbf{v}}\mathbf{v}' + \mathbf{f}_p p') - \\ &- \Delta t\mathbf{B}\bar{p}^n - \Delta t\theta_p \mathbf{B}(\bar{p}^{\diamond} + p' - \bar{p}^n) \end{aligned} \quad (36)$$

After re-grouping, the momentum equation becomes:

$$\left[\Delta t \theta M^{-1} \left(\mathbf{f}_p - \frac{\theta_p}{\theta} \mathbf{B} \right) \right] p' + \left[1 - \Delta t \theta M^{-1} (K - A(\rho, \mathbf{v}) + \mathbb{F}_v) \right] \mathbf{v}' = -\mathbf{res}_v \quad (37)$$

where

$$\begin{aligned} \mathbf{res}_v = & \bar{\mathbf{v}}^\diamond - \bar{\mathbf{v}}^n - \Delta t M^{-1} \left((1 - \theta) K \bar{\mathbf{v}}^n + \theta K \bar{\mathbf{v}}^\diamond \right) + \\ & + \Delta t M^{-1} \left((1 - \theta) A(\rho, \mathbf{v}) \bar{\mathbf{v}}^n + \theta A(\rho, \mathbf{v}) \bar{\mathbf{v}}^\diamond \right) - \\ & - \Delta t M^{-1} \left((1 - \theta) \mathbf{F}^n + \theta \mathbf{F}^\diamond \right) + \\ & + \Delta t M^{-1} \left(\mathbf{B} \bar{p}^n + \theta_p \mathbf{B} (\bar{p}^\diamond - \bar{p}^n) \right) \end{aligned} \quad (38)$$

From eq.(34),

$$\bar{\mathbf{v}}^\diamond + \mathbf{v}' = \bar{\mathbf{v}}^* - \frac{\theta_p \Delta t}{\rho} \nabla \left(\bar{p}^\diamond + p' - \bar{p}^n \right) \quad (39)$$

which can be plugged into eq.(36) to get a counterpart of the momentum “predictor” equation (23):

$$\begin{aligned} & [M - \Delta t \theta (K - A(\rho, \mathbf{v}) + \mathbb{F}_v)] \bar{\mathbf{v}}^* = \\ & = [M + \Delta t (1 - \theta) (K - A(\rho, \mathbf{v}))] \bar{\mathbf{v}}^n - \boxed{\Delta t \theta K \left(\frac{\Delta t \theta_p}{\rho} \nabla \left(\bar{p}^\diamond + p' - \bar{p}^n \right) \right)} + \\ & \quad + \boxed{\Delta t \theta A(\rho, \mathbf{v}) \left(\frac{\Delta t \theta_p}{\rho} \nabla \left(\bar{p}^\diamond + p' - \bar{p}^n \right) \right)} + \\ & \quad + \Delta t (1 - \theta) \mathbf{F}^n + \Delta t \theta \left(\mathbf{F}^\diamond - \mathbb{F}_v \left(\frac{\Delta t \theta_p}{\rho} \nabla \left(\bar{p}^\diamond - \bar{p}^n \right) + \bar{\mathbf{v}}^\diamond \right) \right) - \\ & \quad - \Delta t \theta \left[\mathbb{F}_v \frac{\Delta t \theta_p}{\rho} \nabla - \mathbf{f}_p \right] p' - \\ & - \Delta t \mathbf{B} \bar{p}^n - \boxed{\left(\Delta t \theta_p \mathbf{B} (\bar{p}^\diamond + p' - \bar{p}^n) - \frac{\Delta t \theta_p}{\rho} \nabla \left(\bar{p}^\diamond + p' - \bar{p}^n \right) \right)} \end{aligned} \quad (40)$$

