

Characterization of the forcing and sub-filter scale terms in the volume-filtering immersed boundary method.

Himanshu Dave^a, Marcus Herrmann^a, Peter Brady^b, M. Houssem Kasbaoui^{a,*}

^a*School for Engineering of Matter, Transport and Energy, Arizona State University, Tempe, 85281, AZ, USA*

^b*CCS-2, Los Alamos National Laboratory, Los Alamos, 87545, NM, USA*

Abstract

We present a characterization of the forcing and the sub-filter scale terms produced in the volume-filtering immersed boundary (VF-IB) method by Dave et al. [5]. The process of volume-filtering produces bodyforces in the form of surface integrals to describe the boundary conditions at the interface. Furthermore, the approach also produces unclosed terms called τ_{sfs} . The level of contribution from τ_{sfs} on the numerical solution depends on the filter width δ_f . In order to understand these terms better we take a 2 dimensional, varying coefficient hyperbolic equation shown by Brady and Livescu [3]. This case is chosen for two reasons. First, the case involves 2 distinct regions separated by an interface, making it an ideal case for the VF-IB method. Second, an existing analytical solution allows us to properly investigate the contribution from τ_{sfs} for varying δ_f . The filter width controls how well resolved the interface is. The smaller the filter width, the more resolved the interface will be. A thorough numerical analysis of the method is presented, as well as the effect of τ_{sfs} on the numerical solution. In order to perform a direct comparison, the numerical solution is compared to the filtered analytical solution. Through this we highlight three important points. First, we present a methodical approach to volume filtering a hyperbolic PDE. Second, we show that the VF-IB method exhibits second order convergence with respect to decreasing δ_f (i.e. making the interface sharper). Finally, we show that τ_{sfs} scales with δ_f^2 . Large filter widths would require a modeling approach to sufficiently resolve τ_{sfs} . However for finer filter widths that have a sufficiently sharp interface, τ_{sfs} can be ignored without any significant reduction in the accuracy of solution.

Keywords: Volume filtering, Immersed boundary method

1. Introduction

Most computational simulations to model physical systems involve bounding surfaces with complex topological interfaces. In the case of fluid flows, examples include airfoils, stirring tanks, turbines and fluidized beds. Accuracy of numerical simulations for such systems hinges on the solver's ability to preserve the fundamental physics, and accurately resolving the interface. Furthermore, considering practical computational costs requires the solver to be robust, scalable and quick. Historically, body-conformal meshes have been the tool of choice to simulate such systems [2, 18, 7, 9, 22]. While an accurate approach, mesh generation can become a cumbersome process, particularly for complex topologies. In addition, this mesh needs to be regenerated at every timestep for moving interfaces [12], making it a computationally expensive process in large scale simulations. Immersed boundary methods (IBM), originally proposed by Peskin [10, 14, 15, 16] and later improved by several investigators [20, 8, 11, 4, 6, 13] have become an attractive option compared to body-fitted methods for two reasons. First, Cartesian grids are used to represent the fluid, alleviating the need for complex mesh creation around topologically complex interfaces. Second, since the interface is ‘immersed’ within the grid using a cloud of Lagrangian markers, for moving interfaces only the markers need to be transported [20]. The Lagrangian forcing that defines the boundary condition at the interface is transformed onto the Eulerian field using convolutions with a regularized Dirac delta [17]. Despite the popularity of IBMs, certain questions still remain due to the ad-hocness of the forcing term at the immersed boundary, since it does not correspond to

*Corresponding author, email: houssem.kasbaoui@asu.edu

any physical term in the Navier-Stokes equations. Furthermore, the development of internal flow within solid bodies and how it affects the hydrodynamic force at the interface is also a point of discussion [8, 20, 19].

Recently Dave et al. [5] presented a novel IB method using the volume filtering technique of Anderson and Jackson [1] called the Volume-Filtered Immersed Boundary (VF-IB) method. Using this approach, they provide sound answers to questions regarding the IB method such as (i) an analytical expression for the immersed boundary forcing term, (ii) Elucidating the role of internal flow, and (iii) a more accurate approach to calculating the Lagrangian marker volumes. The transport equations are derived by filtering the original point-wise equations in order to obtain a new set of filtered equations. The boundary conditions which are normally imposed on the fluid-solid interface are converted into bodyforces that apply on the right-hand side of the filtered transport equations as surface integrals. The process is mathematically and physically rigorous, and does not depend on any numerical considerations. They show that by doing this they remove any ad-hoc numerical fixes such as retraction of the immersed boundary to get accurate hydrodynamic forces [4]. Lastly, they show that the Lagrangian marker volumes depend on the local topology of the interface, the choice of filter kernel and the local curvature of the interface. Accurately calculating the Lagrangian marker volume can help improve the calculation of the hydrodynamic force due to more accurate interpolation and extrapolation operations. They also show an efficient procedure to compute the volume fractions of the different regions using a Poisson equation, required to accurately calculate the stresses at the interface.

While the VF-IB method helped answer several longstanding questions, it also has some questions that arise out of the volume-filtering procedure which require careful consideration. The process of volume-filtering the point-wise equations produces unclosed terms, including a subfilter scale tensor τ_{sfs} , similar in spirit to the Large Eddy Simulations [5]. The level of contribution from τ_{sfs} depends on the size of the filter width. Dave et al. [5] ignore this term and show good accuracy for sharp interface resolution. However, this value cannot be ignored as the filter width is increased. In order to make the VF-IB method scalable, simulations need to be accurately run at coarse resolution, thus making it important to understand the role of τ_{sfs} in comparison to the other terms in the filtered equation.

To understand τ_{sfs} and F_{IB} , we inspect the VF-IB method using an equation that has an analytical solution. This can help us properly understand the contribution of τ_{sfs} and other terms for varying parameters, allowing us to properly characterize the method. To do this we use a varying coefficient hyperbolic equation [3]. This is an archetypal problem to solve using the VF-IB method and ideal to help characterize it for two reasons. First, the problem at hand has two distinct regions separated by a circular interface, making it a perfect candidate for the VF-IB method. Second, an existing analytical solution allows us to properly perform an error analysis and understand the effects of τ_{sfs} and other terms in the filtered equation for varying parameters.

In section 2, we introduce the test case, show the computational domain, and investigate the analytical solution. In section 3 we introduce the volume filtering procedure and the prerequisites of the method. With the mathematical framework in place, we show the derivation of the filtered equations from the point wise equations in section 3.1. In section 3.2, we show how the volume fraction computation is performed. Accurately calculating the volume fraction is necessary to correctly resolve the interface and help distinguish between the two regions. The temporal and spatial schemes used in the numerical method are shown in section 4. We then conduct a thorough numerical analysis to look at the the order of convergence of the VF-IB method. The results are addressed in section 5. Section 5.1 shows an Apriori analysis of the case. We examine the filtered analytical solution which is the basis of comparison to the numerical solution produced by the VF-IB method in order to perform a more direct comparison. Furthermore, we also examine the different terms within the filtered equation and how they compare with each other. This section also explores how the magnitude of τ_{sfs} varies with varying filter width. Additionally, we also show how the subgrid resolution $\delta_f/\Delta x_f$ affects the accuracy of the filtered quantity. We then conduct an aposteriori analysis in section 5.2. Here we compare the numerical solution produced by the VF-IB method and the filtered analytical solution, both for varying filter width and varying grid resolution while keeping τ_{sfs} turned off. Lastly we investigate how including τ_{sfs} affects the solution for varying filter width. Finally, we give the concluding remarks in section 6.

2. Varying coefficient hyperbolic equation

In order to test the VF-IB method, we consider the two-dimensional test case of a varying coefficient hyperbolic equation shown in Brady and Livescu [3]. The governing equation in non-dimensional form is given as follows,

$$\frac{\partial u}{\partial t} + \nabla G \cdot \nabla u = 0, \quad (1)$$

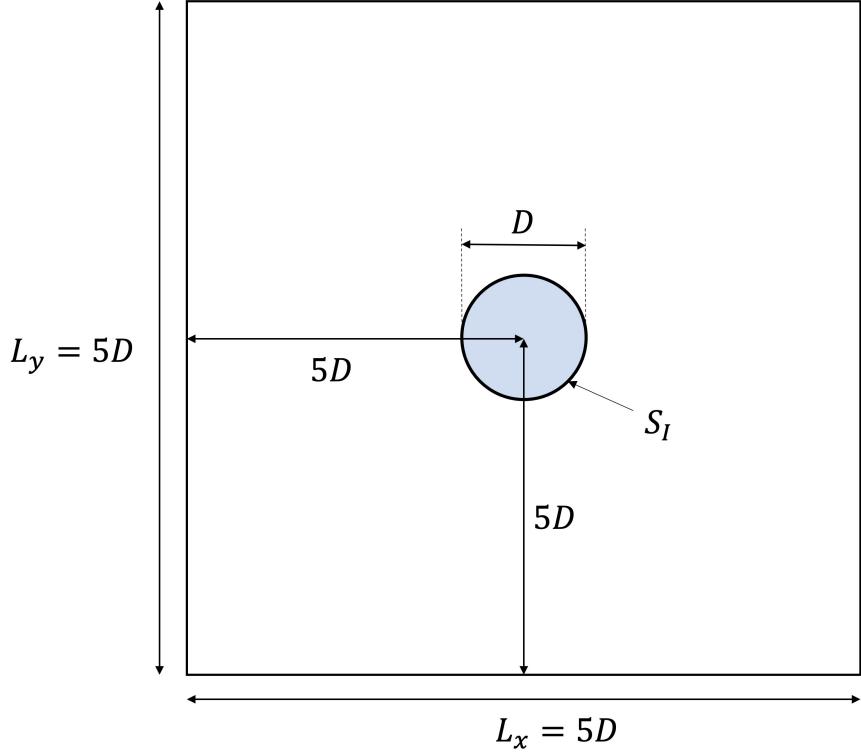


Figure 1: Computational domain for the test case of a varying coefficient hyperbolic equation similar to the domain used by Brady and Livescu [3].

where $G(x, y)$ is,

$$G(x, y) = \sqrt{(x - x_c)^2 + (y - y_c)^2} - r, \quad (2)$$

$$\Omega_2 = \{(x, y) \mid G(x, y) < 0\}, \quad (3)$$

$$S_I = \{(x, y) \mid G(x, y) = 0\}. \quad (4)$$

In equation (2), x_c and y_c are the center coordinates of the circle and r is the radius. S_I represents the interface between the two regions and Ω_2 represents the region inside the circle. The computational domain is shown in figure 1. we take a domain size of $L_x = L_y = 2$ and $D = 0.4$. The circle is located at the center of the computational domain. The initial and boundary conditions are given by,

$$u(x, y, t = 0) = \sin(2\pi G), \quad (5)$$

$$u_I(x, y, t) |_{G=0} = -\sin(2\pi t), \quad (6)$$

and the boundary conditions at the edge of the computational domain is an outflow condition. The analytical solution for the case is a circular pulse radiating out from the circle with a period of 1 such that,

$$u(x, y, t) = \sin(2\pi(G - t)). \quad (7)$$

Figure 2 shows the analytical solution for one period of the circular radiating pulse. A clear circular pulse radiating out from the circle is shown. Furthermore, we display the graph of u vs x along the horizontal direction centered in the vertical direction. The radiating pulse is symmetric in all directions.

3. Volume-filtering approach

In this section, we show the volume filtering approach by Anderson and Jackson [1] and how it can be applied in conjunction with the immersed boundary method. Figure 3 shows an illustration of the volume filtering approach.

