Combining the Volume of Fluid Method with the Drift Flux Model within the OpenFOAM Framework – The vvpfFoam Solver

Jon Elvar Wallevik

ICI Rheocenter

Innovation Center Iceland

April 8, 2020

Report No. NMI 20-01

Abstract

This report describes the theoretical background behind a concrete casting solver named `vvpfFoam`. The work is done within the OpenFOAM framework [1] and can (at least) be compiled on versions from 2.2.0 to 2.2.2 (see Appendix B). OpenFOAM is licensed under the GNU General Public License (Version 3) and as such, the same applies to the solver `vvpfFoam`. That is, you may use, distribute and copy the solver `vvpfFoam` under the terms of GNU General Public License version 3, which is displayed in Appendix C, or (at your option) any later version.

The aim of the solver is to calculate the coarse aggregate distribution as a function of time, with the objective to predict the effect of segregation by gravity as well as by the shear (rate) induced particle migration. The solver encompasses two theories. The first one is the Volume of Fluid Method (VOF), while the second is the Drift Flux Model (DFM). It should be noted that the starting developing point of `vvpfFoam` is the `interFoam` solver and as such the current solver could have been named `interDFMFoam`, `dfmInterFoam`, `driftFluxInterFoam` or similar.

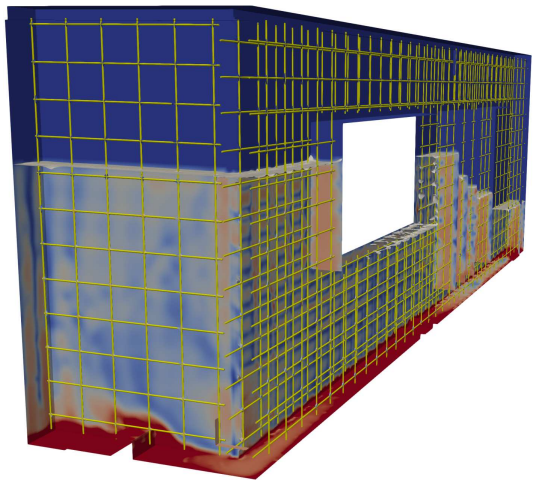
One of the aims with the solver is to simulate operational problems related to uncertainties in casting predictions of fresh concrete (i.e. of newly mixed concrete). These problems are segregation by gravity as well as segregation by shear (rate) induced particle migration. Improved prediction accuracy of fresh concrete flow allows for the design of more complex and durable concrete structures and additionally allows ready-mix plants to investigate the effect of stability variation during casting of a large/difficult structure. This is extremely important because uneven aggregate distribution can increase the local porosity and thus the permeability of concrete. Varying content of mortar causes heterogeneous shrinkage and creep in a given element. Moreover, high heterogeneity will increase the probability that these phenomena yield high internal stress gradients and thus cracking.

Although the solver is designed with fresh concrete in mind, it is not limited to this. It can be used with other high viscous materials which behaves in a laminar manner (i.e. in a non-turbulent manner). Example of this would be the flow of aluminum particles submerged in viscous oil. Also, other types of cement based materials can be analyzed, like the flow of sand particles submerged in cement paste (i.e. investigation of flow and segregation of mortar).

Innovation Center Iceland
Arleynir 2-8
112 Reykjavik
Iceland
ISBN: 978-9935-463-59-3

Many of the simulations done in this work were performed on resources provided by the Icelandic High Performance Computing Centre at the University of Iceland.

This work was supported by the Icelandic Research Fund (IRF) – grant number 163382-05.



◇ **To cite this report**:

Jon Elvar Wallevik, Combining the Volume of Fluid Method with the Drift Flux Model within the OpenFOAM Framework - The vvpfFoam Solver, Innovation Center Iceland, Report No. NMI 20-01, 2020 (ISBN: 978-9935-463-59-3).

Contents

Chapter 1

Introduction

1.1 The Solver

A multiphase transient simulator, named `vvpfFoam`, has been developed that models the dynamics of multiple fluid phases during casting of viscous fluid like the fresh concrete (i.e. newly mixed concrete). The development is realized within the OpenFOAM framework, which uses the finite volume method (FVM). One of the aims with the solver is to simulate operational problems related to uncertainties in casting predictions of fresh concrete. This includes the effect of the settlement of aggregates by gravity (i.e. segregation) as well as the effect of shear (rate) induced particle migration [2, 3, 4]. Improved prediction accuracy of transient multiphase flow allows for the design of more complex and durable concrete structures and additionally allows ready-mix plants to investigate the effect of stability variation during casting of a large/difficult structure.

The `vvpfFoam` encompasses two theories. The first one is the Computational Fluid Dynamics (CFD) of transient viscoplastic fluid with open (free) boundary, thus dividing the system between the atmospheric air and a mixture fluid (e.g. fresh concrete). This is what could be considered as a standard Volume of Fluid approach (VOF) [5]. This subject is treated in Chapter 3 (Page 26). The second theory is the implementation of field equation for particle distribution into the numerical framework to be able to calculate segregation/settling within the mixture fluid (e.g. segregation of fresh concrete), including the shear (rate) induced particle migration [2, 3, 4]. The approach used in treating the segregation is based on the Drift Flux Model (DFM[1]), which is derived from the so-called two-fluid model [6, 7]. This subject is treated in Chapter 2, Page 16.

The solver can be compiled on OpenFOAM 2.2.0 to 2.2.2 and is designed for high viscous fluid only (i.e. laminar flow) and thus turbulence is not included. Although designed with fresh concrete in mind, it can be used with other high viscous materials as well, e.g. aluminum particles submerged in viscous oil. As the atmospheric air has no real stress related interactions with the mixture (e.g. with the fresh concrete), the former is assumed to behave in a non-turbulent manner as well (i.e. atmospheric laminar flow). If turbulent

[1] In some literature, the term *"Drift Flux Method"* is used, rather than *"Drift Flux Model"*.

analysis is required for the mixture (e.g. small contaminant particles submerged in water), the user must add it to the solver.

To reiterate, in the `vvpfFoam` solver, a standard VOF approach is mixed with the DFM and the approach is described in Chapter 4, Page 31. The former calculates the flow of two fluids that do not generally intermix (immiscible) and thus usually have a clear boundary between them, e.g. concrete and atmospheric air. For the second part of the solver, where the DFM is applied, the phases are generally in an intermixed (miscible) state, e.g. coarse aggregates suspended in mortar. Three phases are involved in the solver, of which atmospheric air is the first phase (with a volume fraction α_2). The mixture fluid (volume fraction α_1), which could for example represent fresh concrete, is divided between a matrix phase (also, continuous phase) and a particle phase (also, dispersed phase), which constitute the second and the third phase, respectively (with volume fractions α_c and α_d). In addition to α_c and α_d, the solid concentration[2] of the continuous and the dispersed phases are also designated with $\beta_c = \alpha_c/\alpha_1$ and $\beta_d = \alpha_d/\alpha_1$, respectively (see Chapter 4).

1.2 Service Life of the Concrete Structure

The load carrying capacity and service life of concrete structure is very much dependent on the quality and success of concrete placement into formwork at jobsite [8, 9, 10, 11]. In recent years numerical modeling of concrete placement has showed great potentials to become an important tool for optimization of such process [12]. Only recently, researchers from various part of the world have started to work on such casting prediction tools using different CFD softwares [8]. But lot of work is still to be done to understand the large scale behavior of the involved flow processes [8], especially in terms of calculating the coarse aggregate concentration as a function of location and time [13]. In particular, variation in aggregate distribution can increase the local porosity and thus the permeability of concrete. This can also cause heterogeneous shrinkage and creep in a given concrete element. Moreover, high heterogeneity will increase the probability that this phenomenon yields high internal stress gradients and thus cracking with the reduction in load carrying capacity of the concrete structure as a result [11].

The Self Compacting Concrete (SCC) is a very fluid concrete and thus was expected to be the answer to casting problems. However, experience has shown that even for such type of material, there will always exist a formwork and steel bars configuration in which casting problems may occur [8]. Furthermore, these casting problems may not be fully resolved unless one can calculate the aggregate concentration as a function of time and location and especially its response to different types of obstacles.

[2] The term *"solid concentration"* is also designated as *"phase volume"* [2] and sometimes as *"volume fraction"* or *"solid fraction"*. All these terms will be used interchangeably in this work.

1.3 Casting of a Wall Section – An Example

Several different types of formwork geometries have been used for testing the `vvpfFoam` solver. An example of such is shown in Fig. 1.1 and consists of a wall section. The concrete is being pumped into the formwork from the left side and the diameter of the hose is about 20 cm. The inflow rate is such that it takes about 80 seconds to fill the formwork. The number of cells in the current case is about 1.8 million and calculations were performed on resources provided by the Icelandic High Performance Computing.

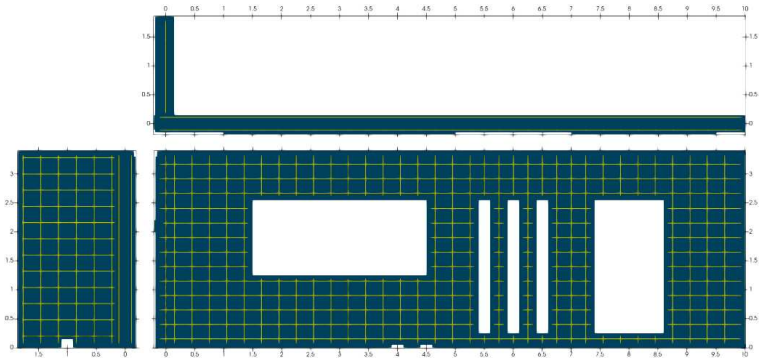


Figure 1.1: Geometry of the wall section (with reinforcing steel) used in some of the simulation tests in this work.

The length of the wall is 10 m, height is 3.4 m, thickness is 30 cm and the length of the small side wall (i.e. the one pointing into the overall structure) is 1.9 m. In the front (main) wall is a double layer steel reinforcement with cover of 34 mm, but in the side wall is a single layer reinforcement located at the wall center. The diameter of the rebar[3] is 12 mm, while the size of the reinforcement mesh is 250×250 mm.

In Fig. 1.2 the concrete is being pumped from the base of the formwork, i.e. in 34 cm height from the ground (see the arrow). Another case where the concrete is pumped from above is shown in Fig. 1.3, with a fall height of 2.1 m. The latter approach is more seldom used at jobsite, but is included to put a certain strain on the solver. In Figs. 1.2 and 1.3, the continuous phase consists of mortar/fine concrete (here, all materials below 11 mm in diameter), while the dispersed phase consists of coarse aggregates (in this case, the 11 - 16 mm aggregates).

The color bar shown in Figs. 1.2 and 1.3 applies to both illustrations (a) and (b) and represents the value β_{d} at the cell closest to the wall (i.e. `wallBetaD`). In accordance with the previous text, the colorbar describes the solid concentration of coarse aggregates (here, the 11 - 16 mm aggregate phase). The dark red color, namely $\beta_{\mathrm{d}} = 0.3$, represents high compaction (or concentration) of coarse aggregates, while the dark blue color, $\beta_{\mathrm{d}} = 0$, represents area that is completely absent of coarse aggregates. At such location, only mortar (i.e. fine concrete < 11 mm) remains. In the light brown color range, the solid concentration is close to $\beta_{\mathrm{d}} = 0.2$, which in this case means homogeneous concrete (this

[3] *Rebar* is short for reinforcing bar, and also known as reinforcing steel, steel bars, reinforcement steel or just reinforcement, among other designations.

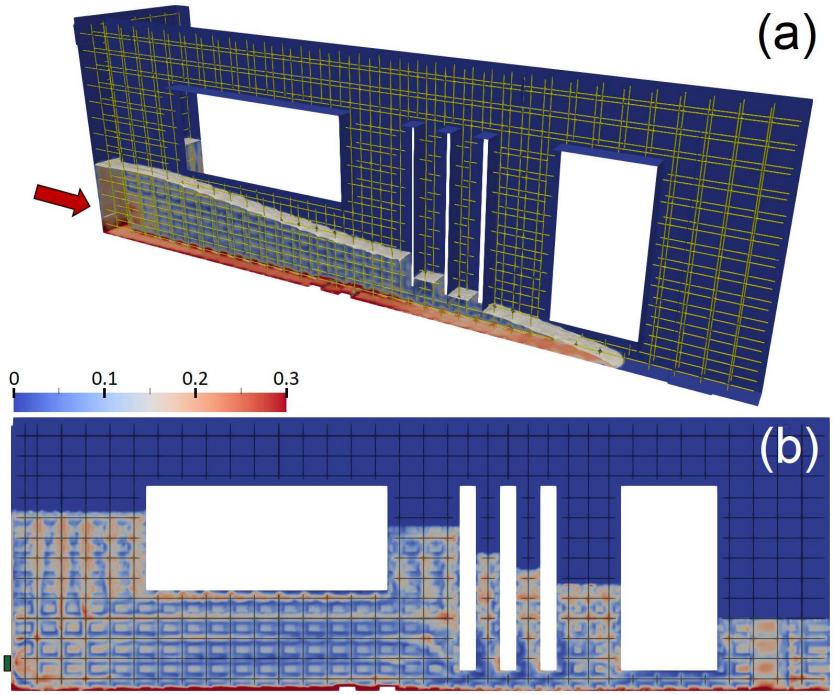


Figure 1.2: Solid concentration of coarse aggregates (i.e. phase volume), during the pumping of fresh concrete into a formwork from the ground (see the arrow) at 20 s (a) and 40 s (b) after start of pumping.

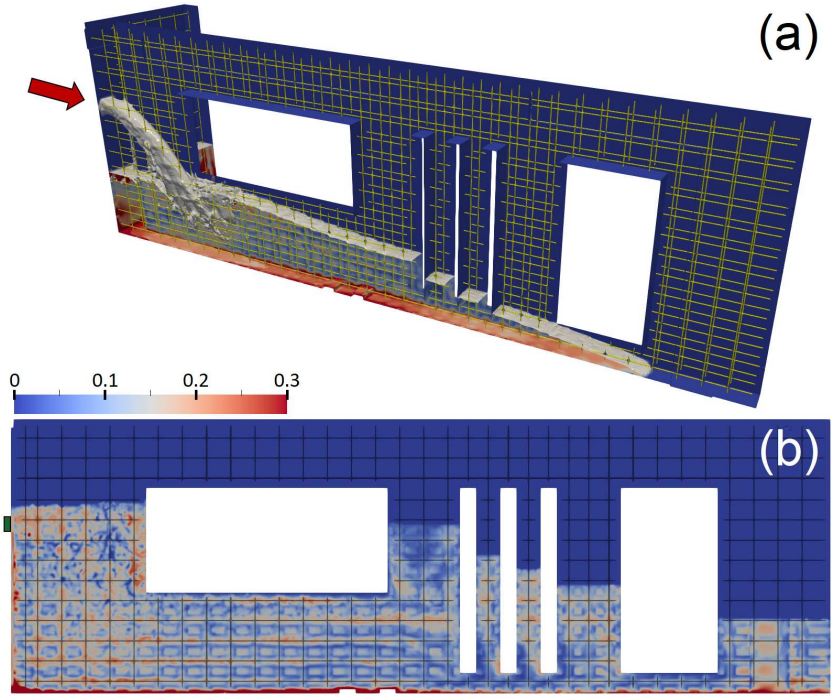


Figure 1.3: Solid concentration of coarse aggregates (i.e. phase volume), during pumping of fresh concrete into a formwork from above, with a fall height of 2.1 m (see the arrow).

value depends on the mixture proportions), or rather an initial state of concentration of coarse aggregates. Finally, the blue color above the concrete represents the atmospheric air, or α_2.

The same type of concrete is used in Figs. 1.2 and 1.3. It has a low apparent viscosity η_1 and is highly prone to segregation as is clearly visible with the red color at the bottom of each formwork. That is, settlement by gravity (i.e. "segregation") as well as by shear (rate) induced particle migration are allowed to occur simultaneously. These two processes are also allowed to affect the apparent viscosity $\eta_1 = \eta_1(\beta_\mathrm{d})$, which again affects the flow and thus the two pre-mentioned settlement types. Including the large segregation that is clearly visible at the base in Figs. 1.2 and 1.3, reinforcement shadows are also present near the rebars as a consequence of the particular concrete type used.

1.4 Testing the VOF Part of the Solver

As mentioned in Section 1.1, the `vvpfFoam` solver is a mixture of VOF and DFM. As a part of the code verification, the output of the current solver is compared with the outcome of the standard `interFoam` solver. Initially, the latter was used as a template in the beginning of the code development of the former. As the `interFoam` is a VOF solver only, the drift velocity $\mathbf{V}_\mathrm{dj}$ (see Chapter 6) must be set equal to zero in the `vvpfFoam` solver. Also, the apparent viscosity η_1 must be set to something that both solvers can use.

Here, a standard Bingham model is applied, with plastic viscosity of $\mu = 50\,\mathrm{Pa}\cdot\mathrm{s}$ and yield stress of $\tau_0 = 10\,\mathrm{Pa}$. For the `interFoam`, the density is set as $\rho_1 = 2300\,\mathrm{kg/m^3}$, while for the `vvpfFoam` the mixture density is implemented as $\rho_1 = \beta_\mathrm{d}\rho_\mathrm{d} + \beta_\mathrm{c}\rho_\mathrm{c} = 2700\,\mathrm{kg/m^3}\cdot 0.2 + 2200\,\mathrm{kg/m^3}\cdot 0.8 = 2300\,\mathrm{kg/m^3}$ (see Eq. (4.14), Page 34).

1.4.1 A "Cake Break" Problem

In this case, a certain type of "*dam break*" problem is tested. The geometry is shown in Fig. 1.4, and consists of quarter of a "*cake*" about 45 cm in height and radius of 55 cm, which is released to flow by its own weight within a large box (90 cm x 1.4 m x 50 cm). In front of the cake are 16 small pillar obstacles about 12 cm in diameter, between which the material has to flow.

With the test setup shown in Fig. 1.4, the `vvpfFoam` manages to reproduce the `interFoam` results exactly. A demonstration of this is shown in Fig. 1.5b. More precisely, this case consists of two simulation results cut in half with the ParaView visual software. The `vvpfFoam` result is marked with "III", while the `interFoam` result with "IV". If any difference exists between the two results, such would be clearly visible where the two results meet in the middle. To demonstrate this point, an example of such difference is produced in Fig. 1.5a, with `interFoam`: The case marked with "II" the same result as marked "IV" in Fig. 1.5b, while the case marked with "I" is an `interFoam` result with 20% higher rheological values. In the early code development, such type of difference

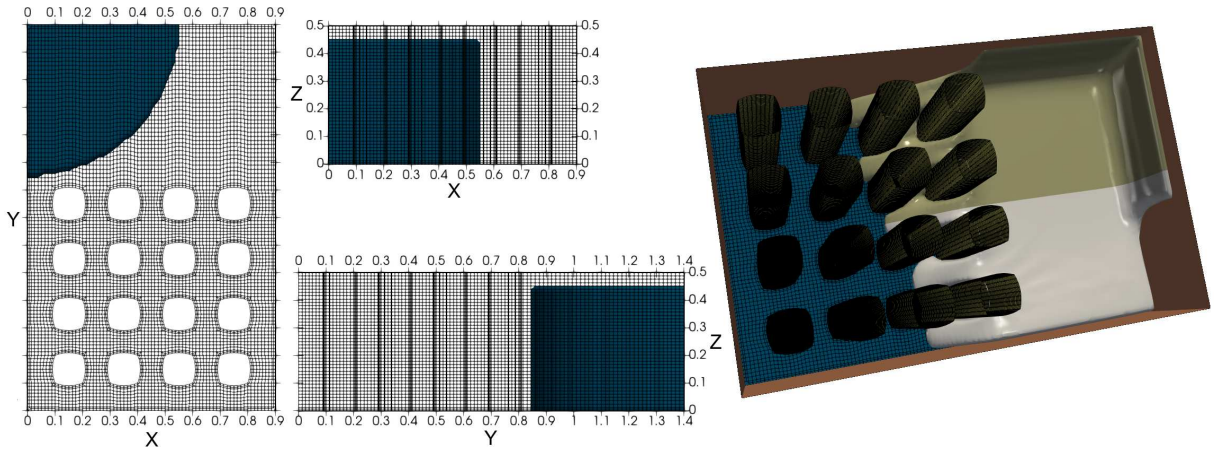


Figure 1.4: The geometry and mesh of the "cake break" problem (with 317,684 cells).

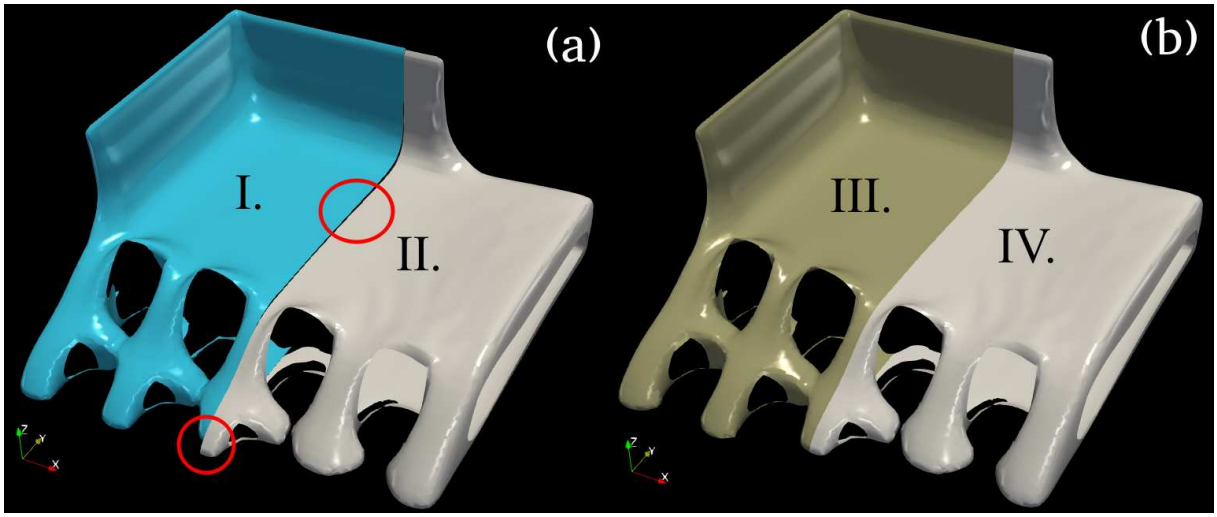


Figure 1.5: Simulation results with the setup in Fig. 1.4. To the left are two `interFoam` results using different rheological values (I and II), while to the right is a comparison of `vvpfFoam` (III) and `interFoam` (IV).

emerged between `vvpfFoam` and `interFoam`, however in smaller degree than demonstrated in Fig. 1.5a.

