Supplementary Materials

Reduced modeling and global instability of finite-Reynolds-number flow in compliant rectangular channels

Xiaojia Wang, Ivan C. Christov

School of Mechanical Engineering, Purdue University, West Lafayette, Indiana, 47907, USA

SM.1 Numerical scheme for the 1D FSI model

Here, we introduce a numerical scheme used for solving the coupled problem of deformation-induced tension, wall deformation, and flow. This scheme is also applied to the simpler case of constant tension with given θ_t. The spatial domain is discretized using the pseudospectral method (Boyd, 2000), with which the governing equations are satisfied at preassigned Gauss-type-quadrature nodes. In our case, the Gauss–Lobatto points are chosen. Therefore, the method is also referred to as "Chebyshev pseudospectral method" or "Chebyshev collocation method." Note that in some literature (Shen et al., 2011), the pseudospectral method is specifically referred to as a Galerkin-type method with the numerical integration using a Gauss-type quadrature, which is not the case for the present method.

The Gauss–Lobatto points

$$\tilde{Z}_j = -\cos\left(\frac{j\pi}{N}\right), \quad j = 0, 1, \ldots, N \tag{SM.1.1}$$

are defined on the domain $[-1, 1]$. (Note that $\tilde{Z}_0 = -1$ and $\tilde{Z}_N = 1$.) Consequently, we use a change of variables, $\tilde{Z} = 2Z - 1$, to transform the computational domain from $\{Z \,|\, Z \in [0, 1]\}$ to $\{\tilde{Z} \,|\, \tilde{Z} \in [-1, 1]\}$. Then, $\mathrm{d}^m/\mathrm{d}Z^m = 2^m \mathrm{d}^m/\mathrm{d}\tilde{Z}^m$. There are two major advantages of choosing the Gauss–Lobatto points as the collocation points. First, the Gauss–Lobatto points are nonuniformly distributed and clustered near the endpoints $\tilde{Z} = \pm 1$, with the spacing scaling as $\mathcal{O}(N^{-2})$, which helps resolve the deformation boundary layer near the channel outlet. Second, it is convenient to compute derivatives at the Gauss–Lobatto points. Essentially, the Chebyshev pseudospectral method finds a high-order polynomial-based, valid in the whole domain, to approximate the actual solution. As long as the functional values are known at the $N + 1$ collocated points, an N-order polynomial can be uniquely determined.

The Lagrange basis, $l_j(Z)$ $(j = 0, 1, \ldots, N)$, is a convenient choice for the interpolating polynomial since the coefficients are just the functional values. Here, l_j denotes the Lagrange polynomial which takes the value of 1 at $\tilde{Z} = Z_j$ while being 0 for Z_k with $k \neq j$. Importantly, the derivatives of the Lagrange basis at the Gauss–Lobatto points are known analytically. For example, taking $Q_k \approx Q|_{\tilde{Z}=\tilde{Z}_k}$, $k = 0, 1, \ldots, N$, then $\mathrm{d}Q/\mathrm{d}\tilde{Z}|_{\tilde{Z}=\tilde{Z}_k} \approx \sum_{j=0}^{N} D_{kj}^{(1)} Q_j$. Here, the components of the first-order differentiation matrix $\boldsymbol{D}^{(1)}$ are:

$$D_{kj}^{(1)} = \left.\frac{\mathrm{d}l_j}{\mathrm{d}\tilde{Z}}\right|_{\tilde{Z}=\tilde{Z}_k} = \begin{cases} -\dfrac{2N^2+1}{6}, & k = j = 0, \\ \dfrac{\widetilde{c}_k}{\widetilde{c}_j}\dfrac{(-1)^{k+j}}{\tilde{Z}_k - \tilde{Z}_j}, & k \neq j, \;\; 0 \leq k, j \leq N, \\ -\dfrac{\tilde{Z}_k}{2(1-\tilde{Z}_k^2)}, & k = j, \;\; 1 \leq k, j \leq N-1, \\ \dfrac{2N^2+1}{6}, & k = j = N, \end{cases} \tag{SM.1.2}$$

where $\widetilde{c}_0 = \widetilde{c}_N = 2$ and $\widetilde{c}_j = 1$ for $1 \le j \le N-1$. The higher-order differentiation matrix is just the matrix multiplication of the lower ones, *i.e.*, $\mathrm{d}^m/\mathrm{d}\tilde{Z}^m|_{\tilde{Z}=\tilde{Z}_k} \approx \boldsymbol{D}^{(m)} = [\boldsymbol{D}^{(1)}]^m = \boldsymbol{D}^{(1)} \times \boldsymbol{D}^{(1)} \times \cdots \times \boldsymbol{D}^{(1)}$.