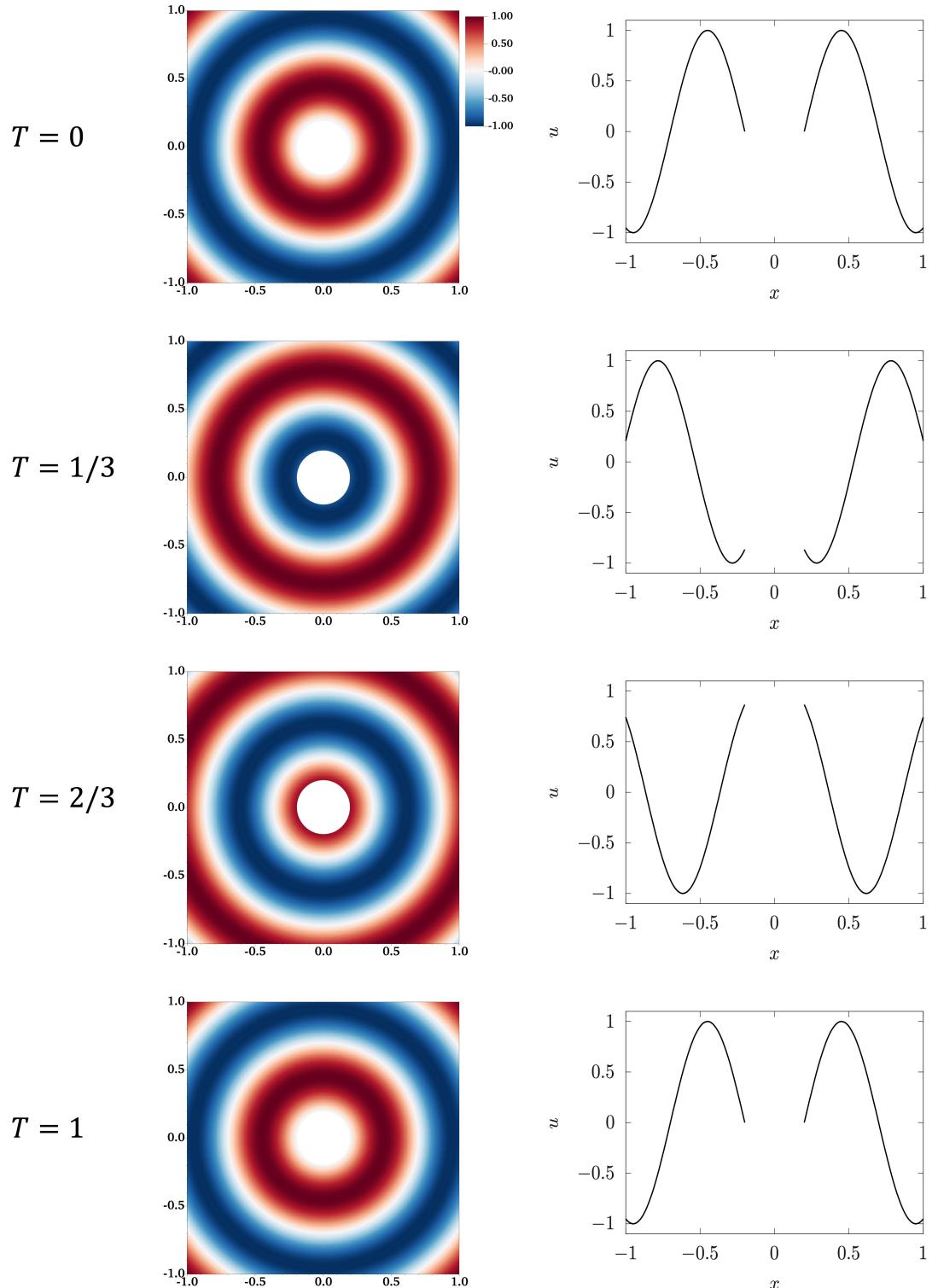


Figure 2: Analytical solution for a varying coefficient hyperbolic equation. The isocontours of u at different time periods T (left) and the value of u vs x in the horizontal direction, centered in the vertical direction (right). There is no velocity field within the region inside the circle.

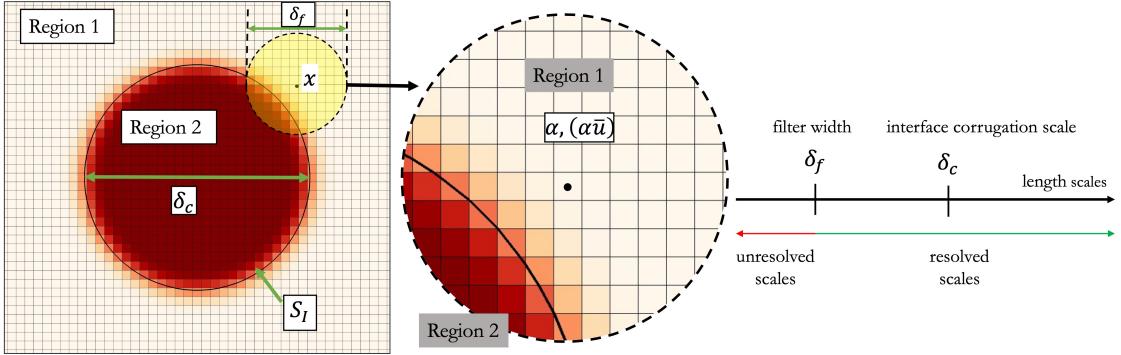


Figure 3: Illustration of the volume-filtering approach. Filtering the point-wise fields allows the extraction of the volume fraction α , and the averaged point-wise fields $(\alpha\bar{u})$ for region 1. The averaging is performed within a region of size δ_f . The immersed boundary is well resolved when the characteristic corrugation scale δ_c of the interface is much larger than the filter width δ_f . (δ_f is not to scale in the figure, but rather shown much larger for easier understanding of the concept of volume-filtering).

Here two distinct regions are separated by an immersed boundary surface S_I . While there is a sharp discontinuity between the regions analytically, their respective volume fraction fields are a smeared indicator function. The degree of smearing is dependent on the size of the filter width δ_f . For illustration, S_I in figure 3 is chosen by the iso-level $\alpha = 0.5$, where α represents the volume fraction related to region 1.

To understand how the volume filtering process works, let us take the point-wise quantity u defined in region 1. We now take a point x close to the immersed boundary surface but within region 1 as shown in figure 3. u can then be filtered at point x . The size of the area within which the filtering process takes place to obtain an averaged quantity at x depends on the size of the filter width δ_f . Through this process, we obtain the volume fraction α , and the averaged point-wise quantity $(\alpha\bar{u})$ at x . While u only exists in region 1, the volume-filtered field $(\alpha\bar{u})$ exists everywhere. As we move past the interface within region 2, $(\alpha\bar{u})$ smoothly decays to zero. There exists a region $\delta_f/2$ away from the interface within region 2 where $(\alpha\bar{u})$ is still non-zero. This is because for points within region 2 close to the interface, there exists a partial area under the filter width that is within region 1. In the case of figure 3, we show the volume fraction field α . The size of the region considered when averaging a point-wise quantity depends on the filter width δ_f . Hence, smaller the filter width, the faster α and $(\alpha\bar{u})$ will decay to zero as we move into region 2. Furthermore, how well resolved the immersed boundary is dependent on the ratio between the interface corrugation length scale δ_c and the filter kernel width δ_f . In order to have a well resolved immersed boundary we require that $\delta_f \ll \delta_c$.

In order to formalize the idea of volume filtering, we consider a filter kernel g that satisfies,

$$\iiint_{y \in \mathbb{R}^3} g(y) dV = 1, \quad (\text{unitary}) \quad (8)$$

$$g(-y) = g(y), \quad (\text{symmetric}) \quad (9)$$

$$g(y) = 0 \text{ if } \|y\| \geq \delta_f/2. \quad (\text{compact}) \quad (10)$$

The property of symmetry is important since it helps eliminate artificial anisotropy. Compactness helps for fast numerical integration of g on surfaces. The integration is considered over the entire space.

The volume fraction at any arbitrary location x is given by

$$\alpha(x, t) = \iiint_{y \in \mathbb{R}^3} \mathbb{1}(y, t) g(x - y) dV. \quad (11)$$

Here $\mathbb{1}(y, t)$ is an indicator function equal to 1 if y is in region 1 and 0 otherwise. $\alpha(x)$ represents the total region 1 that exists within the support of the filter kernel. If x is far away from region 2 such that the entire area under the filter kernel support is exclusively within region 1, the value of $\alpha(x) = 1$. If the probe is moved closer to the interface, such that the area within the support of the kernel is under both the regions, the value of $\alpha(x)$ will be somewhere between $0 < \alpha(x) < 1$. Through this, we eliminate the discontinuous effects across the interface by smoothing out the volume

fraction field based on the scale of the filter width δ_f chosen. In a similar approach we obtain the filtered quantity for the point-wise quantity u that exists at any arbitrary point \mathbf{x} , which is defined as,

$$\alpha(\mathbf{x}, t)\bar{u}(\mathbf{x}, t) = \iiint_{\mathbf{y} \in \mathbb{R}^3} \mathbb{1}(\mathbf{y}, t)u(\mathbf{y}, t)g(\mathbf{x} - \mathbf{y})dV. \quad (12)$$

The volume-filtered quantity ($\alpha\bar{u}$) is continuous and exists everywhere. This value tends smoothly to 0 a distance $\delta_f/2$ away from the interface within region 2.

Following the work of Anderson and Jackson [1] we obtain the volume-filtered equations from the governing point-wise equations. In order to do this, we apply the filtering operations to the point-wise equations. Using the property of symmetry and the divergence theorem, we obtain the filtered gradient and time derivative operators,

$$\alpha(\mathbf{x})\bar{\nabla}u(\mathbf{x}) = \nabla(\alpha\bar{u}) - \iint_{\mathbf{y} \in S_I} \mathbf{n}u(\mathbf{y}, t)g(\mathbf{x} - \mathbf{y})dS, \quad (13)$$

$$\alpha(\mathbf{x})\frac{\partial \bar{u}}{\partial t}(\mathbf{x}) = \frac{\partial(\alpha\bar{u})}{\partial t} + \iint_{\mathbf{y} \in S_I} (\mathbf{n} \cdot \mathbf{u}_{IB})u(\mathbf{y}, t)g(\mathbf{x} - \mathbf{y})dS. \quad (14)$$

Here \mathbf{n} is the normal vector pointing from region 2 to region 1 at the interface S_I . In equation (14), \mathbf{u}_{IB} is the velocity of the immersed boundary, not to be confused with the boundary condition u_I , at the interface. For static boundaries, this value will be zero. The process of volume filtering removes the notion of a boundary, since the filtered quantities exist everywhere in \mathbb{R}^3 . The information of the boundary conditions emerge in the surface integrals that arise in equations (13) and (14).

3.1. Derivation of the volume-filtered equations

We now take equation (1) and perform volume-filtering operations on it to obtain the filtered governing equation. Filtering the first term in equation (1) leads to,

$$\alpha\frac{\partial \bar{u}}{\partial t} = \frac{\partial}{\partial t}(\alpha\bar{u}). \quad (15)$$

For a fixed interface, volume-filtering the time derivative leads to no surface integrals since the interface velocity, \mathbf{u}_{IB} is zero. In order to volume filter the second term in equation (1), we expand the filtered quantity as follows,

$$\alpha\bar{\nabla}G \cdot \bar{\nabla}u = \alpha\bar{\nabla}G \cdot \bar{\nabla}u + (\alpha\bar{\nabla}G \cdot \bar{\nabla}u - \alpha\bar{\nabla}G \cdot \bar{\nabla}u). \quad (16)$$

Volume filtering the first term on the right hand side of equation (16) leads to,

$$\alpha\bar{\nabla}G \cdot \bar{\nabla}u = \bar{\nabla}G \cdot \nabla(\alpha\bar{u}) - \bar{\nabla}G \cdot \iint_{\mathbf{y} \in S_I} u_I \mathbf{n}g(\mathbf{x} - \mathbf{y})dS. \quad (17)$$

Here u_I is the prescribed quantity at the interface. In equation (16), we see the emergence of the sub-filter scale term in the last term on the right hand side such that,

$$\tau_{sfs} = \alpha\bar{\nabla}G \cdot \bar{\nabla}u - \alpha\bar{\nabla}G \cdot \bar{\nabla}u. \quad (18)$$

This term is similar in fashion to the sub-grid scale terms that emerge from the Large-Eddy Simulation (LES) equation of the Navier-Stokes equation. Combining all the terms, we get the final volume-filtered governing equation for a varying coefficient hyperbolic equation as,

$$\frac{\partial}{\partial t}(\alpha\bar{u}) + \bar{\nabla}G \cdot \nabla(\alpha\bar{u}) = F_I - \tau_{sfs}. \quad (19)$$

Here $F_I = \bar{\nabla}G \cdot \iint_{\mathbf{y} \in S_I} u_I \mathbf{n}g(\mathbf{x} - \mathbf{y})dS$. The boundary conditions at the interface arise as surface integrals.

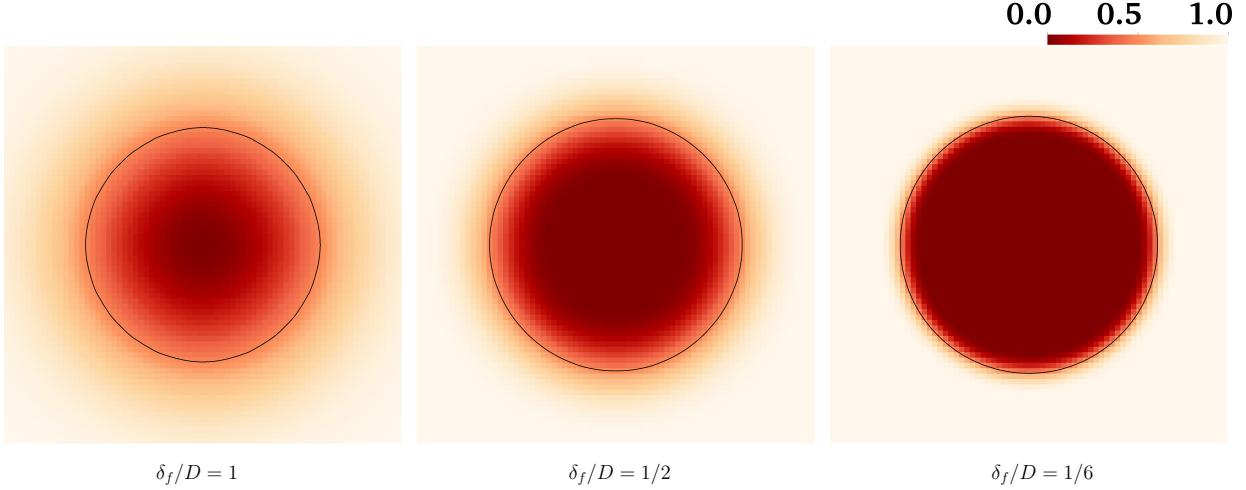


Figure 4: Volume fraction α , at three different filter widths with respect to the circle diameter ($\delta_f/D = 1, 1/2$ and $1/6$). The black contour line represents the Immersed Boundary (IB) surface located at $\alpha = 0.5$ in the limit $\delta_f/D \rightarrow 0$.