1.4.2 Complete Wall Section

In this section, the same formwork is used as in Fig. 1.1, however now without rebars. The results are shown relative to the center cross-section of the front wall as demonstrated in Fig. 1.6. The concrete is being pumped into the formwork from one side, in which the drop height is 2.1 m. As before, the inflow rate is such that it takes about 80 seconds to fill the formwork. Placing the hose at this height is done to put a certain strain on the solver. The number of cells in the current case is about 420,000.

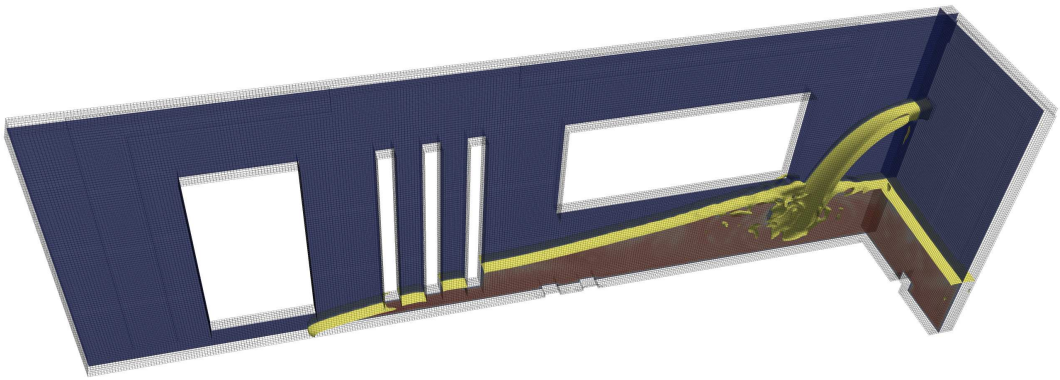


Figure 1.6: Geometry of the wall section used in the simulation tests.

Figs. 1.7a-f show the solid concentration of the overall mixture α_1 (i.e. of the fresh concrete). As before, the Bingham model is applied, with plastic viscosity of 50 Pa · s and yield stress of 10 Pa. Illustrations (a) (c) and (e) are results generated by `interFoam` and refer to 20, 40 and 60 seconds after start of pumping. Illustrations (b), (d) and (f) are results of α_1 at the same time points, generated by the `vvpfFoam`. As shown, the outcome of the two solvers are almost exactly the same.

As mentioned before, the concrete is being pumped into the formwork with a drop height about 2.1 m. This makes the flowing system more "volatile", especially near and around the hose. Any tiniest difference generated near the hose will evolve with the flow and thus grow into larger differences downstream.

When comparing the results in Figs. 1.7a-f, with carefully observation, one can see small differences in results between the two solvers. However, it should be noted that such difference is not observable when placing the hose near the base of the formwork as done in Fig. 1.2, nor was any difference observed for the case in Section 1.4.1.

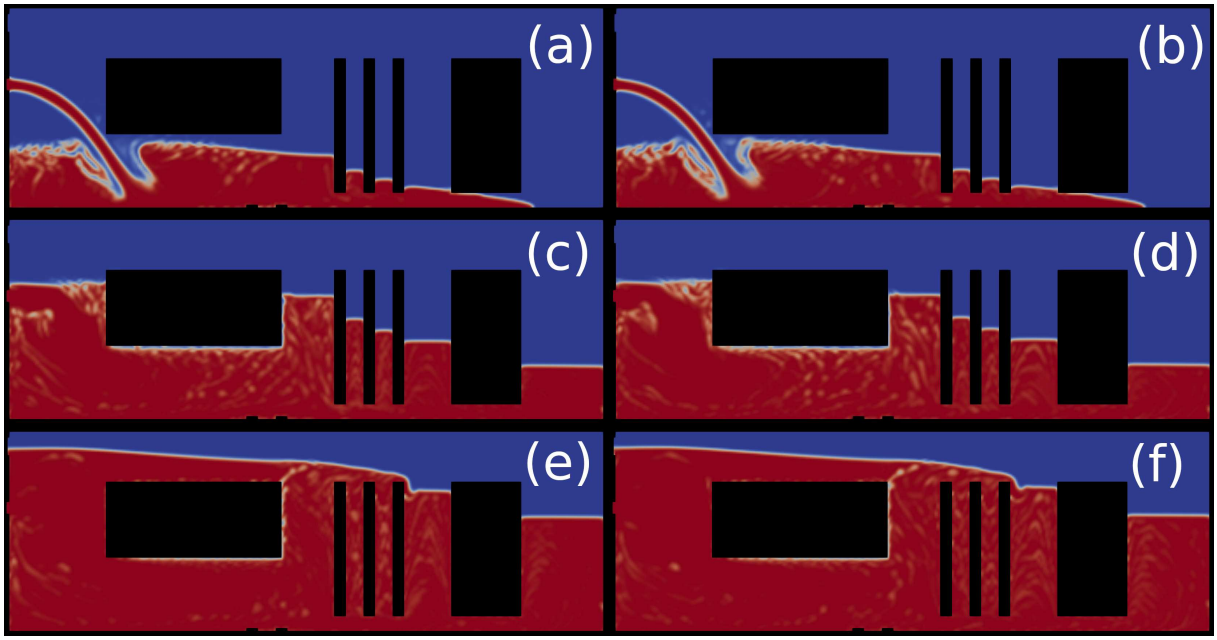


Figure 1.7: Comparison of a standard VOF solver `interFoam` (illustrations (a), (c) and (e)) with the `vvpfFoam` solver (illustrations (b), (d) and (f)).

1.5 Testing the DFM Part of the Solver

1.5.1 Settling by Gravity (Section 6.3)

In this section, the simulation setup is a vertical settling tank shown in Fig. 1.8. Its total height is 1 m in z direction and the width is 20 cm both in x and y directions. In this case, there is no velocity drive, i.e. no inflow, outflow, motion by gravity and so forth, meaning $\mathbf{U} = 0$. As shown in Fig. 1.8a, the mixture height is 0.8 m. The solid concentration at time $t = 0\,\mathrm{s}$ is $\beta_\mathrm{d} = 0.2$ and the maximum possible packing in this case is 0.4. The number of cells used in Fig. 1.8 is 2,560,000.

The settling velocity $\mathbf{V}_\mathrm{s}$ is calculated by Eq. (6.6) on Page 60, using $\mu_\mathrm{N} = 7.67\,\mathrm{Pa}\cdot\mathrm{s}$, $\rho_\mathrm{c} = 2200\,\mathrm{kg/m^3}$, $\rho_\mathrm{d} = 2700\,\mathrm{kg/m^3}$ and $D_\mathrm{a} = 13\,\mathrm{mm}$, which results in $-6\,\mathrm{mm/s}\,\mathbf{i}_\mathrm{z}$. By Section 6.1.4 (Page 60), this means a constant drift velocity of $\mathbf{V}_\mathrm{dj} = -6\,\mathrm{mm/s}\,\mathbf{i}_\mathrm{z}$. The implementation in `gravitySegregation.H` is as follows:

```
const dimensionedScalar constVsGR
(
    "constVsGR",
    dimensionSet(0,0,1,0,0,0,0),
    // scalar(1.0194e-4) // *g = -1 mm/s
    // scalar(3.0581e-4) // *g = -3 mm/s
    scalar(6.1162e-4) // *g = -6 mm/s
);
```

```
tmp<volVectorField> VsGR = mag(alpha1)*constVsGR*g;

#ifdef GRAVITY_SEGREGATION
VdjGR =
    slowDown2
    (
        alphaD, // alphaD, or betaD, depending on user preference!
        alphaDMIN,
        alphaDMAX
    )*(1.0*VsGR);
#else
VdjGR = zeroVelocity;
#endif
```

As the drift velocity $\mathbf{V}_{\mathrm{dj}}$ is constant, the outcome of this experiment is not dependent on mixture density ρ_1 nor on the apparent viscosity η_1.

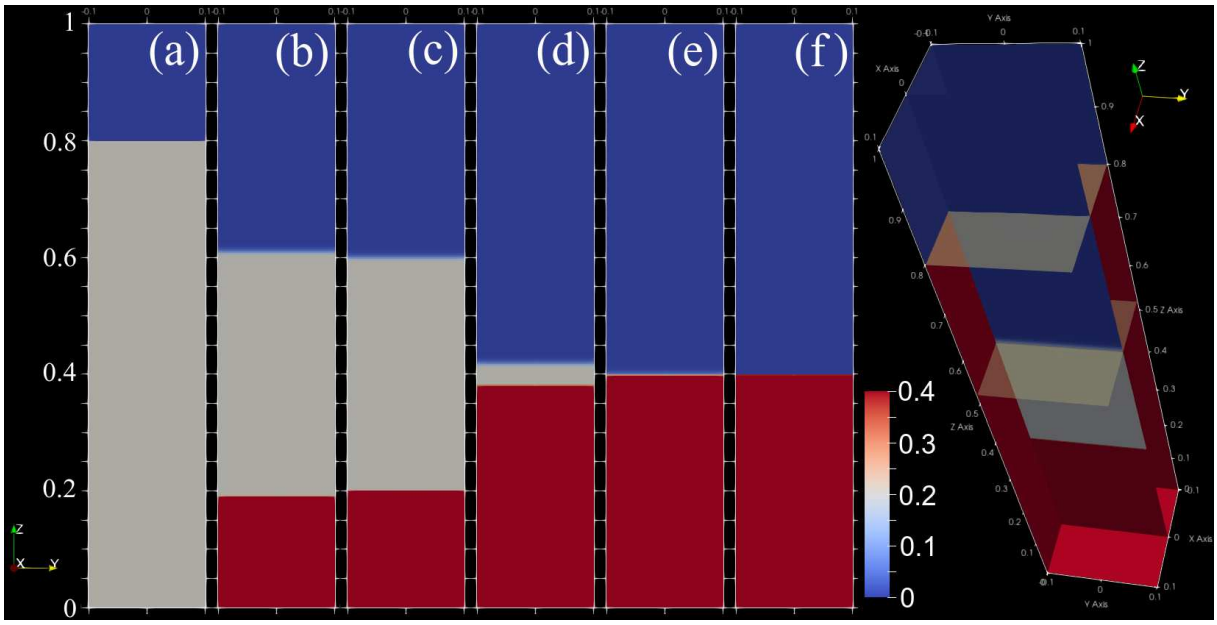


Figure 1.8: Solid concentration of the dispersed phase β_{d} at 0 s (a), 33.3 s (b), 35.0 s (c), 66.7 s (d), 70.0 s (e) and 72.0 s (f), respectively (with $\mathbf{R} = 0$, c.f. Section 4.9).

In Fig. 1.8a, the initial condition $\beta_{\mathrm{d}} = 0.2$ is shown, valid at time $t = 0\,\mathrm{s}$, while in subsequent figures (b), (c), (d), (e) and (f), are the simulation results for β_{d} at time 33.3, 35.0, 66.7, 70.0, and 72.0 s, respectively. The result in Fig. 1.8f applies also at 170 s, which is the end of simulation (i.e. the result at time $t = 72.0\,\mathrm{s}$ is the same as at time 170 s).

Assuming that the observed settling of $\mathbf{V}_{\mathrm{s}} = -6\,\mathrm{mm/s}\,\mathbf{i}_{\mathrm{z}}$ is equal to the drift velocity $\mathbf{V}_{\mathrm{dj}}$ during the whole simulation (see Section 6.1.4 on Page 60) and with mixture column of $L = 0.8\,\mathrm{m}$, the time it should take for the suspended particle to settle should be about $\Delta t = (L/2)/\left|\mathbf{V}_{\mathrm{s}}\right| = (0.4\,\mathrm{m})/(6\,\mathrm{mm/s}) = 66.7\,\mathrm{s}$. But as shown in Fig. 1.8e, with $\mathbf{V}_{\mathrm{dj}} = -6\,\mathrm{mm/s}\,\mathbf{i}_{\mathrm{z}}$, this condition does not occur until at 70 s, meaning a 5% time difference.

At the time of writing, it is unclear if this difference is due to an issue with the code or simply a natural difference between the drift velocity $\mathbf{V}_{\mathrm{dj}}$ and the observed settling $\mathbf{V}_{\mathrm{s}}$ (see Eq. (6.4)), or due to a theoretical solver difficulty in calculating static cases, i.e. a case with $\mathbf{U} = 0$.

1.5.2 Settling by Shear Induced Particle Migration (Section 6.4)

To check the solver relative to settling by shear (rate) induced particle migration, the numerical experiment done by Fang and Phan-Thien [14] is reproduced. The theory applied is in accordance with Section 6.4, Page 63, using the apparent viscosity of Krieger and Dougherty[4], $\eta_1 = \mu(\alpha_{\mathrm{d}})$, where $\mu(\alpha_{\mathrm{d}})$ is given by Eq. (5.12), Page 54. Here, $\mu(0)$ is set equal to $1\,\mathrm{Pa}\cdot\mathrm{s}$. The apparent viscosity is implemented in `apparentViscosity.H` (through `return viscous_6()`), reproduced with the following code ($\alpha_{\mathrm{d}} = \varphi =$ `varPhi`, c.f. the last paragraph in Section 5.1):

```
tmp<volScalarField> viscous_6
(
    mag(alpha1)*
    (
        mu*pow(mag(scalar(1) - varPhi/scalar(0.68)),scalar(-1.82))
    )
  + mag(scalar(1) - alpha1)*eta2
);
```

In [14], a coaxial cylinders rheometer is used, with a rotating inner cylinder. The dimensionless criteria consists of $R_{\mathrm{i}}/R_{\mathrm{o}} = 0.25$, where the term R_{i} represents the radius of the inner cylinder and R_{o} radius of the outer cylinder. To accommodate this criteria, the inner cylinder is set as $R_{\mathrm{i}} = 8.0375\,\mathrm{mm}$, while the outer cylinder as $R_{\mathrm{o}} = 32.1500\,\mathrm{mm}$. The total height of the rheometer is set as $h = 56.2625\,\mathrm{mm}$. The overall geometry of the rheometer is shown in Fig. 1.9.

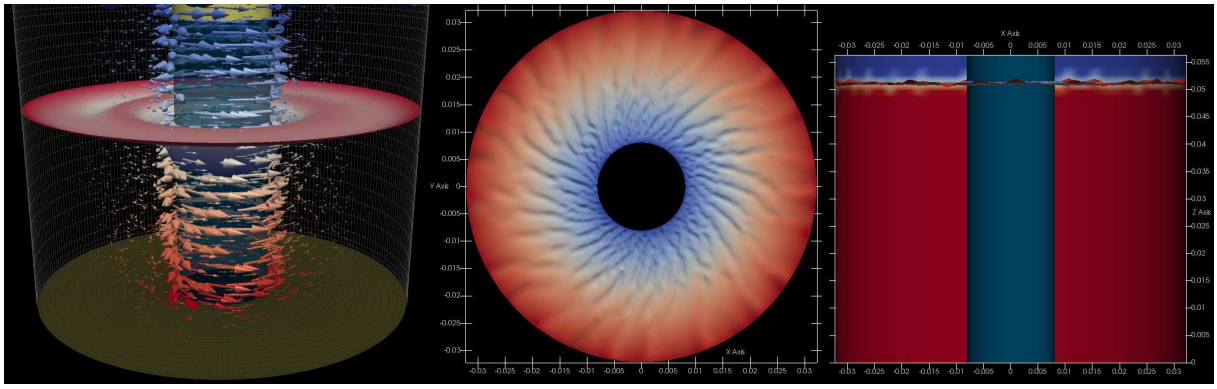


Figure 1.9: The overall geometry of the rheometer used in the current test.

[4]The difference between α_1, α_{d} and φ will not be completely clear until reading the whole report. Thus for the current text, assume $\alpha_{\mathrm{d}} = \beta_{\mathrm{d}} = \varphi$.

The test material consists of small inert neutrally buoyant particles with almost mono-sized particle size distribution, with diameter of $D_a = 1350\,\mu$m [14] (radius $a = D_a/2$). The corresponding dimensional number is $D_a/R_o = 0.042$. The maximum packing fraction is $\varphi_m = 0.68$ while the intrinsic viscosity is set as $[\eta] = 1.82$ [14] (see also Eq. (5.12) about φ_m and $[\eta]$). The density of the suspended particle is equal $\rho_d = 1188\,\mathrm{kg/m^3}$ and with a neutrally buoyant suspension, then $\rho_c = \rho_d = \rho_1$. The solid concentration is initially uniformly distributed at $\alpha_d = 0.5$. The inner cylinder is set to rotate at time $t = 0\,$s with the angular velocity of $\omega = 1\,\mathrm{rad/s}$. Although the theory of Section 6.4 is being used, the equation used in this test is not Eq. (6.8) (Page 63), but as always, Eq. (4.20) (Page 35).

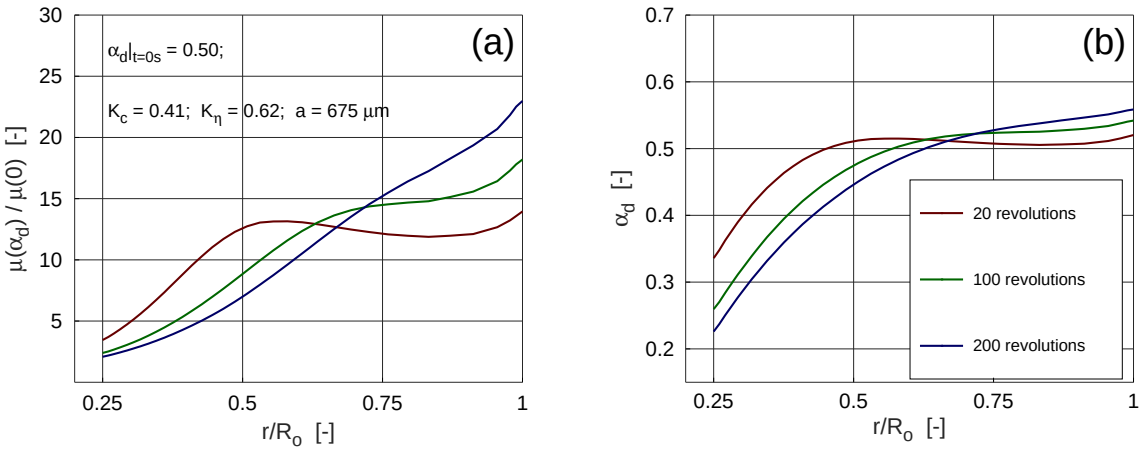


Figure 1.10: Dimensionless viscosity $\eta_1/\mu(0) = \mu(\alpha_d)/\mu(0)$ (a) and solid concentration α_d (b) as a function of dimensionless radius r/R_o, on completion of different inner cylinder revolutions (at the height $z = 2.5\,$cm).

Fig. 1.10a shows a plot of the dimensionless viscosity $\eta_1/\mu(0) = \mu(\alpha_d)/\mu(0)$ as a function of dimensionless radius r/R_o at the height of $z = 2.5\,$cm. Likewise, Fig. 1.10b shows the plot of the solid concentration α_d as a function of r/R_o at the same height. The legend in the latter applies for both illustrations and shows the corresponding number of turns. More precisely, the first plot applies after 20 revolutions of the inner cylinder, while the last plot after 200 revolutions. The results shown in Fig. 1.10b are an exact match of the simulation results given by Fig. 4 in [14].

It should be noted that although the number of cells in this case is only about 230,000, the calculation took about 10 days using one computer node (24 cores). This is a rather long calculation time and is probably due to instability, which could be attributed to how the term $\nabla\eta_1/\eta_1$ is currently evaluated in the solver, i.e. when calculating Eq. (6.14) on Page 64. At the time of writing, there was not sufficient time to investigate this further, but the user might have to change how the term $\nabla\eta_1/\eta_1$ is calculated.

1.6 Conservation of Material Volume

For the case in Fig. 1.8, the total amount of mixture is $0.2\,\mathrm{m} \cdot 0.2\,\mathrm{m} \cdot 0.8\,\mathrm{m} = 32\,\mathrm{l}$ and the amount of dispersed phase β_d is $32\,\mathrm{l} \cdot 0.2 = 6.4\,\mathrm{l}$. In Fig. 1.11 are shown the volume conservation for α_1 and α_d, for this case. The mixture concentration α_1 is solved by Eq. (4.23), Page 36, while the concentration for the dispersed phase α_d is solved by Eq. (4.20), Page 35. As shown, change in either α_1 or α_d is basically nonexistent. Also shown in Fig. 1.11 is the volume change in β_d, but as demonstrated with Eq. (4.8), Page 33, this term is calculated and not solved. Moreover, as discussed in Section 4.7, there can be abnormal changes in this value, especially at the interface between air and mixture that is mostly of little importance. The point is, that the changes in β_d shown in the right illustration of Fig. 1.8 is much less important relative to the changes in α_d.

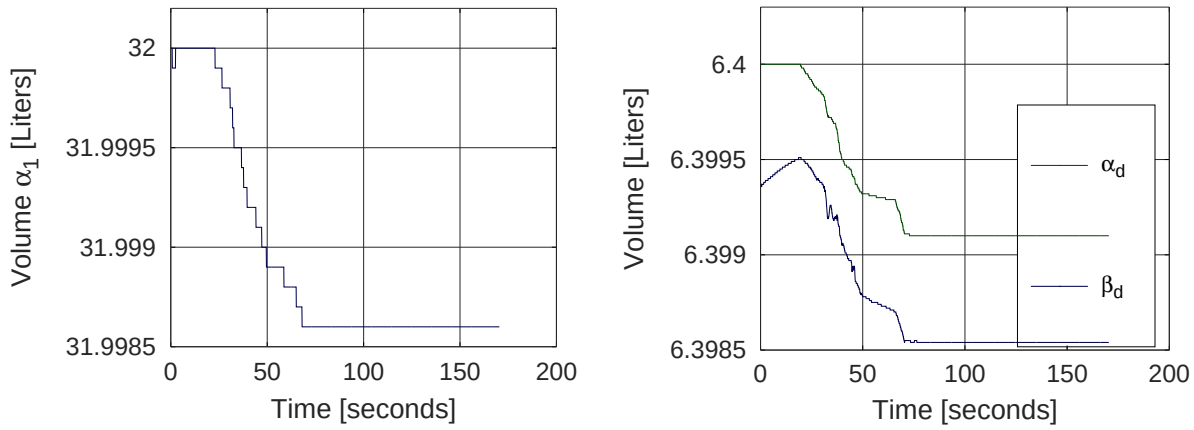


Figure 1.11: Volume conservation for α_1 (left) and α_d (right), for the case in Fig. 1.8.

For the above case, the mixture velocity is zero $\mathbf{U} = 0$. To examine the conservation of α_1 and α_d for a moving mixture, the cake break problem shown in Section 1.4.1 is examined (see Figs. 1.4 and 1.5). Here, a constant drift velocity of $\mathbf{V}_\mathrm{dj} = -3\,\mathrm{mm/s}\,\mathbf{i}_\mathrm{z}$ is applied and the corresponding settling is shown in Fig. 1.12.

