Notes on the PF mini-app within the SSNS web app

Ted Malliaris

September 7, 2025

The typical fluid system — on which the Rigid Planar Flow (PF) mechanical system is based — is discussed, e.g., in §9.2 of Kundu, Fluid Mechanics, 6th ed.[1] There, the fluid flow is incompressible, viscous, laminar, steady, and unidirectional. Our boundaries are the familiar pair of infinite parallel plates with fixed separation h, but we allow both the top and bottom boundary plates to move (rather than just the top one) and denote their respective fixed velocities by U_{top} and U_{bottom}. For this simple setup, it is straightforward to solve the fluid equations exactly, and one obtains the steady-state velocity profile with a familiar parabolic functional form:

$$u(y) = U_{\text{bottom}} + \frac{U_{\text{top}} - U_{\text{bottom}}}{h} y - \frac{\mathrm{d}p/\mathrm{d}x}{2\mu} y(h-y).$$

Like all SSNS mini-apps, the PF mini-app offers the ability to explore a wide variety of time-dependent behavior. Beyond that, however, PF aims to reproduce that parabolic steady-state velocity profile, but with a purely mechanical system. Thus, we replace the layers of fluid with a stack of N infinite rigid slabs, each $\ell \times w \times h/N$. Laminar flow is thus built in, and, in place of shear stress $\tau = \mu\, \partial u/\partial y$, we have a frictional force between neighboring slabs that is linear in their relative velocity Δv. [2] The pressure gradient $\mathrm{d}p/\mathrm{d}x$ — which turns out to be constant in the fluid problem — is replaced by an external pressure difference per length $\Delta p/\ell$ applied across the upstream/downstream faces.

We have the following expressions for the mass, pressure force, and frictional force — all per unit length:

$$\begin{aligned}
\frac{m}{\ell} &= \frac{\ell w h}{N}\rho\frac{1}{\ell} = \frac{wh\rho}{N} \\
\frac{F_p}{\ell} &= \frac{wh}{N}(-\Delta p)\frac{1}{\ell} = -\frac{wh\Delta p}{N\ell} \\
\frac{F_\mu}{\ell} &= A\tau\frac{1}{\ell} = w\ell\mu\frac{\partial u}{\partial y}\frac{1}{\ell} = w\mu\frac{\partial u}{\partial y} \longrightarrow \frac{w\mu\Delta v}{h/N} = \frac{w\mu N\Delta v}{h}
\end{aligned}$$

Combining them, we have the net-force-per-length and acceleration of the ith slab:

$$\begin{aligned}
\frac{\Sigma F_i}{\ell} &= \frac{F_{\mu,i+}}{\ell} + \frac{F_{\mu,i-}}{\ell} + \frac{F_{p,i}}{\ell} = \frac{w\mu N}{h}(v_{i+1} - v_i) + \frac{w\mu N}{h}(v_{i-1} - v_i) - \frac{wh\Delta p}{N\ell} \\
a_i &= \frac{\Sigma F_i}{m_i} = \frac{\Sigma F_i}{m} = \frac{\Sigma F_i/\ell}{m/\ell} = \frac{N}{wh\rho}\left[\frac{w\mu N}{h}(v_{i+1} - v_i) + \frac{w\mu N}{h}(v_{i-1} - v_i) - \frac{wh\Delta p}{N\ell}\right] \\
&= \frac{\mu N^2}{\rho h^2}\left(v_{i+1} + v_{i-1} - 2v_i\right) - \frac{\Delta p}{\rho\ell} \\
&= \frac{\mu N^2}{\rho h^2}\left[\left(v_{i+1} + v_{i-1} - 2v_i\right) - \frac{h^2}{\mu N^2}\frac{\Delta p}{\ell}\right] \\
&= \frac{\mu N^2}{\rho h^2}\left[v_{i+1} + v_{i-1} - 2v_i - \alpha\right],
\end{aligned}$$

[1] doi.org/10.1016/C2012-0-00611-4

[2] We will reuse μ as the constant of proportionality, but keep in mind that it is not a true viscosity. Also, while we keep the fluid system's stream-wise $\hat{x}$ and cross-stream $\hat{y}$ axes, we prefer to use v for the $\hat{x}$-velocities rather than u.

where $\alpha \equiv \frac{h^2}{\mu N^2}\frac{\Delta p}{\ell}$ is a useful combination of parameters that has dimensions of velocity and quantifies the balance between the pressure-derived normal stresses and friction-derived shear stresses.