As for the time integration, a second-order backward-difference formula is used for the flow equations. Let the time step be ΔT, and a subscript denote the functional value at the corresponding grid point, while a superscript indicates the time step. Then, equations (5.3a) and (5.3b) are discretized as:

$$Q_j^{n+1} = 1 - \int_{-1}^{\tilde{Z}_j} \frac{1}{2}\beta \left(\dot{\bar{U}}_Y\right)^{n+1} \mathrm{d}\tilde{Z}, \tag{SM.1.3}$$

$$P_j^{n+1} = \int_1^{\tilde{Z}_j} \frac{1}{2} \left\{ -\frac{\hat{Re}}{\bar{H}^{n+1}} \frac{3Q^{n+1} - 4Q^n + Q^{n-1}}{2\Delta T} - \frac{6\hat{Re}}{5\bar{H}^{n+1}} 2D^{(1)} \frac{(Q^{n+1})^2}{\bar{H}^{n+1}} - \frac{12Q^{n+1}}{(\bar{H}^{n+1})^3} \right\} \mathrm{d}\tilde{Z}. \tag{SM.1.4}$$

Both of these equations have been integrated in space, with $Q_0 \equiv Q|_{Z=0} = 1$ imposed in equation (SM.1.3) and $P_N \equiv P|_{Z=1} = 0$ imposed in equation (SM.1.4). The integral is to be evaluated numerically using the trapezoidal rule. Note $\mathrm{d}\tilde{Z} = 2\mathrm{d}Z$ due to the change of variables introduced above. Also, in equation (SM.1.3), $\dot{\bar{U}}_Y$ denotes the velocity of the interface, which can be obtained from equation (SM.1.7b) below.

The so-called Newmark–β method is applied to the governing solid equation (4.9). Then, the spatially discretized equation (4.9) is written as

$$\boldsymbol{M}\ddot{\bar{U}}_Y + \boldsymbol{K}\bar{U}_Y = P, \tag{SM.1.5}$$

with

$$\boldsymbol{M} = \theta_I \boldsymbol{I}, \quad \boldsymbol{K} = \boldsymbol{I} - \theta_t \boldsymbol{D}^{(2)}. \tag{SM.1.6}$$

Here, θ_t is evaluated from equation (4.13). To ensure the accuracy of the numerical integration required to evaluate θ_t, the kernel, $(\mathrm{d}\bar{H}/\mathrm{d}Z)^2 \approx (2\boldsymbol{D}^{(1)}\bar{H})^2$, is interpolated on the finer grid of $N = 1024$ using the `barycentric_interpolate` subroutine in SciPy. Then, a Gauss–Lobatto quadrature is applied on the finer grids to calculate the integral in equation (4.13).

With the coefficients in equation (SM.1.5) determined, the acceleration, velocity and displacement of the interface are calculated as

$$\ddot{\bar{U}}_Y^{n+1} = \left(\boldsymbol{M} + \phi_2 \Delta T^2 \boldsymbol{K}\right)^{-1} \left\{ P^{n+1} - \boldsymbol{K} \left[\bar{U}_Y^n + \Delta T \dot{\bar{U}}_Y^n + \left(\frac{1}{2} - \phi_2\right) \Delta T^2 \ddot{\bar{U}}_Y^n \right] \right\}, \tag{SM.1.7a}$$

$$\dot{\bar{U}}_Y^{n+1} = \dot{\bar{U}}_Y^n + (1-\phi_1)\Delta T \ddot{\bar{U}}_Y^n + \phi_1 \Delta T \ddot{\bar{U}}_Y^{n+1}, \tag{SM.1.7b}$$

$$\bar{U}_Y^{n+1} = \bar{U}_Y^n + \Delta T \dot{\bar{U}}_Y^n + \left(\frac{1}{2} - \phi_2\right) \Delta T^2 \ddot{\bar{U}}_Y^n + \phi_2 \Delta T^2 \ddot{\bar{U}}_Y^{n+1}, \tag{SM.1.7c}$$

where ϕ_1 and ϕ_2 are two adjustable parameters. The Newmark–β scheme is unconditionally stable and second-order accurate if $\phi_1 = 1/2$ and $\phi_2 = 1/4$. However, to damp out numerically-induced high-frequency oscillations, $\phi_1 > 1/2$ is usually needed (Subbaraj and Dokainish, 1989). In our simulations, we use $\phi_1 = 1.0$ and $\phi_2 = 0.5625$.