In Fig. 1.13 are shown the volume conservation for α_1 and α_d, for the case of Fig. 1.12. Here, the conservation is slightly less than applies for the case of Fig. 1.8, or 0.25% for α_1 and 0.47% for α_d for the whole simulation time.

The calculation of compressibility for this case (by Eqs. (4.66) and (4.67)) is shown in Fig. 4.4, Page 48, in connection with discussion of the so-called pressure equation.

1.7 Known Issues

1. For closed system, the volume of α_d is not necessarily 100% conserved, c.f. Fig. 1.13. This has to do with the term "$(\alpha_\mathrm{d}\rho_\mathrm{c}/\rho_1)\mathbf{V}_\mathrm{dj}$" in Eq. (4.20), Page 35, which can behave as a source term.

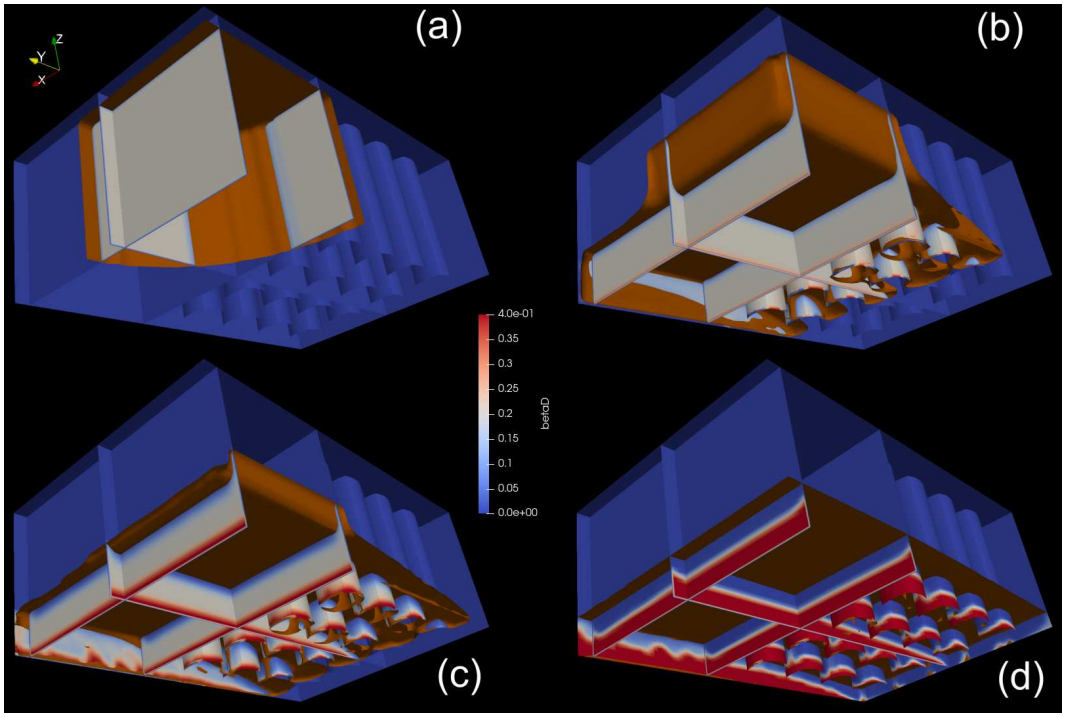


Figure 1.12: Solid concentration of the dispersed phase β_d at 0 s (a), 3 s (b), 10 s (c) and 30 s (d), respectively (with $\mathbf{R} = 0$, c.f. Section 4.9).

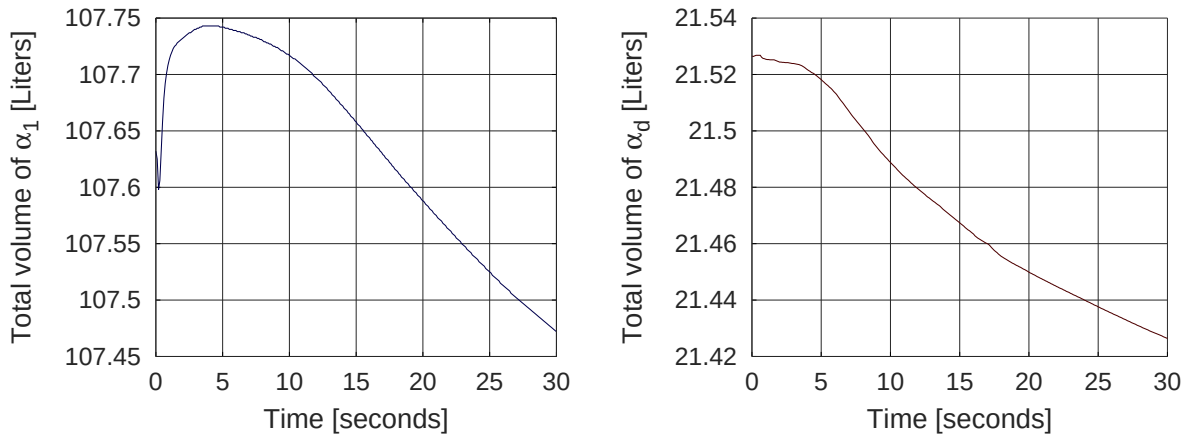


Figure 1.13: Volume conservation for α_1 (left) and α_d (right), for the case in Fig. 1.12.

2. For closed system, the volume of α_1 is in some cases not 100% conserved, c.f. Fig. 1.13. In such cases, putting the drift velocity $\mathbf{V}_{\mathrm{dj}}$ equal to zero results in a better volume conservation. Although the drift velocity $\mathbf{V}_{\mathrm{dj}}$ is not directly present in Eq. (4.23), Page 36, it will affect this equation through the changes in ρ_1 (see Eq. (4.14)), which might bring about an artificial source. Setting $\rho_1 = 1$ in Eq. (4.23) with $\mathbf{V}_{\mathrm{dj}} \neq 0$, results in better conservation. This can be done by commenting the line `#define ALPHA1RHO_SOLVE` in the source file `macroDefinitions.H` and recompile the solver.

3. During a pure vertical settlement (see the case in Section 1.5), the first three bottom cells are simultaneously filled with α_{d} when using `MULES::implicitSolve`[5]. After that, subsequent cells are filled one by one, as expected. To resolve (or patch) this unfortunate behavor, the first row of cells must be divided into three for a given case, as shown in the right illustration of Fig. 1.14. These three cells then behave as one, relative to α_{d}.

4. A custom version of the `settlingFoam` was created to make comparison with the `vvpfFoam`. The customization generally consisted of implementing the same drift velocity and apparent viscosity as used in the `vvpfFoam` solver, as well as disabling turbulence in the former. The case setup was the same in both cases, to the extent possible. Unfortunately, after the modifications, the `settlingFoam` solver always crashed when starting the filling of α into the second row of cells (see Fig. 1.14a). Up to that point, both solvers behaved alike (however, only when the first row of cells are split into three rows for `vvpfFoam` as discussed in item 3 above). Thus, to date, there is no complete comparison with the `settlingFoam` solver.

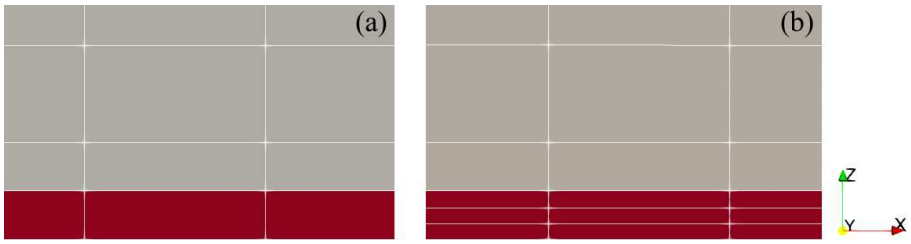


Figure 1.14: Comparison of results by `settlingFoam` (a) with `vvpfFoam` (b).

Most certainly, as with all source codes, there are other unknown issues beyond what is mentioned here. Since this solver is open and licensed under the GNU General Public License (as applies with OpenFOAM), the user has the opportunity to repair current and future issues. The user can modify the code, add new capabilities and otherwise enhance it to the specification needed. However, before committing to such a task, it is important to read and understand this document. Mixing VOF with DFM is not straightforward. This solver should not be confused with capability of other existing solvers within

[5]The first four bottom cells are simultaneously filled with α_{d} when using `MULES::explicitSolve`.

the OpenFOAM framework, like the `multiphaseInterFoam`, `interMixingFoam` or the `twoPhaseEulerFoam`. The current solver deals with the treatment of immiscible fluids (i.e. fluids that do not intermix), in combination with the treatment of miscible fluids (i.e. fluid that do intermix). In addition to the intermixture of phases, the solver has capability to allow for slip between phases, which is an important aspect to allow for segregation by gravitational settling and/or by other means, like by the shear (rate) induced particle migration.

Chapter 2

Drift Flux Model – DFM

2.1 Introduction

The Volume of Fluid Method (VOF) will be treated in Chapter 3. This particular method calculates the flow of two fluids that do not generally intermix (immiscible) and thus usually have a clear boundary between them, e.g. fresh concrete and atmospheric air. However, for the current topic, namely the Drift Flux Model (DFM), the phases are generally in an intermixed (miscible) state, e.g. coarse aggregates suspended in mortar.

Although there exist various documents and reports about the DFM, these are often fragmented with missing in-between mathematical derivations and less relevant to the current topic. Because of this, a whole chapter is dedicated to the subject in this report.

2.2 Two-Fluid Method

The starting point for the formulation of the Drift Flux Model (DFM) in Section 2.3, is the Multi-Fluid Method, using two phases. With only two phases involved, the latter method is also known as the Two-Fluid Method. The Multi-Fluid Method solves mass and momentum equations for each phase. The basic equations of the Two-Fluid Method are shown in Sections 2.2.2 and 2.2.3, and follows the representation that has previously been given in [15, 16].

2.2.1 Fundamental Relations

The primary variable in the Multi-Fluid Method is the volume fraction[6] of phase k, which is represented with the term[7] β_k and defined in Eq. (2.1). In this equation, the variable V_m designates the (local) volume of the mixture, i.e. volume of all phases combined, while

[6] To reiterate Footnote 2, the term *"volume fraction"* is also designated as *"phase volume"* and sometimes as *"solid concentration"* [2]. All these terms will be used interchangeably in this work.

[7] Commonly, the volume fraction of phase k is rather designated with the term α_k (instead of β_k), but because of the complexity of the current work, the term α needs to be reserved for later usage.

the term V_k represents the volume of phase k within this mixture volume (i.e. $V_\mathrm{k} \leq V_\mathrm{m}$). More specifically, the following always applies

$$\beta_\mathrm{k} = \frac{V_\mathrm{k}}{V_\mathrm{m}} \quad \wedge \quad V_\mathrm{m} = \sum_{\mathrm{k}=1}^{\mathrm{n}} V_\mathrm{k} \quad \Rightarrow \quad \sum_{\mathrm{k}=1}^{\mathrm{n}} \beta_\mathrm{k} = 1 \tag{2.1}$$

As shown above, summarizing the volume of all phases, namely V_k, gives the mixture volume V_m. From this, by summarizing the volume fraction β_k of each phase k results in the value 1.

In this work, the mixture volume V_m is also represented with the term V_1 and the reason for doing this will be clear in Section 4.2 (i.e. $V_\mathrm{m} \equiv V_1$).

The density of phase k is represented with ρ_k, while m_k is its mass, meaning $\rho_\mathrm{k} = m_\mathrm{k}/V_\mathrm{k}$. From this, the following relations are obtained

$$\beta_\mathrm{k}\rho_\mathrm{k} = \frac{V_\mathrm{k}\rho_\mathrm{k}}{V_\mathrm{m}} = \frac{m_\mathrm{k}}{V_\mathrm{m}} \tag{2.2}$$

The mixture density is represented with ρ_m, while $m_\mathrm{m} = \sum m_\mathrm{k}$ is its mass. The relationship between density, volume and mass for the mixture is $\rho_\mathrm{m} = m_\mathrm{m}/V_\mathrm{m}$. From this, the following relations are obtained

$$\rho_\mathrm{m} = \frac{m_\mathrm{m}}{V_\mathrm{m}} = \frac{\sum_{\mathrm{k}=1}^{\mathrm{n}} m_\mathrm{k}}{V_\mathrm{m}} = \sum_{\mathrm{k}=1}^{\mathrm{n}} \frac{m_\mathrm{k}}{V_\mathrm{m}} = \sum_{\mathrm{k}=1}^{\mathrm{n}} \beta_\mathrm{k}\rho_\mathrm{k} \tag{2.3}$$

The last part in the above equation was obtained with help from Eq. (2.2). In this work, the mixture density ρ_m and mass m_m are also represented with the terms ρ_1 and m_1, respectively. The reason for doing this will be clear in Section 4.2 (i.e. $\rho_\mathrm{m} \equiv \rho_1 \wedge m_\mathrm{m} \equiv m_1$).

2.2.2 Mass Conservation of Each Phase k

The mass conservation of each phase k (i.e. equation for the volume fraction of phase k) is given by [15] (see also [16])

$$\frac{\partial \beta_\mathrm{k}\rho_\mathrm{k}}{\partial t} + \nabla \cdot (\beta_\mathrm{k}\rho_\mathrm{k}\mathbf{V}_\mathrm{k}) = \Gamma_\mathrm{k} \tag{2.4}$$

The term Γ_k is the rate of mass generation of phase k and $\mathbf{V}_\mathrm{k}$ is the center of mass velocity of phase k. That is, if the phase k is composed of N particles[8], each of mass $m_{\mathrm{k,i}}$, and velocity of $\mathbf{V}_{\mathrm{k,i}}$, the velocity of phase k is given by (see [3], Section 2.2)

$$\mathbf{V}_\mathrm{k} = \frac{\sum_{\mathrm{i}=1}^{\mathrm{N}} m_{\mathrm{k,i}}\mathbf{V}_{\mathrm{k,i}}}{\sum_{\mathrm{i}=1}^{\mathrm{N}} m_{\mathrm{k,i}}} \quad \wedge \quad m_\mathrm{k} = \sum_{\mathrm{i}=1}^{\mathrm{N}} m_{\mathrm{k,i}} \tag{2.5}$$

[8] That is, particles, molecules or whatever that is relevant for the physical system to be investigated.

It should be clear that Eq. (2.5) is never explicitly used in the current document and serves only as a philosophical foundation[9].

The term $\mathbf{V}_\mathrm{m}$ represents the center of mass velocity of the mixture (also, "mixture velocity") and is defined in the same manner as above, by (here, assuming that the mixture consists of n phases)

$$\mathbf{V}_\mathrm{m} = \frac{\sum_{\mathrm{k}=1}^{\mathrm{n}} m_\mathrm{k}\mathbf{V}_\mathrm{k}}{\sum_{\mathrm{k}=1}^{\mathrm{n}} m_\mathrm{k}} \quad \wedge \quad m_\mathrm{m} = \sum_{\mathrm{k}=1}^{\mathrm{n}} m_\mathrm{k} \tag{2.6}$$

Continuing further with the above equation, the following is obtained

$$\mathbf{V}_\mathrm{m} = \frac{\sum_{\mathrm{k}=1}^{\mathrm{n}} (m_\mathrm{k}/V_\mathrm{m})\,\mathbf{V}_\mathrm{k}}{\sum_{\mathrm{k}=1}^{\mathrm{n}} m_\mathrm{k}/V_\mathrm{m}} = \frac{\sum_{\mathrm{k}=1}^{\mathrm{n}} \beta_\mathrm{k}\rho_\mathrm{k}\mathbf{V}_\mathrm{k}}{\sum_{\mathrm{k}=1}^{\mathrm{n}} \beta_\mathrm{k}\rho_\mathrm{k}} = \frac{\sum_{\mathrm{k}=1}^{\mathrm{n}} \beta_\mathrm{k}\rho_\mathrm{k}\mathbf{V}_\mathrm{k}}{\rho_\mathrm{m}} \tag{2.7}$$

In the last part of the above equation, Eq. (2.3) was used.

In this work, the mixture velocity $\mathbf{V}_\mathrm{m}$ is also represented with $\mathbf{U}_1$ and the reason for doing this will be clear in Section 4.2 (i.e. $\mathbf{V}_\mathrm{m} \equiv \mathbf{U}_1$).

By summing the mass conservation Eq. (2.4) over all phases k, the following is obtained

$$\frac{\partial}{\partial t}\sum_{\mathrm{k}=1}^{\mathrm{n}} (\beta_\mathrm{k}\rho_\mathrm{k}) + \nabla\cdot\sum_{\mathrm{k}=1}^{\mathrm{n}} (\beta_\mathrm{k}\rho_\mathrm{k}\mathbf{V}_\mathrm{k}) = \sum_{\mathrm{k}=1}^{\mathrm{n}} \Gamma_\mathrm{k} \tag{2.8}$$

Because the total mass is conserved, the right hand side of Eq. (2.8) must vanish [15]. By using Eqs. (2.3) and (2.7) in Eq. (2.8), the continuity equation for the mixture can be derived and is given by Eq. (2.9).

$$\frac{\partial \rho_\mathrm{m}}{\partial t} + \nabla\cdot(\rho_\mathrm{m}\mathbf{V}_\mathrm{m}) = 0 \tag{2.9}$$

2.2.3 Conservation of Momentum for Each Phase k

For the Multi-Fluid Method, the momentum equation for each phase k, is given by [6, 7, 15, 16]

$$\frac{\partial}{\partial t}(\beta_\mathrm{k}\rho_\mathrm{k}\mathbf{V}_\mathrm{k}) + \nabla\cdot(\beta_\mathrm{k}\rho_\mathrm{k}\mathbf{V}_\mathrm{k}\mathbf{V}_\mathrm{k}) = \nabla\cdot(\beta_\mathrm{k}\boldsymbol{\sigma}_\mathrm{k}) + \beta_\mathrm{k}\rho_\mathrm{k}\mathbf{g} + \mathbf{M}_\mathrm{k} \tag{2.10}$$

The term $\mathbf{g}$ is the gravity, while β_k, ρ_k and $\mathbf{V}_\mathrm{k}$ have already been defined in the previous sections. The term $\boldsymbol{\sigma}_\mathrm{k}$ is the stress tensor for phase k. For many fluids, this constitutive equation is represented as [2],

$$\boldsymbol{\sigma}_\mathrm{k} = -p_\mathrm{k}\,\mathbf{I} + \mathbf{T}_\mathrm{k} \tag{2.11}$$

[9] As demonstrated in [3] (Section 2.2), in order to reconstruct the Cauchy equation of motion, the fluid velocity must be defined in this manner. This reconstruction is based on particle–particle interactions between particles that make up the continuum particle [3]. Being a "child" of the Cauchy equation, the same applies for the Navier–Stokes equations.

where the second order tensor $\mathbf{T}_k$ is known as the extra stress tensor, $\mathbf{I}$ is the unit dyadic, and p_k is the pressure (the pressure of each phase will be slightly discussed in Section 2.3.4). Since the subject of the current work is a high viscous fluid with low Reynolds numbers, the term $\mathbf{T}_k$ does not contain any turbulent components.

In Eq. (2.10), the term $\mathbf{M}_k$ is the average interfacial momentum source for phase k. That is, it represents transfer of momentum from one phase to the the next, like drag, lift, surface tension and so forth [7, 15, 16]. Its determination represents the largest uncertainties in the results of Multi-Fluid Method [16].

2.3 Drift Flux Model - DFM

As previously mentioned, the Drift Flux Model is based on the Multi-Fluid Method, using two phases, then also known as the Two-Fluid Method. As such, many of the above equations will be reused in the following sections.

2.3.1 Fundamental Relations

As in the Two-Fluid Method, the Drift Flux Model (DFM) assumes that the fluid mixture consists of two phases. It is assumed that the two fluids can intermix as well as separate, depending on the flow conditions involved. The latter phenomenon is also known as *phase separations*, *slip between phases*, *settling*, *segregation*, and so forth, depending on the field of science in which the DFM is applied (i.e. waste water treatment, sewage treatment, flow of fresh concrete, etc.).

Here, the fluid mixture is a suspension that consists of a continuous phase (i.e. a matrix) and a dispersed phase (i.e. suspended particles). The continuous phase will have the symbol c, while the dispersed phase has the symbol d. Thus, the numbering of phases is in terms[10] of $k = c, d$. With this "numbering scheme", the following is obtained from Eq. (2.1)

$$\beta_c = \frac{V_c}{V_m} \quad \wedge \quad \beta_d = \frac{V_d}{V_m} \quad \wedge \quad V_m = V_c + V_d \quad \Rightarrow \quad \beta_c + \beta_d = 1 \tag{2.12}$$

The **mixture density** is given by Eq. (2.3) and with two phases it is calculated as

$$\rho_m = \beta_c \rho_c + \beta_d \rho_d \tag{2.13}$$

Likewise, the calculation of **mixture velocity** is shown in Eq. (2.7) and with two phases it is given by

$$\mathbf{V}_m = \frac{\beta_c \rho_c \mathbf{V}_c + \beta_d \rho_d \mathbf{V}_d}{\rho_m} \tag{2.14}$$

[10] Often, the numbering of phases is in terms of numbers like $k = 1, 2$, but because of the complexity of the current work (see Chapter 4), such labeling scheme needs to be reserved for later usage.

The **relative velocity**[11] between the two phases is defined by [7]

$$\mathbf{V}_\mathrm{r} = \mathbf{V}_\mathrm{c} - \mathbf{V}_\mathrm{d} \tag{2.15}$$

The **diffusion velocity** is defined by the following equation [7]

$$\mathbf{V}_\mathrm{km} = \mathbf{V}_\mathrm{k} - \mathbf{V}_\mathrm{m} \quad \forall \quad k = c, d. \tag{2.16}$$

The term $\mathbf{V}_\mathrm{k}$ is the center of mass velocity of phase k (see Eq. (2.5)). From Eq. (2.16), the diffusion velocity of each phase, namely c (the continuous phase) and d (the dispersed phase), are given by the following

$$\mathbf{V}_\mathrm{cm} = \mathbf{V}_\mathrm{c} - \mathbf{V}_\mathrm{m} \tag{2.17}$$

$$\mathbf{V}_\mathrm{dm} = \mathbf{V}_\mathrm{d} - \mathbf{V}_\mathrm{m} \tag{2.18}$$

Below are derivations of some important relationships that are used later. These are obtained from previous derivations and definitions.

