

A Equivalence of Pressure Force Representations

To connect the global vector and the particle-wise representations, we derive the particle-wise pressure force (Eq. (7)) directly from the global-vector representation (Eq. (8)). We start by extracting the i -th block component of Eq. (8) as a summation over all N particles:

$$\mathbf{f}_i^p = - \sum_{k=1}^N \kappa_k \frac{m_k}{\rho_k} \left(\frac{\partial \rho_k}{\partial \mathbf{x}_i} \right)^\top. \quad (31)$$

The derivative $\frac{\partial \rho_k}{\partial \mathbf{x}_i}$ is non-zero only in two cases: (1) $k = i$, i.e., when k is the particle i itself, or (2) $k = j$, i.e., when k is a neighbor of i . Accordingly, we split the summation into two parts:

$$\mathbf{f}_i^p = - \frac{m_i \kappa_i}{\rho_i} \left(\frac{\partial \rho_i}{\partial \mathbf{x}_i} \right)^\top - \sum_j \frac{m_j \kappa_j}{\rho_j} \left(\frac{\partial \rho_j}{\partial \mathbf{x}_i} \right)^\top. \quad (32)$$

Following Eq. (9), the gradient of particle i 's density with respect to its own position in the first term is given by $\left(\frac{\partial \rho_i}{\partial \mathbf{x}_i} \right)^\top = \sum_j m_j \nabla W_{ij}$. Similarly, the gradient of neighbor j 's density with respect to $\mathbf{x}_i$ in the second term is given by $\left(\frac{\partial \rho_j}{\partial \mathbf{x}_i} \right)^\top = -m_i \nabla W_{ji}$, which simplifies to $m_i \nabla W_{ij}$, yielding the particle-wise symmetric pressure force:

$$\begin{aligned} \mathbf{f}_i^p &= - \frac{m_i \kappa_i}{\rho_i} \left(\sum_j m_j \nabla W_{ij} \right) - \sum_j \frac{m_j \kappa_j}{\rho_j} (m_i \nabla W_{ij}) \\ &= -m_i \sum_j m_j \left(\frac{\kappa_i}{\rho_i} + \frac{\kappa_j}{\rho_j} \right) \nabla W_{ij}. \end{aligned}$$

B Explicit Fixed-Point Update

We derive the explicit fixed-point update for velocity in Eq. (23), starting from the dual problem of Eq. (20):

$$(\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^\top)\boldsymbol{\lambda}^l = \frac{\mathbf{r}^l}{\Delta t^2}, \quad \boldsymbol{\lambda}^l \geq \mathbf{0}, \quad (33)$$

where $\mathbf{r}^l = \mathbf{c} + \Delta t \mathbf{J} \mathbf{v}^l$ is the constraint residual for the l -th subproblem. Applying the standard PRJ to solve Eq. (33) yields:

$$\begin{aligned} \boldsymbol{\lambda}^{l,m+1} &= \boldsymbol{\lambda}^{l,m} + \omega \mathbf{D}^{-1} \left(\frac{\mathbf{r}^l}{\Delta t^2} - \mathbf{J}\mathbf{M}^{-1}\mathbf{J}^\top \boldsymbol{\lambda}^{l,m} \right), \\ \boldsymbol{\lambda}^{l,m+1} &\leftarrow \boldsymbol{\phi}(\boldsymbol{\lambda}^{l,m+1}), \end{aligned} \quad (34)$$

where m denotes the PRJ iteration count. A single iteration with $\boldsymbol{\lambda}^{l,0} = \mathbf{0}$, Eq. (34) can be simplified as follows:

$$\boldsymbol{\lambda}^l \approx \boldsymbol{\lambda}^{l,1} = \frac{\omega}{\Delta t^2} \mathbf{D}^{-1} \boldsymbol{\phi}(\mathbf{r}^l), \quad (35)$$

where we use the fact that $\mathbf{D}^{-1}$ is diagonal with positive entries, allowing $\boldsymbol{\phi}$ to be applied directly to $\mathbf{r}^l$. Finally, the Karush–Kuhn–Tucker conditions for Eq. (20) yield the explicit fixed-point update:

$$\mathbf{v}^{l+1} = \mathbf{v}^l - \Delta t \mathbf{M}^{-1} \mathbf{J}^\top \boldsymbol{\lambda}^l. \quad (36)$$