Finally, the discretized interface equation (4.7) is simply

$$\bar{H}_j^{n+1} = 1 + \beta(\bar{U}_Y)_j^{n+1}. \tag{SM.1.8}$$

SM.1.1 Steady-state simulation

As mentioned in the main text, the case with given constant tension can be easily solved using ScPpy's `solve_bvp`. However, in the case of the deformation-dependent tension, since θ_t is unknown, `solve_bvp` is not as robust and usually has difficulty reaching convergence. Instead, SciPy's `newton_krylov` method is applied to resolve the steady-state solution.

At steady state, all of the terms involving ΔT can be neglected. Further, we drop the subscripts on the spatial discretizations for convenience. Then, equations (SM.1.3), (SM.1.4), (SM.1.5) and (SM.1.8) comprise a nonlinear algebraic problem. Then, given $\bar{U}_Y$, equations (SM.1.8) and (SM.1.4) allow us to evaluate the

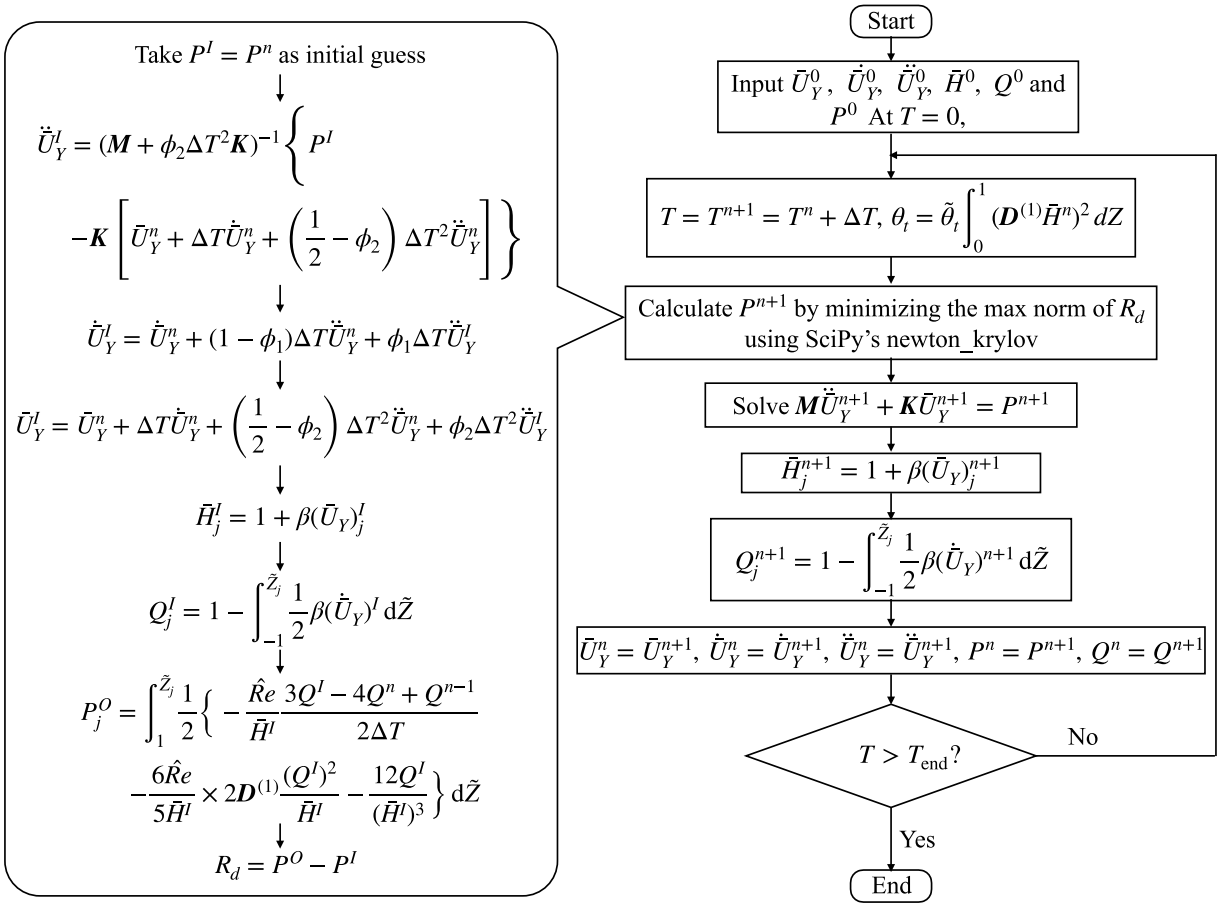

Figure SM.1: Flow chart of the numerical scheme for the dynamic simulations. To the left of the cell where SciPy's `newton_krylov` solver is called, the details of the construction of the residual, R_d, is shown. The superscript I indicates that the quantity is calculated based on P^I.