The first relationship is as follows

$$\beta_\mathrm{c}\rho_\mathrm{c}\mathbf{V}_\mathrm{cm} + \beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{dm} = 0 \tag{2.19}$$

The proof of this is given by Eq. (2.20) shown below.

$$\begin{aligned}
&\beta_\mathrm{c}\rho_\mathrm{c}\mathbf{V}_\mathrm{cm} + \beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{dm} = \beta_\mathrm{c}\rho_\mathrm{c}(\mathbf{V}_\mathrm{c} - \mathbf{V}_\mathrm{m}) + \beta_\mathrm{d}\rho_\mathrm{d}(\mathbf{V}_\mathrm{d} - \mathbf{V}_\mathrm{m}) = \\
&\beta_\mathrm{c}\rho_\mathrm{c}\mathbf{V}_\mathrm{c} - \beta_\mathrm{c}\rho_\mathrm{c}\mathbf{V}_\mathrm{m} + \beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{d} - \beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{m} = \\
&\beta_\mathrm{c}\rho_\mathrm{c}\mathbf{V}_\mathrm{c} + \beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{d} - (\beta_\mathrm{c}\rho_\mathrm{c} + \beta_\mathrm{d}\rho_\mathrm{d})\mathbf{V}_\mathrm{m} = \beta_\mathrm{c}\rho_\mathrm{c}\mathbf{V}_\mathrm{c} + \beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{d} - \rho_\mathrm{m}\mathbf{V}_\mathrm{m} = \\
&\rho_\mathrm{m}\mathbf{V}_\mathrm{m} - \rho_\mathrm{m}\mathbf{V}_\mathrm{m} = 0
\end{aligned} \tag{2.20}$$

In the above equation, Eqs. (2.17) and (2.18) were used, while in its third line, Eq. (2.3) was used. Finally, in its fourth line, Eq. (2.7) was used.

The second relationship is obtained from Eq. (2.17), with the concomitant use of Eqs. (2.3) and (2.7):

$$\begin{aligned}
\mathbf{V}_\mathrm{cm} &= \mathbf{V}_\mathrm{c} - \mathbf{V}_\mathrm{m} = \mathbf{V}_\mathrm{c} - \frac{\beta_\mathrm{c}\rho_\mathrm{c}\mathbf{V}_\mathrm{c} + \beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{d}}{\beta_\mathrm{c}\rho_\mathrm{c} + \beta_\mathrm{d}\rho_\mathrm{d}} \\
&= \frac{\beta_\mathrm{c}\rho_\mathrm{c}\mathbf{V}_\mathrm{c} + \beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{c} - \beta_\mathrm{c}\rho_\mathrm{c}\mathbf{V}_\mathrm{c} - \beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{d}}{\beta_\mathrm{c}\rho_\mathrm{c} + \beta_\mathrm{d}\rho_\mathrm{d}} \\
&= \frac{\beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{c} - \beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{d}}{\rho_\mathrm{m}} = \frac{\beta_\mathrm{d}\rho_\mathrm{d}}{\rho_\mathrm{m}}(\mathbf{V}_\mathrm{c} - \mathbf{V}_\mathrm{d}) = \frac{\beta_\mathrm{d}\rho_\mathrm{d}}{\rho_\mathrm{m}}\mathbf{V}_\mathrm{r}
\end{aligned} \tag{2.21}$$

In the last line of Eq. (2.21), the relative velocity according to Eq. (2.15) was used. By rearranging terms in Eq. (2.19), the third relationship is obtained:

$$\mathbf{V}_\mathrm{dm} = -\frac{\beta_\mathrm{c}\rho_\mathrm{c}}{\beta_\mathrm{d}\rho_\mathrm{d}}\mathbf{V}_\mathrm{cm} = -\frac{\beta_\mathrm{c}\rho_\mathrm{c}}{\rho_\mathrm{m}}(\mathbf{V}_\mathrm{c} - \mathbf{V}_\mathrm{d}) = -\frac{\beta_\mathrm{c}\rho_\mathrm{c}}{\rho_\mathrm{m}}\mathbf{V}_\mathrm{r} \tag{2.22}$$

[11] In some literature, the relative velocity is defined as $\mathbf{V}_\mathrm{r} = \mathbf{V}_\mathrm{d} - \mathbf{V}_\mathrm{c}$ instead of $\mathbf{V}_\mathrm{r} = \mathbf{V}_\mathrm{c} - \mathbf{V}_\mathrm{d}$. Irrespective of which is used, the difference is not consequential because other terms will just change accordingly and the physics of the Drift Flux Model will remain unchanged.

In the above, Eqs. (2.15) and (2.21) were used.

The the **volumetric fluxes** of each phase is defined as (see for example [7])

$$\mathbf{j}_k = \beta_k \mathbf{V}_k \tag{2.23}$$

Hence the **total volumetric flux** is given by the following

$$\mathbf{j} = \sum_{k=1}^{n} \beta_k \mathbf{V}_k = \beta_c \mathbf{V}_c + \beta_d \mathbf{V}_d \tag{2.24}$$

The **drift velocity** $\mathbf{V}_{kj}$ is the velocity of phase k relative to that of the volume center of the mixture and is given by the following [17] (see also [7, 15])

$$\mathbf{V}_{kj} = \mathbf{V}_k - \mathbf{j} \tag{2.25}$$

By using the above equation, the fourth relationship can be derived for phase c (as previously mentioned, phase c is the continuous phase, i.e. the matrix part of the overall mixture):

$$\begin{aligned} \mathbf{V}_{cj} &= \mathbf{V}_c - \mathbf{j} = \mathbf{V}_c - \beta_c \mathbf{V}_c - \beta_d \mathbf{V}_d = (1-\beta_c)\mathbf{V}_c - \beta_d \mathbf{V}_d \\ &= \beta_d \mathbf{V}_c - \beta_d \mathbf{V}_d = \beta_d \mathbf{V}_r \end{aligned} \tag{2.26}$$

By using Eq. (2.25), the fifth relationship can be derived for phase d (as previously mentioned, phase d is the dispersed phase, i.e. the suspended particles):

$$\begin{aligned} \mathbf{V}_{dj} &= \mathbf{V}_d - \mathbf{j} = \mathbf{V}_d - \beta_c \mathbf{V}_c - \beta_d \mathbf{V}_d = (1-\beta_d)\mathbf{V}_d - \beta_c \mathbf{V}_c \\ &= \beta_c \mathbf{V}_d - \beta_c \mathbf{V}_c = -\beta_c \mathbf{V}_r \end{aligned} \tag{2.27}$$

The above is the drift velocity of the dispersed phase. That is, the term $\mathbf{V}_{dj}$ represents the velocity of the dispersed phase relative to that of the volume center of the mixture (see Eq. (2.25)).

The sixth relationship is as follows

$$\beta_c \mathbf{V}_{cj} + \beta_d \mathbf{V}_{dj} = 0 \tag{2.28}$$

The proof of this relationship is given by Eq. (2.29), shown below:

$$\begin{aligned} &\beta_c \mathbf{V}_{cj} + \beta_d \mathbf{V}_{dj} = \beta_c(\mathbf{V}_c - \mathbf{j}) + \beta_d(\mathbf{V}_d - \mathbf{j}) = \\ &\beta_c \mathbf{V}_c + \beta_d \mathbf{V}_d - \beta_c \mathbf{j} - \beta_d \mathbf{j} = \mathbf{j} - (\beta_c + \beta_d)\mathbf{j} = \mathbf{j} - \mathbf{j} = 0 \end{aligned} \tag{2.29}$$

Rearrangement of Eq. (2.28) gives

$$\mathbf{V}_{dj} = -\frac{\beta_c}{\beta_d} \mathbf{V}_{cj} \tag{2.30}$$

From Eq. (2.22), the seventh relationship is obtained:

$$\mathbf{V}_{dm} = -\frac{\beta_c \rho_c}{\rho_m} \mathbf{V}_r = -\frac{\beta_c \rho_c}{\rho_m} \left(-\frac{\mathbf{V}_{dj}}{\beta_c} \right) = \frac{\rho_c}{\rho_m} \mathbf{V}_{dj} \tag{2.31}$$

In the above, the relation from Eq. (2.27) was used. Continuing further with Eq. (2.31), the following can be obtained

$$\mathbf{V}_{\mathrm{dm}} = \frac{\rho_{\mathrm{c}}}{\rho_{\mathrm{m}}}\mathbf{V}_{\mathrm{dj}} = \frac{\rho_{\mathrm{c}}}{\rho_{\mathrm{m}}}\left(-\frac{\beta_{\mathrm{c}}}{\beta_{\mathrm{d}}}\mathbf{V}_{\mathrm{cj}}\right) = -\frac{\beta_{\mathrm{c}}}{\beta_{\mathrm{d}}}\frac{\rho_{\mathrm{c}}}{\rho_{\mathrm{m}}}\mathbf{V}_{\mathrm{cj}} \tag{2.32}$$

In the above, the relation from Eq. (2.30) was used.

2.3.2 Mass Conservation of the Dispersed Phase

Using the mass conservation equation for the Two-Fluid Method Eq. (2.4), one can obtain the mass conservation for both the continuous phase and the dispersed phase, namely with

$$\frac{\partial \beta_{\mathrm{c}}\rho_{\mathrm{c}}}{\partial t} + \nabla\cdot(\beta_{\mathrm{c}}\rho_{\mathrm{c}}\mathbf{V}_{\mathrm{c}}) = \Gamma_{\mathrm{c}}, \tag{2.33}$$

$$\frac{\partial \beta_{\mathrm{d}}\rho_{\mathrm{d}}}{\partial t} + \nabla\cdot(\beta_{\mathrm{d}}\rho_{\mathrm{d}}\mathbf{V}_{\mathrm{d}}) = \Gamma_{\mathrm{d}} \tag{2.34}$$

In this work, no mass formation or destruction is occurring (i.e. no phase changes), meaning $\Gamma_{\mathrm{c}} = \Gamma_{\mathrm{d}} = 0$ (see also [15] about a general assumption of this).

If the densities of the two phases are constant, i.e. ρ_{c} = constant and ρ_{d} = constant, one can simplify the above equations further

$$\frac{\partial \beta_{\mathrm{c}}}{\partial t} + \nabla\cdot(\beta_{\mathrm{c}}\mathbf{V}_{\mathrm{c}}) = 0 \tag{2.35}$$

$$\frac{\partial \beta_{\mathrm{d}}}{\partial t} + \nabla\cdot(\beta_{\mathrm{d}}\mathbf{V}_{\mathrm{d}}) = 0 \tag{2.36}$$

By adding Eqs. (2.35) and (2.36) and keeping in mind that $\beta_{\mathrm{c}} + \beta_{\mathrm{d}} = 1$ = constant (see Eq. (2.12)), one obtains the following

$$\frac{\partial(\beta_{\mathrm{c}} + \beta_{\mathrm{d}})}{\partial t} + \nabla\cdot(\beta_{\mathrm{c}}\mathbf{V}_{\mathrm{c}} + \beta_{\mathrm{d}}\mathbf{V}_{\mathrm{d}}) = 0 + \nabla\cdot(\beta_{\mathrm{c}}\mathbf{V}_{\mathrm{c}} + \beta_{\mathrm{d}}\mathbf{V}_{\mathrm{d}}) = 0 \tag{2.37}$$

By comparing the above result with Eq. (2.24), it becomes clear that the divergence of the total volumetric flux $\mathbf{j}$ is zero:

$$\nabla\cdot\mathbf{j} = 0 \tag{2.38}$$

It should be kept in mind that the above result is hinged on the above-mentioned constraints: ρ_{c} = constant and ρ_{d} = constant, which are valid for the current work (see Eq. (4.11), Page 33).

From the diffusion velocity for the dispersed phase $\mathbf{V}_{\mathrm{dm}}$, namely Eq. (2.18), with the concomitant use of Eq. (2.31), the following is obtained

$$\mathbf{V}_{\mathrm{d}} = \mathbf{V}_{\mathrm{dm}} + \mathbf{V}_{\mathrm{m}} = \frac{\rho_{\mathrm{c}}}{\rho_{\mathrm{m}}}\mathbf{V}_{\mathrm{dj}} + \mathbf{V}_{\mathrm{m}} \tag{2.39}$$

Using the above in Eq. (2.34) gives

$$\frac{\partial \beta_\mathrm{d}\rho_\mathrm{d}}{\partial t} + \nabla \cdot \left(\beta_\mathrm{d}\rho_\mathrm{d} \left[\frac{\rho_\mathrm{c}}{\rho_\mathrm{m}} \mathbf{V}_\mathrm{dj} + \mathbf{V}_\mathrm{m} \right] \right) = 0 \tag{2.40}$$

Likewise, using Eq. (2.39) in Eq. (2.36) results in

$$\frac{\partial \beta_\mathrm{d}}{\partial t} + \nabla \cdot (\beta_\mathrm{d} \mathbf{V}_\mathrm{m}) + \nabla \cdot \left(\frac{\beta_\mathrm{d}\rho_\mathrm{c}}{\rho_\mathrm{m}} \mathbf{V}_\mathrm{dj} \right) = 0 \tag{2.41}$$

2.3.3 Divergence of Velocity

From Eqs (2.14) and (2.24), the difference between the mixture velocity and total volumetric flux is as follows

$$\mathbf{V}_\mathrm{m} - \mathbf{j} = \frac{\beta_\mathrm{c}\rho_\mathrm{c}\mathbf{V}_\mathrm{c} + \beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{d}}{\rho_\mathrm{m}} - \beta_\mathrm{c}\mathbf{V}_\mathrm{c} + \beta_\mathrm{d}\mathbf{V}_\mathrm{d} \tag{2.42}$$

The above can be evaluated further as shown with Eq. (2.43).

$$\begin{aligned}
(\mathbf{V}_\mathrm{m} - \mathbf{j})\,\rho_\mathrm{m} &= \beta_\mathrm{c}\rho_\mathrm{c}\mathbf{V}_\mathrm{c} + \beta_\mathrm{d}\rho_\mathrm{d}\mathbf{V}_\mathrm{d} - \beta_\mathrm{c}\rho_\mathrm{m}\mathbf{V}_\mathrm{c} - \beta_\mathrm{d}\rho_\mathrm{m}\mathbf{V}_\mathrm{d} \\
&= \beta_\mathrm{c}\mathbf{V}_\mathrm{c}(\rho_\mathrm{c} - \rho_\mathrm{m}) + \beta_\mathrm{d}\mathbf{V}_\mathrm{d}(\rho_\mathrm{d} - \rho_\mathrm{m}) \\
&= \beta_\mathrm{c}\mathbf{V}_\mathrm{c}(\rho_\mathrm{c} - \beta_\mathrm{c}\rho_\mathrm{c} - \beta_\mathrm{d}\rho_\mathrm{d}) + \beta_\mathrm{d}\mathbf{V}_\mathrm{d}(\rho_\mathrm{d} - \beta_\mathrm{c}\rho_\mathrm{c} - \beta_\mathrm{d}\rho_\mathrm{d}) \\
&= \beta_\mathrm{c}\mathbf{V}_\mathrm{c}\left[\rho_\mathrm{c}(1 - \beta_\mathrm{c}) - \beta_\mathrm{d}\rho_\mathrm{d}\right] + \beta_\mathrm{d}\mathbf{V}_\mathrm{d}\left[\rho_\mathrm{d}(1 - \beta_\mathrm{d}) - \beta_\mathrm{c}\rho_\mathrm{c}\right] \\
&= \beta_\mathrm{c}\mathbf{V}_\mathrm{c}\left[\rho_\mathrm{c}\beta_\mathrm{d} - \beta_\mathrm{d}\rho_\mathrm{d}\right] + \beta_\mathrm{d}\mathbf{V}_\mathrm{d}\left[\rho_\mathrm{d}\beta_\mathrm{c} - \beta_\mathrm{c}\rho_\mathrm{c}\right] \\
&= \beta_\mathrm{c}\mathbf{V}_\mathrm{c}\beta_\mathrm{d}\left(\rho_\mathrm{c} - \rho_\mathrm{d}\right) + \beta_\mathrm{d}\mathbf{V}_\mathrm{d}\beta_\mathrm{c}\left(\rho_\mathrm{d} - \rho_\mathrm{c}\right) \\
&= \beta_\mathrm{c}\mathbf{V}_\mathrm{c}\beta_\mathrm{d}\left(\rho_\mathrm{c} - \rho_\mathrm{d}\right) - \beta_\mathrm{d}\mathbf{V}_\mathrm{d}\beta_\mathrm{c}\left(\rho_\mathrm{c} - \rho_\mathrm{d}\right) \\
&= \beta_\mathrm{c}\beta_\mathrm{d}\left(\rho_\mathrm{c} - \rho_\mathrm{d}\right)\left(\mathbf{V}_\mathrm{c} - \mathbf{V}_\mathrm{d}\right)
\end{aligned} \tag{2.43}$$

In the third line, Eq. (2.13) was used (i.e. $\rho_\mathrm{m} = \beta_\mathrm{c}\rho_\mathrm{c} + \beta_\mathrm{d}\rho_\mathrm{d}$), while in the fifth line, Eq. (2.12) was used (i.e. $\beta_\mathrm{c} + \beta_\mathrm{d} = 1$). Using Eqs. (2.15) and (2.27), the above can be further refined:

$$(\mathbf{V}_\mathrm{m} - \mathbf{j})\,\rho_\mathrm{m} = \beta_\mathrm{c}\beta_\mathrm{d}\left(\rho_\mathrm{c} - \rho_\mathrm{d}\right)\mathbf{V}_\mathrm{r} = -\beta_\mathrm{c}\beta_\mathrm{d}\left(\rho_\mathrm{c} - \rho_\mathrm{d}\right)\frac{\mathbf{V}_\mathrm{dj}}{\beta_\mathrm{c}} = \beta_\mathrm{d}\left(\rho_\mathrm{d} - \rho_\mathrm{c}\right)\mathbf{V}_\mathrm{dj} \tag{2.44}$$

Finally, rearranging the above, gives

$$\mathbf{V}_\mathrm{m} = \mathbf{j} + \beta_\mathrm{d}\left(\frac{\rho_\mathrm{d} - \rho_\mathrm{c}}{\rho_\mathrm{m}}\right)\mathbf{V}_\mathrm{dj} \tag{2.45}$$

Taking the divergence of Eq. (2.45) and keeping in mind the result of Eq. (2.38), namely that $\nabla \cdot \mathbf{j} = 0$, the following is obtained

$$\nabla \cdot \mathbf{V}_\mathrm{m} = \nabla \cdot \left(\beta_\mathrm{d} \left[\frac{\rho_\mathrm{d} - \rho_\mathrm{c}}{\rho_\mathrm{m}} \right] \mathbf{V}_\mathrm{dj} \right) \tag{2.46}$$

The above can also be expressed with the following

$$\nabla \cdot \mathbf{V}_\mathrm{m} = \nabla \cdot \mathbf{R} \tag{2.47}$$

where the vector $\mathbf{R}$ is defined with

$$\mathbf{R} = \left(\beta_\mathrm{d} \left[\frac{\rho_\mathrm{d} - \rho_\mathrm{c}}{\rho_\mathrm{m}} \right] \mathbf{V}_\mathrm{dj} \right) \tag{2.48}$$

2.3.4 Mixture Momentum Equation

From Eq. (2.10), the mixture momentum equation for each phase $k = c, d$ is given by

$$\frac{\partial}{\partial t}(\beta_c \rho_c \mathbf{V}_c) + \nabla \cdot (\beta_c \rho_c \mathbf{V}_c \mathbf{V}_c) = \nabla \cdot (\beta_c \boldsymbol{\sigma}_c) + \beta_c \rho_c \mathbf{g} + \mathbf{M}_c \tag{2.49}$$

$$\frac{\partial}{\partial t}(\beta_d \rho_d \mathbf{V}_d) + \nabla \cdot (\beta_d \rho_d \mathbf{V}_d \mathbf{V}_d) = \nabla \cdot (\beta_d \boldsymbol{\sigma}_d) + \beta_d \rho_d \mathbf{g} + \mathbf{M}_d \tag{2.50}$$

The mixture momentum equation for the DFM is obtained by the summation of each part of Eqs. (2.49) and (2.50). Summarizing the first part and using Eq. (2.7) (Page 18) gives

$$\frac{\partial}{\partial t}(\beta_c \rho_c \mathbf{V}_c) + \frac{\partial}{\partial t}(\beta_d \rho_d \mathbf{V}_d) = \frac{\partial}{\partial t}(\rho_m \mathbf{V}_m) \tag{2.51}$$

For the stress components, the following can be obtained by use of Eq. (2.11)

$$\beta_c \, \boldsymbol{\sigma}_c + \beta_d \, \boldsymbol{\sigma}_d = -(\beta_c \, p_c + \beta_d \, p_d) \, \mathbf{I} + (\beta_c \, \mathbf{T}_c + \beta_d \, \mathbf{T}_d) = -p_m \, \mathbf{I} + \mathbf{T}_m \tag{2.52}$$

...meaning...

$$\nabla \cdot (\beta_c \, \boldsymbol{\sigma}_c) + \nabla \cdot (\beta_d \, \boldsymbol{\sigma}_d) = \nabla \cdot (-p_m \, \mathbf{I} + \mathbf{T}_m) = -\nabla p_m + \nabla \cdot \mathbf{T}_m \tag{2.53}$$

As shown in Eq. (2.52), the mixture pressure is given by $p_m = \beta_c p_c + \beta_d p_d$ (see [7]). Also shown there, the mixture extra stress tensor is given by $\mathbf{T}_m = \beta_c \, \mathbf{T}_c + \beta_d \, \mathbf{T}_d$ (see [15]).

In practice, the phase pressures are often taken to be equal, i.e. $p_c = p_d$, meaning $p_m = \beta_c p_c + \beta_d p_d = (\beta_c + \beta_d) \, p_c = p_c = p_d$. This assumption is considered to be valid except in the case of expanding bubbles [18] (see also [15]).