Note that the terms in boxes are dropped in eq.(25). Next, we further re-group this equation as:

$$\mathfrak{P} \bar{\mathbf{v}}^* = \mathbf{m}^\diamond - \mathfrak{G} p' \quad (41)$$

where

$$\begin{aligned} \mathbf{m}^\diamond &= [M + \Delta t(1 - \theta)(K - A(\rho, \mathbf{v}))] \bar{\mathbf{v}}^n + \\ &+ \Delta t(1 - \theta) \mathbf{F}^n + \Delta t \theta \left(\mathbf{F}^\diamond - \mathbb{F}_v \left(\bar{\mathbf{v}}^\diamond + \frac{1}{\rho} \nabla \lambda^\diamond \right) \right) - \Delta t \mathbf{B} \bar{p}^n - \\ &- \Delta t \theta [K - A(\rho, \mathbf{v})] \left(\frac{1}{\rho} \nabla \lambda^\diamond \right) - \mathbf{B} \lambda^\diamond + \frac{1}{\rho} \nabla \lambda^\diamond \end{aligned} \quad (42)$$

$$\mathfrak{G} \equiv \Delta t \theta \left[\mathbb{F}_v \frac{\Delta t \theta_p}{\rho} \nabla - \mathbf{f}_p + \frac{\theta_p}{\theta} \mathbf{B} - \frac{\Delta t \theta_p}{\rho \theta} \nabla + (K - A(\rho, \mathbf{v})) \frac{\Delta t \theta_p}{\rho} \nabla \right] \quad (43)$$

$$\mathfrak{P} \equiv [M - \Delta t \theta (K - A(\rho, \mathbf{v})) + \mathbb{F}_v] \quad (44)$$

and we used equation (35) for definition of λ^\diamond . Thus, “predictor” velocity can be computed as

$$\bar{\mathbf{v}}^* = \mathfrak{P}^{-1} \left(\mathbf{m}^\diamond - \mathfrak{G} p' \right) \quad (45)$$

Finally, we can form *incremental PPE* by taking divergence of eq.(34) and using eq.(45):

$$\nabla \cdot \frac{1}{\rho} \nabla \lambda' + \frac{1}{\theta_p \Delta t} \nabla \cdot \mathfrak{G} \lambda' = -res_\lambda \quad (46)$$

where

$$res_\lambda = \nabla \cdot \frac{1}{\rho} \nabla \lambda^\diamond - \nabla \cdot \left(\mathfrak{P}^{-1} \mathbf{m}^\diamond \right) \quad (47)$$

With this, linear iterations of a Newton-based algorithm are defined by the following equation:

$$\underbrace{\begin{bmatrix} \nabla \cdot \frac{1}{\rho} \nabla + \frac{1}{\theta_p \Delta t} \nabla \cdot \mathfrak{G} & 0 \\ \frac{\theta}{\theta_p} M^{-1} \left(\mathbf{f}_p - \frac{\theta_p}{\theta} \mathbf{B} \right) & 1 - \Delta t \theta M^{-1} (K - A(\rho, \mathbf{v})) + \mathbb{F}_v \end{bmatrix}}_{\text{Jacobian, } \mathbb{J}_v} \underbrace{\begin{bmatrix} \lambda' \\ \mathbf{v}' \end{bmatrix}}_{\mathbf{v}'} = - \underbrace{\begin{bmatrix} res_\lambda \\ \mathbf{res}_v \end{bmatrix}}_{\mathbf{res}_v} \quad (48)$$

Non-linear residuals res_λ and \mathbf{res}_v are supplied to PETSC-SNES [29] for JFNK implementation.

6. Preconditioning

6.1. General strategy

Consider the following modification of eq.(48):

$$\underbrace{\mathbb{J}_{\mathbf{V}} \mathbb{P}^{-1}}_{\mathbb{J}_{\mathbf{P}}} \underbrace{\mathbb{P} \mathbf{V}'}_{\mathbf{V}''} = \underbrace{-r \vec{e}_{s_{\mathbf{V}}}}_{\vec{b}}(\mathbf{V}') \quad (49)$$

where \mathbb{P} symbolically represents the preconditioning matrix (or process), and \mathbb{P}^{-1} is its inverse. Thus, the solution procedure is splitted into two processes:

1. Solving for

$$\mathbb{J}_{\mathbf{P}} \mathbf{V}'' = \vec{b} \quad (50)$$

(this is what actually crunched by GMRES), and

2. Preconditioning:

$$\mathbf{V}' = \mathbb{P}^{-1} \mathbf{V}'' \quad (51)$$

While one refers to the matrix/process \mathbb{P} , operationally the algorithm only requires the action of \mathbb{P}^{-1} on a vector. The main requirement is that \mathbb{P} designed properly, to enable clustering eigenvalues of the $\mathbb{J}_{\mathbf{P}}$, making the solution of eq.(50) to converge faster.