3.2. Volume-fraction computation

Computing the volume fraction of the different regions within the computational domain serves two goals pertaining to the test case presented above: (i) distinguishing between the interior and exterior points, (ii) computing the total volume occupied by the immersed solid. Additionally, the process of volume filtering produces the volume filtered quantity in the form of $(\alpha \bar{u})$. In certain cases we may be required to extract \bar{u} . This requires us to compute the volume fraction in order to perform the operation $(\alpha \bar{u})/\alpha$.

We first compute the region 1 indicator function within the field such that,

$$\mathbb{1}(\mathbf{x}, t) = \begin{cases} 1, & \text{if } \sqrt{(x - x_c)^2 + (y - y_c)^2} > R. \\ 0, & \text{otherwise.} \end{cases} \quad (20)$$

where R is the radius of the circle. The center of the circle is defined by x_c and y_c . The indicator function is then filtered, in order to obtain the volume fraction such that,

$$\alpha(\mathbf{x}, t) = \iiint_{y \in \mathbb{R}^3} \mathbb{1}(\mathbf{y}, t) g(\mathbf{x} - \mathbf{y}) dV. \quad (21)$$

Figure 4 shows the volume fraction field at 3 different δ_f/D values of $1, 1/2$ and $1/6$. Here, we show the diffuse nature of the volume fraction field. The distance it takes to go from $\alpha = 1$ to $\alpha = 0$ depends on the filter kernel support. The larger the filter width, as in the case of $\delta_f/D = 1$ shown in figure 4, the greater the smearing. This is visually observed when looking at the transition range from $\alpha = 1 \rightarrow 0$ for the $\delta_f/D = 1$ case. As we make the filter width smaller, i.e. $\delta_f/D = 1/6$, the transition occurs within a shorter distance. Hence, the sharpness of the filtered quantities depends on the filter width chosen. The smaller the filter width with respect to the circle, the sharper the representation of the interface. Furthermore, we also observe that as α transitions from one to zero, there does exist a region $\delta_f/2$ away from the interface but within region 2 where α is non-zero. In order to highlight the smearing effect of volume filtering more clearly we show a contour line at $\alpha = 0.5$ which is chosen to represent the IB surface in the limit $\delta_f/D \rightarrow 0$.

This approach works well for the case chosen due to the simplicity of the geometry together with having a fixed interface. For more complex topological surfaces and/or moving interfaces a more sophisticated approach should be chosen to compute the volume fraction. Dave et al. [5] presents an approach that entails solving a Poisson equation in order to compute the volume fraction such that,

$$\nabla^2 \alpha(\mathbf{x}, t) = \nabla \cdot \iint_{y \in S_I} \mathbf{n} g(\mathbf{x} - \mathbf{y}) dS. \quad (22)$$

This is similar to the computation of the phase-indicator functions with the front-tracking method of Unverdi and Tryggvason [21]. Dave et al. [5] choose an algebraic multigrid solver with Dirichlet boundary conditions to solve equation (22). The volume fraction is computed everywhere in the domain for simplicity. However, it would suffice to solve the Poisson equation in a narrow band with a thickness that is equal to the filter width δ_f similar to what is done by Unverdi and Tryggvason [21]. In the case of a fixed interface, the Poisson solver would only need to be solved once.

4. Numerical implementation

The VF-IB method is implemented in a library called LEAP. In this section we describe the numerical implementation of the VF-IB approach in LEAP pertaining to the test case shown in section 2.

4.1. Spatial discretization of the interface

We will now examine equation (19) and focus on the discretization of the interface. The IB interface is generated using a mesh of ‘m’ discrete elements with a surface area A_m , having a centroid \mathbf{x}_m and an outward pointing normal \mathbf{n}_m . The IB forcing can then be written as a sum of all discrete contributions from each element of the mesh.

The forcing on the interface is performed by the surface integral term in equation (19),

$$F_I(\mathbf{x}) = \overline{\nabla G}(\mathbf{x}) \cdot \sum_{m=1}^N \iint_{y \in S_I} u_I \mathbf{n} |_y g(\mathbf{x} - \mathbf{y}) dS. \quad (23)$$

Using the midpoint rule and assuming a typical mesh width of $O(\Delta x)$, equations (23) leads to,

$$F_I(\mathbf{x}) = \overline{\nabla G}(\mathbf{x}) \cdot \sum_{m=1}^N \{(u_I \mathbf{n})|_{\mathbf{x}_m} g(\mathbf{x} - \mathbf{x}_m) A_m\}. \quad (24)$$

Here \mathbf{x}_m is the centroid location of the mesh element S_m and its surface area is A_m . The centroids are portrayed as Lagrangian forcing points as shown in Dave et al. [5] and many other IB methods. The interface forcing is then extrapolated onto the Eulerian field. Details of the extrapolation procedure are shown in Dave et al. [5].

4.2. Temporal discretization

The time integration scheme is based on a strong-stability preserving Runge-Kutta 3rd order (SSP-RK3) scheme. The steps below describe the update from t^n to t^{n+1} .

Step 1: At this step the time integration loop is started. The first step is to make $u^{(0)}$ equal to u^n ,

$$(\bar{u})^{(0)} = (\bar{u})^n. \quad (25)$$

Step 2: Next, we compute the immersed boundary forcing term, i.e. equation (24), for step 1 of the SSP-RK3 scheme,

$$F_I^n(\mathbf{x}) = \overline{\nabla G} \cdot \sum_{m=1}^N \{(u_{I,m}^n \mathbf{n})|_{\mathbf{x}_m} g(\mathbf{x} - \mathbf{x}_m) A_m\}, \quad (26)$$

The updated immersed boundary forcing is then extrapolated onto the Eulerian field in order to proceed with the time integration scheme. The first step velocity is computed as,

$$(\bar{u})^{(1)} = \Delta t \left(-\overline{\nabla G} \cdot \nabla (\bar{u})^{(0)} + F_I^n - \tau_{\text{sfs}}^n \right) + (\bar{u})^{(0)}. \quad (27)$$

Step 3: In similar fashion to expressions (26), we compute the immersed boundary forcing term for step 2 of the SSP-RK3 scheme for time t^{n+1} . The second step velocity is computed as,

$$(\bar{u})^{(2)} = \frac{3}{4}(\bar{u})^{(0)} + \frac{1}{4} \left\{ \Delta t \left(-\overline{\nabla G} \cdot \nabla (\bar{u})^{(1)} + F_I^{n+1} - \tau_{\text{sfs}}^{n+1} \right) + (\bar{u})^{(1)} \right\}. \quad (28)$$

Step 4: Finally, we compute the final velocity u^{n+1} as,

$$(\bar{u})^{n+1} = \frac{1}{3}(\bar{u})^{(0)} + \frac{2}{3} \left\{ \Delta t \left(-\overline{\nabla G} \cdot \nabla (\bar{u})^{(2)} + F_I^{n+1/2} - \tau_{\text{sfs}}^{n+1/2} \right) + (\bar{u})^{(2)} \right\}. \quad (29)$$

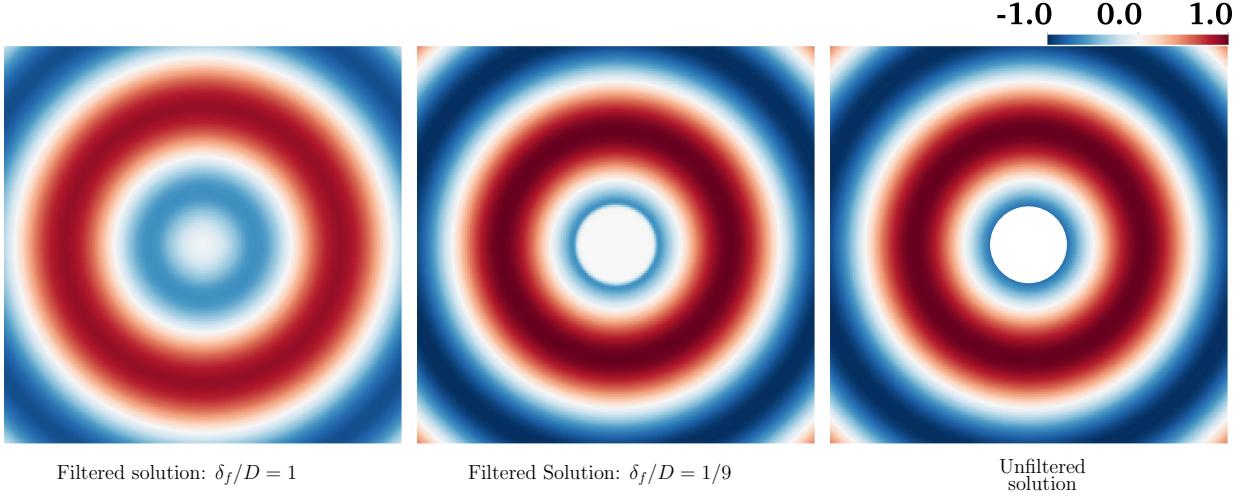


Figure 5: Isocontours of the filtered solution $(\bar{u})_e$, and the unfiltered solution u_e at time period $T = 1/4$. At this time the forcing is at its absolute maximum value. $(\bar{u})_e$ is shown at 2 different filter resolution of $\delta_f/D = 1$ and $\delta_f/D = 1/9$.

5. Results

In this section we show the comparison of the results obtained from the filtered analytical solution and solution obtained using the VF-IB method. The domain and cylinder radius are kept consistent with what is shown in section 2. First we conduct an apriori analysis (section 5.1) to understand the difference between the analytical solution shown in section 2 and the filtered analytical solution. We then examine the various terms within the filtered governing equation to show how each term plays a role. Furthermore, we show how τ_{sfs} changes as δ_f/D is varied. Secondly in section 5.2, we perform an error analysis and obtain and order of convergence for the VF-IB method for both spatial quantities, $\delta_f/\Delta x$ and δ_f/D . We also investigate how including τ_{sfs} in the solution affects it for varying filter width. In order to distinguish between the different quantities, u_e will be the unfiltered analytical solution, $(\bar{u})_e$ is the filtered analytical solution and (\bar{u}) is the numerical solution produced using the VF-IB method.

5.1. Apriori analysis

The accuracy of the filtered quantity in comparison to the unfiltered value is dependent on the filter width δ_f . The smaller the filter width, the sharper the representation of the filtered quantity is and the more closer it is to the unfiltered value. As the filter width is increased, the filtered field becomes more smeared and digresses from the unfiltered value. In order to show this, figure 5 shows the isocontours for $(\bar{u})_e$ at two different filter resolutions of $\delta_f/D = 1$ and $\delta_f/D = 1/9$. The results are at the same time, $T = 1/4$. We also show the unfiltered solution u_e at the same time for comparison. From the instantaneous visualizations, it is immediately clear that $(\bar{u})_e$ exists everywhere in space. The quantity smoothly decays to zero as we move past the interface towards the center of the circle. The distance past the interface at which $(\bar{u})_e$ is zero is $\delta_f/2$. In comparison, u_e only exists in region 1 and does not go past the interface. As shown in figure 5, at a filter width of $\delta_f/D = 1$ the smearing of the field is greater than at $\delta_f/D = 1/9$. Hence, for larger filter widths the transition region will be larger, as we move from region 1 into region 2. The smaller the filter width the sharper the field looks and the closer it is to the unfiltered solution.

In order to quantitatively highlight the differences between the filtered and unfiltered solution, figure 6 shows $(\bar{u})_e$ at different filter resolutions alongside the unfiltered solution u_e . We choose three different filter widths ranging from $\delta_f/D = 1$ to $\delta_f/D = 1/9$. $(\bar{u})_e$ and u_e are compared along the horizontal axis, centered in the vertical axis. We show the graph at three instances in time. Figure 6a is at $T = 0$, corresponding to the time when the forcing is at its absolute minimum value. In figure 6b, we show the results at $T = 1/8$, when the forcing is half of its maximum absolute value. Finally, we show the results when the forcing term is at its absolute maximum value at $T = 1/4$ in figure 6c. Again, we see that while the filtered quantity exists past the boundary, the unfiltered solution only exists outside the circle. Furthermore, we notice that as we reduce the filter width, the value of $(\bar{u})_e$ becomes sharper. The results obtained