pressure, denoted as P^F. At the same time, equation (SM.1.5) can also be used to evaluate the pressure, denote as P^S. Now, we define a residual as

$$R_s = P^F - P^S. \tag{SM.1.9}$$

SciPy's `newton_krylov` solver is used to minimize the max-norm of R_s, which yields an approximate evaluation for $\bar{U}_Y$ at steady state. The tolerance used was 6×10^{-6}.

With $\bar{U}_Y$ obtained, θ_t is calculated from equation (4.13) using the Gauss–Lobatto quadrature, as discussed before. The steady-state solution is then validated with `solve_bvp` by solving equation (6.1), where both the initial guess and θ_t are based on the outputs of `newton_krylov`.

SM.1.2 Dynamic simulation

The dynamic problem is solved in a similar manner. At each time step, the nonlinear system of equations (SM.1.3), (SM.1.4), (SM.1.5) and (SM.1.8) must be solved. However, to get the Newmark–β time integration (SM.1.7) involved, the residual, R_d, is defined with the pressure as the input, denoted as P^I. Then, starting from equation (SM.1.5), solved with the scheme (SM.1.7), $\bar{U}_Y$ and $\dot{\bar{U}}_Y$ are obtained. Then, $\bar{H}$ and Q are evaluated from equations (SM.1.8) and (SM.1.3), respectively. Equation (SM.1.4) gives another evaluation of the pressure, denoted as, P^O. The residual is thus evaluated as