For effective preconditioning of the fully-implicit projection algorithm, we can use semi-implicit algorithm described in Section 4. The strategy with involving a legacy (e.g., operator-splitting) algorithm for preconditioning is commonly referred to as *Physics-(Process)-based preconditioning (PBP)* [30, 31, 32, 33, 34], to be contrasted to the *Matrix-(Math)-based preconditioning (MBP)* algorithms. The later include different flavors of SOR, SSOR, ILU, MILU, ILUT, ILUTP, ILUS, ILUC, etc. preconditioners, see [35] for review. In these cases, the preconditioning matrix \mathbb{P} is required, as a suitable approximation for $\mathbb{J}_{\mathbf{V}}$.

In the following section, we will describe details of our implementation of the *semi-implicit projection* as PBP, emphasizing all differences relative to the using this algorithm as a solver (in an operator-splitting OS mode).

6.2. Semi-Implicit Projection as Physics-Based Preconditioning

At the input of the preconditioning step, we have current Newton iteration values of $\bar{\mathbf{v}}^\diamond$, \bar{p}^\diamond and λ^\diamond , and current update values $\bar{\mathbf{v}}''$, \bar{p}'' and λ'' . In the OS splitting mode, these are:

$$\bar{\mathbf{v}}^\diamond = \bar{\mathbf{v}}^n, \quad \bar{p}^\diamond = \bar{p}^n, \quad \lambda^\diamond = 0, \quad \bar{\mathbf{v}}'' = 0, \quad \bar{p}'' = 0 \quad \text{and} \quad \lambda'' = 0$$

The task of the preconditioning is to convert these into $\bar{\mathbf{v}}^\heartsuit$, \bar{p}^\heartsuit , λ^\heartsuit , $\bar{\mathbf{v}}'$, \bar{p}' and λ' , where

$$\begin{aligned} \bar{\mathbf{v}}^\heartsuit &= \bar{\mathbf{v}}^\diamond + \bar{\mathbf{v}}' \\ \bar{p}^\heartsuit &= \bar{p}^\diamond + \bar{p}' \\ \lambda^\heartsuit &= \lambda^\diamond + \lambda' \end{aligned} \quad (52)$$

In the OS mode, $\Phi^\heartsuit = \Phi^{n+1}$, where $\Phi = \bar{\mathbf{v}}$, \bar{p} and λ .

We define Helmholtz decomposition as

$$\bar{\mathbf{v}}^{\heartsuit,*} = \underbrace{\bar{\mathbf{v}}^\diamond + \bar{\mathbf{v}}'}_{\text{Divergence-free part, } \bar{\mathbf{v}}^\heartsuit} + \frac{1}{\rho} \nabla \left(\underbrace{\lambda^\diamond + \lambda'}_{\lambda^\heartsuit} \right) \quad (53)$$

Non-incremental Form.

1. The first step would be to solve for non-solenoidal (“predictor”) velocity field, $\bar{\mathbf{v}}^{\heartsuit,*}$, using one of the following options.