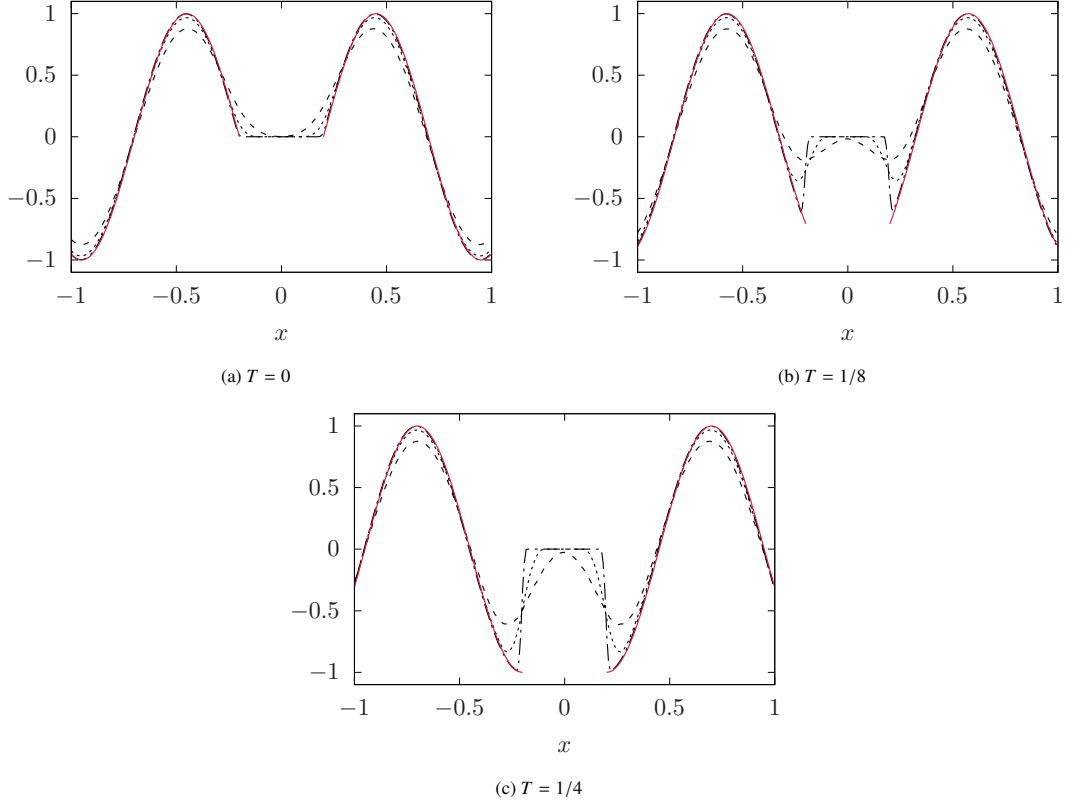


Figure 6: Value of $(\alpha \bar{u})_e$ vs x at $T = 0$, $T = 1/8$ and $T = 1/4$. The result shown is a line cut horizontally and centered in the vertical axis. We show three different filter resolutions of $\delta_f/D = 1$ (---), $\delta_f/D = 1/2$ (.....) and $\delta_f/D = 1/9$ (-·-). We also show u_e (—) for reference.

using a filter width of $\delta_f/D = 1/9$ follows the unfiltered solution well. However, with $\delta_f/D = 1$, we see greater smearing of the results. This is particularly visible close to the boundary ($x = -0.2$ and $x = 0.2$) and near the peaks of the solution.

The filtered equation has four terms that govern the solution. The time derivative $\partial(\alpha \bar{u})/\partial t$ represents the rate of change of the solution in time. $\bar{\nabla}G \cdot \nabla(\alpha \bar{u})$ represents an advection term. The two other terms that govern the solution are the forcing term F_I and the sub-filter scale term τ_{sfs} . We examine how F_I varies with the filter width. The expression for F_I is,

$$F_I = \bar{\nabla}G \cdot \iint_{y \in S_I} u_I n g(x - y) dS = \bar{\nabla}G \cdot \left(u_I \iint_{y \in S_I} n g(x - y) dS \right) = u_I (\bar{\nabla}G \cdot \nabla \alpha), \quad (30)$$

where we have used the identity $\iint_{y \in S_I} n g(x - y) dS = \nabla \alpha$ and the fact that u_I does not vary along the boundary. Since $\nabla \alpha$ scales as $1/\delta_f$, equation (30) shows that F_I also scales with $1/\delta_f$. Figure 7 shows the isocontours of the forcing term $|F_I|$ at varying δ_f/D . The isocontours are shown when the forcing at the interface is at its maximum ($T = 1/4$) and minimum ($T = 0$) absolute magnitude as well as a time in between ($T = 1/8$). This term is computed in a purely analytical fashion using equation (6). In order to highlight how the filter width affects the forcing field, we show two different cases at $\delta_f/D = 1/3$ and $\delta_f/D = 1/9$. We observe that the forcing term F_I is zero everywhere apart from a region within $\delta_f/2$ of the interface. This is inline with what would be expected since F_I is computed by extrapolating the quantity at the interface onto the nearby cells within a region $\delta_f/2$ away from the interface. Furthermore, F_I changes in a sinusoidal manner dependent on the boundary condition at the interface, dictated by u_I which is defined by equation (6). The larger the filter width the more smeared F_I is and hence has a larger area of influence to the field. As δ_f is reduced we see a sharper representation of F_I . While the interface is a sharp discontinuity, the volume-filtering

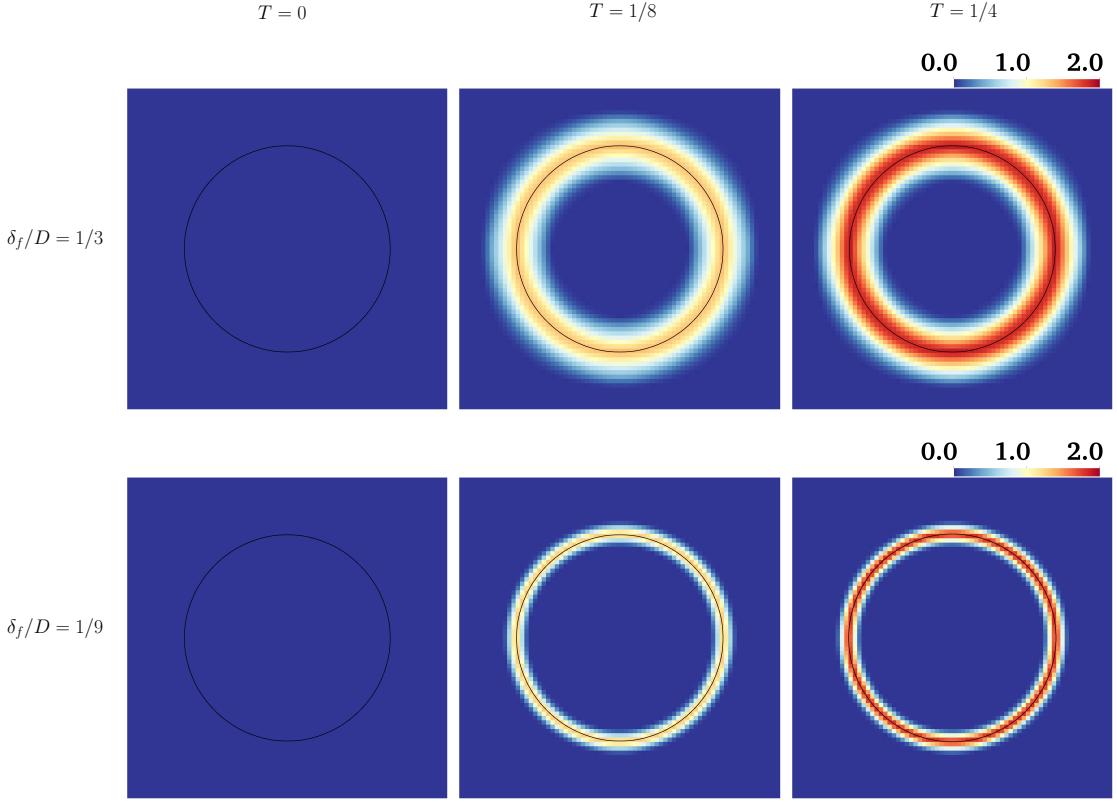


Figure 7: Isocontours of $|\delta_f F_I|$ at the absolute minimum value ($T = 0$) and at its absolute maximum value ($T = 1/4$). We also show F_I at a middle point between the two extremes at $T = 1/8$. The forcing is sinusoidal in nature with respect to time. F_I is shown at 2 different filter resolution of $\delta_f/D = 1/3$ and $\delta_f/D = 1/9$.

approach removes the notion of the sharp boundary. Instead, the boundary condition is enforced by the forcing term F_I over a narrow band whose width is dependent on the filter width δ_f .

τ_{sfs} is the sub-grid term that comes out of the volume-filtering process and is given in equation (18). Expanding τ_{sfs} using a Taylor series gives us,

$$\tau_{\text{sfs}} = \iiint_{y \in \mathbb{R}^3} \mathbb{1}(y, t) (\nabla G(y) - \nabla G(x)) \cdot ((y - x) \cdot \nabla(\nabla u)(x)) g(x - y) dV + \text{HOT}, \quad (31)$$

where HOT refers to the higher order terms in the Taylor series expansion. The equation above shows that τ_{sfs} scales with δ_f^2 . Figure 8 shows the isocontours for $|\tau_{\text{sfs}}/\delta_f^2|$. The results are shown for both the extreme cases when F_I is at its maximum amplitude and when F_I is zero. The snapshots are at the same time as those shown in figure 7. We observe that τ_{sfs} is maximum near the interface and gradually reduces as we go away from it while following a wave-like pattern. This is because τ_{sfs} is a function of $\bar{\nabla}G$ which is a function that decreases as we move away from the interface. The wave like pattern is due to the dot product between ∇G and ∇u . At this resolution and interface sharpness, for both maximum and minimum absolute forcing at the interface the value of τ_{sfs} is 3-4 orders of magnitude smaller than the advection term or the forcing term. From this, we infer that for a sharp enough interface the contribution of τ_{sfs} is negligible and does not affect the accuracy of the filtered solution.

Before comparing τ_{sfs} to the other terms in the equation, we first observe how the L_∞ -norm of τ_{sfs} varies with varying $\delta_f/\Delta x_f$. Here, Δx_f is the subgrid mesh that we employ within the filtering procedure in order to remove numerical errors from the filtering procedure as much as possible and in order to compute an accurate volume fraction field and hence a more accurate τ_{sfs} value at any point x . We take a filter width of $\delta_f/D = 1/6$ and compute $\|\tau_{\text{sfs}}/\delta_f^2\|_\infty$ and show the values at $T = 1/2$ and $T = 1/4$. Figure 9 shows that the value converges when $\delta_f/\Delta x_f = 32$.

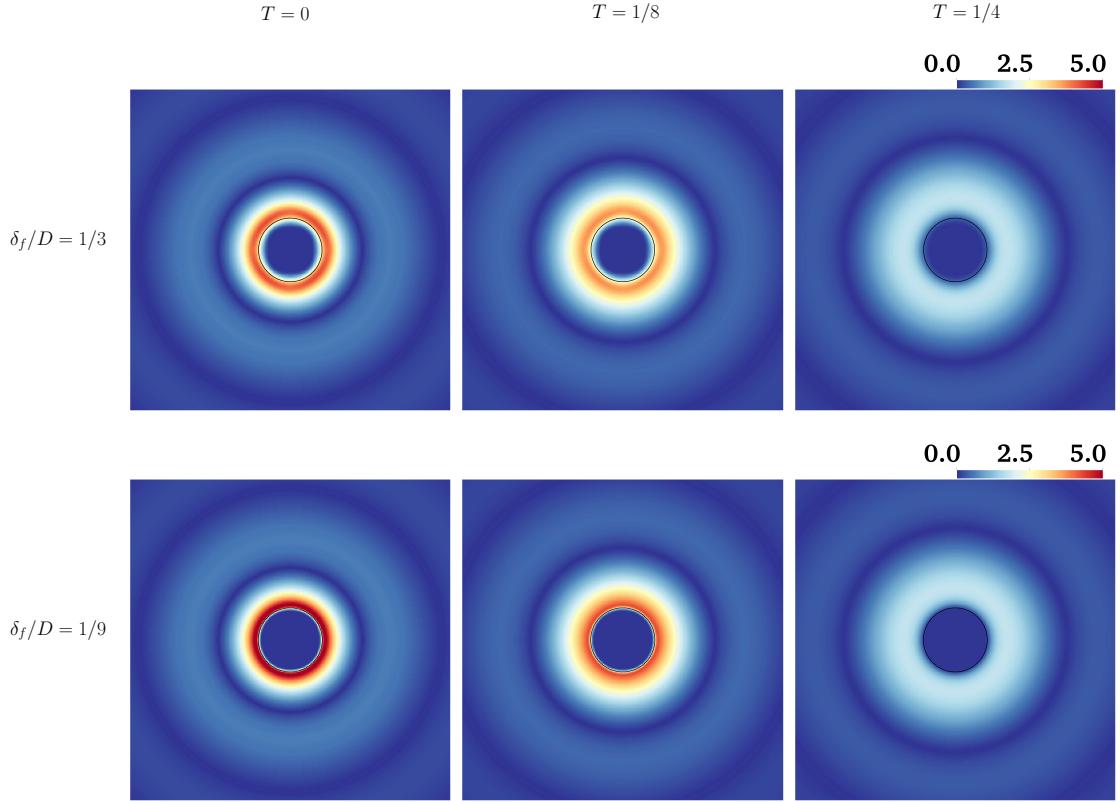


Figure 8: Isocontours of $|\tau_{\text{sfs}}/\delta_f^2|$ is shown three different time snapshots. The results are computed purely analytically apart from the filtering procedure. We show two different filter widths, $\delta_f/D = 1/3$ and $\delta_f/D = 1/9$.