From Eq. (2.3) on Page 17, the following is obtained

$$\beta_c \rho_c \mathbf{g} + \beta_d \rho_d \mathbf{g} = \rho_m \mathbf{g} \tag{2.54}$$

By the summation of Eqs. (2.49) and (2.50) and thereafter using the results of Eqs. (2.51) to (2.56) with $\mathbf{M}_c + \mathbf{M}_d = \mathbf{M}_m$, one can obtain the mixture momentum equation, given by

$$\frac{\partial}{\partial t}(\rho_m \mathbf{V}_m) + \nabla \cdot (\beta_c \rho_c \mathbf{V}_c \mathbf{V}_c + \beta_d \rho_d \mathbf{V}_d \mathbf{V}_d) = -\nabla p_m + \nabla \cdot \mathbf{T}_m + \rho_m \mathbf{g} + \mathbf{M}_m \tag{2.55}$$

In Eqs. (2.49) and (2.50), the terms $\mathbf{M}_c$ and $\mathbf{M}_d$ represents transfer of momentum from one phase to the the other by phenomena such as drag, lift or surface tension effects [7]. By Newton's third law of motion (action and reaction), these effects are always opposite and equal, meaning $\mathbf{M}_c = -\mathbf{M}_d$. Thus, the overall effect $\mathbf{M}_m = \mathbf{M}_c + \mathbf{M}_d$ is always summarized to zero and is therefore not of concern, something that is not possible for the Two-Fluid Method (see Eq. (2.10)). That is, the following applies [19].

$$\mathbf{M}_c + \mathbf{M}_d = \mathbf{M}_m = 0 \tag{2.56}$$

In order to make Eq. (2.55) usable, the sum $\beta_c\rho_c\mathbf{V}_c\mathbf{V}_c+\beta_d\rho_d\mathbf{V}_d\mathbf{V}_d$ has to be calculated further. By applying the diffusion velocity of the phases, namely Eqs. (2.17) and (2.18), one obtains $\mathbf{V}_c = \mathbf{V}_m + \mathbf{V}_{cm}$ and $\mathbf{V}_d = \mathbf{V}_m + \mathbf{V}_{dm}$. By using these two relationships, including Eqs. (2.3) and (2.19), the following is obtained[12]:

$$\begin{aligned}
&\beta_c\rho_c\mathbf{V}_c\mathbf{V}_c + \beta_d\rho_d\mathbf{V}_d\mathbf{V}_d = \beta_c\rho_c(\mathbf{V}_m + \mathbf{V}_{cm})(\mathbf{V}_m + \mathbf{V}_{cm})+\\
&\beta_d\rho_d(\mathbf{V}_m + \mathbf{V}_{dm})(\mathbf{V}_m + \mathbf{V}_{dm}) =\\
&\beta_c\rho_c(\mathbf{V}_m\mathbf{V}_m + \mathbf{V}_m\mathbf{V}_{cm} + \mathbf{V}_{cm}\mathbf{V}_m + \mathbf{V}_{cm}\mathbf{V}_{cm})+\\
&\beta_d\rho_d(\mathbf{V}_m\mathbf{V}_m + \mathbf{V}_m\mathbf{V}_{dm} + \mathbf{V}_{dm}\mathbf{V}_m + \mathbf{V}_{dm}\mathbf{V}_{dm}) =\\
&(\beta_c\rho_c + \beta_d\rho_d)\mathbf{V}_m\mathbf{V}_m + \mathbf{V}_m(\beta_c\rho_c\mathbf{V}_{cm} + \beta_d\rho_d\mathbf{V}_{dm}) + (\beta_c\rho_c\mathbf{V}_{cm} + \beta_d\rho_d\mathbf{V}_{dm})\mathbf{V}_m+\\
&\beta_c\rho_c\mathbf{V}_{cm}\mathbf{V}_{cm} + \beta_d\rho_d\mathbf{V}_{dm}\mathbf{V}_{dm} =\\
&\rho_m\mathbf{V}_m\mathbf{V}_m + \mathbf{V}_m\cdot 0 + 0\cdot\mathbf{V}_m + \beta_c\rho_c\mathbf{V}_{cm}\mathbf{V}_{cm} + \beta_d\rho_d\mathbf{V}_{dm}\mathbf{V}_{dm} =\\
&\rho_m\mathbf{V}_m\mathbf{V}_m + \textstyle\sum_{k=1}^{n}\beta_k\rho_k\mathbf{V}_{km}\mathbf{V}_{km}
\end{aligned} \tag{2.57}$$

By rearranging terms in Eq. (2.19), Page 20, and thereafter using Eq. (2.31), Page 21, the following is obtained

$$\mathbf{V}_{cm} = -\frac{\beta_d\rho_d}{\beta_c\rho_c}\mathbf{V}_{dm} = -\frac{\beta_d\rho_d}{\beta_c\rho_c}\left(\frac{\rho_c}{\rho_m}\mathbf{V}_{dj}\right) = -\frac{\beta_d\rho_d}{\beta_c\rho_m}\mathbf{V}_{dj} \tag{2.58}$$

By using Eqs. (2.31) and (2.58), the last part of Eq. (2.57) can be extrapolated further, which is done in Eq. (2.59).

$$\begin{aligned}
&\beta_c\rho_c\mathbf{V}_{cm}\mathbf{V}_{cm} + \beta_d\rho_d\mathbf{V}_{dm}\mathbf{V}_{dm} = \beta_c\rho_c\left(\tfrac{\beta_d\rho_d}{\beta_c\rho_m}\right)^2\mathbf{V}_{dj}\mathbf{V}_{dj} + \beta_d\rho_d\left(\tfrac{\rho_c}{\rho_m}\right)^2\mathbf{V}_{dj}\mathbf{V}_{dj} =\\
&\left[\beta_c\rho_c\left(\tfrac{\beta_d\rho_d}{\beta_c\rho_m}\right)^2 + \beta_d\rho_d\left(\tfrac{\rho_c}{\rho_m}\right)^2\right]\mathbf{V}_{dj}\mathbf{V}_{dj} = \left[\tfrac{\beta_c\rho_c\beta_d^2\rho_d^2}{\beta_c^2\rho_m^2} + \tfrac{\beta_d\rho_d\rho_c^2}{\rho_m^2}\right]\mathbf{V}_{dj}\mathbf{V}_{dj} =\\
&\left[\tfrac{\rho_c\beta_d^2\rho_d^2}{\beta_c\rho_m^2} + \tfrac{\beta_d\rho_d\rho_c^2}{\rho_m^2}\right]\mathbf{V}_{dj}\mathbf{V}_{dj} = \tfrac{\beta_d\rho_c\rho_d}{\rho_m^2}\left(\tfrac{\beta_d\rho_d}{\beta_c} + \tfrac{\beta_c\rho_c}{\beta_c}\right)\mathbf{V}_{dj}\mathbf{V}_{dj} =\\
&\tfrac{\beta_d\rho_c\rho_d}{\rho_m^2}\left(\tfrac{\beta_c\rho_c+\beta_d\rho_d}{\beta_c}\right)\mathbf{V}_{dj}\mathbf{V}_{dj} = \left(\tfrac{\beta_d\rho_c\rho_d}{\rho_m^2}\tfrac{\rho_m}{1-\beta_d}\right)\mathbf{V}_{dj}\mathbf{V}_{dj} = \left(\tfrac{\beta_d}{1-\beta_d}\tfrac{\rho_c\rho_d}{\rho_m}\right)\mathbf{V}_{dj}\mathbf{V}_{dj}
\end{aligned} \tag{2.59}$$

In the above, the relationships by Eqs. (2.1) and (2.3) were also used.

By putting the results of Eqs. (2.56), (2.57) and (2.59) into Eq. (2.55), the final form for the mixture momentum equation for the DFM is obtained

$$\frac{\partial(\rho_m\mathbf{V}_m)}{\partial t} + \nabla\cdot(\rho_m\mathbf{V}_m\mathbf{V}_m) + \nabla\cdot\left(\left[\frac{\beta_d}{1-\beta_d}\frac{\rho_c\rho_d}{\rho_m}\right]\mathbf{V}_{dj}\mathbf{V}_{dj}\right) = -\nabla p_m + \nabla\cdot\mathbf{T}_m + \rho_m\mathbf{g} \tag{2.60}$$

[12] Note that the tensor product $\otimes$ is not used in this report. Rather, the same presentation method is used as given by Mase [20] and Malvern [21]. For example, in order to demonstrate the non-commutative behavior of a tensor product (or outer product) of two vectors, it would be shown as $\mathbf{V}_m\mathbf{V}_{cm} \neq \mathbf{V}_{cm}\mathbf{V}_m$ and not as $\mathbf{V}_m \otimes \mathbf{V}_{cm} \neq \mathbf{V}_{cm} \otimes \mathbf{V}_m$.

Chapter 3

Volume of Fluid Method – VOF

3.1 Introduction

For the current work, it is important to divide the system between the atmospheric air and the fluid mixture (e.g. fresh concrete). This is done with a so–called *free interface*. Numerical methods that can manage such division are classified into two groups depending on the fundamental type of mesh used [22]. These are moving mesh (Lagrangian mesh) and fixed mesh (Eulerian mesh). Although the moving mesh approach allows a sharp interface definition it encounters serious problems in cases when the interface undergoes large deformations where the moving mesh may become severely distorted [23]. Because of this, the Eulerian mesh approach is preferred in many cases, like the volume-of-fluid [5], the level set [22, 24] or the marker and cell [24] methods. In this work, the volume-of-fluid method (VOF) is used and thus the text in this chapter refers to that specific theory.

3.2 Fundamental Relations

Here, the volume fraction (also, solid concentration or phase volume) of the fluid mixture (e.g. fresh concrete) within each computational cell[13] is represented with α_1, while the volume fraction of atmospheric air is represented with α_2. More precisely, $\alpha_1 = V_1/V_\mathrm{P}$, where V_P is the volume of the cell and V_1 is the volume of concrete within the cell (i.e. $V_1 \leq V_\mathrm{P}$). When $\alpha_1 = 1$, the computational cell is filled only with fluid mixture, while if $\alpha_1 = 0$, the cell is filled only with atmospheric air. For the interface between air and mixture, the following applies $0 < \alpha_1 < 1$. In general, the value of α_1 can range from 0 to 1. In this text, the fluid mixture (α_1) will also have standard VOF designations like[14] *phase 1* or *fluid 1*. The same applies for the atmospheric air (α_2), i.e. *phase 2* or *fluid 2*.

The mixed fluid's properties, density ρ and apparent viscosity η, are weighted by the

[13] The volume occupied by the system is divided into discrete cells. All these cells make up the mesh.

[14] Relative to Section 2.3.1, Page 19, then within *phase 1*, namely within the fluid mixture (e.g. fresh concrete), the phase designations are *phase c* for the continuous phase (the matrix, e.g. mortar/mini concrete) and *phase d* for the dispersed phase (the suspended particles, e.g. coarse aggregates).

volume fractions α_1 and α_2 of the two fluids given by Eqs. (3.1) and (3.2) [25, 26]

$$\rho = \alpha_1 \rho_1 + \alpha_2 \rho_2 \tag{3.1}$$

$$\eta = \alpha_1 \eta_1 + \alpha_2 \eta_2 \tag{3.2}$$

In each and every computational cell, the following is always valid

$$\alpha_1 + \alpha_2 = 1, \tag{3.3}$$

meaning that if the quantity of phase 1 is known and given by α_1, then so is the quantity of phase 2 by $\alpha_2 = 1 - \alpha_1$. This means that it is sufficient to calculate only the interface advection for α_1. This interface is moved through the mesh and is captured by a *phase transport equation*. Relative to this specific equation, the VOF can be divided between two families, namely the *direct methods* and the *reconstruction methods* [25]. For the latter approach, the phase transport equation is approximated typically in two steps, first by a geometric interface reconstruction step and thereafter by an interface propagation step [25]. Examples of such approaches are the PLIC [27] and SLIC [28].

Unlike geometric interface reconstruction methods, the *direct methods* do not introduce geometrical representation of the interface, but rather try to maintain sharply defined interface by properly chosen discretization scheme, commonly known as *compressive differencing scheme* [29]. Example of such are the CICSAM [30, 31] and HRIC [32]. Another method, which could be considered to belong to the *direct methods* is the so-called Weller-scheme [33] (see also [19]). However, instead of using compressive differencing scheme like done in CICSAM, the compression of the interface is achieved by applying an extra compression term directly into the phase transport equation [19, 24]. This approach is used here and thus explained below.

3.3 Velocity

In VOF, the mixed velocity $\mathbf{U}$ is generally given (or defined) with the following equation [19, 23, 25, 33]

$$\mathbf{U} = \alpha_1 \mathbf{U}_1 + \alpha_2 \mathbf{U}_2 \tag{3.4}$$

Instead of the above equation, one could rather consider of use the mixed velocity given by Eq. (2.7) on Page 18. This would result in the following equation

$$\mathbf{U} = \frac{\alpha_1 \rho_1 \mathbf{U}_1 + \alpha_2 \rho_2 \mathbf{U}_2}{\alpha_1 \rho_1 + \alpha_2 \rho_2} \tag{3.5}$$

However, since the VOF is about the treatment of immiscible fluids (i.e. fluids that do not generally intermix), the difference between Eqs. (3.4) and (3.5) is only present at the thin interface region, between phase 1 and 2, namely at $0 < \alpha_1 < 1$. This is in contrary to the DFM in Chapter 2, which is about the treatment of miscible fluids (i.e. fluid that

intermix), meaning that the difference between Eqs. (3.4) and (3.5) is more or less always present everywhere in the fluid mixture.

Regardless of which equation is considered more appropriate, Eq. (3.4) or Eq. (3.5), it is important to note that except for the short derivation below, neither are explicitly used in VOF, meaning that the above-mentioned difference is not of paramount importance. This becomes clear below Eq. (3.15), where it is explained how the mixed velocity $\mathbf{U}$ is always solved as a single entity and not as presented with either of the two above equations.

The relative velocity between the phases 1 and 2 when using Eq. (3.4) is given by Eq. (3.6) (see also Eq. (2.15), Page 20, for the case of DFM).

$$\mathbf{U}_\mathrm{r} = \mathbf{U}_1 - \mathbf{U}_2 \tag{3.6}$$

When using Eq. (3.5), the above equation has to be defined in a different manner, as discussed in Footnote 15.

In the solver `interFoam`, the relative velocity $\mathbf{U}_\mathrm{r}$ is used to compress the interface between phase 1 and 2 (i.e. at $0 < \alpha_1 < 1$). However, since neither $\mathbf{U}_1$ nor $\mathbf{U}_2$ are actually resolved in VOF, $\mathbf{U}_\mathrm{r}$ is not calculated as nominally defined in Eq. (3.6). It is rather calculated by a semi-empirical formula as shown in the source code `alphaEqn.H` through the field variable `phir` $= \phi_r = \mathbf{U}_\mathrm{r} \cdot \mathbf{S}$, where the term $\mathbf{S}$ is the face area vector [34] (see also Footnote 22 about the vector direction of $\mathbf{S}$.).

3.4 Phase Transport Equation

The transport equation of each volume fraction α_1 and α_2 in a compressible two–fluid VOF system can be extracted from the Two-Fluid Method in Section 2.2, or more precisely from Eq. (2.4), Page 17. In terms of VOF quantities (e.g. $\beta_\mathrm{k} \to \alpha_\mathrm{i}$, $\rho_\mathrm{k} \to \rho_\mathrm{i}$ and $\mathbf{V}_\mathrm{k} \to \mathbf{U}_\mathrm{i}$), as well as putting $\Gamma_\mathrm{i} = 0$, where $i = 1, 2$, the following is obtained

$$\frac{\partial(\alpha_\mathrm{i}\,\rho_\mathrm{i})}{\partial t} + \nabla \cdot (\alpha_\mathrm{i}\,\rho_\mathrm{i}\,\mathbf{U}_\mathrm{i}) = 0 \tag{3.7}$$

From the above equation, the transport equation in a incompressible two–fluid VOF system (i.e. $\rho_\mathrm{i} = \text{constant}$) becomes as follows, where $i = 1, 2$ (see also [25])

$$\frac{\partial \alpha_\mathrm{i}}{\partial t} + \nabla \cdot (\alpha_\mathrm{i}\,\mathbf{U}_\mathrm{i}) = 0 \tag{3.8}$$

However, with $\alpha_2 = 1 - \alpha_1$ (c.f. Eq. (3.3)), it is sufficient to consider the transport equation of α_1 only. Therefore, with $i = 1$, Eq. (3.8) gives

$$\frac{\partial \alpha_1}{\partial t} + \nabla \cdot (\alpha_1\,\mathbf{U}_1) = 0 \tag{3.9}$$

To solve this transport equation, the velocity of phase 1 is needed, namely $\mathbf{U}_1$. In the much used original VOF method by Hirt and Nichols [5], the velocity $\mathbf{U}_1$ is assumed to

be equal to the mixed velocity $\mathbf{U}$ [5, 25]. This is only valid if α_1 is maintained as a step function throughout the domain, for example, numerical diffusion at the interface is not allowed [25].

By using the velocity $\mathbf{U}$ given by Eq. (3.4) and multiply it with α_1 as well as applying Eq. (3.3), the following is obtained

$$\alpha_1 \mathbf{U} = \alpha_1^2 \mathbf{U}_1 + \alpha_1(1-\alpha_1)\,\mathbf{U}_2 \tag{3.10}$$

Furthermore, by multiplying Eq. (3.6) with $\alpha_1(1-\alpha_1)$, one obtains:

$$\alpha_1(1-\alpha_1)\,\mathbf{U}_\mathrm{r} = \alpha_1(1-\alpha_1)\,(\mathbf{U}_1 - \mathbf{U}_2) \tag{3.11}$$

By adding Eq. (3.10) and Eq. (3.11) together, it can be shown after few steps[15] that [25]

$$\alpha_1\,\mathbf{U}_1 = \alpha_1\,\mathbf{U} + \mathbf{U}_\mathrm{r}\,\alpha_1\,(1-\alpha_1), \tag{3.12}$$

...or equally...

$$\mathbf{U}_1 = \mathbf{U} + \mathbf{U}_\mathrm{r}\,(1-\alpha_1) \tag{3.13}$$

With the above result, Eq. (3.9) can be converted to Eq. (3.14).

$$\frac{\partial \alpha_1}{\partial t} + \nabla\cdot(\alpha_1\,\mathbf{U}) + \nabla\cdot(\mathbf{U}_\mathrm{r}\,\alpha_1\,(1-\alpha_1)) = 0 \tag{3.14}$$

With the multiplication term $\alpha_1\,(1-\alpha_1)$, the compression term $\mathbf{U}_\mathrm{r}\,\alpha_1\,(1-\alpha_1)$ is only active in the thin interface region, between the fluid mixture and the atmospheric air $0 < \alpha_1 < 1$.

One of the critical issues with Eq. (3.14) is the discretization of the advection term $\nabla\cdot(\alpha_1\,\mathbf{U})$. Lower order schemes like the first order upwind method smear the interface due to numerical diffusion and higher order schemes are unstable, resulting in numerical oscillations [24]. Thus, it is necessary to apply special advection schemes that can contribute to a sharper interface and produce better monotonic profiles of the volume fraction α_1 [24]. To do this, the Flux Corrected Transport technique (FCT) is applied, which was introduced by Boris and Book [35] and later enhanced by Zalesak [36]. OpenFOAM implementation of FCT is named MULES (Multidimensional Universal Limiter for Explicit Solution) [16]. It is based on a similar concept relative to Zalesak's limiter λ, but its determination is iterative [16].

The FCT can be considered to be a *compressive differencing scheme* and thus has been used on Eq. (3.9), with $\mathbf{U}_1 = \mathbf{U}$, to maintain a sharp interface [24]. Therefore, with the special compression term in Eq. (3.14) (i.e. by $\mathbf{U}_\mathrm{r}$) and with the use of MULES (i.e. FCT), a double compression is actually being applied in the last-mentioned equation. Here, the FCT is applied on both advection terms in Eq. (3.14).

[15] Using the same procedure with Eq. (3.5) and defining the relative velocity between phases with the following $\rho\mathbf{U}_\mathrm{r} = \rho_1\mathbf{U}_1 - \rho_2\mathbf{U}_2$, one obtains $\alpha_1\rho_1\mathbf{U}_1 = \alpha_1\rho\,\mathbf{U} + \mathbf{U}_\mathrm{r}\,\rho\,\alpha_1\,(1-\alpha_1)$ in the end. Using this equation instead of Eq. (3.12), in Eq. (3.7) did not give any beneficial outcome for the tested simulations.

3.5 Governing Equation

For non-Newtonian fluids, like what applies to cement based materials, the governing equation is the Cauchy equation of motion, given by Eq. (3.15) [20, 21]. Being a "parent" of the Navier–Stokes equations, the Cauchy equation is also fully valid for Newtonian fluids like the atmospheric air.

$$\frac{\partial(\rho\mathbf{U})}{\partial t} + \nabla\cdot(\rho\,\mathbf{U}\mathbf{U}) = -\nabla p + \nabla\cdot\mathbf{T} + \rho\,\mathbf{g} + \mathbf{F}_{\mathrm{s}} \tag{3.15}$$

Since the VOF method is a single pressure system [23], the pressure p in Eq. (3.15) is not calculated separately for each of the phases α_1 and α_2 (i.e. for concrete and air). The same applies for the velocity $\mathbf{U}$, which is strictly speaking given by Eq. (3.4) (or by Eq. (3.5), depending on preference). That is, the mixed velocity $\mathbf{U}$ is solved as a single entity by Eq. (3.15). The density ρ in Eq. (3.15) is given by Eq. (3.1), the term t represents the time and $\mathbf{g}$ is the gravity. The term $\mathbf{F}_{\mathrm{s}}$ is the force by surface tension between the two phases α_1 and α_2 (e.g. fresh concrete and atmospheric air), and is calculated in accordance with the Continuum-Surface-Force (CSF) model of Brackbill et al. [37]. The above terms will be further discussed in Section 4.8. The extra stress tensor $\mathbf{T}$ is explained in Section 3.6.

3.6 Constitutive Equation

The constitutive equation consists of the Generalized Newtonian Model [38], or in short GNM and is given by $\mathbf{T} = 2\,\eta\,\dot{\boldsymbol{\varepsilon}}$ [2]. The term $\dot{\boldsymbol{\varepsilon}} = \frac{1}{2}(\nabla\mathbf{U} + (\nabla\mathbf{U})^{\mathrm{T}})$ is known as the rate–of–deformation tensor [20, 21, 39]. Here, the apparent viscosity η by is given by Eq. (3.2), in which the fluid mixture (or phase 1, e.g. fresh concrete) is rheologically modeled through η_1, while the atmospheric air (i.e. phase 2) is always set as a Newtonian fluid $\eta_2 =$ constant. The computational implementation of η_1 into the source code is by the regularization approach [3, 40, 41, 42, 43, 44, 45, 46]. See also Section 5.2, Page 50, for further information about the regularization approach.

Chapter 4

Combining DFM with VOF

4.1 Introduction

This chapter deals with combining the theories of the Drift Flux Model, or DFM (Chapter 2) and the Volume of Fluid Method, or VOF (Chapter 3). That is, this chapter represents the basis for the `vvpfFoam` solver. As a short clarification, the VOF tackles the treatment of immiscible fluids (i.e. fluids that do not intermix), while DFM tackles the treatment of miscible fluids (i.e. fluid that intermix). In addition to this, the DFM has the capability to allow slip between phases (see Section 2.3.1, Page 19). This is an important aspect to allow the fluid mixture to segregate by gravitational settling and/or by other means, like by the shear (rate) induced particle migration.

4.2 The Mixture of Phase 1

In Chapter 3, the phase 1 (i.e. the fluid mixture) and phase 2 (e.g. atmospheric air) were treated by the VOF. To reiterate, the volume fraction of phase 1 within each computational cell is represented with α_1, while the volume fraction of phase 2 is represented with α_2. More precisely, $\alpha_1 = V_1/V_\mathrm{P}$, where V_P is the volume of the computational cell and V_1 is the volume of mixture (e.g. fresh concrete) within the cell (i.e. $V_1 \leq V_\mathrm{P}$). Likewise, $\alpha_2 = V_2/V_\mathrm{P}$, where V_2 is the volume of atmospheric air within the same cell (i.e. $V_2 \leq V_\mathrm{P}$), as shown in Fig. 4.1.