So we have a system of linear equations for the slab accelerations a_i in terms of the slab velocities v_i. It is convenient to incorporate the boundary velocities into the slab velocity vector as "ghost cells" $v_0 = U_{\text{top}}$ and $v_{N+1} = U_{\text{bottom}}$, respectively. In both paper expressions and computer arrays, the N slabs will be indexed from $i = 1$ to $i = N$. For the time evolution of the system, we use numerical integration as described below, with the length $N+2$ vector $\boldsymbol{v}$ being used to calculate the middle N entries of the length $N+2$ acceleration vector $\boldsymbol{a}$. The end entries a_0 and a_{N+1} are set equal to zero, which makes sense since the boundary velocities are fixed. For the time-independent steady state (also described below), we find it easier to deal with length N vectors and matrices, and will fold U_{top} and U_{bottom} into the 1st and Nth entries of a vector $\boldsymbol{\alpha}$.

For numerical integration, we've implemented both forward Euler and Runge-Kutta (RK4) schemes. Since accuracy in the approach-to-steady-state trajectory is not terribly important, there's no compelling reason to use RK4, so Euler is the default choice [3]. The forward Euler scheme to update the velocity of the ith slab is:

$$a_i = \frac{\mathrm{d}v_i}{\mathrm{d}t} \cong \frac{\Delta v_i}{\Delta t} \implies v_{i,\text{new}} \cong v_i + a_i \Delta t,$$

where a_i is given by the expression above [4]. Of course, numerical stability is an issue and dictates our choice for the size of the time step Δt. We employ a simple heuristic to determine the default value: $\Delta t_{\text{def.}} = 0.1/(\mu N^2)$. The Δt entry field is populated with this value on app load, but that's the extent of the guidance — subsequent changes to Δt or parameter values may result in instability or needlessly slow approach to steady state.

Intuitively, we sense that — for fixed parameter values — the system will eventually reach a state of force balance where each slab has a fixed (but generally nonzero) velocity. This steady state (or steady flow) can be found analytically by setting $a_i = 0$:

$$\begin{aligned} 0 = a_i &= \frac{\mu N^2}{\rho h^2}\left[v_{i+1} + v_{i-1} - 2v_i - \alpha\right], && 1 \le i \le N \\ \implies v_{i+1} + v_{i-1} - 2v_i &= \alpha, && 1 \le i \le N \end{aligned}$$

As mentioned above, the equations for $a_{i=1}$ and $a_{i=N}$ involve $v_0 = U_{\text{top}}$ and $v_{N+1} = U_{\text{bottom}}$, respectively. However, length N vectors/matrices will suffice in this case, rather than length $N+2$. Let $\boldsymbol{v_s}$ be the length N state vector we seek, and let A be the $N \times N$ tridiagonal matrix with elements $A_{ij} = \delta_{i,i+1} + \delta_{i,i-1} - 2\delta_{ii}$. The boundary slab speeds can be brought to the opposite side and incorporated into a length N vector $\boldsymbol{\alpha}$, giving us the following vector equation to solve:

$$\boldsymbol{A v_s} = \boldsymbol{\alpha} \equiv \begin{pmatrix} \alpha - U_{\text{top}} \\ \alpha \\ \alpha \\ \vdots \\ \alpha \\ \alpha - U_{\text{bot.}} \end{pmatrix}$$

The inverse matrix A^{-1} can be found by standard techniques. It is symmetric (but not tridiagonal) and — on the diagonal and upper right — has elements:

$$(A^{-1})_{ij} = \frac{-i(N-j+1)}{N+1}, \qquad 1 \le i \le j \le N.$$

[3] The setting is easy to change, just not through the UI.

[4] In the `PF` implementation we again use forward Euler to get position information by integrating $v_i = \mathrm{d}x_i/\mathrm{d}t$, but this is only for the qualitative visualization of the moving slabs — the general focus is on the slab velocities.

The lower left elements are given by $(A^{-1})_{ij} = (A^{-1})_{ji}$, and the steady state is found as $\boldsymbol{v}_s = A^{-1}A\boldsymbol{v}_s = A^{-1}\boldsymbol{\alpha}$. Writing out the expression for the ith element $v_{s,i}$, we have:

$$
\begin{aligned}
v_{s,i} &= \sum_{j=1}^{N}(A^{-1})_{ij}\alpha_j \\
&= \sum_{j=1}^{N}(A^{-1})_{ij}\left[\alpha - U_{\text{top}}\delta_{j,1} - U_{\text{bot.}}\delta_{j,N}\right] \\
&= \alpha\sum_{j=1}^{N}(A^{-1})_{ij} - (A^{-1})_{i1}U_{\text{top}} - (A^{-1})_{iN}U_{\text{bot.}} \\
&= \alpha\sum_{j=1}^{i-1}(A^{-1})_{ji} + \alpha\sum_{j=i}^{N}(A^{-1})_{ij} - (A^{-1})_{1i}U_{\text{top}} - (A^{-1})_{iN}U_{\text{bot.}} \\
&= -\alpha\sum_{j=1}^{i-1}\frac{j(N-i+1)}{N+1} - \alpha\sum_{j=i}^{N}\frac{i(N-j+1)}{N+1} + \frac{1(N-i+1)}{N+1}U_{\text{top}} + \frac{i(N-N+1)}{N+1}U_{\text{bot.}} \\
&= \frac{1}{N+1}\left[(N-i+1)U_{\text{top}} + iU_{\text{bot.}} - \alpha(N-i+1)\sum_{j=1}^{i-1}j - \alpha i\sum_{j=i}^{N}(N-j+1)\right] \\
&= \frac{1}{N+1}\left[(N-i+1)U_{\text{top}} + iU_{\text{bot.}} - \alpha(N-i+1)\frac{i(i-1)}{2} - \alpha i\sum_{j=1}^{N-i+1}(N-(j+i-1)+1)\right] \\
&= \frac{1}{N+1}\left[(N-i+1)U_{\text{top}} + iU_{\text{bot.}} - \alpha(N-i+1)\frac{i(i-1)}{2} - \alpha i\sum_{j=1}^{N-i+1}(N-i+2-j)\right] \\
&= \frac{1}{N+1}\left[(N-i+1)U_{\text{top}} + iU_{\text{bot.}} - \alpha(N-i+1)\frac{i(i-1)}{2} - \alpha i\left\{(N-i+1)(N-i+2) - \frac{(N-i+1)(N-i+2)}{2}\right\}\right] \\
&= \frac{1}{N+1}\left[(N-i+1)U_{\text{top}} + iU_{\text{bot.}} - \alpha(N-i+1)\frac{i(i-1)}{2} - \alpha i\frac{(N-i+1)(N-i+2)}{2}\right] \\
&= \frac{1}{N+1}\left[(N-i+1)U_{\text{top}} + iU_{\text{bot.}} - \frac{\alpha i(N-i+1)}{2}\left\{i-1+N-i+2\right\}\right] \\
&= \frac{1}{N+1}\left[(N-i+1)U_{\text{top}} + iU_{\text{bot.}} - \frac{\alpha i(N-i+1)}{2}(N+1)\right] \\
&= \frac{1}{N+1}\left[(N-i+1)U_{\text{top}} + iU_{\text{bot.}} - \frac{\alpha i(N-i+1)}{2}(N+1)\right] \\
&= \frac{(N-i+1)U_{\text{top}} + iU_{\text{bot.}}}{N+1} - \frac{\alpha i(N-i+1)}{2}
\end{aligned}
$$

This expression can be extended back to length $N+2$ to reproduce the boundary conditions $v_{s,i=0} = U_{\text{top}}$ and $v_{s,i=N+1} = U_{\text{bottom}}$. Further, if we use the definition of α above and make the substitutions [5]:

$$
\frac{i}{N} \longrightarrow 1 - \frac{y}{h} \qquad \text{or} \qquad \frac{N-i}{N} \longrightarrow \frac{y}{h} \qquad \text{and} \qquad \frac{\Delta p}{\ell} \longrightarrow \frac{\mathrm{d}p}{\mathrm{d}x},
$$

while taking $N \to \infty$, we recover our original parabolic velocity profile:

[5] U_{top} corresponds to $i = 0$, but $y = h$, so we must reflect either the discrete or continuous coordinate to get the other.

$$
\begin{aligned}
\lim_{N\to\infty} v_{s,i} &= \lim_{N\to\infty}\left[\frac{(N-i+1)U_{\text{top}} + iU_{\text{bottom}}}{N+1} - \frac{\alpha i(N-i+1)}{2}\right] \\
&= \lim_{N\to\infty}\left[\frac{(N-i)U_{\text{top}} + iU_{\text{bottom}}}{N} - \frac{\alpha i(N-i)}{2}\right] \\
&= \frac{y}{h}U_{\text{top}} + \left(1-\frac{y}{h}\right)U_{\text{bottom}} - \frac{1}{2}\lim_{N\to\infty}\frac{h^2\Delta p/\ell}{\mu N^2} i(N-i) \\
&= U_{\text{bottom}} + \frac{U_{\text{top}} - U_{\text{bottom}}}{h} y - \frac{\mathrm{d}p/\mathrm{d}x}{2\mu}\lim_{N\to\infty} h^2\frac{i}{N}\frac{N-i}{N} \\
&= U_{\text{bottom}} + \frac{U_{\text{top}} - U_{\text{bottom}}}{h} y - \frac{\mathrm{d}p/\mathrm{d}x}{2\mu} y(h-y) \\
&= u(y)
\end{aligned}
$$