On Fractional-Step Schemes for Incompressible Inviscid Flow

Derek T. Steinmoeller*

I. Introduction

The divergence-free constraint in incompressible flow

$$\nabla \cdot \mathbf{u} = 0 \,, \tag{1}$$

has weak form

$$\int_\Omega q^h \nabla \cdot \mathbf{u} \, d\Omega = 0 \,. \tag{2}$$

where q^h lies in the space of test functions. In the standard finite element method, the test functions lie in the same space as the basis functions of the unknown flow variable. Due to the well-known LBB stability constraint that says that the velocity and pressure variables do not, in general, lie within the same space, it is often argued that mixed-order spatial discretizations for the velocity and pressure fields (or stable element pairs) are required to recover a numerically stable scheme. For the sake of simplicity, it is desirable to have a stable equal-order numerical scheme. Is this possible?

II. Methods

To derive a stable equal-order method, we will apply the Pressure-Stabilized Petrov-Galerkin method [1] that replaces the weak form (2) with the stabilized (or penalized form)

$$\int_\Omega q^h \nabla \cdot \mathbf{u} \, d\Omega + \sum_{e=1}^{N_e} \int_{\Omega_e} \tau \nabla q^h \cdot \mathbf{R} = 0 \,. \tag{3}$$

Here,

$$\mathbf{R} = \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \nabla p \,, \tag{4}$$

is the residual of the momentum equation, and N_e is the number of elements used in the spatial discretization. If we assume that $\tau \geq 0$ is constant throughout the domain, then the summation can be replaced by a single integral and we can integrate by parts:

$$\int_\Omega q^h \nabla \cdot \mathbf{u} d\Omega + \tau \left[\int_{\partial\Omega} q^h \mathbf{R} \cdot \hat{\mathbf{n}} - \int_\Omega q^h \nabla \cdot \mathbf{r} \right] = 0 \,. \tag{5}$$

Assuming a closed or periodic boundary, our boundary integral vanishes. If we then apply the standard Galerkin approach where the flow variables are expanded in terms of basis functions and the test function q^h lies in the space of basis functions, we find the discrete formulation

$$M_{ij} D_{ij} \mathbf{u} - \tau M_{ij} D_{ij} \mathbf{R} = 0 \,. \tag{6}$$

Here D_{ij} is the discrete divergence operator, M_{ij} is the global mass matrix that is invertible by construction.

Left-multiplying this expression by the inverse mass matrix M_{ij}^{-1} and returning to continuous formulation, at any particular discrete time value (say $t_{n+1} = (n+1)\Delta t$) we have

$$\nabla \cdot \mathbf{u}^{n+1} - \tau \nabla \cdot \left(\frac{\partial \mathbf{u}}{\partial t}^{n+1} + [(\mathbf{u} \cdot \nabla)\mathbf{u}]^{n+1} + \nabla p^{n+1} \right) = 0 \,, \tag{7}$$

a penalized form of the divergence-free constraint

$$\nabla \cdot \mathbf{u}^{n+1} = 0 \,. \tag{8}$$

III. Time-Stepping Strategy

We notice that equation (7) when combined with the momentum equation

$$\frac{\partial \mathbf{u}}{\partial t}^{n+1} + [(\mathbf{u} \cdot \nabla)\mathbf{u}]^{n+1} = -\nabla p^{n+1} \tag{9}$$

is strongly coupled in time and one might be tempted to use a strongly-implicit time-stepper to solve for the flow variables at time-level t_{n+1}. However, it should be noticed that if we "lag" the stabilization term to $t_{n+\frac{1}{2}} = (n+\frac{1}{2})\Delta t$ and apply the second-order accurate approximation

$$\frac{\partial \mathbf{u}}{\partial t} = \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} + O(\Delta t^2) \,, \tag{10}$$

we find

$$\nabla \cdot \mathbf{u}^{n+1} - \tau \nabla \cdot \left(\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} + [(\mathbf{u} \cdot \nabla)\mathbf{u}]^{n+\frac{1}{2}} + \nabla p^{n+\frac{1}{2}} \right) = 0 \,. \tag{11}$$

Now, if we make the choice of $\tau = \Delta t$ we have

$$\nabla \cdot \left(\mathbf{u}^n - \Delta t \, [(\mathbf{u} \cdot \nabla)\mathbf{u}]^{n+\frac{1}{2}} - \Delta t \nabla p^{n+\frac{1}{2}} \right) = 0 \,. \tag{12}$$

For later convenience, we define

$$\mathbf{u}^* = \mathbf{u}^n - \Delta t \, [(\mathbf{u} \cdot \nabla)\mathbf{u}]^{n+\frac{1}{2}} - \Delta t \nabla p^{n+\frac{1}{2}} \,. \tag{13}$$