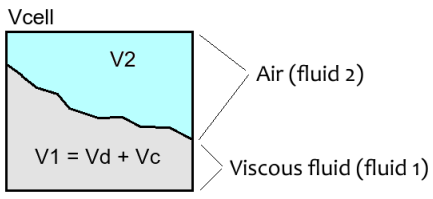


Figure 4.1: The division of a computational cell between phase 2 (atmospheric air) and phase 1 (mixture). The latter is further divided between the dispersed phase d and the continuous phase c.

Since phase 1 will be treated with the DFM as described in Chapter 2, the main subscript of that chapter, namely m for "mixture" (e.g. V_m for mixture volume) will also have the subscript 1, in accordance with the scheme of VOF in Chapter 3 (e.g. V_1 for the volume of phase 1). This means that the mixture mass m_m is also the mass of phase 1, namely m_1, the mixture velocity $\mathbf{V}_\mathrm{m}$ is also the velocity of phase 1, which is designated with $\mathbf{U}_1$ in Chapter 3. That is, $V_\mathrm{m} \equiv V_1$, $m_\mathrm{m} \equiv m_1$, $\mathbf{V}_\mathrm{m} \equiv \mathbf{U}_1$, $\rho_\mathrm{m} \equiv \rho_1$, and so forth.

4.3 Fundamental Relations

As indicated above, *phase 1* will consist of two phases, namely the continuous phase, or *phase c* (i.e. the matrix) and the dispersed phase, or *phase d* (i.e. suspended particles). As explained in Chapter 2, their corresponding volume fractions are represented with β_c and β_d. More precisely, from Eq. (2.12), Page 19, then $\beta_\mathrm{c} = V_\mathrm{c}/V_\mathrm{m}$ and $\beta_\mathrm{d} = V_\mathrm{d}/V_\mathrm{m}$, in which $V_\mathrm{m} \equiv V_1$, c.f. Section 4.2. The term V_c is the volume of the continuous phase within the cell (i.e. $V_\mathrm{c} \leq V_1 \leq V_\mathrm{P}$) and V_d is the volume of the dispersed phase within the cell (i.e. $V_\mathrm{d} \leq V_1 \leq V_\mathrm{P}$).

Summarizing the fundamental relations for α_1 and α_2 (from Chapter 3):

$$V_\mathrm{P} = V_1 + V_2 \quad \wedge \quad \alpha_1 = \frac{V_1}{V_\mathrm{P}} \quad \wedge \quad \alpha_2 = \frac{V_2}{V_\mathrm{P}} \tag{4.1}$$

$$\alpha_1 + \alpha_2 = \frac{V_1}{V_\mathrm{P}} + \frac{V_2}{V_\mathrm{P}} = \frac{V_1 + V_2}{V_\mathrm{P}} = \frac{V_\mathrm{P}}{V_\mathrm{P}} = 1 \tag{4.2}$$

Summarizing the fundamental relations for β_c and β_d (from Chapter 2):

$$V_\mathrm{m} \equiv V_1 = V_\mathrm{c} + V_\mathrm{d} \quad \wedge \quad \beta_\mathrm{c} = \frac{V_\mathrm{c}}{V_1} \quad \wedge \quad \beta_\mathrm{d} = \frac{V_\mathrm{d}}{V_1} \tag{4.3}$$

$$\beta_\mathrm{d} + \beta_\mathrm{c} = \frac{V_\mathrm{d}}{V_1} + \frac{V_\mathrm{c}}{V_1} = \frac{V_\mathrm{d} + V_\mathrm{c}}{V_1} = \frac{V_1}{V_1} = 1 \tag{4.4}$$

Because of the combination of VOF with DFM, additional definitions are needed for the continuous phase (i.e. phase c) and the dispersed phase (i.e. phase d). These are α_c and α_d, representing the volume fractions relative to the volume of a cell V_P, given by

$$\alpha_\mathrm{c} = V_\mathrm{c}/V_\mathrm{P} \quad \wedge \quad \alpha_\mathrm{d} = V_\mathrm{d}/V_\mathrm{P} \tag{4.5}$$

The sum of these two new quantities is equal to the volume fraction of the mixture, namely

$$\alpha_\mathrm{c} + \alpha_\mathrm{d} = \frac{V_\mathrm{c}}{V_\mathrm{P}} + \frac{V_\mathrm{d}}{V_\mathrm{P}} = \frac{V_\mathrm{c} + V_\mathrm{d}}{V_\mathrm{P}} = \frac{V_1}{V_\mathrm{P}} = \alpha_1 \tag{4.6}$$

Repeating the outcome of Eqs. (4.2), (4.4) and (4.6), gives:

$$\alpha_1 + \alpha_2 = 1 \quad \wedge \quad \beta_\mathrm{d} + \beta_\mathrm{c} = 1 \quad \wedge \quad \alpha_\mathrm{d} + \alpha_\mathrm{c} = \alpha_1 \tag{4.7}$$

Finally, the relationship between the two quantities $\beta_d = V_d/V_1$ and $\alpha_d = V_d/V_P$ is as follows

$$\beta_d = \frac{V_d}{V_1} = \frac{V_P}{V_P}\frac{V_d}{V_1} = \frac{V_P}{V_1}\frac{V_d}{V_P} = \frac{1}{\alpha_1}\,\alpha_d \tag{4.8}$$

The same can be concluded for the continuous phase, meaning $\beta_c = \alpha_c/\alpha_1$.

4.4 Mass and Density

The term m_P represents the total mass of materials inside a single computational cell and is equal to m_1+m_2. The term m_1 represents the mass of phase 1 (i.e. mixture) within each computational cell, while the term m_2 represents the mass of phase 2 (i.e. atmospheric air) within the same cell.

The mass of phase 1, namely m_1, is equal to $m_c + m_d$. The term m_c represents the mass of *phase c* (i.e. matrix) within the above-mentioned computational cell, while the term m_d represents the mass of *phase d* (i.e. suspended particles) within the same cell. Summarizing the above:

$$m_P = m_1 + m_2 \qquad \wedge \qquad m_1 = m_c + m_d \tag{4.9}$$

The term ρ_1 is the density of phase 1, while the term ρ_2 represents the density of phase 2:

$$\rho_1 = \frac{m_1}{V_1} \equiv \frac{m_m}{V_m} = \rho_m \quad (Mixture) \qquad \wedge \qquad \rho_2 = \frac{m_2}{V_2} = \text{constant} \quad (Air) \tag{4.10}$$

In the above, the following were used $m_m \equiv m_1$ and $V_m \equiv V_1$, c.f. Section 4.2.

The term ρ_c is the density of *phase c* (matrix), while the term ρ_d represents the density of *phase d* (suspended particles):

$$\rho_c = \frac{m_c}{V_c} = \text{constant} \qquad \wedge \qquad \rho_d = \frac{m_d}{V_c} = \text{constant} \tag{4.11}$$

A typical density value for the coarse aggregates is $\rho_d = 2700\,\text{kg/m}^3$ (dispersed phase), while $\rho_c = 2200\,\text{kg/m}^3$ for the mortar/mini concrete (continuous phase).

In this work, the term ρ represents the total density of the cell (that is, including both mixture and atmospheric air), given by

$$\begin{aligned}\rho &= \frac{m_P}{V_P} = \frac{m_1+m_2}{V_P} = \frac{m_1}{V_P} + \frac{m_2}{V_P} = \frac{m_1}{V_P}\frac{V_1}{V_1} + \frac{m_2}{V_P}\frac{V_2}{V_2} = \\ &= \frac{V_1}{V_P}\frac{m_1}{V_1} + \frac{V_2}{V_P}\frac{m_2}{V_2} = \alpha_1\,\rho_1 + \alpha_2\,\rho_2\end{aligned} \tag{4.12}$$

The above result corresponds to Eq. (3.1). Likewise, the density of phase 1 (e.g. of the fresh concrete), can be calculated by

$$\begin{aligned}\rho_1 &= \frac{m_1}{V_1} = \frac{m_d+m_c}{V_P\,\alpha_1} = \frac{V_d\,\rho_d + V_c\,\rho_c}{V_P\,\alpha_1} = \frac{(V_d/V_P)\,\rho_d + (V_c/V_P)\,\rho_c}{\alpha_1} \\ &= \frac{\alpha_d\,\rho_d + \alpha_c\,\rho_c}{\alpha_1} = \frac{\alpha_d\,\rho_d + (\alpha_1-\alpha_d)\,\rho_c}{\alpha_1}\end{aligned} \tag{4.13}$$

An alternative method of calculating the density of phase 1 is by

$$\begin{aligned}\rho_1 &= \frac{\alpha_\mathrm{d}\,\rho_\mathrm{d} + \alpha_\mathrm{c}\,\rho_\mathrm{c}}{\alpha_1} = \frac{\frac{V_1}{V_1}\frac{V_\mathrm{d}}{V_\mathrm{P}}\,\rho_\mathrm{d} + \frac{V_1}{V_1}\frac{V_\mathrm{c}}{V_\mathrm{P}}\,\rho_\mathrm{c}}{\alpha_1} = \frac{\frac{V_\mathrm{d}}{V_1}\frac{V_1}{V_\mathrm{P}}\,\rho_\mathrm{d} + \frac{V_\mathrm{c}}{V_1}\frac{V_1}{V_\mathrm{P}}\,\rho_\mathrm{c}}{\alpha_1} \\ &= \frac{\frac{V_\mathrm{d}}{V_1}\,\alpha_1\,\rho_\mathrm{d} + \frac{V_\mathrm{c}}{V_1}\,\alpha_1\,\rho_\mathrm{c}}{\alpha_1} = \beta_\mathrm{d}\,\rho_\mathrm{d} + \beta_\mathrm{c}\,\rho_\mathrm{c} = \beta_\mathrm{d}\,\rho_\mathrm{d} + (1-\beta_\mathrm{d})\,\rho_\mathrm{c}\end{aligned} \tag{4.14}$$

In the above derivations, Eqs. (4.1) to (4.7) were used.

4.5 Mass Conservation of the Dispersed Phase

4.5.1 Original State of Equation

The mass conservation of the dispersed phase, *phase d*, was derived in Section 2.3.2, Page 22, and is reproduced below (see Eq. (2.41)):

$$\frac{\partial \beta_\mathrm{d}}{\partial t} + \nabla\cdot(\beta_\mathrm{d}\mathbf{U}_1) + \nabla\cdot\left(\frac{\beta_\mathrm{d}\rho_\mathrm{c}}{\rho_\mathrm{m}}\mathbf{V}_\mathrm{dj}\right) = 0 \tag{4.15}$$

In the above, the mixture velocity $\mathbf{V}_\mathrm{m}$ is replaced by the velocity of phase 1, namely by $\mathbf{U}_1$ (see Section 4.2). Moreover, by using Eq. (3.13), Page 29, the above equation is converted into the following

$$\frac{\partial \beta_\mathrm{d}}{\partial t} + \nabla\cdot(\beta_\mathrm{d}\mathbf{U} + \beta_\mathrm{d}\frac{\rho_\mathrm{c}}{\rho_1}\mathbf{V}_\mathrm{dj}) + \nabla\cdot(\mathbf{U}_\mathrm{r}\,\beta_\mathrm{d}\,(1-\alpha_1)) = 0 \tag{4.16}$$

The problem with the term "$\mathbf{U}_\mathrm{r}\,\beta_\mathrm{d}\,(1-\alpha_1)$" in the above equation, is that β_d is a variable that is in general not at its minimum (e.g. 0) nor maximum (e.g. 0.4) value. That is, it can (and should be) in any range between the two extremes anywhere in the mixture and this includes the interface. With this property, this term cannot contribute to interface compression. This is contrary to the variable α_1 which is fundamentally either equal to 0 or 1 and means that the traditional `interFoam`–term "$\mathbf{U}_\mathrm{r}\,\alpha_1\,(1-\alpha_1)$" is at maximum at the surface and thus can contribute to interface compression. But the problem is that this term will serve as an artificial source/sink inside Eq. (4.16) and thus cannot be used as is. To solve this, an empirical modification of the term "$\mathbf{U}_\mathrm{r}\,\beta_\mathrm{d}\,(1-\alpha_1)$" is rather applied, in which the term β_d is replaced with $\xi_\mathrm{d} = \xi_\mathrm{d}(\alpha_1, \alpha_\mathrm{d})$. More precisely, the following conversion is done in Eq. (4.16)

$$\mathbf{U}_\mathrm{r}\,\beta_\mathrm{d}\,(1-\alpha_1) \Rightarrow \mathbf{U}_\mathrm{r}\,\xi_\mathrm{d}\,(1-\alpha_1) \tag{4.17}$$

At the time of writing, the function ξ_d has the following form

$$\xi_\mathrm{d}(\alpha_1, \alpha_\mathrm{d}) = \frac{\alpha_1\,\alpha_\mathrm{d}}{\alpha_\mathrm{d}^{\mathrm{MAX}}} \tag{4.18}$$

The above function is based on trial and error, with the constraint to obtain good interface compression and also with emphasis on obtaining the best mass conservation. At present, the same interface compression velocity $\mathbf{U}_\mathrm{r}$ is used in Eq. (4.17) as in Eq. (3.14), Page 29.

4.5.2 Dispersed Phase Relative to V_P

With Eqs. (4.8) and (4.17), Eq. (4.16) is transformed into the following

$$\frac{\partial(\alpha_d/\alpha_1)}{\partial t} + \nabla\cdot(\frac{\alpha_d}{\alpha_1}\mathbf{U} + \frac{\alpha_d}{\alpha_1}\frac{\rho_c}{\rho_1}\mathbf{V}_{dj}) + \nabla\cdot(\mathbf{U}_r\,\xi_d\,(1-\alpha_1)) = 0 \tag{4.19}$$

With Eq. (4.19), the variable α_d (which is relative to the volume of the cell V_P) is solved and not the variable β_d (which is relative to the volume of phase 1, namely V_1).

In this work, attempt has been made to solve Eq. (4.19) for α_d, while keeping α_1 constant at a value from the previous iteration. The solver does run, but mass conservations are not kept. Because of this, a changed version of Eq. (4.19) is rather used, obtained by fixing α_1 to 1. This modified version[16] is shown below:

$$\frac{\partial\alpha_d}{\partial t} + \nabla\cdot(\alpha_d\mathbf{U} + \alpha_d\frac{\rho_c}{\rho_1}\mathbf{V}_{dj}) + \nabla\cdot(\mathbf{U}_r\,\xi_d\,(1-\alpha_1)) = 0 \tag{4.20}$$

The term "$\mathbf{U}_r\,\xi_d\,(1-\alpha_1)$" acts to some extent as a source/sink, and thus is responsible for artificial changes in quantity of α_d. In general, with more abrupt changes in the flow $\mathbf{U}$, the larger variation of $\mathbf{U}_r$ becomes, contributing to a larger source "$\nabla\cdot(\mathbf{U}_r\,\xi_d\,(1-\alpha_1))$" (see Figs. 1.11 and 1.13, Page 12).

If the term "$\mathbf{U}_r\,\xi_d\,(1-\alpha_1)$" is omitted in Eq. (4.20), a better conservation is obtained. However, without it, then for volatile/plunging flow like shown in Fig 1.3, Page 4, the term α_d can push through the interface, appearing to evaporate and float in the atmospheric air (the code shown later in Section 4.7 is also meant to assist in avoiding this for the case of such abrupt flow).

It should be clear that Eq. (4.20) is correct within the phase 1 fluid, where $\alpha_1 = 1$ (i.e. is identical to Eq. (4.19) for such a case). However, at the interface between the atmospheric air (phase 2) and the mixture fluid (phase 1), namely at $0 < \alpha_1 < 1$, it becomes theoretically less correct, which could introduce a slight error in the simulation.

4.5.3 Dispersed Phase Relative to V_1 or V_P

It should be clear that an extensive[17] cell based variable (i.e. a variable calculated at a nodal point P) is relative to the cell volume V_P (unless a special treatment is applied to it). Thus, with the attempt to solve for β_d through Eq. (4.16), one would actually solve for α_d with that exact same equation. This is because the former variable is relative to V_1, while the latter variable is relative to V_P, a normalization that is native to FVM. In Chapter 2, when only treating the DFM, the mixture volume $V_1 \equiv V_m$ is always equal to the cell volume V_P, making the above discussion irrelevant.

[16] In the file `alphaDEqn.H`, a source term $S_p\,\alpha_d + S_u$ is added to the right side of Eq. (4.20), in which both S_p and S_u are set equal to zero (units of S_p and S_u are $[s^{-1}]$).

[17] An extensive variable is one whose magnitude is dependent of the size of the system (examples: volume, mass, heat capacity). An intensive variable is one whose magnitude is independent of the size of the system (examples: temperature, pressure, specific heat capacity).

The reason for going through the steps of generating Eq. (4.19) and thereafter transforming it to Eq. (4.20), is out of necessity to maintain conservation of the quantity of the dispersed phase (i.e. keeping it more or less constant throughout the simulation). This also helps to understand the (necessary) physical error involved in the solver `vvpfFoam`.

4.6 Phase Transport Equation

The transport equation of each volume fraction α_1 and α_2 can be extracted from Eq. (3.7) in Section 3.4 and is reproduced below:

$$\frac{\partial(\alpha_\mathrm{i}\,\rho_\mathrm{i})}{\partial t} + \nabla\cdot(\alpha_\mathrm{i}\,\rho_\mathrm{i}\,\mathbf{U}_\mathrm{i}) = 0 \tag{4.21}$$

As mentioned in Section 3.4, with $\alpha_2 = 1-\alpha_1$, it is sufficient to consider the transport equation of α_1 only and as such, only applying $i = 1$ to Eq. (4.21), results in

$$\frac{\partial(\alpha_1\,\rho_1)}{\partial t} + \nabla\cdot(\alpha_1\,\rho_1\,\mathbf{U}_1) = 0 \tag{4.22}$$

Furthermore, by applying Eq. (3.12), the following is obtained

$$\frac{\partial(\alpha_1\,\rho_1)}{\partial t} + \nabla\cdot(\alpha_1\,\rho_1\,\mathbf{U}) + \nabla\cdot(\mathbf{U}_\mathrm{r}\,\rho_1\,\alpha_1\,(1-\alpha_1)) = 0 \tag{4.23}$$

The above equation is solved[18] in `alpha1EqnRho.H`. By applying it, there emerges a slight volume change in phase 1, say 0.3% or so, during a typical simulation run (see Figs. 1.11 and 1.13, Page 12). If this is unacceptable, the user can rather use `alpha1Eqn.H`, which returns better conservation. However, that file solves Eq. (4.23) with $\rho_1 =$ constant, which is not in accordance with Eq. (4.13).

It should be clear that there is no difference visually observable in the overall flow, when comparing the use of `alpha1Eqn.H` with the use of `alpha1EqnRho.H`. An example of this is shown in Fig. 4.2, which demonstrates the "cake break" flow through pillars obstacles (see Section 1.4.1 on Page 5, about the case setup). The simulation is split into two parts, in which the left part (green) is solved by Eq. (4.23) as is (i.e. with `alpha1EqnRho.H`), while the right part (white) is solved with the same equation however with $\rho_1 =$ constant (i.e. with `alpha1Eqn.H`). As shown, the left (green) and the right (white) part meet at the center, exactly. The overall volume change for phase 1 (i.e. of α_1) is less than 0.2% for the left simulation, while 0% for the right simulation. In these two simulations, no difference is observed in $\alpha_\mathrm{d} = \alpha_\mathrm{d}(x,y,z,t)$, solved by Eq. (4.20) (as well as none in terms of $\beta_\mathrm{d} = \alpha_\mathrm{d}/\alpha_1$ by Eq. (4.8)). The total simulation time is 30 s, which means "Time index: 300" relative to Fig. 4.2.

Whichever source is used, the change is made in the file `macroDefinitions.H`, set by the macro definition `ALPHA1RHO_SOLVE`. If defined, then `alpha1EqnRho.H` is used, and if not, `alpha1Eqn.H` is used instead. Unless otherwise stated, the results shown in this report are based on use of `alpha1EqnRho.H`.

[18] In the file `alpha1EqnRho.H`, a source term $S_\mathrm{p}\,\alpha_1 + S_\mathrm{u}$ is added to the right side of Eq. (4.23), in which both S_p and S_u are set equal to zero (units of S_p and S_u are $[\mathrm{kg}\cdot\mathrm{m}^{-3}\cdot\mathrm{s}^{-1}]$).

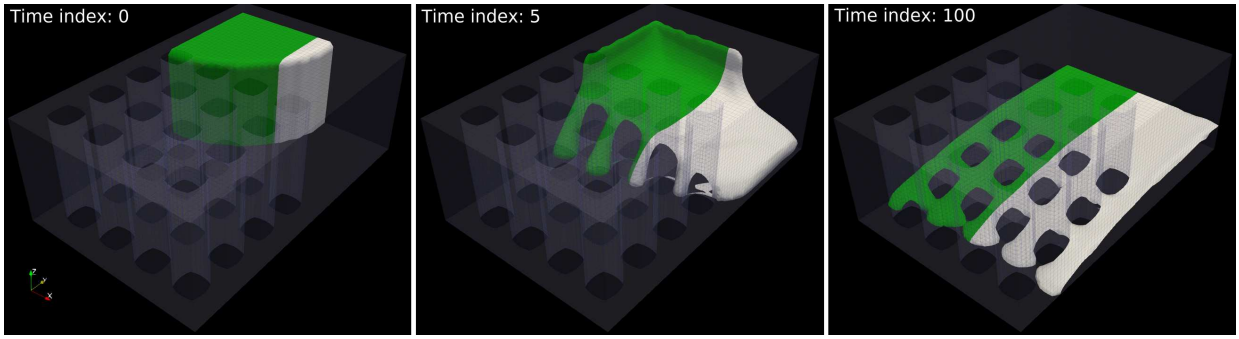


Figure 4.2: Simulation of a "cake break" flow through pillars obstacles (see also Fig. 1.4). The left part (green) is solved with `alpha1EqnRho.H`, while the right part (white) is solved with `alpha1Eqn.H`.