Option-A:

$$\begin{aligned} & [M - \Delta t \theta (K - A(\rho, \mathbf{v}) + \mathbb{F}_v)] \bar{\mathbf{v}}^{\heartsuit,*} = \\ & = [M + \Delta t(1 - \theta)(K - A(\rho, \mathbf{v}))] \bar{\mathbf{v}}^n - \boxed{\Delta t \theta K \left(\frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond + p'' - \bar{p}^n) \right)} + \\ & \quad + \boxed{\Delta t \theta A(\rho, \mathbf{v}) \left(\frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond + p'' - \bar{p}^n) \right)} + \\ & \quad + \Delta t(1 - \theta) \mathbf{F}^n + \Delta t \theta \left(\mathbf{F}^\diamond - \mathbb{F}_v \left(\frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond - \bar{p}^n) + \bar{\mathbf{v}}^\diamond \right) \right) - \\ & \quad - \Delta t \theta \left[\mathbb{F}_v \frac{\Delta t \theta_p}{\rho} \nabla - \mathbf{f}_p \right] p'' - \\ & - \Delta t \mathbf{B} \bar{p}^n - \boxed{\left(\Delta t \theta_p \mathbf{B} (\bar{p}^\diamond + p'' - \bar{p}^n) - \frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond + p'' - \bar{p}^n) \right)} \end{aligned} \quad (54)$$

In the OS mode, $\bar{\mathbf{v}}^{\heartsuit,*} = \bar{\mathbf{v}}^*$ and eq.(54) reduces to eq.(25).

Option-B:

Equation (54) is of advection-diffusion type, which is not well amenable to multigrid algorithm, and solved in `Hydra` by ILU-based solver. Another viable option would be to convert it into the parabolic equation, by taking out advection operator on the left-hand-side (leaving it to GMRES to deal with). Thus, the parabolic equation would be:

$$\begin{aligned}
[M - \Delta t \theta (K + \mathbb{F}_v)] \bar{\mathbf{v}}^{\heartsuit,*} &= [M + \Delta t(1 - \theta) (K - A(\rho, \mathbf{v}))] \bar{\mathbf{v}}^n - \\
&- \Delta t \theta A(\rho, \mathbf{v}) (\bar{\mathbf{v}}^\diamond + \bar{\mathbf{v}}'') - \boxed{\Delta t \theta K \left(\frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond + p'' - \bar{p}^n) \right)} + \\
&+ \boxed{\Delta t \theta A(\rho, \mathbf{v}) \left(\frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond + p'' - \bar{p}^n) \right)} + \quad (55) \\
&+ \Delta t(1 - \theta) \mathbf{F}^n + \Delta t \theta \left(\mathbf{F}^\diamond - \mathbb{F}_v \left(\frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond - \bar{p}^n) + \bar{\mathbf{v}}^\diamond \right) \right) - \\
&\quad - \Delta t \theta \left[\mathbb{F}_v \frac{\Delta t \theta_p}{\rho} \nabla - \mathbf{f}_p \right] p'' - \\
&- \Delta t \mathbf{B} \bar{p}^n - \boxed{\left(\Delta t \theta_p \mathbf{B} (\bar{p}^\diamond + p'' - \bar{p}^n) - \frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond + p'' - \bar{p}^n) \right)}
\end{aligned}$$

- The second step would be to form and solve PPE. Taking divergence of eq.(53) leads to the following PPE:

$$\underbrace{\nabla \cdot \frac{1}{\rho} \nabla \lambda^\heartsuit}_{K_p \lambda^\heartsuit} = \underbrace{\nabla \cdot \bar{\mathbf{v}}^{\heartsuit,*}}_D \quad (56)$$

which reduces to eq.(26) in the OS mode.

- Pressure is computed from the new Lagrange multiplier as:

$$\bar{p}^\heartsuit = \bar{p}^n + \frac{1}{\theta_p \Delta t} \lambda^\heartsuit \quad (57)$$

- Next, we project the cell-centered velocities as

$$\bar{\mathbf{v}}^\heartsuit = \bar{\mathbf{v}}^{\heartsuit,*} - \frac{1}{\rho} \mathbf{B} \lambda^\heartsuit \quad (58)$$

5. Finally, we can compute

$$\begin{aligned}\bar{\mathbf{v}}' &= \bar{\mathbf{v}}^\heartsuit - \bar{\mathbf{v}}^\diamond \\ \bar{p}' &= \bar{p}^\heartsuit - \bar{p}^\diamond \\ \lambda' &= \lambda^\heartsuit - \lambda^\diamond\end{aligned}\quad (59)$$

and these are the values which are returned to PETSC-SNES. As mentioned above, in the OS mode, this step is absent, as $\Phi^\heartsuit = \Phi^{n+1}$.