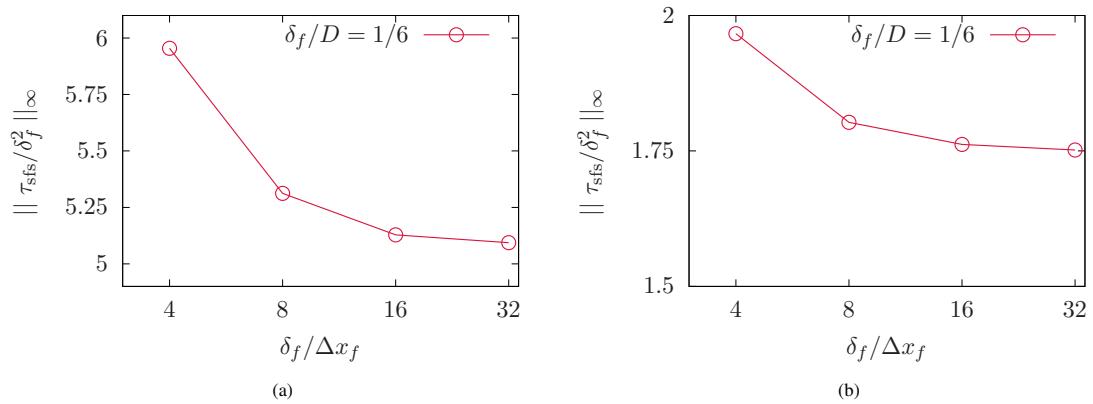


Figure 9: $\|\tau_{\text{sfs}}/\delta_f^2\|_\infty$ at $\delta_f/D = 1/6$ for varying $\delta_f/\Delta x_f$. We show the results are the two extreme time periods in the sinusoidal solution (a) $T = 1/2$ and (b) $T = 1/4$. We see that as we increase the number of subgrid points used for the filtering procedure, the value of $\|\tau_{\text{sfs}}/\delta_f^2\|_\infty$ converges at $\delta_f/\Delta x_f = 32$.

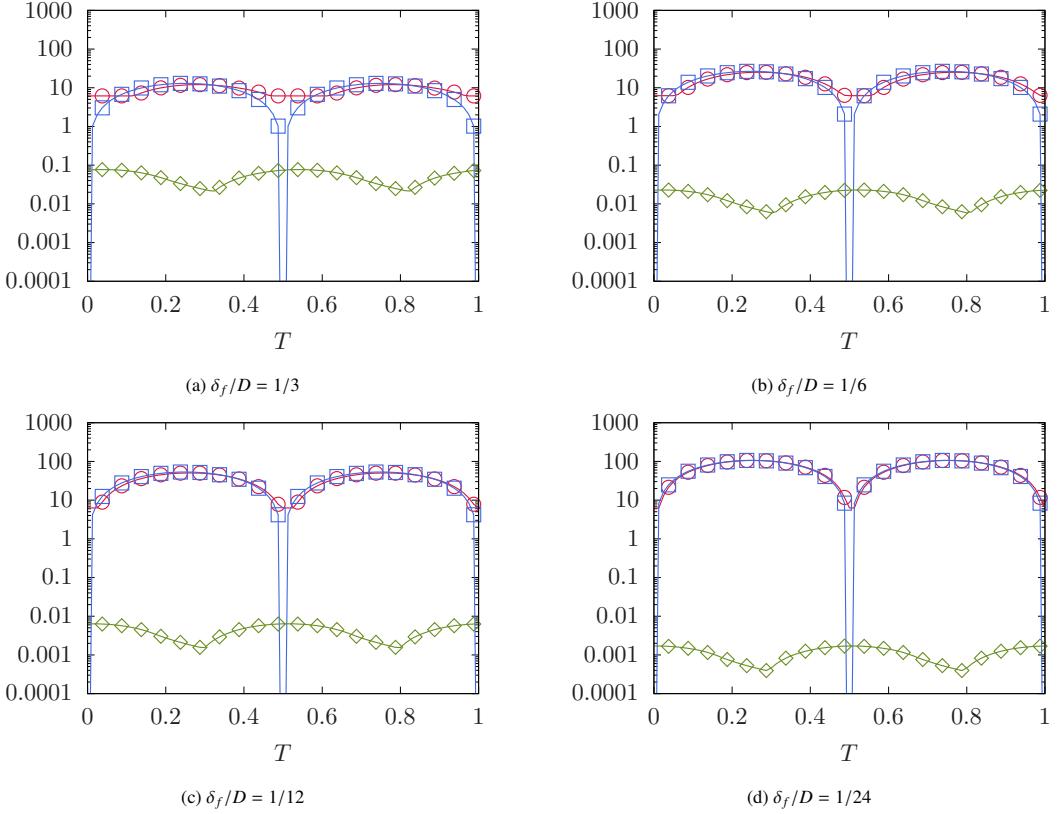


Figure 10: Time series of the L_∞ norms is shown for the different terms in equation 19. $\| F_I \|_\infty$ denoted by (\square). $\| \bar{G} \cdot (\alpha \bar{u}) \|_\infty$ is denoted by (\circ). Lastly $\| \tau_{\text{sfs}} \|_\infty$ denoted by (\diamond). The terms are computed using the analytical solution. We show the results for four different filter widths ranging from $\delta_f/D = 1/3$ to $\delta_f/D = 1/24$.

To shed more light on how the contribution of τ_{sfs} changes in comparison to the advection term and forcing term, figure 10 shows the L_∞ -norm of these terms different interface sharpness ranging from the coarsest interface sharpness of $(\delta_f/D)^{-1} = 3$ to the sharpest interface of $(\delta_f/D)^{-1} = 24$. We keep the subgrid mesh width constant at $\delta_f/\Delta x_f = 32$. The results clearly validate what is observed from the visual snapshots shown in figure 8. As we decrease δ_f , τ_{sfs} reduces, thus reducing its contribution to the solution. Furthermore, we see that τ_{sfs} peaks when the forcing term F_I is zero and is at its minimum when F_I is at its maximum amplitude. Figure 10 shows that at the coarsest interface sharpness τ_{sfs} is around 2 orders of magnitude smaller. As the interface sharpness is refined this number reduces to 4-5 orders of magnitude compared to the other two terms in the governing equations. We infer that for cases with coarser interface sharpness, a modeling approach would be warranted in order to obtain higher accuracy. However, for a fine enough interface sharpness, τ_{sfs} is negligible.

In order to confirm that τ_{sfs} scales with δ_f^2 , we plot $\| \tau_{\text{sfs}} \|_2$ and $\| \tau_{\text{sfs}} \|_\infty$ for increasing $(\delta_f/D)^{-1}$, i.e. increasing sharpness. In doing so, we calculate the order of convergence with respect to $(\delta_f/D)^{-1}$. The results are shown when τ_{sfs} is at its absolute maximum at $T = 1/2$ and when it is at its absolute minimum at $T = 1/4$ in figure 11. We show the order of convergence for the $\| \tau_{\text{sfs}} \|_2$ and $\| \tau_{\text{sfs}} \|_\infty$ norm when τ_{sfs} is at its maximum and when τ_{sfs} is at its minimum. Similar to above the ratio of $\delta_f/\Delta x_f = 32$ is kept constant. We observe that τ_{sfs} scales with δ_f^2 , hence giving us a second order rate of convergence for τ_{sfs} with δ_f .

5.2. *A posteriori* analysis

We now solve numerically the volume-filtered equations derived in section 3.1, and compare the numerical solution with the filtered analytical solution $(\alpha \bar{u})_e$. For now, we keep τ_{sfs} turned off.

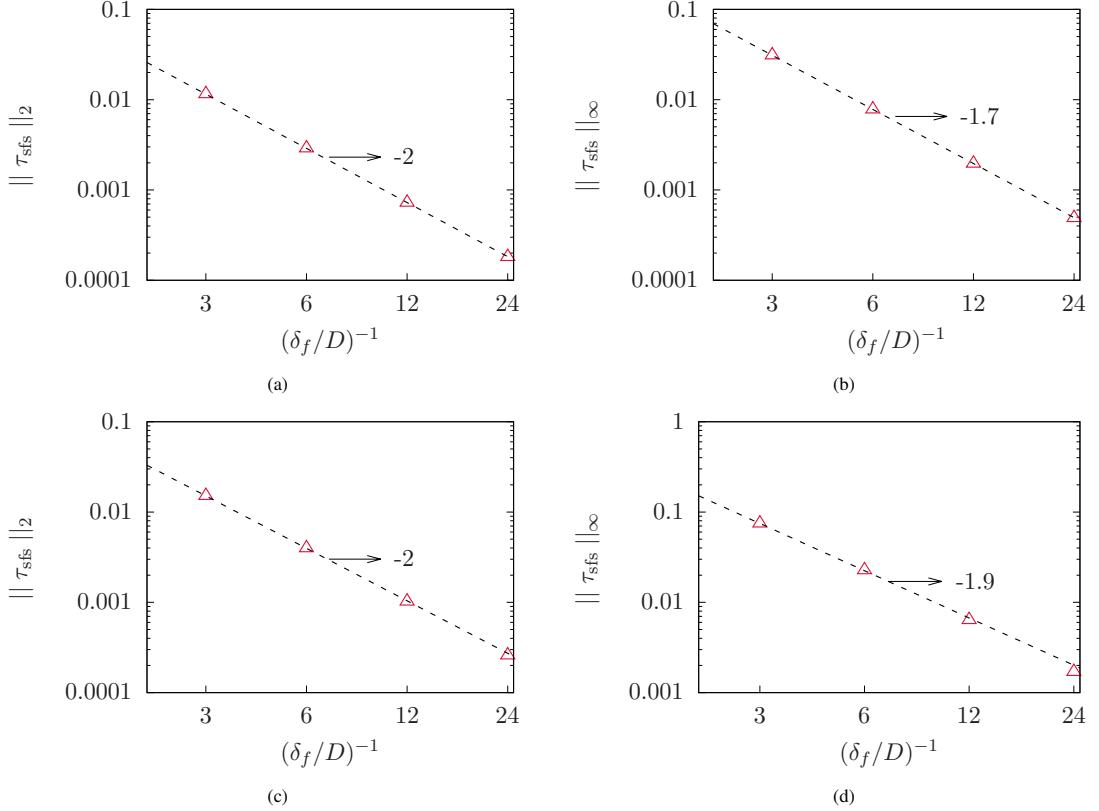


Figure 11: $\|\tau_{\text{sfs}}\|_2$ and $\|\tau_{\text{sfs}}\|_\infty$ at $T = 1/4$ (top) and $T = 1/2$ (bottom). The results are shown for varying filter width (δ_f/D) while keeping $\delta_f/\Delta x_f = 32$. The terms are computed analytically.

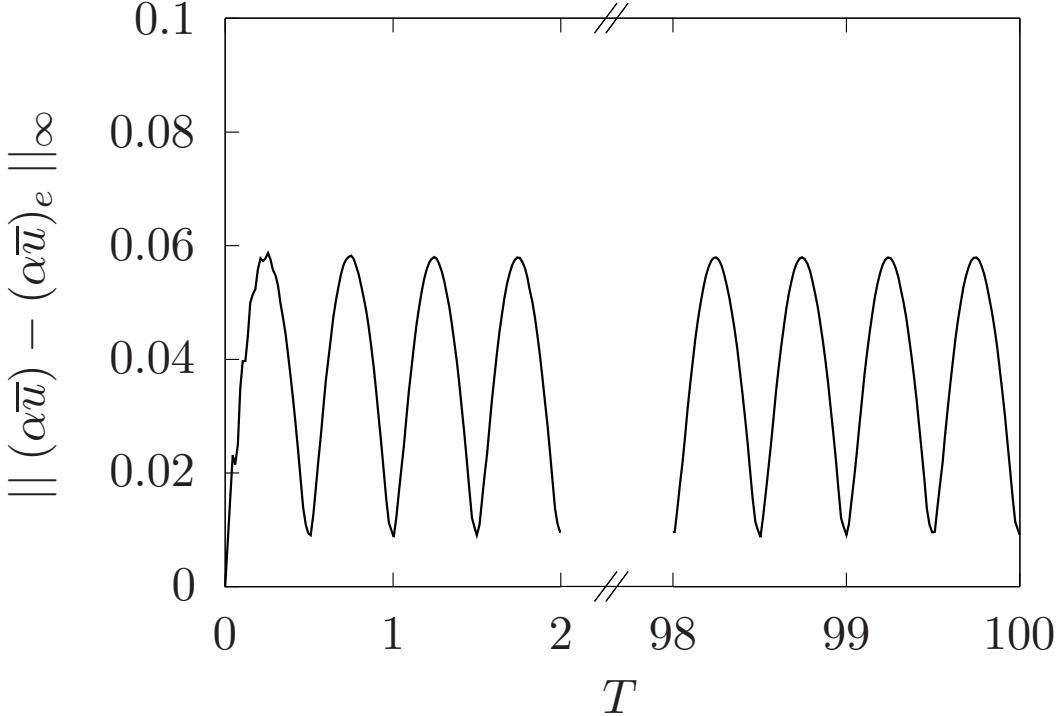


Figure 12: $\|(\bar{\alpha u}) - (\bar{\alpha u})_e\|_\infty$ for 100 time periods in order to show stability of the numerical solution. The result is run at $\delta_f/D = 1/12$, $\delta_f/\Delta x = 4$ and a CFL = 0.5. The subgrid mesh is kept at $\delta_f/\Delta x_f = 32$

In order to show the stability of the method we run the case at a filter width of $\delta_f/D = 1/12$ and a spatial resolution of $\delta_f/\Delta x = 4$. The simulation is run at a Courants-Friedrichs Lewy (CFL) number of $CFL = 0.5$. We run the simulation for a total of 100 periods and we report the $\|(\bar{\alpha u}) - (\bar{\alpha u})_e\|_\infty$ error in figure 12. The results show a constant sinusoidal graph, showing that the simulation is stable for long periods of time and does not blow up numerically. Furthermore, we observe that the error is at its maximum when the forcing term is greatest, at points $T = 1/4$ and $T = 3/4$. The error is at its minimum when the forcing is zero at the interface, $T = 0.5$ and $T = 1$.