C Matrix-free $\mathbf{A}^l \mathbf{q}$ Implementation

Given the system matrix $\mathbf{A}^l = \mathbf{M} + \omega \mathbf{J}^\top \mathbf{D}^{-1} \frac{\partial \boldsymbol{\phi}}{\partial \mathbf{r}} \Big|_{\mathbf{r}^l} \mathbf{J}$ defined in Eq. (29) and an input vector $\mathbf{q} \in \mathbb{R}^{3N}$, we compute the matrix-vector product $\mathbf{A}^l \mathbf{q}$ as

$$\mathbf{A}^l \mathbf{q} = \mathbf{M} \mathbf{q} + \omega \left(\mathbf{J}^\top \mathbf{D}^{-1} \frac{\partial \boldsymbol{\phi}}{\partial \mathbf{r}} \Big|_{\mathbf{r}^l} \mathbf{J} \right) \mathbf{q}. \quad (37)$$

The first term $\mathbf{M} \mathbf{q}$ is trivial due to the block-diagonal structure of the mass matrix $\mathbf{M}$. For each particle i , it is computed locally as:

$$(\mathbf{M} \mathbf{q})_i = m_i \mathbf{q}_i. \quad (38)$$

We focus on the second term, which is evaluated in three sequential steps without explicitly assembling the global matrices. First, we multiply the Jacobian $\mathbf{J}$ by the input vector $\mathbf{q}$. The i -th component of the resulting vector $\mathbf{s}_1$ represents the local density variation induced by $\mathbf{q}$. Using the density gradient formulation of Eq. (9), $s_{1,i}$ is computed by summing the relative contributions from neighboring particles j :

$$s_{1,i} = (\mathbf{J} \mathbf{q})_i = \sum_j m_j (\mathbf{q}_i - \mathbf{q}_j) \cdot \nabla W_{ij}. \quad (39)$$

Next, we apply the diagonal scaling matrix $\mathbf{D}^{-1} \frac{\partial \boldsymbol{\phi}}{\partial \mathbf{r}} \Big|_{\mathbf{r}^l}$. The i -th component of the intermediate vector $\mathbf{s}_2$ is then obtained as:

$$s_{2,i} = \mathbf{D}_{ii}^{-1} \frac{\partial \phi_i}{\partial r_i} \Big|_{r_i^l} s_{1,i}, \quad (40)$$

where $\frac{\partial \phi_i}{\partial r_i} \Big|_{r_i^l}$ is the Heaviside step function that equals 1 for active constraints (i.e., $r_i^l > 10^{-12}$) and 0 otherwise, as described in Sec. 6.1. Note that $s_{2,i}$ is non-zero only for particles within the active constraint set. Finally, multiplying by $\mathbf{J}^\top$ redistributes these scalar values back into the vector space. This step is structurally equivalent to the symmetric pressure force evaluation in Eq. (7). The contribution is accumulated as:

$$\mathbf{s}_{3,i} = \left(\mathbf{J}^\top \mathbf{D}^{-1} \frac{\partial \boldsymbol{\phi}}{\partial \mathbf{r}} \Big|_{\mathbf{r}^l} \mathbf{J} \mathbf{q} \right)_i = \sum_j (m_j s_{2,i} + m_i s_{2,j}) \nabla W_{ij}. \quad (41)$$

Combining these terms with the mass component from Eq. (37), the i -th block of the final output vector $\mathbf{A}^l \mathbf{q}$ is:

$$(\mathbf{A}^l \mathbf{q})_i = m_i \mathbf{q}_i + \omega \mathbf{s}_{3,i}. \quad (42)$$

Since all steps rely solely on particle-wise operations and neighborhood sums, this matrix-free implementation is efficiently parallelized on the GPU. Moreover, the operations in Eqs. (39) and (41) correspond to standard SPH routines for density gradient evaluation and symmetric pressure force computation, respectively, making the method straightforward to integrate into existing SPH frameworks.