Incremental Form. One can re-write eq.(53) as:

$$\bar{\mathbf{v}}^{\heartsuit,*} = \bar{\mathbf{v}}^\diamond + \mathbf{v}^{*'} \quad (60)$$

where

$$\mathbf{v}^{*'} = \mathbf{v}' + \frac{1}{\rho} \nabla (\lambda^\diamond + \lambda') \quad (61)$$

1. Solve for non-solenoidal velocity increment $\mathbf{v}^{*'}$.

Option-A:

$$\begin{aligned}[M - \Delta t \theta (K - A(\rho, \mathbf{v}) + \mathbb{F}_v)] \mathbf{v}^{*'} &= -[M - \Delta t \theta (K - A(\rho, \mathbf{v}) + \mathbb{F}_v)] \bar{\mathbf{v}}^\diamond + \\ &+ [M + \Delta t(1 - \theta)(K - A(\rho, \mathbf{v}))] \bar{\mathbf{v}}^n - \boxed{\Delta t \theta K \left(\frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond + p'' - \bar{p}^n) \right)} + \\ &+ \boxed{\Delta t \theta A(\rho, \mathbf{v}) \left(\frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond + p'' - \bar{p}^n) \right)} + \\ &+ \Delta t(1 - \theta) \mathbf{F}^n + \Delta t \theta \left(\mathbf{F}^\diamond - \mathbb{F}_v \left(\frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond - \bar{p}^n) + \bar{\mathbf{v}}^\diamond \right) \right) - \\ &\quad - \Delta t \theta \left[\mathbb{F}_v \frac{\Delta t \theta_p}{\rho} \nabla - \mathbf{f}_p \right] p'' - \\ &- \Delta t \mathbf{B} \bar{p}^n - \boxed{\left(\Delta t \theta_p \mathbf{B} (\bar{p}^\diamond + p'' - \bar{p}^n) - \frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond + p'' - \bar{p}^n) \right)}\end{aligned}\quad (62)$$

Option-B:

$$\begin{aligned}
[M - \Delta t \theta (K + \mathbb{F}_v)] \mathbf{v}^{*'} &= - [M - \Delta t \theta (K + \mathbb{F}_v)] \bar{\mathbf{v}}^\diamond + \\
&\quad + [M + \Delta t (1 - \theta) (K - A(\rho, \mathbf{v}))] \bar{\mathbf{v}}^n - \\
&\quad - \Delta t \theta A(\rho, \mathbf{v}) (\bar{\mathbf{v}}^\diamond + \bar{\mathbf{v}}'') - \boxed{\Delta t \theta K \left(\frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond + p'' - \bar{p}^n) \right)} + \\
&\quad + \boxed{\Delta t \theta A(\rho, \mathbf{v}) \left(\frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond + p'' - \bar{p}^n) \right)} + \tag{63} \\
&\quad + \Delta t (1 - \theta) \mathbf{F}^n + \Delta t \theta \left(\mathbf{F}^\diamond - \mathbb{F}_v \left(\frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond - \bar{p}^n) + \bar{\mathbf{v}}^\diamond \right) \right) - \\
&\quad - \Delta t \theta \left[\mathbb{F}_v \frac{\Delta t \theta_p}{\rho} \nabla - \mathbf{f}_p \right] p'' - \\
&\quad - \Delta t \mathbf{B} \bar{p}^n - \boxed{\left(\Delta t \theta_p \mathbf{B} (\bar{p}^\diamond + p'' - \bar{p}^n) - \frac{\Delta t \theta_p}{\rho} \nabla (\bar{p}^\diamond + p'' - \bar{p}^n) \right)}
\end{aligned}$$

2. Solve incremental PPE:

$$\nabla \cdot \frac{1}{\rho} \nabla \lambda' = \nabla \cdot \mathbf{v}^{*'} - \nabla \cdot \frac{1}{\rho} \nabla \lambda^\diamond \tag{64}$$