Figure 13 shows the isocontours of $(\bar{\alpha u})$ and $(\bar{\alpha u})_e$ at $\delta_f/D = 1$ and $\delta_f/D = 1/24$. We show the results at $T = 1/4$, both cases are run at a $CFL = 0.25$, $\delta_f/\Delta x = 4$. We make sure to maintain the subgrid mesh at $\delta_f/\Delta x_f = 32$. The numerical solution approaches the filtered analytical solution as we decrease the filter width δ_f/D . For a more quantitative comparison, figure 14 plots the value of $(\bar{\alpha u})$ and $(\bar{\alpha u})_e$ as a line cut horizontally and centered in the vertical axis. We show the results for both a coarse filter width ($\delta_f/D = 1$) and a fine filter width ($\delta_f/D = 1/24$). We clearly see that as the filter width is reduced, the numerical solution is much closer to the filtered analytical solution. This is because as we coarsen the filter, the contribution from the sub-filter scale term, τ_{sfs} increases. Hence, for larger filter widths, Errors can be expected unless a modeling approach is utilized in order to account for the contributuon from τ_{sfs} . For finer filter widths, the contribution of τ_{sfs} is negligible, and therefore neglecting the term does not affect the quality of the solution.

In order to accurately calculate the order of convergence for $(\bar{\alpha u})$, we first need to eliminate all numerical errors as much as possible. One of the spatial parameters that govern the numerical solution is the ratio between the filter width and the mesh spacing $\delta_f/\Delta x$. The subgrid filtering mesh is kept constant at $\delta_f/\Delta x_f = 32$ and τ_{sfs} is still turned off. Figure 15 shows $\|(\bar{\alpha u}) - (\bar{\alpha u})_e\|_\infty$ for varying $\delta_f/\Delta x$ ranging from $\delta_f/\Delta x = 4$ to $\delta_f/\Delta x = 32$. We see from the graph that the results converge at $\delta_f/\Delta x = 16$. We obtain the order of convergence for varying δ_f/D . In order to highlight how interface sharpness affects the results, figure 16 shows the $\|(\bar{\alpha u}) - (\bar{\alpha u})_e\|_2$ and $\|(\bar{\alpha u}) - (\bar{\alpha u})_e\|_\infty$ error. All simulations are run at $CFL = 0.1$. The ratio of the filter width to the mesh width is fixed at $\delta_f/\Delta x = 16$. Currently we turn off τ_{sfs} and only consider the advection and forcing term. Looking at the order of convergence for the L_2 and L_∞

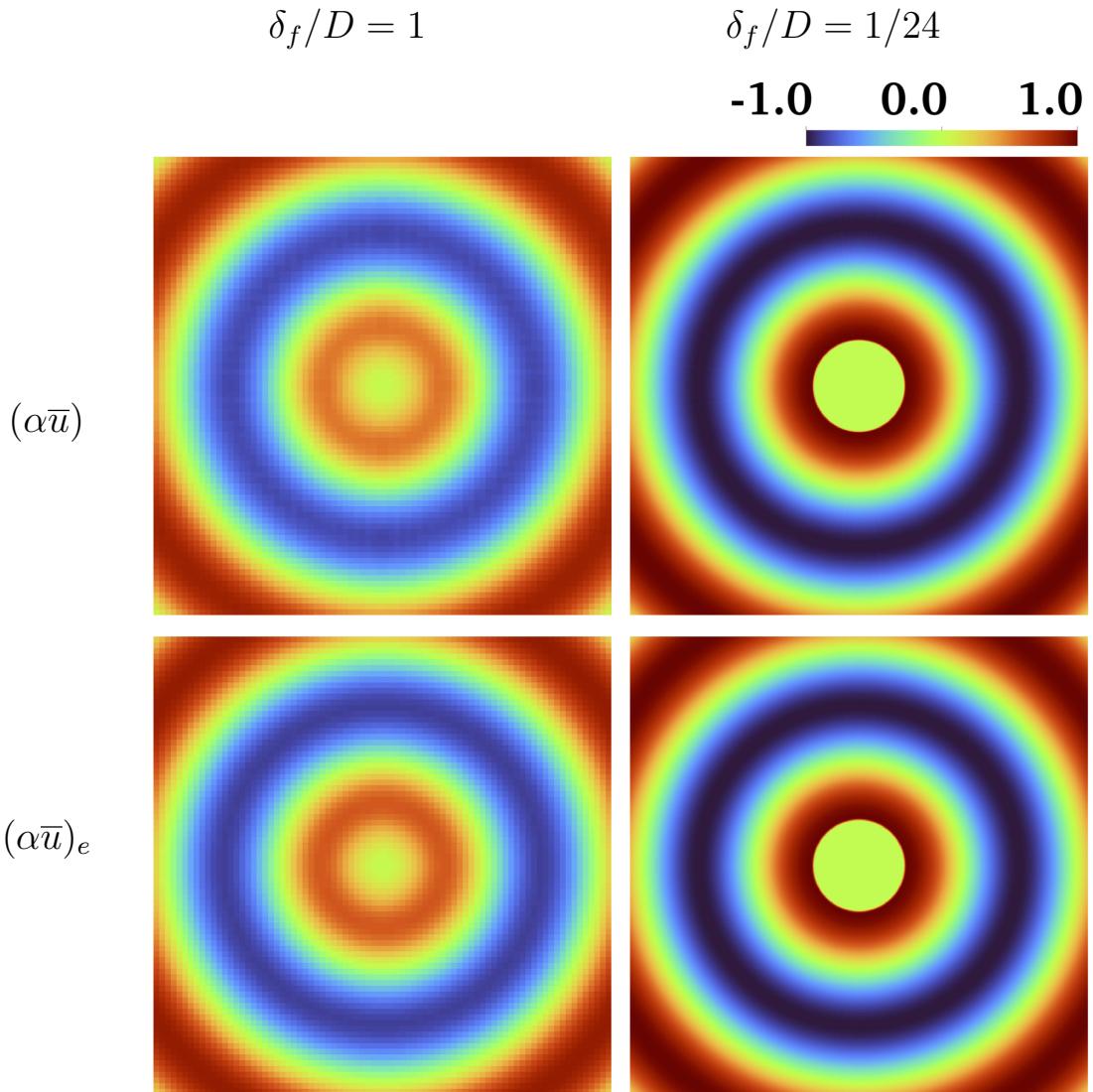


Figure 13: Isocontours of the numerical solution $(\alpha \bar{u})$ and the analytically filtered solution $(\alpha \bar{u})_e$ for filter width $\delta_f/D = 1$ (left) and $\delta_f/D = 1/24$ (right). The results are shown for phase $T = 1/4$. The simulations are run at $CFL = 0.25$ and the subgrid mesh is kept such that $\delta_f/\Delta x_f = 32$. The main grid spatial resolution is set at $\delta_f/\Delta x = 16$.

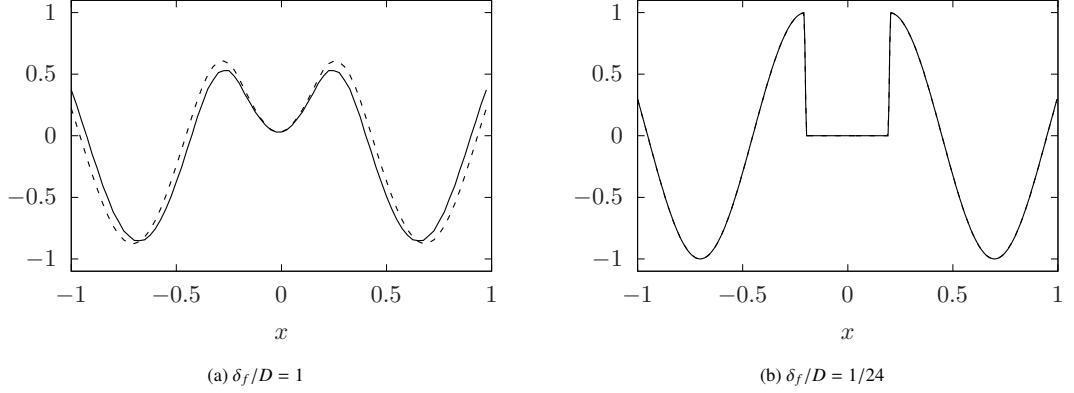


Figure 14: Value of $(\bar{\alpha}u)$ (—) vs x at $T = 1/4$. For comparison we also plot the analytically filtered solution $(\bar{\alpha}u)_e$ (---). The result shown is a line cut horizontally and centered in the vertical axis. We show the results for a coarse filter width of $\delta_f/D = 1$ and a fine filter width of $\delta_f/D = 1/24$.

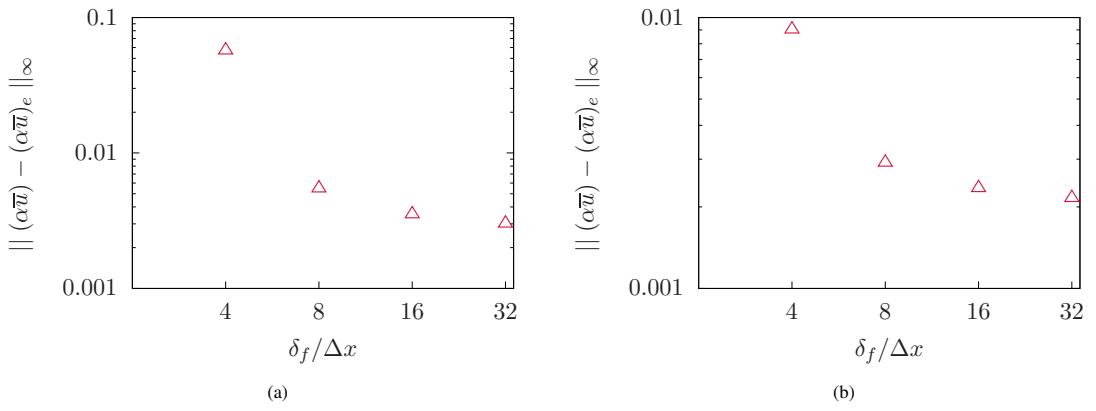


Figure 15: $\|(\bar{\alpha}u) - (\bar{\alpha}u)_e\|_\infty$ norm for varying $\delta_f/\Delta x$ at $(\delta_f/D)^{-1} = 1/12$. (a) shows the error when the interface forcing is at its maximum and (b) show the error when the interface forcing is at its minimum. We show that the error converges to stationary value as we increase the number of grid points.

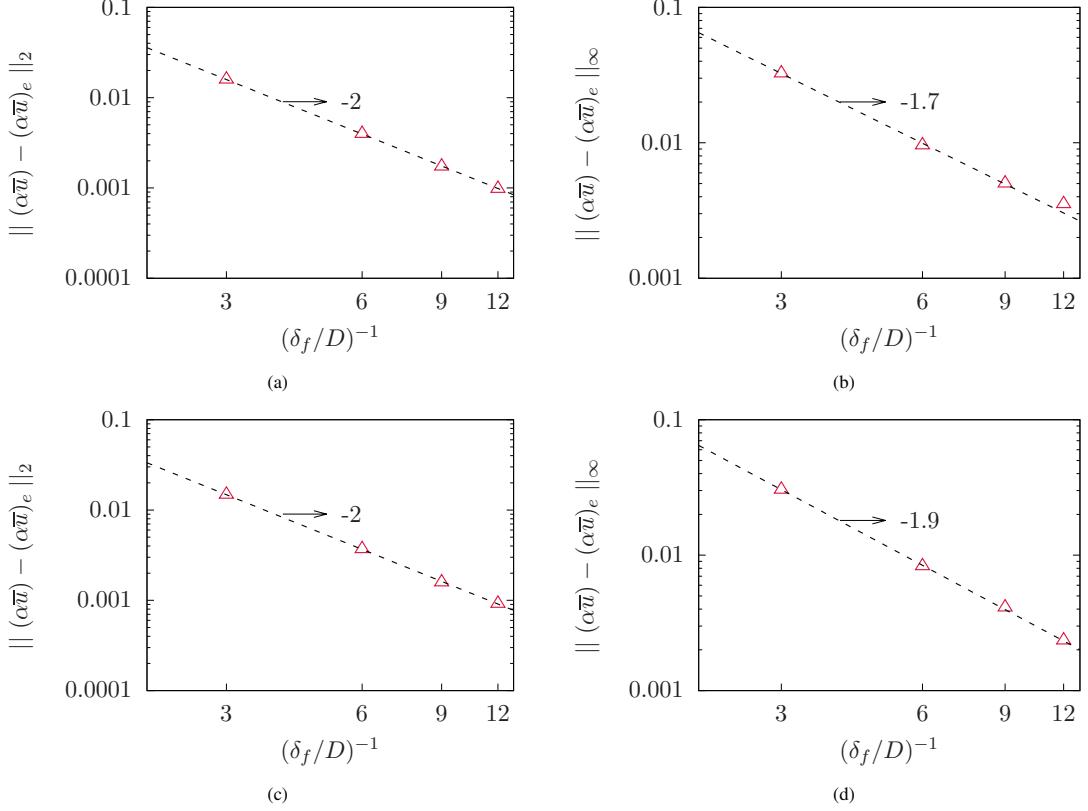


Figure 16: $\|(\alpha\bar{u}) - (\alpha\bar{u})_e\|_2$ and $\|(\alpha\bar{u}) - (\alpha\bar{u})_e\|_\infty$ norm when the interface forcing F_I is at its absolute maximum (top) and when it is at its absolute minimum (bottom). The results are shown for increasing $(\delta_f/D)^{-1}$ while keeping $\delta_f/\Delta x = 16$. All simulations are run at CFL = 0.1.

norm error we observe that the slope obtained is similar to that of the order of convergence for τ_{sfs} . This shows that the VF-IB method is a stable second order method and performs well even under spatially coarse conditions.