$$R_d = P^O - P^I. \tag{SM.1.10}$$

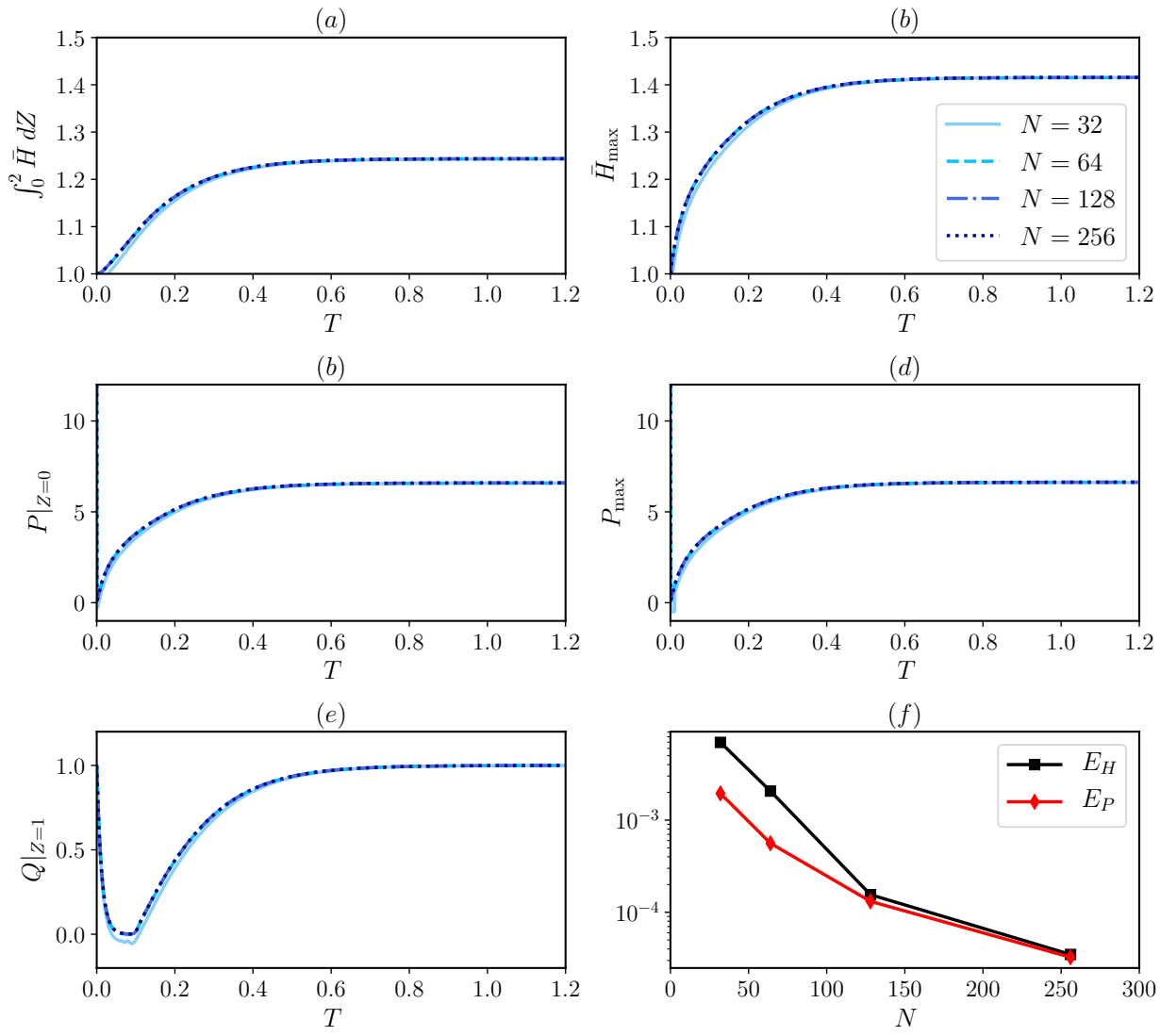

Figure SM.2: Grid independence study on the dynamic simulations of FSI in the microchannel with $E = 2$ MPa under a flow rate of $q = 1500$ µL min^{-1} (case C4). The time histories of the representative quantities in the system are shown in panels (a) to (e). Panel (f) shows the two-norm of the difference between the simulated steady state solution and the "exact" solution, which is computed from the steady simulation with $N = 2048$ Gauss–Lobatto points using the scheme described in § SM.1.1. The tolerance used in SciPy's `newton_krylov` was 10^{-8}. The errors E_H and E_P are computed via equation (SM.1.11).

At each time step, SciPy's `newton_krylov` is used to minimize the max-norm of R_d. The details of this numerical procedure are summarized in the flow chat in figure SM.1.

Note that at time step $n + 1$, θ_t is evaluated as $\theta_t = \tilde{\theta}_t \int_0^1 (\boldsymbol{D}^{(1)} \bar{H}^n)^2 \, \mathrm{d}Z$. The integral is approximated using the Gauss–Lobatto quadrature after interpolating the kernel on the finer grid of $N = 1024$. Here, we use $\bar{H}^n$ instead of $\bar{H}^{n+1}$ in order to avoid another nonlinear problem requiring "internal iterations" on θ_t^{n+1}. We have verified that the results using $\bar{H}^n$ instead of $\bar{H}^{n+1}$ do not differ from those evaluated based on more involved method using $\bar{H}^{n+1}$.