4.7 Visualization of β_d Relative to V_P

When looking at the transport equation of phase 1, namely Eq. (4.23), it is clear that α_1 is moved (i.e. convected) by the velocity $\mathbf{U}$. But, when looking at the transport equation of the dispersed phase, namely Eq. (4.20), its quantity α_d is not convected by $\mathbf{U}$ alone, but rather by the sum $\mathbf{U} + (\rho_c/\rho_1)\,\mathbf{V}_{dj}$. Usually, the convection is almost the same in both cases of α_1 and α_d since generally $\mathbf{U} \approx \mathbf{U} + (\rho_c/\rho_1)\,\mathbf{V}_{dj}$. But in some part of the flow, the difference between $\mathbf{U}$ and $\mathbf{U} + (\rho_c/\rho_1)\,\mathbf{V}_{dj}$ can be such that the variables α_d and α_1 propagate differently. Although the consequence of this for either α_1 or α_d is none[19], the effect of this for $\beta_d = \alpha_d/\alpha_1$ (Eq. (4.8), Page 33) is that it can reach an abnormal high value. This applies especially near/at the interface, where the variation in α_1 is greatest. This can also apply where the difference between $\mathbf{U}$ and $\mathbf{U} + (\rho_c/\rho_1)\,\mathbf{V}_{dj}$ is high, as a consequence of a new additional interface compression scheme used for the dispersed phase α_d (beyond what is explained in Section 4.5.2). This new interface compression is implemented in `driftVelocity.H` and `gravitySegregation.H` and is as follows:

```
volVectorField gradAlpha1(fvc::grad(alpha1));
surfaceVectorField gradAlpha1f(fvc::interpolate(gradAlpha1));
surfaceVectorField interfaceNormal(gradAlpha1f/(mag(gradAlpha1f) + deltaN));

forAll(alpha1.internalField(), celli)
{
    if
    (
        alpha1[celli] > lowerCrit.value()
     && alpha1[celli] < upperCrit.value()
     && alphaD[celli] > criteriaD.value()
    )
    {
        VdjGR[celli] = (1.0 - alpha1[celli])*0.2100*interfaceNormal[celli];
```

[19] Both α_d and α_1 have their own partial differential equation, namely Eqs. (4.20) and (4.23), that can in a sense operate independent of each other.

```
    }
    else if (alpha1[celli] <= lowerCrit.value())
    {
        VdjGR[celli] = (1.0 - alpha1[celli])*0.0306*g.value();
    }
}
```

Convection by $\mathbf{U}$ from a cell depleted of α_1 into a neighboring cell, with the concomitant convection by $\mathbf{U} + (\rho_c/\rho_1)\,\mathbf{V}_{dj}$, from an another cell full of α_d, into the same neighboring cell, results in build up of $\beta_d = \alpha_d/\alpha_1$. This particular build up has mostly no consequence other than being visually annoying (i.e. since β_d is calculated and not solved by a partial differential equation, as applies for α_1 and α_d).

An example of the above-mentioned properties of β_d is shown in Fig. 4.3, marked with a green square. For this cell, the volume fraction of phase 1 is $\alpha_1 = 0.5$, while the volume fraction of the dispersed phase is $\alpha_d = 0.2$. This results in too high value of $\beta_d = 0.4$ shown in the figure, which might be incorrectly understood as mass generation at the interface.

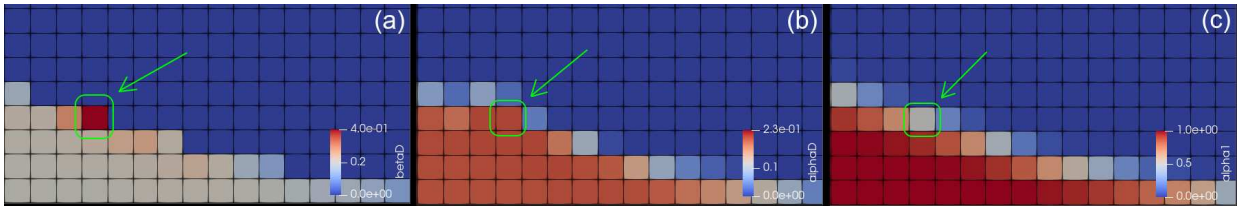


Figure 4.3: Example of β_d (a), α_d (b) and α_1 (c) distribution of the same case.

To avoid singularity in the code `UEqn.H`, the calculation of $\beta_d/(1-\beta_d)$ is performed, as opposed to $\alpha_d/(\alpha_1 - \alpha_d)$, and the above-mentioned behavior of β_d might disturb the simulation somewhat at the interface between atmospheric air and mixture. The relevant part of the `UEqn.H` code is as follows (see also Eq. (4.33)):

```
    volScalarField alphaDrho1Ratio
    (
        "alphaDrho1Ratio",
        (betaD/(scalar(1) - betaD))*((rhoC*rhoD)/rho1)
        // (alphaD/((alpha1 + delta) - alphaD))*((rhoC*rhoD)/rho1)
    );
```

4.8 Governing Equation

4.8.1 Momentum Equation for Phases 1 and 2 (VOF)

The governing equation for the combined system of phases 1 (mixture) and 2 (atmospheric air) is solved by the VOF, given by Eq. (3.15), Page 30, and reproduced below:

$$\frac{\partial(\rho\mathbf{U})}{\partial t} + \nabla\cdot(\rho\,\mathbf{U}\mathbf{U}) = -\nabla p + \nabla\cdot\mathbf{T} + \rho\,\mathbf{g} + \mathbf{F}_s \tag{4.24}$$

As mentioned in Chapter 3, the mixed fluid's properties, density ρ and apparent viscosity η, are weighted by the volume fractions α_1 and α_2 [25, 26]. These were given by Eqs. (3.1), (3.2) and (3.4), and are reproduced below[20]:

$$\rho = \alpha_1 \rho_1 + \alpha_2 \rho_2 \tag{4.25}$$

$$\eta = \alpha_1 \eta_1 + \alpha_2 \eta_2 \tag{4.26}$$

$$\mathbf{U} = \alpha_1 \mathbf{U}_1 + \alpha_2 \mathbf{U}_2 \tag{4.27}$$

With the extra stress tensor $\mathbf{T} = 2\,\eta\,\dot{\boldsymbol{\varepsilon}}$ and the rate–of–deformation tensor $\dot{\boldsymbol{\varepsilon}} = \frac{1}{2}(\nabla\mathbf{U} + (\nabla\mathbf{U})^{\mathrm{T}})$ (see Section 3.6), combined with Eqs. (4.26) and (4.27), at $\alpha_1 = 1$ (i.e. inside the phase 1 fluid), the following is obtained

$$\mathbf{T} = \alpha_1 \mathbf{T}_1 = 2\,\eta_1\,\dot{\boldsymbol{\varepsilon}}_1 = 2\,\eta_1 \left[\frac{1}{2}(\nabla\mathbf{U}_1 + (\nabla\mathbf{U}_1)^{\mathrm{T}})\right] \tag{4.28}$$

Furthermore, inside the mixture fluid, the pressure can be designated with p_1, meaning

$$p|_{\alpha_1=1} = p_1 \tag{4.29}$$

Finally, the term $\mathbf{F}_{\mathrm{s}}$ is the force by surface tension between the two phases α_1 and α_2 [37], and is thus only active at the thin interface region, namely at $0 < \alpha_1 < 1$. That is, inside phase 1 at $\alpha_1 = 1$, then $\mathbf{F}_{\mathrm{s}} = 0$.

$$\mathbf{F}_{\mathrm{s}}|_{\alpha_1=1} = 0 \tag{4.30}$$

With Eqs. (4.25) to (4.30) at $\alpha_1 = 1$ (i.e. inside the phase 1 fluid), Eq. (4.24) is transformed into the following

$$\frac{\partial(\rho_1 \mathbf{U}_1)}{\partial t} + \nabla\cdot(\rho_1\,\mathbf{U}_1\mathbf{U}_1) = -\nabla p_1 + \nabla\cdot\mathbf{T}_1 + \rho_1\,\mathbf{g} \tag{4.31}$$

The above is the governing equation that is valid inside the phase 1 fluid ($\alpha_1 = 1$) relative to the VOF method.

4.8.2 Momentum Equation for Phase 1 (DFM)

The governing equation that is valid inside the phase 1 fluid ($\alpha_1 = 1$) relative to the DFM method is given by Eq. (2.60), Page 25, and reproduced below:

$$\frac{\partial(\rho_1 \mathbf{U}_1)}{\partial t} + \nabla\cdot(\rho_1\mathbf{U}_1\mathbf{U}_1) + \nabla\cdot\left(\left[\frac{\beta_{\mathrm{d}}}{1-\beta_{\mathrm{d}}}\frac{\rho_{\mathrm{c}}\rho_{\mathrm{d}}}{\rho_1}\right]\mathbf{V}_{\mathrm{dj}}\mathbf{V}_{\mathrm{dj}}\right) = -\nabla p_1 + \nabla\cdot\mathbf{T}_1 + \rho_1\mathbf{g} \tag{4.32}$$

In the above, the mixture velocity $\mathbf{V}_{\mathrm{m}}$ has been replaced with the velocity of phase 1, namely with $\mathbf{U}_1$ and the mixture density ρ_{m} with ρ_1. Also, the extra stress tensor $\mathbf{T}_{\mathrm{m}}$ has been replaced with $\mathbf{T}_1$ and mixture pressure p_{m} with p_1. All these changes are in accordance with Section 4.2.

[20] In relation to the velocity $\mathbf{U}$, see also the text in Sections 3.3 and 3.5, Page 27.

4.8.3 Momentum Equation for Phases 1 and 2 (VOF & DFM)

It should be clear that Eq. (4.31) is part of Eq. (4.24). The latter equation applies to the combined system of phase 1 and 2 fluids, while the former applies only for the phase 1 fluid. Moreover, with Eq. (4.31) (and thus Eq. (4.24)), the phase 1 fluid cannot segregate, meaning that slippage between *phase c* and *phase d* cannot occur inside it.

By comparison of Eq. (4.31) (which is a part of Eq. (4.24)) with Eq. (4.32), it can be suggested that the component needed in Eq. (4.24), to allow for slippage between *phase c* and *phase d*, is given by the following

$$\nabla\cdot\left(\left[\frac{\beta_\mathrm{d}}{1-\beta_\mathrm{d}}\frac{\rho_\mathrm{c}\rho_\mathrm{d}}{\rho_1}\right]\mathbf{V}_\mathrm{dj}\mathbf{V}_\mathrm{dj}\right)=\nabla\cdot\left(\left[\frac{\alpha_\mathrm{d}}{\alpha_1-\alpha_\mathrm{d}}\frac{\rho_\mathrm{c}\rho_\mathrm{d}}{\rho_1}\right]\mathbf{V}_\mathrm{dj}\mathbf{V}_\mathrm{dj}\right) \tag{4.33}$$

Including Eq. (4.33) into the left side of Eq. (4.24), results in the following

$$\frac{\partial(\rho\mathbf{U})}{\partial t}+\nabla\cdot(\rho\,\mathbf{U}\mathbf{U})+\nabla\cdot\left(\left[\frac{\beta_\mathrm{d}}{1-\beta_\mathrm{d}}\frac{\rho_\mathrm{c}\rho_\mathrm{d}}{\rho_1}\right]\mathbf{V}_\mathrm{dj}\mathbf{V}_\mathrm{dj}\right)=-\nabla p+\nabla\cdot\mathbf{T}+\rho\,\mathbf{g}+\mathbf{F}_\mathrm{s} \tag{4.34}$$

Because $\beta_\mathrm{d}=0\wedge\alpha_\mathrm{d}=0$ when $\alpha_1=0$, Eq. (4.34) returns to Eq. (4.24) in phase 2 (i.e. in the atmospheric air). Furthermore, for $\alpha_1=1$ (i.e. inside the mixture), Eq. (4.34) returns to Eq. (4.32) (keeping Eqs. (4.29) and (4.30) in mind).

Modified Pressure

Here, the (total) pressure p is substituted by a modified version of it, namely by p_rgh. The implications and benefits of the modified pressure p_rgh is well explained in [19, 23, 25]. The relationship between p and p_rgh is given by

$$p=p\_\mathrm{rgh}+\rho\,\mathbf{g}\cdot\mathbf{x}+\mathrm{pRef} \tag{4.35}$$

The term pRef is a (constant) reference pressure, often set equal to zero and the term $\mathbf{x}=x_\mathrm{k}\mathbf{i}_\mathrm{k}=x_x\mathbf{i}_x+x_y\mathbf{i}_y+x_z\mathbf{i}_z$ is the vector location of a fluid particle in the system. The philosophical understanding of the term $\mathbf{x}$ relative to a fluid particle (also, continuum particle) is well described in [3] in Chapter 2, entitled *Description of Fluid*. Applying the gradient operator on Eq. (4.35), gives the following

$$-\nabla p=-\nabla(p\_\mathrm{rgh}+\rho\,\mathbf{g}\cdot\mathbf{x}+\mathrm{pRef})=-\nabla p\_\mathrm{rgh}-\nabla(\rho\,\mathbf{g}\cdot\mathbf{x}). \tag{4.36}$$

To calculate the last term in the above equation, one can use indicial notation, with summation convention [20, 21, 39] (see also Footnote 12, Page 25). Here, k and p are the running indices (i.e. k = x,y,z or 1,2,3), where x, y and z (or, 1, 2 and 3) are the specific Cartesian coordinates. The term $\mathbf{i}_\mathrm{k}$ is the unit vector in the direction of k.

$$\begin{aligned}&\nabla(\mathbf{g}\cdot\mathbf{x}\,\rho)=\mathbf{i}_z\tfrac{\partial}{\partial x_z}\left(g_\mathrm{k}\mathbf{i}_\mathrm{k}\cdot x_\mathrm{p}\mathbf{i}_\mathrm{p}\,\rho\right)=\\&\left(\mathbf{i}_z\tfrac{\partial}{\partial x_z}(g_\mathrm{k}\mathbf{i}_\mathrm{k})\right)\cdot x_\mathrm{p}\mathbf{i}_\mathrm{p}\,\rho+g_\mathrm{k}\mathbf{i}_\mathrm{k}\cdot\left(\mathbf{i}_z\tfrac{\partial}{\partial x_z}(x_\mathrm{p}\mathbf{i}_\mathrm{p})\right)\rho+g_\mathrm{k}\mathbf{i}_\mathrm{k}\cdot\,x_\mathrm{p}\mathbf{i}_\mathrm{p}\,\mathbf{i}_z\tfrac{\partial\rho}{\partial x_z}=\\&=\nabla\mathbf{g}\cdot\mathbf{x}\,\rho+\mathbf{g}\cdot\nabla\mathbf{x}\,\rho+\mathbf{g}\cdot\mathbf{x}\,\nabla\rho=\mathbf{g}\cdot\nabla\mathbf{x}\,\rho+\mathbf{g}\cdot\mathbf{x}\,\nabla\rho\end{aligned} \tag{4.37}$$

In the above last line, the term $\nabla \mathbf{g}$ is zero, since the gravity $\mathbf{g}$ is a constant. Going further with the above and using the Kronecker delta[21]:

$$\begin{aligned}&\mathbf{g}\cdot\nabla\mathbf{x} = g_{\mathrm{k}}\mathbf{i}_{\mathrm{k}}\cdot\frac{\partial x_{\mathrm{p}}}{\partial x_z}\mathbf{i}_z\mathbf{i}_{\mathrm{p}} = g_{\mathrm{k}}(\mathbf{i}_{\mathrm{k}}\cdot\mathbf{i}_z)\frac{\partial x_{\mathrm{p}}}{\partial x_z}\mathbf{i}_{\mathrm{p}} = g_{\mathrm{k}}\delta_{\mathrm{kz}}\frac{\partial x_{\mathrm{p}}}{\partial x_z}\mathbf{i}_{\mathrm{p}} = \\ &g_{\mathrm{k}}\frac{\partial x_{\mathrm{p}}}{\partial x_{\mathrm{k}}}\mathbf{i}_{\mathrm{p}} = g_{\mathrm{k}}\delta_{\mathrm{pk}}\mathbf{i}_{\mathrm{p}} = g_{\mathrm{k}}\mathbf{i}_{\mathrm{k}} = \mathbf{g}\end{aligned} \tag{4.38}$$

Thus, from Eqs. (4.37) and (4.38), the following conclusion can be obtained

$$\nabla(\rho\,\mathbf{g}\cdot\mathbf{x}) = \mathbf{g}\,\rho + \mathbf{g}\cdot\mathbf{x}\,\nabla\rho. \tag{4.39}$$

Finally, by using the above result in Eq. (4.36) the following is arrived at

$$-\nabla p = -\nabla p\_\mathrm{rgh} - \rho\,\mathbf{g} - \mathbf{g}\cdot\mathbf{x}\,\nabla\rho. \tag{4.40}$$

Surface Tension Force

To account for the surface tension between the atmospheric air and mixture, the Continuum-Surface-Force (CSF) model of Brackbill is used [37] (see also [26, 30]). Brackbill interpreted the surface tension as a continuous, three-dimensional effect across an interface. In this approach, the interface is neither tracked explicitly nor are shape or location known [26]. Therefore, an exact boundary condition cannot be applied to the interface [26]. The surface tension force that applies for the CSF model is given by

$$\mathbf{F}_{\mathrm{s}} = \sigma\,\kappa\,\nabla\alpha_1 \tag{4.41}$$

The terms σ and κ are the surface tension and the curvature of the interface, respectively.

Final Governing Equation

Now, using Eqs. (4.40) and (4.41) in Eq. (4.34), the following emerges

$$\begin{aligned}&\tfrac{\partial(\rho\mathbf{U})}{\partial t} + \nabla\cdot(\rho\,\mathbf{U}\mathbf{U}) + \nabla\cdot\left(\left[\tfrac{\beta_{\mathrm{d}}}{1-\beta_{\mathrm{d}}}\tfrac{\rho_{\mathrm{c}}\rho_{\mathrm{d}}}{\rho_1}\right]\mathbf{V}_{\mathrm{dj}}\mathbf{V}_{\mathrm{dj}}\right) = \\ &-\nabla p\_\mathrm{rgh} + \nabla\cdot\mathbf{T} - \mathbf{g}\cdot\mathbf{x}\,\nabla\rho + \sigma\,\kappa\,\nabla\alpha_1\end{aligned} \tag{4.42}$$

Including Single Reference Frame (SRF)

If needed, the so–called single reference frame (SRF) approach [47] can be activated. This is done by uncommenting the line `#define SINGLE_REFERENCE_FRAME` in the source file `macroDefinitions.H` and thereafter recompile the solver. When taking this step, the computational domain represents no longer an inertial reference frame [48]. With this, the Coriolis force $\mathbf{F}_{\mathrm{cor}} = 2\,\rho\,\boldsymbol{\omega}\times\mathbf{U}$ and the centrifugal force $\mathbf{F}_{\mathrm{cen}} = \rho\,\boldsymbol{\omega}\times(\boldsymbol{\omega}\times\mathbf{x})$ have to be included into the governing equation [48] as shown below

$$\begin{aligned}&\tfrac{\partial(\rho\mathbf{U})}{\partial t} + \nabla\cdot(\rho\,\mathbf{U}\mathbf{U}) + \nabla\cdot\left(\left[\tfrac{\beta_{\mathrm{d}}}{1-\beta_{\mathrm{d}}}\tfrac{\rho_{\mathrm{c}}\rho_{\mathrm{d}}}{\rho_1}\right]\mathbf{V}_{\mathrm{dj}}\mathbf{V}_{\mathrm{dj}}\right) + \mathbf{F}_{\mathrm{cor}} + \mathbf{F}_{\mathrm{cen}} = \\ &-\nabla p\_\mathrm{rgh} + \nabla\cdot\mathbf{T} - \mathbf{g}\cdot\mathbf{x}\,\nabla\rho + \sigma\,\kappa\,\nabla\alpha_1\end{aligned} \tag{4.43}$$

[21] The Kronecker delta is written as δ_{ij} where $\delta_{\mathrm{ij}} = 1$ if i = j and $\delta_{\mathrm{ij}} = 0$ if i $\neq$ j [20].

Except for the angular velocity $\boldsymbol{\omega}$ [rad/s], all variables (i.e. α_1, $\dot{\gamma}$, $\mathbf{U}$, p_rgh and etc.) are relative to the non–inertial (i.e. rotating) reference. Scalar equations like Eqs. (4.20) and (4.23) remains unchanged [48] in this framework. Furthermore, scalar quantities like ρ, α_1, α_d, $\dot{\gamma}$ or p_rgh appear the same in inertial and non–inertial reference frames, as does their material derivative [48]. However, for a vector quantity like the velocity $\mathbf{U}$, the transformation between the two reference frames is $\mathbf{U}_\text{in} = \mathbf{U} + \boldsymbol{\omega} \times \mathbf{x}$, where $\mathbf{U}_\text{in}$ is the inertial velocity [48]. The transformation is done in the source code `enableFieldControl.H`, as shown with the following code:

```
#ifdef SINGLE_REFERENCE_FRAME
Uin = U + (Omega ^ mesh.C());
#endif
```

Governing Equation in a Semi-Discretized Form

Writing Eq. (4.42)/(4.43) in a semi-discretized form at the nodal point P of a computational cell, results in

$$a_\text{P}\mathbf{U}_\text{P} = \mathbf{H}(\mathbf{U}) - \nabla p\_\text{rgh} - \mathbf{g}\cdot\mathbf{x}\,\nabla\rho + \sigma\,\kappa\,\nabla\alpha_1 \tag{4.44}$$

As indicated, the term $\mathbf{U}_\text{P}$ is the velocity at the nodal point. The vector $\mathbf{H}(\mathbf{U})$ includes everything on the left side (i.e. on the first line) of Eq. (4.42) except for any terms containing velocity of the current time step, located on the diagonal of the array generated by `fvVectorMatrix UEqn()`. However, $\mathbf{H}(\mathbf{U})$ does include $\mathbf{U}_\text{P}$ from previous time step, namely $\mathbf{U}_\text{P}^\text{old}$ (i.e. from $\partial(\rho\mathbf{U})/\partial t \approx (\rho_\text{P}\mathbf{U}_\text{P} - \rho_\text{P}^\text{old}\mathbf{U}_\text{P}^\text{old})/\Delta t$, or similar).

Isolating $\mathbf{U}_\text{P}$ in Eq. (4.44) results in the following equation

$$\mathbf{U}_\text{P} = \frac{\mathbf{H}(\mathbf{U})}{a_\text{P}} - \frac{\nabla p\_\text{rgh}}{a_\text{P}} - \frac{\mathbf{g}\cdot\mathbf{x}\,\nabla\rho}{a_\text{P}} + \frac{\sigma\,\kappa\,\nabla\alpha_1}{a_\text{P}} \tag{4.45}$$

4.9 Pressure Equation

4.9.1 Continuity Equation

In accordance with Eq. (2.9), the combined continuity equation for the mixture and atmospheric air is as follows

$$\frac{\partial\rho}{\partial t} + \nabla\cdot(\rho\,\mathbf{U}) = \frac{\partial\rho}{\partial t} + \mathbf{U}\cdot\nabla\rho + \rho\nabla\cdot\mathbf{U} = \frac{d\rho}{dt} + \rho\nabla\cdot\mathbf{U} = 0 \tag{4.46}$$

The above can also be derived from standard continuum mechanics (see for example [20, 21] as well as [3], Pages 386 to 387). Now, rearranging Eq. (4.46) into the following

$$\nabla\cdot\mathbf{U} = -\frac{1}{\rho}\frac{d\rho}{dt} \tag{4.47}$$

The above can be used together with Eq. (4.45) to generate the so-called pressure equation (see also [25, 49] about the pressure equation). In this work, this was attempted. A pressure equation similar to what is implemented in the `settlingFoam` solver was tested. Unfortunately, the implementation resulted in frequent run crashes and extreme sensitivity towards the use of higher solid concentration values for α_d. That is, α_d values higher than say 10^{-2} started to result in run crashes. Thus, another approach was needed in generating the pressure equation.