The contribution of the sub-filter scale term τ_{sfs} on the numerical solution depends on δ_f/D . For large δ_f/D values, the contribution from τ_{sfs} is large enough that including it significantly improves the accuracy of the solution compared to the filtered analytical solution $(\alpha\bar{u})_e$. Figure 17 shows the isocontours of $(\alpha\bar{u})$ when τ_{sfs} is turned on and when we ignore it. We also show the filtered analytical solution $(\alpha\bar{u})_e$ for comparison. The results are shown at a coarse filter width of $\delta_f/D = 1$. The subgrid mesh is kept constant at $\delta_f/\Delta x_f = 32$ and the main grid resolution is $\delta_f/\Delta x = 16$. The cases are run at CFL = 0.1. From the results, we observe that when we include τ_{sfs} the error is significantly lower than when it is excluded. This shows that for coarser filter widths, including τ_{sfs} helps improve the accuracy of the solution. In order to show this in a quantitative manner, figure 18 shows the value of $\|(\alpha\bar{u}) - (\alpha\bar{u})_e\|_\infty$ as a line cut horizontally and centered in the vertical direction. We show the results, both when τ_{sfs} is turned off and then it is turned on. This is further proof that inclusion of τ_{sfs} at coarser filter widths helps produce a more accurate numerical solution.

Figure 19 shows how $\|(\alpha\bar{u}) - (\alpha\bar{u})_e\|_\infty$ varies with changing δ_f/D . The results are run at $\delta_f/\Delta x = 16$ and a subgrid resolution of $\delta_f/\Delta x_f = 32$. In order to highlight how refining the interface grid resolution $D/\Delta x$ affects the solution we run an extra case at $D/\Delta x = 32$ at $\delta_f/D = 1$ when τ_{sfs} is turned on. The results show that as we reduce δ_f/D , the contribution from τ_{sfs} significantly reduces. We observe that at $\delta_f/D = 1/4$ the error is negligible enough. At this resolution or finer resolutions, τ_{sfs} can be ignored without any reduction in the accuracy of the solution. Furthermore, we show the importance of resolving the interface grid resolution. At a $\delta_f/D = 1$ we witness that the error is significantly lower at $D/\Delta x = 32$ than it is at $D/\Delta x = 16$. We therefore conclude that for coarser filter resolutions modeling τ_{sfs} will improve the accuracy, however as we refine δ_f/D , we can ignore the term. Furthermore, refining the grid resolution $D/\Delta x$ will also improve the accuracy of the solution.

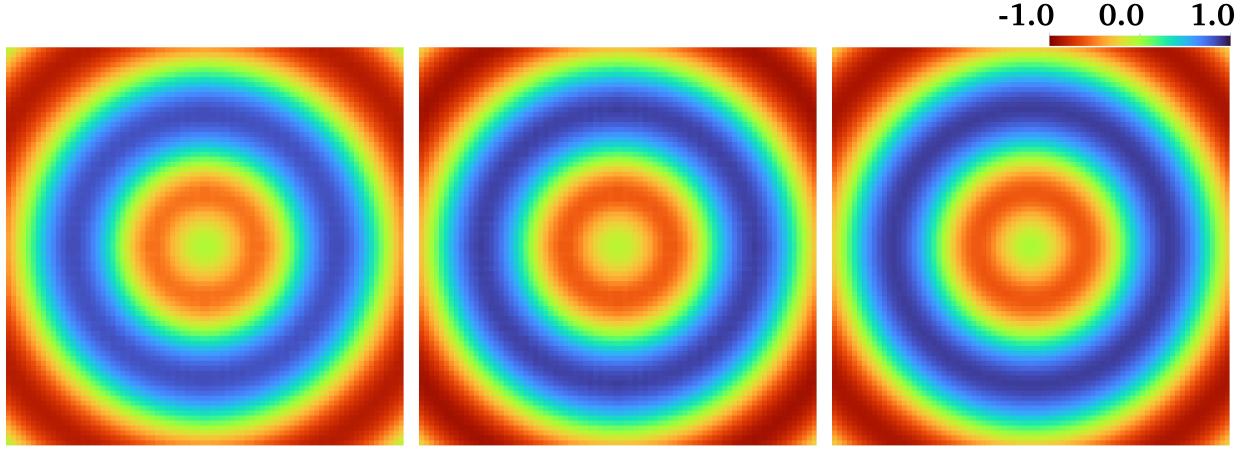


Figure 17: Isocontours of $(\alpha\bar{u})$ when τ_{sfs} is turned off (left) and then τ_{sfs} is turned on (middle). We also show the filtered analytical solution $(\alpha\bar{u})_e$ for comparison (right). The results are shown at the filter width of $\delta_f/D = 1$ to show how including τ_{sfs} makes the numerical solution closer to the filtered analytical solution at coarse filter sizes.

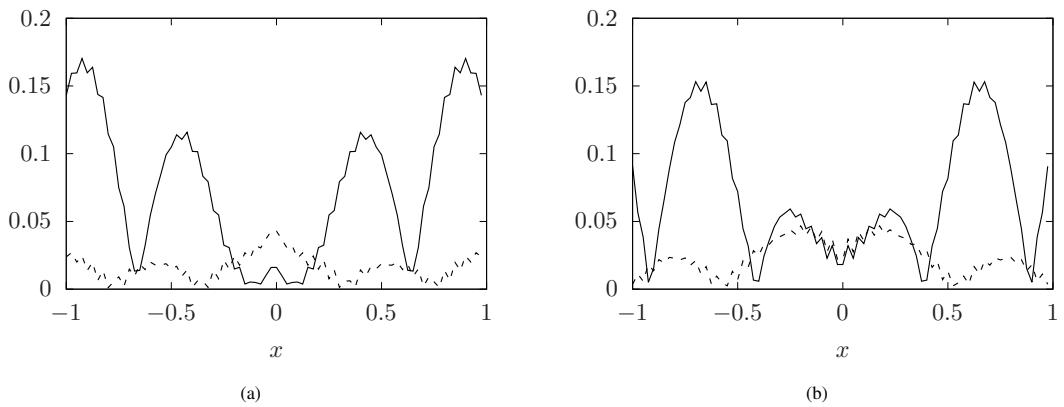


Figure 18: Value of $\|(\alpha\bar{u}) - (\alpha\bar{u})_e\|_\infty$ vs x at (a) $T = 1/4$ and (b) $T = 1/2$. The result shown is a line cut horizontally and centered in the vertical axis. We show the result when τ_{sfs} is turned off (—) and when τ_{sfs} is turned on (····). We run the simulations at $\delta_f/D = 1$.

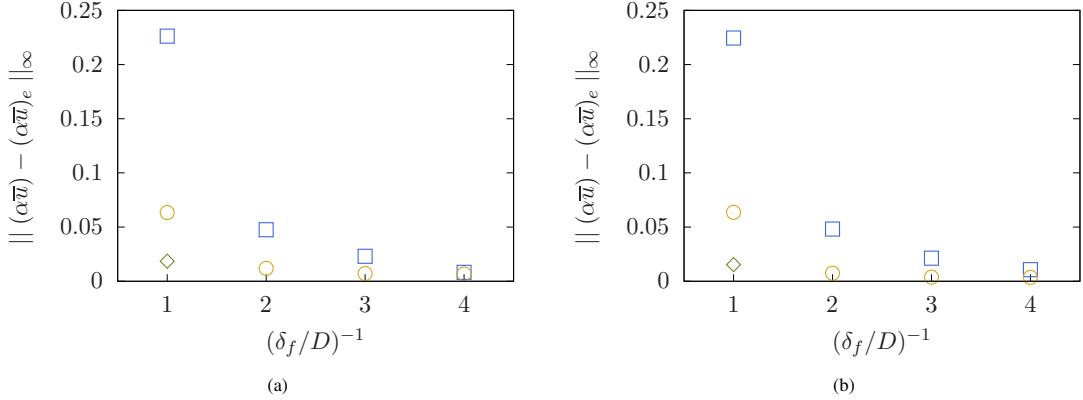


Figure 19: $\|(\alpha\bar{u}) - (\alpha\bar{u})_e\|_\infty$ norm for varying $(\delta_f/D)^{-1}$ at $\delta_f/\Delta x = 16$. (□) correspond to the result when τ_{sfs} is turned off and (○) correspond to the result when τ_{sfs} is turned on. In order to show how $D/\Delta x$ affects the solution, (◇) shows the results at $D/\Delta x = 32$ when τ_{sfs} is turned on. (a) shows the error when the interface forcing is at its maximum and (b) show the error when the interface forcing is at its minimum. We show that for coarse filter widths, turning on τ_{sfs} helps significantly reduce the error. This error can be further reduced by resolving the interface grid resolution $D/\Delta x$.

6. Conclusion

In this paper we extend the novel Volume-Filtered Immersed Boundary method (VF-IB) by Dave et al. [5] through characterizing the unclosed term τ_{sfs} and the forcing term F_I for varying spatial parameters. This helps us better understand how unclosed terms play a role in the VF-IB method and how varying filter width affects the quality of the solution.

In order to show this, we take the case of a varying coefficient hyperbolic equation. In a purely theoretical fashion, through the process of volume filtering [1] we are able to convert the point-wise equations into filtered transport equations. The immersed boundary plays a role in the forcing term, which appears as a surface integral on the right hand side of the transport equation. We show that using the volume-filtering process within an immersed boundary framework, we are able to theoretically transform the PDE that involves boundary conditions on surfaces into a filtered equation to solve, where the surface integrals define the boundary conditions at the interface. This process is not tied to any discretization and is viable for any topologically complex boundary. Furthermore, through the process of volume-filtering arise the sub-filter scale terms, namely τ_{sfs} . This paper studies the effect that interface sharpness has on the magnitude of τ_{sfs} , in comparison to the terms that are present in the volume-filtered transport equations. Furthermore, we also explore how τ_{sfs} contributes to the solution for varying δ_f/D .

We first study the separate terms shown in the filtered transport equations in a purely analytical fashion as shown in section 5.1. We see that for finer interface sharpness values the contribution of τ_{sfs} is small, and can be considered to be negligible. Hence, the term can be ignored without any significant reduction in accuracy of the solution. As the interface sharpness becomes coarse this contribution increases, making τ_{sfs} significant. At these coarse resolutions a modeling approach may be warranted in order to output high orders of accuracy. Furthermore, we take a look at the order of convergence for τ_{sfs} and show that the sub-filter scale term scales with δ_f^2 , making the VF-IB method theoretically second order. We then conduct an a posteriori analysis as shown in section 5.2 and perform the simulations numerically using the VF-IB method. We initially ignore the effects of τ_{sfs} and turn off the term. Again, we conduct simulations from the coarsest interface sharpness of $\delta_f/D = 1/3$ and gradually increase the sharpness to $\delta_f/D = 1/12$. The results are compared to the analytically filtered solution and the results are as expected. We observe a second order rate of convergence that is in line with what was observed during the a priori analysis. For sharp interface cases such as at $\delta_f/D = 1/12$. The results are comparable to the second order results in the cut-cell method by Brady and Livescu [3]. Hence, we can say that for a sufficiently sharp interface, τ_{sfs} can be ignored while still keeping obtaining a high order of accuracy.