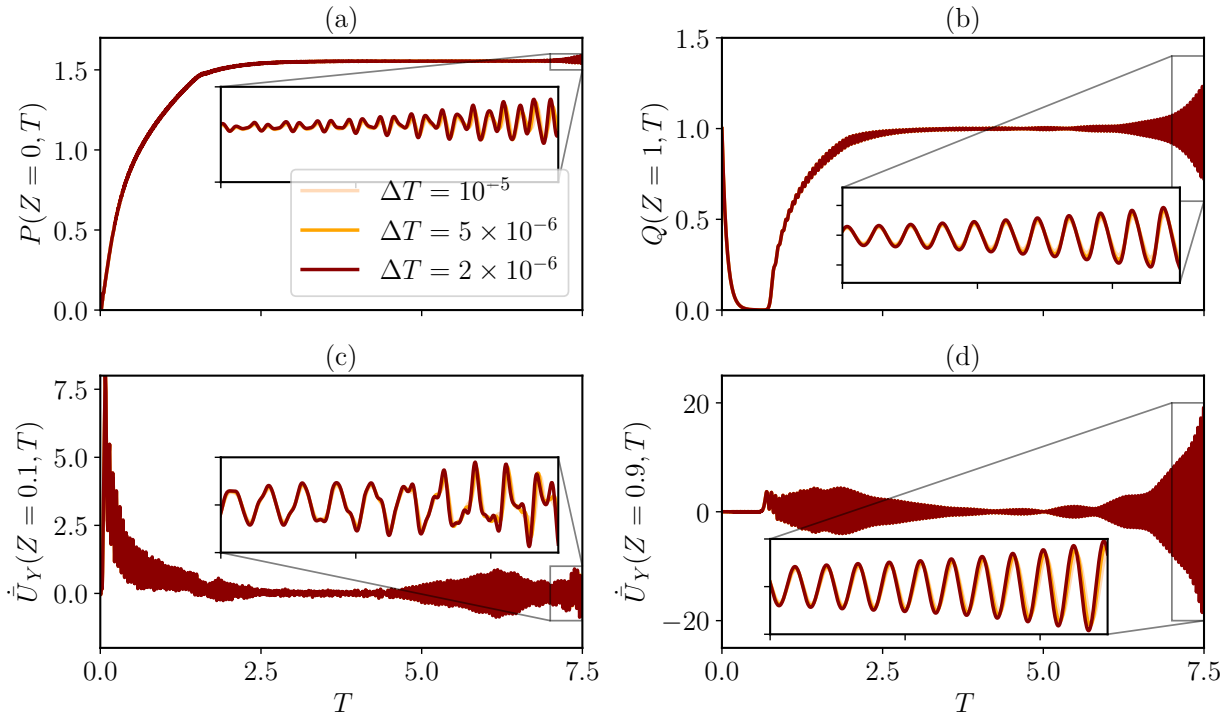

Figure SM.3: Time histories of (a) inlet pressure $P(0,T)$, (b) outlet flow rate $Q(1,T)$, the vertical velocity $\dot{\bar{U}}_Y$ of the fluid–solid interface at (c) $Z=0.1$ and (d) $Z=0.9$. All panels are for case C3 but using different time step sizes ΔT. The spatial grid is fixed to have $N=128$ nodes.

SM.1.2.1 Grid independence study

Next, we verify the grid independence of the numerical results shown in the main text. The case chosen to perform the grid independence study is C4 from table 5 (corresponding to $E=2$ MPa and $q=1500$ μL min^{-1}). For case C4, we have shown in § 7 that the inflated steady state is linearly stable to infinitesimal perturbations. If the flat initial condition (5.6) is used, then the system will reach the steady state eventually.

The end time for the simulation is $T_{\text{end}}=2.0$. However, for the clarity of the presentation, only the results between $T=0$ and $T=1.2$ are shown. Furthermore, the ratio of the smallest grid size to the time step is fixed for every simulation. Since the smallest spacing of the Gauss–Lobatto grid points goes as $\mathcal{O}(N^{-2})$, then as N is doubled, ΔT is decreased by a factor 4 accordingly. The time step for $N=32$ is $\Delta T = 4\times10^{-4}$. As shown figure SM.2$(a)$ and (e), all of the representative quantities agree well with each other as ΔT is refined, except on the courses grid with $N=32$,

After the simulation has reached $T=T_{\text{end}}$, the deformed interface shape, $\bar{H}^{\text{end}}(Z)$, and the pressure distribution within the channel, $P^{\text{end}}(Z)$ are compared with an "exact" steady-state solution. The latter are denoted as $\bar{H}^{e}(Z)$ and $P^{e}(Z)$, respectively. The "exact" solution is taken to be the steady state of the simulation with $N=2048$, and tolerance for SciPy's `newton_krylov` set to 10^{-8}. We define two L^2-norm based error estimates:

$$E_H = \left(\frac{1}{2}\sum_{j=1}^{N-1}(\bar{H}_j^{\text{end}}-\bar{H}_j^{e})\mathfrak{w}\sqrt{1-\tilde{Z}_j^2}\right)^{1/2}, \tag{SM.1.11a}$$

$$E_P = \left(\frac{1}{2}\sum_{j=1}^{N-1}(P_j^{\text{end}}-P_j^{e})\mathfrak{w}\sqrt{1-\tilde{Z}_j^2}\right)^{1/2}, \tag{SM.1.11b}$$

which are written in the discrete form using the Gauss–Lobatto quadrature. Here, $\mathfrak{w}=\pi/N$ are the weights,

and we choose $N = 2048$. Figure SM.2(f) shows that the error decreases with the increase of N. The cases of $N = 32,\ 64$ and 128 even display an exponential decay for E_H. However, since the "exact" solution is not really exact, both error estimates tend to "saturate" for $N = 256$.