*Thanks to: Marek Stastna, Kevin Lamb, Sumedh Joshi

Inspecting the definition (13) and the pressure-stabilized statement of conservation of mass

$$\nabla \cdot \mathbf{u}^* = 0\,, \tag{14}$$

that is equivalent to (12), the requirements of our time-stepping scheme become clear. We require:

1. An approximation of $\mathbf{u}^*$ via extrapolation from previous time-levels.
2. An accurate computation of $\mathbf{u}^{n+1} = \mathbb{P}(\mathbf{u}^*)$: the enforcement of incompressibility via the projection of $\mathbf{u}^*$ onto the null space of the divergence operator.

The code for an Octave implementation of a fractional-step time-splitting scheme is given in `Code Listing 1`. This technique is based on the approach discussed by Bell and Marcus [2]. Since the calculation of the advective/source-term right-hand side of the equations and the advancement of pressure is dependent on the spatial-discretization used, they are abstracted away into separate function calls labelled `EulerRHS2D` and `pressproj_hnd`. Their implementation details are left out as an exercise for the reader. Similarly, `Filt` represents a filtering matrix with a construction that depends upon the spatial-discretization method being used.

```
%% Octave code listing for 2nd-order fraction step Incompressible Euler equations.
%% Here, Qn = (rho,u,w)^n represents the vector of flow variables at time-level tn=n*dt.
%% where dt is the time-step, and n denotes the time-level.

while (time(n) < FINAL_TIME)

   % Get advective RHS, given the current time, and the boundary condition
   % and the acceleration due to gravity g=9.81 m/s^2.
   rhsQn = EulerRHS2D(Qn,time,BC,g);

   % Advective step (predictor step).
   Qnp1_estimate = Qn + dt*rhsQn - dt*gradp_nmh;

   % Spatially filter velocity:
   for n=2:3
       Qnp1_estimate(:,:,n) = Filt*Qnp1_estimate(:,:,n);
   end

   % Corrector stage, this can be repeated more than once if desired.
   for jj=1:NUM_CORRECTIONS
       Qnph = 0.5*(Qnp1_estimate + Qn);

       % Get estimate for convective/source terms at half step.
       rhsQnph = EulerRHS2D(Qnph,time,BC,g);

       % Extrapolate to get corrected estimate for grad p at n+1/2 step.
       gradp_nph = rhsQnph - rhsQn + gradp_nmh;

       % Advective step (corrector).
       Qnp1_estimate = Qn + dt*rhsQnph - dt*gradp_nph;
   end

   % Filter updated velocity:
   for n=2:3
       Qnp1_estimate(:,:,n) = Filt*Qnp1_estimate(:,:,n);
   end

   ustar = Qnp1_estimate(:,:,2);
   wstar = Qnp1_estimate(:,:,3);
```

```
    % Do pressure projection.
    RHS = [ustar(:); wstar(:)];

    % Pressure projection is carried out given the predicted velocity, the time-step, and
    % the LU-factors of the Laplacian with Neumann boundary conditions on walls.
    % Here, P & Q are pivot and column-reordering matrices such that P*A*Q = L*U.

    result = pressproj_hnd(RHS,dt,L,U,P,Q);

    % Retrieve velocity from stacked result.
    % p (pressure) can be retrieved as a diagnostic tool, but only the gradient
    % of pressure is required to be updated for the time-stepping algorithm.

    u = reshape(result(1:end/3),Np,K);
    w = reshape(result(end/3+1:2*end/3),Np,K);
    p = reshape(result(2*end/3+1:end),Np,K);

    % Update flow variables in packed form.
    Qnp1(:,:,1) = Qnp1_estimate(:,:,1);
    Qnp1(:,:,2) = u;
    Qnp1(:,:,3) = w;

    Qn = Qnp1;
    gradp_nmh = gradp_nph;

    time(n) = time(n) + dt;
    n = n + 1;
end  % End while loop.
```

Listing 1. Octave code illustrating the second-order fractional-step method for solving the Incompressible Euler equations numerically.

IV. Discussion

The issue of enforcing incompressibility for the Euler equations in density-driven flow whilst retaining numerical stability and accuracy can be a troublesome one (see [1] for a discussion). The second-order accurate method of Bell and Marcus [2] offers a method that works quite well for inviscid flow as can be seen in the work therein. Current work by Joshi *et al.* [3] looks to extend these ideas into three-dimensions by weakly enforcing C^0 and C^1 continuity at the same time as enforcing the incompressibility constraint in a weak way. The reader is encouraged to learn more through their own explorations with their numerical code and by reading the paper by Joshi *et al.* [3] – A pre-print is available on `http://arXiv.org`.