In order to obtain high level of accuracy at coarser interface sharpness, the sub-filter scale terms would need to be modeled. To show this we take cases with varying δ_f/D from $\delta_f/D = 1$ to $\delta_f/D = 1/4$ and include τ_{sfs} as part

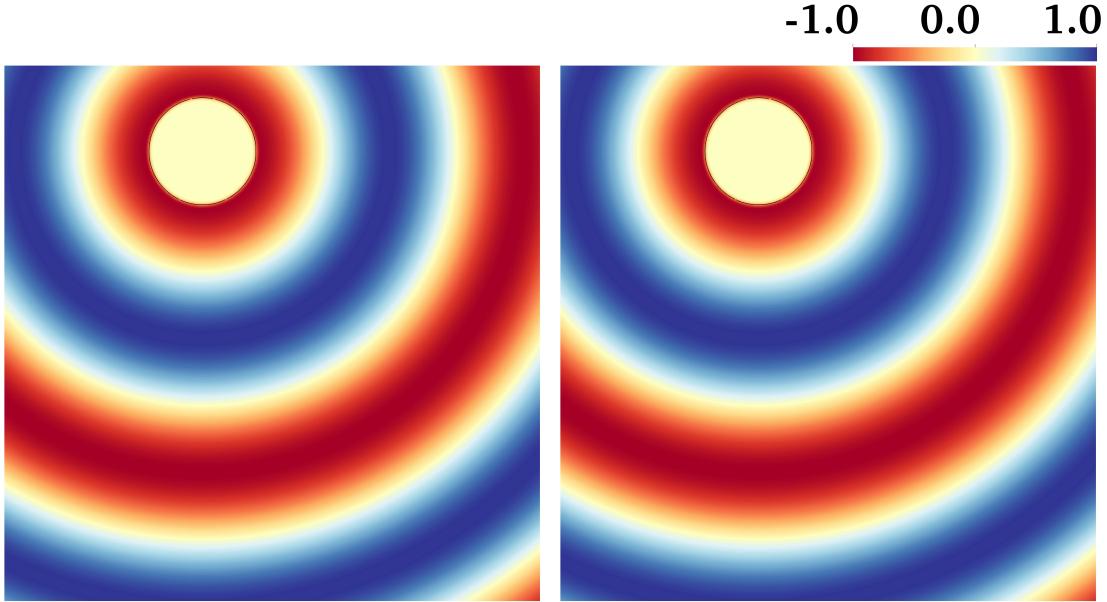


Figure A.20: Isocontours of the numerical solution ($\alpha\bar{u}$) and the analytically filtered solution ($\alpha\bar{u}_e$) for filter width $\delta_f/D = 1/12$. The results are shown for phase $T = 3/4$. The simulations are run at $CFL = 0.1$ and the subgrid mesh is kept such that $\delta_f/\Delta x_f = 32$. The main grid spatial resolution is set at $\delta_f/\Delta x = 16$.

of the solution. From this we conclude that at extremely coarse resolution, τ_{sfs} has a contribution large enough that it cannot be ignored and would have to be modeled in order to achieve good accuracy. However, as we refine δ_f/D this value quickly reduces in magnitude and does not play a major role in the solution. Keep in mind, these resolutions are coarse enough that they can run at an extremely cheap computational cost and obtaining sub-filter scale models can help achieve robustness of the method while still obtaining rather good accuracy against the filtered analytical solution. Furthermore, we also shed light on the interface grid resolution $D/\Delta x$. We show that increasing $D/\Delta x$ significantly improves the results. A combination of accurate τ_{sfs} models together with reasonable grid resolution can provide a robust method that is accurate while keeping the computational cost low.

Finally, we show that this methodical volume-filtering process can be conducted on any arbitrary hyperbolic PDE that involves boundary conditions on surfaces. The process allows us to solve for the transport equations on a cartesian grid using lagrangian markers as a tool to construct the interface. This allows us to solve around any topologically complex interface that can be moving or static.

Acknowledgement

The authors acknowledge support from the US National Science Foundation (award #2028617, CBET-FD and EEC-GOALI award #2216969). Computing resources were provided by Research Computing at Arizona State University.

Appendix A. Convergence analysis for embedded circle placed at different locations

Here we show the results when the circle is placed at 4 different locations. The center locations are such that, $l_1 = (0.5, 0.5)$ is within the top left quadrant. The other three locations are picked at random such that, $l_2 = (0.5, -0.34)$, $l_3 = (-0.61, -0.43)$ and $l_4 = (-0.26, 0.68)$. Figure A.20 shows the isocontours of $(\alpha\bar{u})$ in comparison to $(\alpha\bar{u})_e$ when the circle is placed at location l_4 . The simulation is run at a filter resolution of $\delta_f/D = 1/12$ and a spatial resolution of $\delta_f/\Delta x = 16$. We can visually compare the results and observe that the change in position of the circle does not affect the accuracy of the simulation. To further show this, figure A.21 plots the values of $(\alpha\bar{u})$ and $(\alpha\bar{u})_e$ by ‘cutting’ across

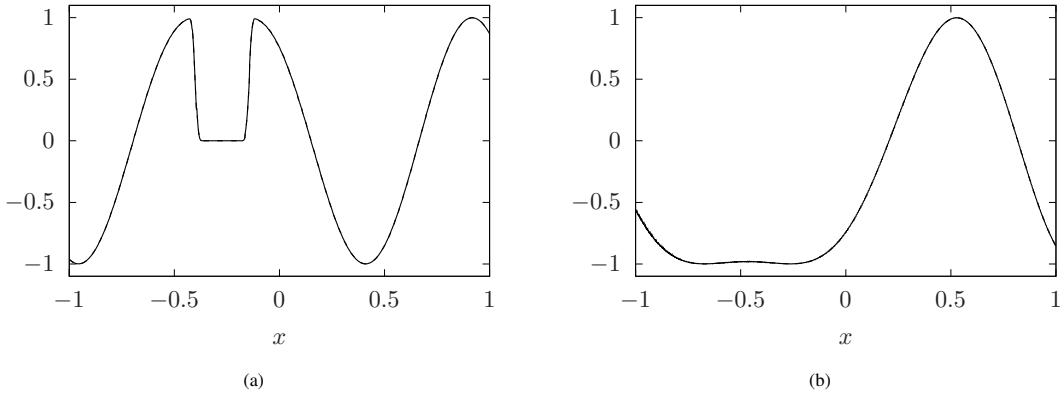


Figure A.21: Value of $(\alpha \bar{u})$ (—) vs x at $T = 3/4$. For comparison we also plot the analytically filtered solution $(\alpha \bar{u})_e$ (---). The results shown are two lines picked at random that cut across the domain. The first line goes from points $(-1, 0.6) \rightarrow (1, 0.4)$ (left) and the second line goes from points $(-1, 0.25) \rightarrow (1, -0.53)$ (right). We show the results for a filter resolution of $\delta_f/D = 1/12$.

the domain using two randomly selected lines. The results show that the numerical solution matches the analytically filtered solution well, thus making the method independent of the immersed boundary placement.

Finally we perform a convergence analysis. For comparison we also show the results when the circle is in the center of the domain, $l_0 = (0, 0)$. Figure A.22 shows that we get a second order convergence and the results are similar even when the IB is placed in different locations of the domain. Hence confirming that the second order convergence is not dependent on the location of the immersed boundary.

References

- [1] T. B. Anderson and Roy Jackson. Fluid Mechanical Description of Fluidized Beds. Equations of Motion. *Industrial & Engineering Chemistry Fundamentals*, 6(4):527–539, November 1967.
 - [2] W. Kyle Anderson. A Grid Generation and Flow Solution Method for the Euler Equations on Unstructured Grids. *Journal of Computational Physics*, 110(1):23–38, January 1994.
 - [3] P. T. Brady and D. Livescu. Foundations for High-Order, Conservative Cut-Cell Methods: Stable Discretizations on Degenerate Meshes. *Journal of Computational Physics*, 426:109794, February 2021.
 - [4] Wim-Paul Breugem. A second-order accurate immersed boundary method for fully resolved simulations of particle-laden flows. *Journal of Computational Physics*, 231(13):4469–4498, May 2012.
 - [5] Himanshu Dave, Marcus Herrmann, and M. Houssem Kasbaoui. The volume-filtering immersed boundary method. *Journal of Computational Physics*, 487:112136, August 2023.
 - [6] R. Glowinski, T. W. Pan, T. I. Hesla, D. D. Joseph, and J. Périaux. A Fictitious Domain Approach to the Direct Numerical Simulation of Incompressible Viscous Flow Past Moving Rigid Bodies: Application to Particulate Flow. *Journal of Computational Physics*, 169(2):363–426, May 2001.
 - [7] Wyatt James Horne and Krishnan Mahesh. A massively-parallel, unstructured overset method to simulate moving bodies in turbulent flows. *Journal of Computational Physics*, 397:108790, November 2019.
 - [8] Tobias Kempe and Jochen Fröhlich. An improved immersed boundary method with direct forcing for the simulation of particle laden flows. *Journal of Computational Physics*, 231(9):3663–3684, May 2012.
 - [9] A. R. Koblitz, S. Lovett, N. Nikiforakis, and W. D. Henshaw. Direct numerical simulation of particulate flows with an overset grid method. *Journal of Computational Physics*, 343:414–431, August 2017.
 - [10] Ming-Chih Lai and Charles S. Peskin. An Immersed Boundary Method with Formal Second-Order Accuracy and Reduced Numerical Viscosity. *Journal of Computational Physics*, 160(2):705–719, May 2000.
 - [11] Kun Luo, Zeli Wang, Jianren Fan, and Kefa Cen. Full-Scale Solutions to Particle-Laden Flows: Multidirect Forcing and Immersed Boundary Method. *Physical Review E*, 76(6):066709, December 2007.
 - [12] D. J. Mavriplis. Unstructured Grid Techniques. *Annual Review of Fluid Mechanics*, 29(1):473–514, 1997.
 - [13] N A Patankar. A Formulation for Fast Computations of Rigid Particulate Flows. *Center of Turbulence Research Annual Briefs 2001*, pages 185–196, 2001.
 - [14] C S Peskin. The Fluid Dynamics of Heart Valves: Experimental, Theoretical, and Computational Methods. *Annual Review of Fluid Mechanics*, 14(1):235–259, 1982.
 - [15] Charles S Peskin. Flow Patterns around Heart Valves: A Numerical Method. *Journal of Computational Physics*, 10(2):252–271, October 1972.
 - [16] Charles S. Peskin. The immersed boundary method. *Acta Numerica*, 11:479–517, January 2002.

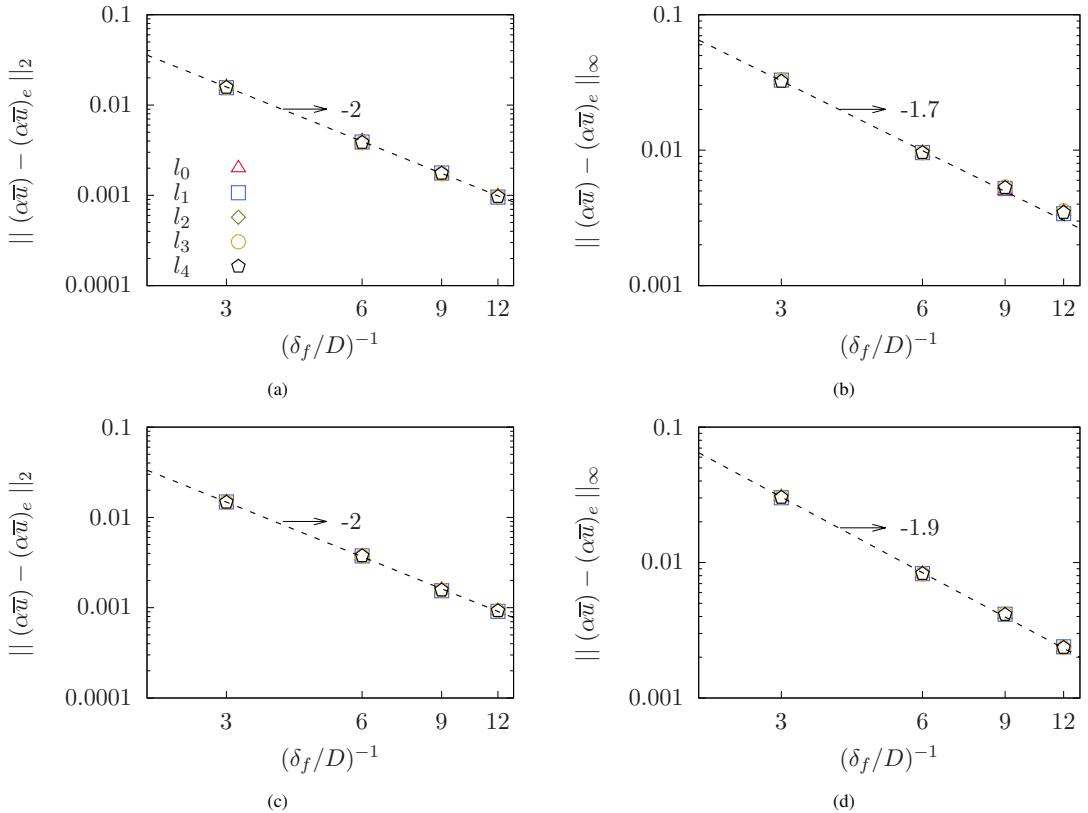


Figure A.22: $\|(\alpha\bar{u}) - (\alpha\bar{u})_e\|_2$ and $\|(\alpha\bar{u}) - (\alpha\bar{u})_e\|_\infty$ norm when the interface forcing F_I is at its absolute maximum (top) and when it is at its absolute minimum (bottom). The results are shown for increasing $(\delta_f/D)^{-1}$ while keeping $\delta_f/\Delta x = 16$. All simulations are run at CFL = 0.1. We show the results for the circle placed at 4 different locations in comparison to the original position.

- [17] Alexandre M Roma, Charles S Peskin, and Marsha J Berger. An Adaptive Version of the Immersed Boundary Method. *Journal of Computational Physics*, 153(2):509–534, August 1999.
- [18] M. H. Saadat, F. Bösch, and I. V. Karlin. Semi-Lagrangian lattice Boltzmann model for compressible flows on unstructured meshes. *Physical Review E*, 101(2):023311, February 2020.
- [19] Silvio Tschigale, Tobias Kempe, and Jochen Fröhlich. A non-iterative immersed boundary method for spherical particles of arbitrary density ratio. *Journal of Computational Physics*, 339:432–452, June 2017.
- [20] Markus Uhlmann. An Immersed Boundary Method with Direct Forcing for the Simulation of Particulate Flows. *Journal of Computational Physics*, 209(2):448–476, November 2005.
- [21] Salih Ozen Unverdi and Grétar Tryggvason. A Front-Tracking Method for Viscous, Incompressible, Multi-Fluid Flows. *Journal of Computational Physics*, 100(1):25–37, May 1992.
- [22] A. W. Vreman. A staggered overset grid method for resolved simulation of incompressible flow around moving spheres. *Journal of Computational Physics*, 333:269–296, March 2017.