Version: draft *Updated: 2025-09-10*

Section 1

Introduction

`fractions.h` provides a toolbox for reconstructing interfaces from the volume fraction field. Its two main purposes are:

1. **Initialization** — assigning volume fraction values from a given level-set field. This step is used to define the initial condition in VOF methods. In addition, it can generate face fraction values to ensure a closed interface, which is specifically required by embedded boundary methods.
2. **Reconstruction** — rebuilding the interface from the volume fraction field for use in VOF advection. Unlike initialization, this reconstruction does not require the interface to be closed.

In addition, the header includes auxiliary functions to support coarsening and refinement of volume fractions across grid levels, visualization of the interface, and computation of the total interface area.

Section 2

Assigning Volume Fractions from Level-Set Fields

Level-set function Φ represents the distance to the interface and are stored discretely at vertices of the gird. Take Φ as input the function fractions computes the volume fraction field and face fraction (if required).Space where $\Phi > 0$ is deemed as inside the interface, the volume fraction is set to 1.0 and vice versa.

In header file `geometry.h` a group of tools have been constructed to obtain volume fractions of single cell given the interface normal $\mathbf{n}$ and the intersection α both in 2D and 3D cases. The overall process to compute volume fraction is therefore twofolds: 1)to obtain $(\mathbf{n}, \alpha)$ 2)pass the parameter to corresponding tools provided by `geometry.h`. The underlying theory shall be briefly introduced in the following.

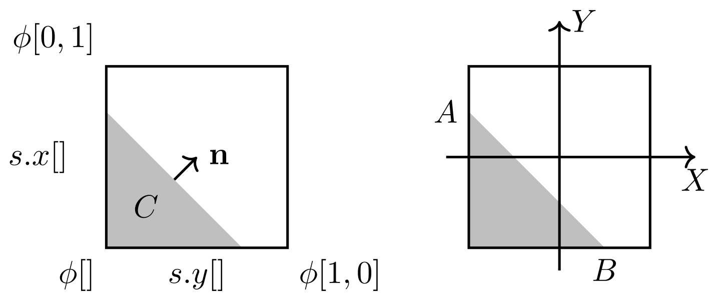

Figure 1: A sketch for level-set based volume fraction initialization. The left panel shows the normal vector $\mathbf{n}$, face fraction s and the vertex level-set value. The right panel illustrates the coordinate system, two intersection point is marked as A, B.

Figure 1 illustrates the geometry of a single cell in 2D. In the left panel the level-set values at the four vertices are denoted as $\phi[]$, $\phi[1,0]$, $\phi[0,1]$ and $\phi[1,1]$. The nonzero face fractions (edge fraction for 3D cases) are $s.x[]$ and $s.y[]$ which can be obtained directly from the level-set values as $s.x[] = \frac{\phi[]}{\phi[]-\phi[1,0]}$ and $s.y[] = \frac{\phi[]}{\phi[]-\phi[0,1]}$.

Consider that $\oint_{\partial\Omega} \mathbf{n} dl = 0$ which equals to

$$\mathbf{n} + (s.x[1] - s.x[])\mathbf{e_x} + (s.y[1] - s.y[])\mathbf{e_y} = 0 \tag{1}$$

Hence the interfacial normal is $\mathbf{n} = (s.x[] - s.x[1], s.y[] - s.y[1])$, the interfacial normal can be computed following same formula with s representing the face fraction on six cell-faces.

As for α, it can be derived from the intersection points A and B in the right panel of figure 1. Consider the origin locates at cell center and the length of the cell as 1, coordinates of A and B are $(-0.5, s.y[] - 0.5), (s.x[] - 0.5, -0.5)$ respectively. The line equation of the interface is $n_x X + n_y Y = \alpha$ where $\mathbf{n} = (n_x, n_y)$, substituting the coordinates of A or B into the equation yields (A as an example)

$$\alpha = -0.5 n_x + n_y(s.y[] - 0.5) \tag{2}$$

The final α is determined by averaging the results from A and B. For 3D cases, the procedure to compute α is the same but with more complicity, since one plane can cut through a cube with minimum three intersection points and maximum six intersection points.

To summarize, the algorithm to compute volume fractions from level-set fields is as follows:

1. Compute face (edge) fractions from level-set values at vertices.
2. Compute interfacial normal $\mathbf{n}$ from face (edge) fractions.
3. Compute intersection α from $\mathbf{n}$ and face (edge) fractions.
4. Compute volume (face) fraction from $\mathbf{n}$ and α using tools provided by `geometry.h` (2D cases ends here).
5. For 3D cases, compute interfacial normal using face fractions in all three directions, then compute α from $\mathbf{n}$ and edge fractions, finally compute volume fraction using tools provided by `geometry.h`.