3. Convert to pressure correction as

$$\bar{p}' = \frac{\lambda'}{\Delta t \theta_p} \tag{65}$$

4. Return to PETSC-SNES a preconditioned solution as

$$\begin{bmatrix} \bar{p}' \text{ (or } \lambda') \\ \mathbf{v}' = \mathbf{v}^{*'} - \frac{1}{\rho} \nabla (\lambda^\diamond + \lambda') \end{bmatrix} \tag{66}$$

7. Concluding Remarks

The main technical contribution of the present report is the formulation of the fully-implicit projection algorithm for implementation in Hydra-TH code. We discussed definition of non-linear residual vector, as well as the strategy for efficient preconditioning of linear (GMRES) solver, utilizing the variation of the currently-available in Hydra-TH semi-implicit projection algorithm. While focusing here on single-phase flow formulation, the basic ideas of the fully-implicit projection should be straightforwardly extendable to multi-fluid flows. These extensions will be presented in future.

Acknowledgement

This work has been authored by Battelle Energy Alliance, LLC under contract No. DE-AC07-05ID14517 (INL/EXT-12-27197) with the U.S. Department of Energy. The United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manual, or allow others to do so, for United States Government purposes.

References

- [1] R.R. Nourgaliev and M. A. Christon. Solution algorithms for multi-fluid-flow averaged equations. Technical Report INL/EXT-12-27187, Idaho National Laboratory, Idaho Falls, Idaho, September 2012.
- [2] Alexandre Joel Chorin. Numerical solution of the Navier-Stokes equations. *Mathematics of Computations*, 22:745–762, 1968.
- [3] J. Van Kan. A second-order accurate pressure-correction scheme for viscous incompressible flow. *SIAM Journal for Scientific and Statistical Computing*, 7:870–891, 1986.
- [4] John B. Bell, Philip Colella, and Harland M. Glaz. A second-order projection method for the incompressible navier-stokes equations. *Journal of Computational Physics*, 85:257–283, 1989.
- [5] Philip 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*, 11:587–620, 1990.
- [6] Philip M. Gresho and Stevens 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*, 11:621–659, 1990.
- [7] Philip M. Gresho, Stevens T. Chan, Mark A. Christon, and Allen C. Hindmarsh. A little more on stabilized $q1q1$ for transient viscous incompressible flow. *International Journal for Numerical Methods in Fluids*, 21:837–856, 1995.

- [8] Philip M. Gresho and Stevens T. Chan. Projection 2 goes turbulent – and fully implicit. *preprint International Journal for Computational Fluid Dynamics*, March 1996. (LLNL UCRL-JC-123727).
- [9] Ann S. Almgren, John B. Bell, Phillip Colella, and Louis H Howell. An adaptive projection method for the incompressible euler equations. In *Eleventh AIAA Computational Fluid Dynamics Conference*, pages 530–539. AIAA, 1993.
- [10] Ann S. Almgren, John B. Bell, and William G. Szymczyk. A numerical method for the incompressible navier-stokes equations based on an approximate projection. *SIAM Journal for Scientific Computing*, 17(2):358–369, March 1996.
- [11] Ann S. Almgren, John B. Bell, and William Y. Crutchfield. Approximate projection methods: Part i. inviscid analysis. *SIAM Journal on Scientific Computing*, 22(4):1139–1159, 2000.
- [12] William J. Rider. The robust formulation of approximate projection methods for incompressible flows. Technical Report LA-UR-3015, Los Alamos National Laboratory, 1994.
- [13] William J. Rider. Filtering nonsolenoidal modes in numerical solutions of incompressible flows. Technical Report LA-UR-3014, Los Alamos National Laboratory, Los Alamos, New Mexico, September 1994.
- [14] W. J. Rider, D. B. Kothe, S. J. Mosso, J. H. Cerutti, and J. I. Hochstein. Accurate solution algorithms for incompressible multiphase flows. Technical Report AIAA-95-0699, AIAA, Reno, Nevada, January 1995.
- [15] William J. Rider. Approximate projection methods for incompressible flow: implementation, variants and robustness. Technical Report LA-UR-2000, Los Alamos National Laboratory, Los Alamos, New Mexico, July 1995.
- [16] Michael L. Minion. A projection method for locally refined grids. *Journal of Computational Physics*, 127:158–178, 1996.
- [17] J.-L. Guermond and L. Quartapelle. Calculation of incompressible viscous flow by an unconditionally stable projection fem. *Journal of Computational Physics*, 132:12–23, 1997.