As for the linearly unstable cases, the errors defined in equation (SM.1.11) are not applicable because the system will not reach steady state. In these cases, each simulation is tested with different time step sizes, and only the converged results are shown. The spatial grid is typically fixed as $N = 128$ for satisfactory accuracy (as shown in figure SM.2). One example for case C3 is shown in figure SM.3. In panel (c) and (d), the vertical velocity of the fluid–solid interface is obtained as follows. First, we substitute the simulated $\bar{H}$ into equation (5.1) to obtain V_Z^{2D}. Then, we compute V_Y^{2D} based on conservation of mass, *i.e.*, $\partial V_Z^{2D}/\partial Z + \partial V_Y^{2D}/\partial Y = 0$. Lastly, we obtain $\dot{\bar{U}}_Y = \beta^{-1} V_Y^{2D}|_{Y=\bar{H}}$ using equations (5.2) and (4.7). The actual simulation time is longer than the time window shown for each case. However, it is observed that after a certain T, the results with different time step sizes begin to diverge, indicating this nonlinear 1D FSI model's dynamic behavior may be chaotic. Understanding such an interesting possibility is beyond of the scope of the current work.

SM.2 A Chebyshev pseudospectral method for the generalized eigenvalue problem

We solve the generalized eigenvalue problem (7.4) using the approach proposed by Inamdar et al. (2020). However, since the boundary conditions for equation (7.4) are different from the boundary conditions of Inamdar et al. (2020), we employ another modified Lagrange polynomial basis, now written as

$$\widetilde{Q}(\tilde{Z}) \approx (1+\tilde{Z}) \sum_{j=1}^{N} \widetilde{Q}_j \frac{\ell_j(\tilde{Z})}{1+\tilde{Z}_j}, \qquad \widetilde{H}(\tilde{Z}) \approx \sum_{j=1}^{N-1} \widetilde{H}_j \ell_j(\tilde{Z}) + \widetilde{H}_N (1-\tilde{Z}^2). \tag{SM.2.1}$$

It is easy to check that $\widetilde{Q}(\tilde{Z}_j) = \widetilde{Q}_j$ and $\widetilde{H}(\tilde{Z}_j) = \widetilde{H}_j$, except that $\widetilde{H}(\tilde{Z}_N) = 0 \neq \widetilde{H}_N$, meaning that $\widetilde{Q}_j$ and $\widetilde{H}_j$ are the collocated function values; $\widetilde{H}_N$ is introduced ensure the satisfaction of the boundary condition. Also note j starts from 1 instead of 0, because equation (SM.2.1) has already satisfied the conditions that $\widetilde{Q} = \mathrm{d}\widetilde{Q}/\mathrm{d}\tilde{Z} = 0$ at $\tilde{Z} = -1$, and $\widetilde{H} = 0$ at $\tilde{Z} = \pm 1$. The two remaining boundary conditions at $\tilde{Z} = 1$ in equation (7.6) are enforced manually.

Next, the generalized eigenvalue problem (7.4) is collocated at the Gauss–Lobatto points from $j = 1, 2, \ldots, N-1$, with the unsatisfied boundary conditions enforced at $j = N$. This approach gives rise to the $2N \times 2N$ matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ in equation (7.4). Due to the imposition of the boundary conditions at $\tilde{Z} = 1$, $\boldsymbol{B}$ is singular. Thus, we invert $\boldsymbol{A}$ to obtain a regular eigenvalue problem, $\boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{\psi} = (\mathrm{i}\omega_G)^{-1}\boldsymbol{\psi}$, which can be solved using SciPy's `eig`.

Finally, to filter out any spurious modes, each calculation has been performed with $N = 600$ and $N = 800$ grid points and cross-checked. Only the converged modes are reported in the text.