V. References

1. T.-P. Fries, H.G. Matthies, *A Review of Petrov–Galerkin Stabilization Approaches and an Extension to Meshfree Methods*, Technical University Braunschweig, **2004-01** (2004).

2. J.B. Bell, D.L. Marcus, *A Second–Order Projection Method for Variable–Density Flows*, Journal of Computational Physics, **101**, 334–348 (1992).

3. S.M. Joshi, P.J. Diamessis, D.T. Steinmoeller, M. Stastna, G.N. Thomsen, *A post-processing technique for stabilizing the discontinuous pressure projection operator in marginally-resolved incompressible inviscid flow*, Submitted to Computers and Fluids (2016).

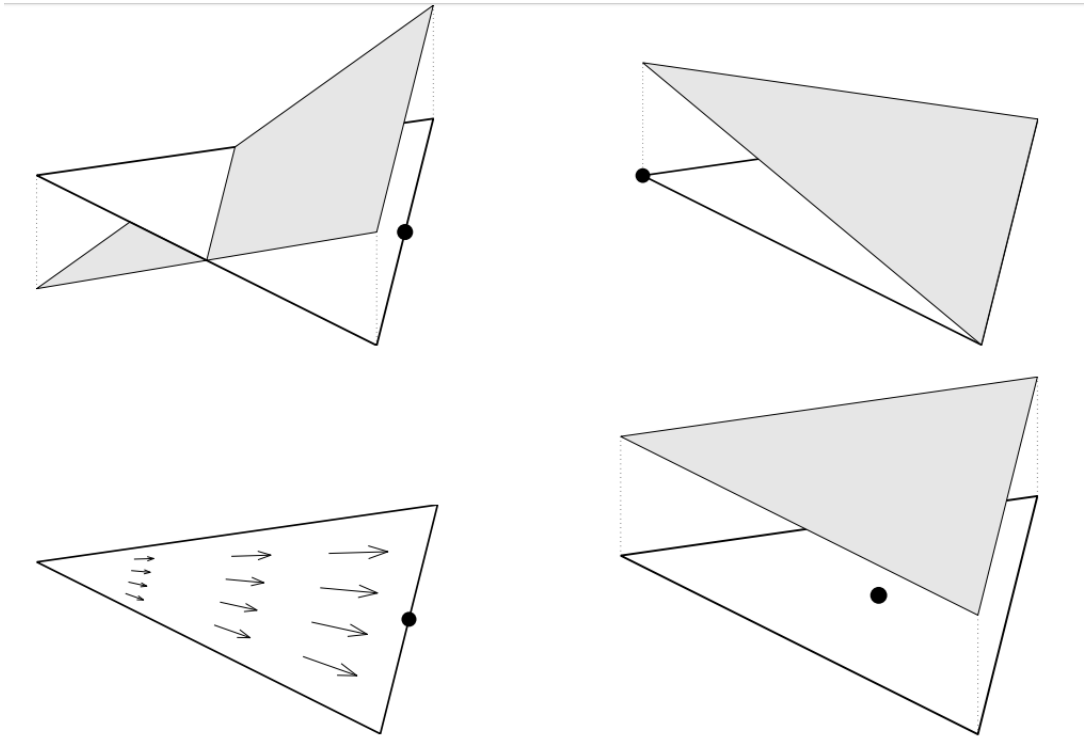

Figure 1: *Cartoon diagram comparing $P_1^{NC} - P_1$ (top) and RT_0 (bottom) element pairs. The velocity shape functions are shown on the left and the elevation/pressure shape functions are shown on the right. The node associated to each of them is represented by "•". Velocity shape functions are scalar for $P_1^{NC} - P_1$ and vectorial for RT_0. Image obtained from Hanert* et. al. *(2008)* A tale of two elements: $P_1^{NC} - P_1$ and RT_0.

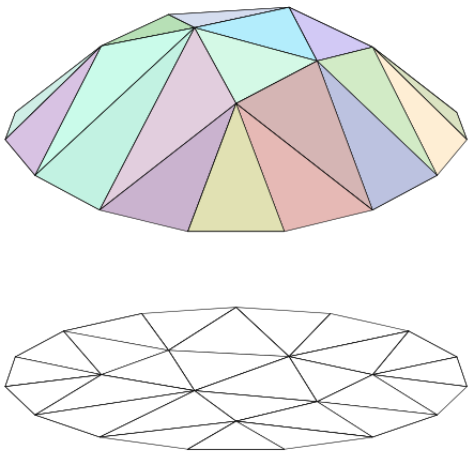

Figure 2: *A piecewise linear function in two dimensions. Image obtained from* `https://en.wikipedia.org/wiki/Finite_element_method`