- [18] Elbridge G. Puckett, Ann S. Almgren, John B. Bell, Daniel L. Marcus, and William J. Rider. A high-order projection method for tracking fluid interfaces in variable density incompressible flows. *Journal of Computational Physics*, 130:269–282, 1997.
- [19] Mark Sussman, Ann S. Almgren, John B. Bell, Phillip Colella, Louis H. Howell, and Michael L. Welcome. An adaptive level set approach for two-phase flows. *Journal of Computational Physics*, 148:81–124, 1999.
- [20] Omar M. Knio, Habib N. Najm, and Peter S. Wyckoff. A semi-implicit numerical scheme for reacting flow. ii. stiff operator-split formulation. *pre-print submitted to Journal of Computational Physics*, 1999.
- [21] David L. Brown and Michael L. Minion. Performance of under-resolved two-dimensional incompressible flow simulations. *Journal of Computational Physics*, 122:165–183, 1995.
- [22] Michael L. Minion and David L. Brown. Performance of under-resolved two-dimensional incompressible flow simulations, ii. *Journal of Computational Physics*, 138:734–765, 1997.
- [23] Brian B. Wetton. Error analysis of pressure increment schemes. *submitted to SINUM*, April 1998.
- [24] Jean-Luc Guermond. Some implementations of projection methods for navier-stokes equations. *Mathematical Modelling and Numerical Analysis*, 30(5):637–667, 1996.
- [25] Jean-Luc Guermond. A convergence result for the approximation of the navier-stokes equations by an incremental projection method. *C. R. Acad. Sci. Paris*, 325:1329–1332, 1997.
- [26] Jean-Luc Guermond and L. Quartapelle. On the approximation of the unsteady navier-stokes equations by finite element projection methods. *Numerische Mathematik*, 80:207–238, 1998.
- [27] Jean-Luc Guermond and L. Quartapelle. On stability and convergence of projection methods based on pressure poisson equation. *International Journal for Numerical Methods in Fluids*, 26:1039–1053, 1998.
- [28] M. A. Christon. Hydra-th theory manual. Technical Report LA-UR-11-05387, Los Alamos National Laboratory, Los Alamos, New Mexico, September 2011.

- [29] Satish Balay, Kris Buschelman, Victor Eijkhout, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 2.1.5, Argonne National Laboratory, 2004.
- [30] D. A. Knoll and D. Keyes. Jacobian-free Newton-Krylov methods: A survey of approaches and applications. *Journal of Computational Physics*, 193:357–397, 2004.
- [31] D. A. Knoll and W. J. Rider. A multigrid preconditioned Newton-Krylov method. *SIAM Journal of Scientific Computing*, 21:691–710, 2000.
- [32] D.A. Knoll, L. Chacon, L.G. Margolin, and V.A. Mousseau. On balanced approximations for time integration of multiple time scales systems. *Journal of Computational Physics*, 185:583–611, 2003.
- [33] D.A. Knoll, P. R. McHugh, and D. E. Keyes. Newton-Krylov methods for low-Mach-number compressible combustion. *AIAA Journal*, 34(5):961, 1996.
- [34] D.A. Knoll, V.A. Mousseau, L. Chacon, and J. M. Reisner. Jacobian-free Newton-Krylov methods for the accurate time integration of stiff wave systems. *SIAM Journal of Scientific Computing*, 25:213–230, 2005.
- [35] Y. Saad and M.H. Schultz. GMRES: a generalized minimal residual algorithm for solving linear systems. *SIAM Journal of Science and Statistical Computing*, 7:856, 1986.