For compressible gas, a 10% variation in density (i.e. $0.1 \cdot \rho$) can occur within a fraction of a second, giving

$$\frac{1}{\rho}\frac{d\rho}{dt} \approx \frac{1}{\rho}\frac{\Delta\rho}{\Delta t} = \frac{1}{\rho}\left(\frac{0.1 \cdot \rho}{0.1\,\mathrm{s}}\right) = 1\,\mathrm{s}^{-1}, \tag{4.48}$$

and thus $\nabla \cdot \mathbf{U} = 0$ cannot be assumed for this particular case. However, for a highly viscous fluid like the fresh concrete, a 10% variation in density ($0.1 \cdot \rho$) by settling (or other means) typically occurs, say, during minutes or longer, resulting in the following condition

$$\frac{1}{\rho}\frac{d\rho}{dt} \approx \frac{1}{\rho}\frac{\Delta\rho}{\Delta t} = \frac{1}{\rho}\left(\frac{0.1 \cdot \rho}{100\,\mathrm{s}}\right) \leq 10^{-3}\,\mathrm{s}^{-1} \tag{4.49}$$

Thus, for such case, one can suggest the use of Eq. (4.50) when generating the pressure equation.

$$\nabla \cdot \mathbf{U} = 0 \tag{4.50}$$

A third approach (Eq. (4.47) being the first, and Eq. (4.50) the second) is possible consisting of using the result of Eq. (2.47), Page 23, reproduced with Eq. (4.51).

$$\nabla \cdot \mathbf{U} = \nabla \cdot \mathbf{R} \tag{4.51}$$

In the above, the mixture velocity $\mathbf{V}_\mathrm{m}$ was replaced by $\mathbf{U}_1$ (see Section 4.2), which again was replaced by $\mathbf{U}$, since $\mathbf{R} = 0$ in the atmospheric air (i.e. in phase 2, meaning $\alpha_2 = 1$), where Eq. (4.50) is retrieved (c.f. $\rho_2 =$ constant). The term $\mathbf{R}$ is defined with Eq. (2.48) and reproduced below:

$$\mathbf{R} = \left(\beta_\mathrm{d}\left[\frac{\rho_\mathrm{d} - \rho_\mathrm{c}}{\rho_\mathrm{m}}\right]\mathbf{V}_\mathrm{dj}\right) \tag{4.52}$$

Using Eq. (4.51) with Eq. (4.45) results in the following

$$\nabla \cdot \frac{\mathbf{H}(\mathbf{U})}{a_\mathrm{P}} - \nabla \cdot \frac{\nabla p\_\mathrm{rgh}}{a_\mathrm{P}} - \nabla \cdot \frac{\mathbf{g} \cdot \mathbf{x}\,\nabla\rho}{a_\mathrm{P}} + \nabla \cdot \frac{\sigma\,\kappa\,\nabla\alpha_1}{a_\mathrm{P}} = \nabla \cdot \mathbf{R} \tag{4.53}$$

Rearranging and integrating over an arbitrary cell volume V_P:

$$\int_{V_\mathrm{P}} \nabla \cdot \left(\tfrac{1}{a_\mathrm{P}}\nabla p\_\mathrm{rgh}\right) dV = \int_{V_\mathrm{P}} \left(\nabla \cdot \tfrac{\mathbf{H}(\mathbf{U})}{a_\mathrm{P}} - \nabla \cdot \tfrac{\mathbf{g}\cdot\mathbf{x}\,\nabla\rho}{a_\mathrm{P}} + \nabla \cdot \tfrac{\sigma\,\kappa\,\nabla\alpha_1}{a_\mathrm{P}} - \nabla \cdot \mathbf{R}\right) dV \tag{4.54}$$

Since the right side of Eq. (4.54) is evaluated with the divergence operator `fvc::div()`, it has to be multiplied with the face area vector[22] $\mathbf{S} = \mathbf{n}|\mathbf{S}|$, where $\mathbf{n}$ is the corresponding unit normal vector. With this step, there is apparently a physical-unit mismatch between between the left side and right side of this equation. But actually, this is not the case since `fvc::div()` operator is overloaded in such manner. After this step, Eq. (4.54) has the following form[23]

$$\begin{aligned} &\int_{V_\mathrm{P}} \nabla \cdot \left(\tfrac{1}{a_\mathrm{P}} \nabla p\_\mathrm{rgh} \right)_f dV = \int_{V_\mathrm{P}} \nabla \cdot \left(\left(\tfrac{\mathbf{H}(\mathbf{U})}{a_\mathrm{P}} \right)_f \cdot \mathbf{S} \right) dV + \\ &\int_{V_\mathrm{P}} \nabla \cdot \left[(\sigma\, \kappa\, \mathbf{n} \cdot \nabla \alpha_1 - \mathbf{g} \cdot \mathbf{x}\, \mathbf{n} \cdot \nabla \rho)_f \left(\tfrac{1}{a_\mathrm{P}} \right)_f |\mathbf{S}| - \mathbf{R}_f \cdot \mathbf{S} \right] dV \end{aligned} \tag{4.55}$$

The right side of the above is calculated by the Gauss's theorem and therefore values that apply at the cell faces are shown. To reiterate the above text about `fvc::div()`, the $\int \nabla \cdot \mathbf{R}\, dV = \int \mathbf{R}_f \cdot \mathbf{dS}$ is calculated as $\sum_f \mathbf{R}_f$ and not as $\sum_f \mathbf{R}_f \cdot \mathbf{S}$ and thus the operation of the face flux on the right side of Eq. (4.55) is necessary (i.e. calculation of $\mathbf{R}_f \cdot \mathbf{S}$ is needed before applying `fvc::div()`). However, for the left hand side, then $\int \nabla \cdot (b \nabla p)\, dV$ is evaluated with $\sum_f (b_f \mathbf{S} \cdot \nabla p)_f$, which means that there is no need to operate the face area flux $\mathbf{S}$ before applying the Laplacian operator. That is, the left side of Eq. (4.55) is evaluated with the `fvm::laplacian()` operator, as shown with the following section of the source code `pEqn.H`:

```
fvScalarMatrix p_rghEqn
(
    fvm::laplacian(rAUf, p_rgh) == fvc::div(phiHbyA)
);
```

in which `laplacian(rAUf, p_rgh)` is given by

$$\texttt{laplacian(rAUf, p\_rgh)} = \left(\tfrac{1}{a_\mathrm{P}} \nabla p\_\mathrm{rgh} \right)_f \tag{4.56}$$

and `phiHbyA` by

$$\texttt{phiHbyA} = \left(\tfrac{\mathbf{H}(\mathbf{U})}{a_\mathrm{P}} \right)_f \cdot \mathbf{S} + (\sigma\, \kappa\, \mathbf{n} \cdot \nabla \alpha_1 - \mathbf{g} \cdot \mathbf{x}\, \mathbf{n} \cdot \nabla \rho)_f \left(\tfrac{1}{a_\mathrm{P}} \right)_f |\mathbf{S}| - \mathbf{R}_f \cdot \mathbf{S} \tag{4.57}$$

As shown in the code `pEqn.H`, the term $-\mathbf{R}_f \cdot \mathbf{S}$ is added to the pressure equation with `phiHbyA += pResidue`, in which $-\mathbf{R}_f$ is calculated as:

```
volScalarField densityVariation
(
    "densityVariation",
```

[22] Usually, the face area vector $\mathbf{S}$ is drawn as pointing outward from a cell. However, this actually depends on the label of the cell in question and the label of the neighboring cell, the vector pointing into the cell of higher label number. The cell with the lower label number is the owner of the face in question.

[23] When operating the face area vector $\mathbf{S}$, it has to be on a face value of a vector (or tensor) instead of on the corresponding nodal point value (i.e. $\mathbf{S} \cdot \mathbf{R}_f$ and not $\mathbf{S} \cdot \mathbf{R}$).

```
    betaD*((rhoD - rhoC)/rho1)
);

surfaceScalarField pResidue
(
   -fvc::interpolate(densityVariation)*phiVdj
);
```

In the above, the term `phiVdj` is calculated with $\mathbf{V}_{\mathrm{dj}}\cdot\mathbf{S} = \mathbf{V}_{\mathrm{dj}}^{\mathrm{GR}}\cdot\mathbf{S}+\mathbf{V}_{\mathrm{dj}}^{\mathrm{SR}}\cdot\mathbf{S}$. More precisely, in `particleMigration.H`, `gravitySegregation.H` and `driftVelocity.H`, respectively, then:

```
phiVdjSR = fvc::interpolate(VdjSR) & mesh.Sf();
phiVdjGR = fvc::interpolate(VdjGR) & mesh.Sf();
phiVdj = phiVdjGR + phiVdjSR;
```

In `pEqn.H`, the total flux ϕ_f = `phi`, through each single face of an arbitrary cell, is calculated with:

```
phi = phiHbyA - p_rghEqn.flux();
```

More specifically, the above code represents the following equation

$$\begin{aligned}\phi_f &= \left(\tfrac{\mathbf{H(U)}}{a_{\mathrm{P}}}\right)_f\cdot\mathbf{S} + (\sigma\,\kappa\,\mathbf{n}\cdot\nabla\alpha_1 - \mathbf{g}\cdot\mathbf{x}\,\mathbf{n}\cdot\nabla\rho)_f\left(\tfrac{1}{a_{\mathrm{P}}}\right)_f|\mathbf{S}| \\ &- \mathbf{R}_f\cdot\mathbf{S} - \left(\tfrac{1}{a_{\mathrm{P}}}\nabla p\_\mathrm{rgh}\right)_f\cdot\mathbf{S}\end{aligned} \tag{4.58}$$

The term `p_rghEqn.flux()` is the off diagonal part of the array in `p_rghEqn`, given by

$$\texttt{p\_rghEqn.flux()} = \left(\tfrac{1}{a_{\mathrm{P}}}\nabla p\_\mathrm{rgh}\right)_f\cdot\mathbf{S} \tag{4.59}$$

By summing the flux Eq. (4.58) of all faces for a single arbitrary cell, the following is obtained

$$\begin{aligned}\textstyle\sum_f\phi_f &= \textstyle\sum_f\left(\tfrac{\mathbf{H(U)}}{a_{\mathrm{P}}}\right)_f\cdot\mathbf{S} + \sum_f(\sigma\,\kappa\,\mathbf{n}\cdot\nabla\alpha_1 - \mathbf{g}\cdot\mathbf{x}\,\mathbf{n}\cdot\nabla\rho)_f\left(\tfrac{1}{a_{\mathrm{P}}}\right)_f|\mathbf{S}| \\ &- \textstyle\sum_f\mathbf{R}_f\cdot\mathbf{S} - \sum_f\left(\tfrac{1}{a_{\mathrm{P}}}\nabla p\_\mathrm{rgh}\right)_f\cdot\mathbf{S} \\ &= \textstyle\int_{\partial V_{\mathrm{P}}}\left(\tfrac{\mathbf{H(U)}}{a_{\mathrm{P}}}\right)_f\cdot d\mathbf{S} + \int_{\partial V_{\mathrm{P}}}(\sigma\,\kappa\,\nabla\alpha_1 - \mathbf{g}\cdot\mathbf{x}\,\nabla\rho)_f\left(\tfrac{1}{a_{\mathrm{P}}}\right)_f d\mathbf{S} \\ &- \textstyle\int_{\partial V_{\mathrm{P}}}\mathbf{R}_f\cdot d\mathbf{S} - \int_{\partial V_{\mathrm{P}}}\left(\tfrac{1}{a_{\mathrm{P}}}\nabla p\_\mathrm{rgh}\right)_f\cdot d\mathbf{S} \\ &= \textstyle\int_{V_{\mathrm{P}}}\nabla\cdot\tfrac{\mathbf{H(U)}}{a_{\mathrm{P}}}\,dV - \int_{V_{\mathrm{P}}}\nabla\cdot\tfrac{\mathbf{g}\cdot\mathbf{x}\,\nabla\rho}{a_{\mathrm{P}}}\,dV + \int_{V_{\mathrm{P}}}\nabla\cdot\tfrac{\sigma\,\kappa\,\nabla\alpha_1}{a_{\mathrm{P}}}\,dV \\ &- \textstyle\int_{V_{\mathrm{P}}}\nabla\cdot\mathbf{R}\,dV - \int_{V_{\mathrm{P}}}\nabla\cdot\left(\tfrac{1}{a_{\mathrm{P}}}\nabla p\_\mathrm{rgh}\right)dV = 0\end{aligned} \tag{4.60}$$

In the last two lines, the Gauss theorem was applied, while the outcome of zero in the end (i.e. $\sum_f\phi_f = 0$) is in accordance with Eq. (4.54). The outcome of Eq. (4.60) means that when calculating the flux by Eq. (4.58), continuity by Eq. (4.51) is automatically fulfilled.

4.9.2 Explicit Velocity Correction

In the source code `pEqn.H`, the term $\phi_g =$ `phig` is calculated as follows:

```
surfaceScalarField phig
(
    (
        fvc::interpolate(sigmaK)*fvc::snGrad(alpha1)
      - ghf*fvc::snGrad(rho)
    )*rAUf*mesh.magSf()
);
```

With `rAUf` $= (1/a_\mathrm{P})_f$, then `phig/rAUf` is equal to

$$\texttt{phig/rAUf} = \left((\sigma\,\kappa\,\mathbf{n}\cdot\nabla\alpha_1 - \mathbf{g}\cdot\mathbf{x}\,\mathbf{n}\cdot\nabla\rho)_f \left(\frac{1}{a_\mathrm{P}}\right)_f |\mathbf{S}| \right) a_\mathrm{P}|_f \tag{4.61}$$

...and through Eq. (4.59), then...

$$\texttt{p\_rghEqn.flux()/rAUf} = \left[\left(\frac{1}{a_\mathrm{P}}\nabla p\_\mathrm{rgh}\right)_f \cdot \mathbf{S} \right] a_\mathrm{P}|_f \tag{4.62}$$

The difference between the two above equations is as follows

$$\begin{aligned} &\texttt{(phig - p\_rghEqn.flux())/rAUf} = \\ &(\sigma\,\kappa\,\mathbf{n}\cdot\nabla\alpha_1 - \mathbf{g}\cdot\mathbf{x}\,\mathbf{n}\cdot\nabla\rho)_f\,|\mathbf{S}| - \nabla p\_\mathrm{rgh}|_f\cdot\mathbf{S} \end{aligned} \tag{4.63}$$

Making the above result valid at nodal point P (instead of at a cell face):

$$\begin{aligned} &\texttt{fvc::reconstruct((phig - p\_rghEqn.flux())/rAUf)} = \\ &\sigma\,\kappa\,\nabla\alpha_1 - \mathbf{g}\cdot\mathbf{x}\,\nabla\rho - \nabla p\_\mathrm{rgh} \end{aligned} \tag{4.64}$$

Thus the final velocity corrector $\mathbf{U}_\mathrm{P}$, calculated at nodal point P, and given by the following code...

```
U = HbyA + rAU*fvc::reconstruct((phig - p_rghEqn.flux())/rAUf);
```

...and is exactly the same as Eq. (4.45), which is reproduced below:

$$\mathbf{U}_\mathrm{P} = \frac{\mathbf{H}(\mathbf{U})}{a_\mathrm{P}} - \frac{\nabla p\_\mathrm{rgh}}{a_\mathrm{P}} - \frac{\mathbf{g}\cdot\mathbf{x}\,\nabla\rho}{a_\mathrm{P}} + \frac{\sigma\,\kappa\,\nabla\alpha_1}{a_\mathrm{P}} \tag{4.65}$$

The influence of $\nabla\cdot\mathbf{R}$ on the above equation is through its use of p_rgh, which is obtained solving the pressure equation Eq. (4.55). If `pimple.momentumPredictor()` is true, then $\nabla\cdot\mathbf{R}$ will also affect the new prediction of `U` (i.e. $\mathbf{U}_\mathrm{P}$) through `solve(UEqn == fvc::reconstruct(... - fvc::snGrad(p_rgh)...))` in `UEqn.H`. Also, since the flux calculation $\phi_f =$ `phi` by Eq. (4.58) is frequently used, including in `fvVectorMatrix UEqn(...)` through `fvm::div(rhoPhi, U)` (which gives $\mathbf{H}(\mathbf{U})$), as well as in the calculation of α_1 and α_d convection in `alpha1EqnRho.H` and `alphaDEqn.H`, the effect of $\nabla\cdot\mathbf{R}$ will basically resonate everywhere in the code. The same consideration would arise if Eq. (4.47) would be used instead of Eq. (4.51).

Part of the source code `pEqn.H` reads as follows:

```
    phiHbyA += phig;


    #ifdef USE_PRESSURE_RESIDUE
    volScalarField densityVariation
    (
        "densityVariation",
        betaD*((rhoD - rhoC)/rho1)
        // alpha1*betaD*((rhoD - rhoC)/rho1)
    );

    surfaceScalarField pResidue
    (
       -fvc::interpolate(densityVariation)*phiVdj
    );

    phiHbyA += pResidue;
    #endif


    while (pimple.correctNonOrthogonal())
    {
        fvScalarMatrix p_rghEqn
        (
            fvm::laplacian(rAUf, p_rgh) == fvc::div(phiHbyA)
        );
    ...
```

As shown in the above code, the user can decide if Eq. (4.55) is solved with $\mathbf{R} = 0$ (i.e. solving for Eq. (4.50)) or with $\mathbf{R}$ as given by Eq. (4.52) (i.e. solving for Eq. (4.51)). This is controlled with the macro definition `USE_PRESSURE_RESIDUE` in the source code `macroDefinitions.H`. The default setup in the source code is using $\mathbf{R} = 0$. This is not a bad choice when considering the result of Eq. (4.49). Most of the simulation results shown in this report are solved in this manner. However, the term $\mathbf{R}$ by Eqs. (4.51) and (4.52), has for example been used in [16]. Thus, applying this last-mentioned approach is apparently neither a bad choice.

4.9.3 Monitoring $\nabla \cdot \mathbf{U}$

In the source code `comprsblContErrs.H`, the information about $\nabla \cdot \mathbf{U}$ is exported to the console. More precisely, the term `magWeightedAverageDivPhi` is shown and calculated as:

$$E_{\mathrm{w}}(t) = \int_0^t \left|\nabla \cdot \mathbf{U}(\mathbf{x}, t^*)\right|_{\mathrm{w}} \, dt^* \tag{4.66}$$

where $\left|\nabla \cdot \mathbf{U}\right|_{\mathrm{w}}$ is the weighted average of $|\nabla \cdot \mathbf{U}|$ relative to mass of materials in each cell (i.e. `weightedAverage(rho*mesh.V())`) and $\mathbf{x}$ represents the coordinates x, y and z.

The term $E_{\mathrm{w}}(t)$ can be divided by the current time t, which then can be considered as measure of incompressibility at time t. The product of this, namely $e_{\mathrm{w}}(t)$ is given by Eq. (4.67). A value of $e(t) = 0$ would represent a complete incompressibility, or $\nabla \cdot \mathbf{U} = 0$.

$$e_{\mathrm{w}}(t) = \frac{E_{\mathrm{w}}(t)}{t} = \frac{1}{t} \int_0^t \left| \nabla \cdot \mathbf{U}(\mathbf{x}, t^*) \right|_{\mathrm{w}} \, dt^* \tag{4.67}$$

Fig. 4.4 shows the calculation results of $\left| \nabla \cdot \mathbf{U} \right|_{\mathrm{w}}$ and e_{w} as a function of time, for the case of Figs. 1.12 and 1.13, Page 13 (note, $\mathbf{R} = 0$ in this case). As shown, the value of $e(t) \approx 10^{-7}\,\mathrm{s}^{-1}$ is obtained, which is fairly close to incompressibility. If the calculations are repeated without magnitude (i.e. without `mag()` in the above code), the result is an order of magnitude less, or $10^{-8}\,[\mathrm{s}^{-1}]$.

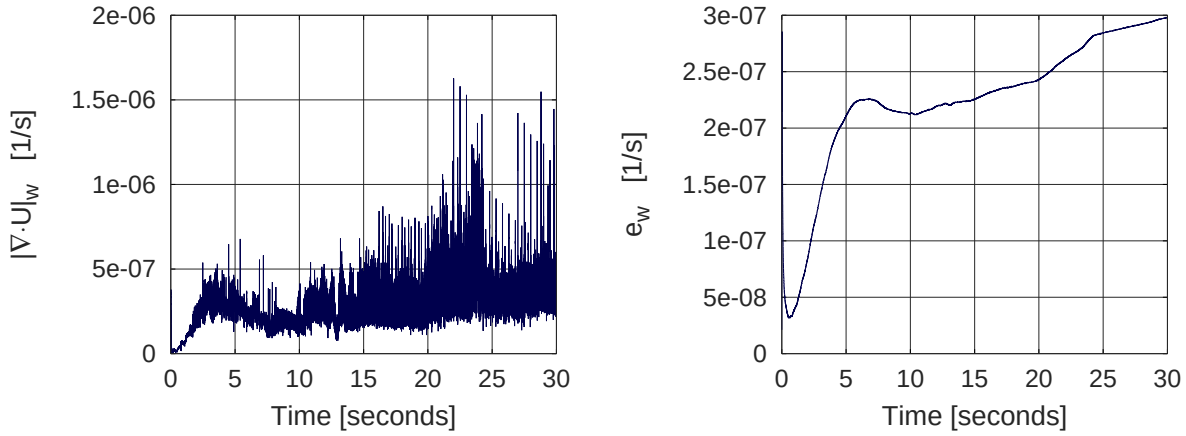


Figure 4.4: Calculations of $\left| \nabla \cdot \mathbf{U} \right|_{\mathrm{w}}$ (shown to the left) and $e_{\mathrm{w}}(t)$ (shown to the right) for the case of Figs. 1.12 and 1.13 (Page 13).

Chapter 5

Rheological Behavior of the Mixture

5.1 Apparent Viscosity

As discussed in Section 4.8.1 (see also Chapter 3), the combined apparent viscosity η of the atmospheric air (phase 2) and the fluid mixture (phase 1) are weighted by the volume fractions α_1 and α_2 [25, 26]

$$\eta = \alpha_1 \, \eta_1 + \alpha_2 \, \eta_2 \tag{5.1}$$

in which the fluid mixture is modeled through η_1, while the atmospheric air (i.e. phase 2) is always set as a Newtonian fluid $\eta_2 = \text{constant}$. As a starting point, the mixture fluid can be (nominally) modelled in the same manner as done by Eq. (5.1), namely[24] with

$$\eta_1 = \beta_\mathrm{c} \, \eta_\mathrm{c} + \beta_\mathrm{d} \, \eta_\mathrm{d} \tag{5.2}$$

where β_c and β_d are the volume fraction of *phase* c and d, respectively, c.f. Chapter 2. The terms η_c and η_d are their respective apparent viscosities. When the mixture fluid is a suspension, the phase d represents suspended particles (e.g. coarse aggregates), while phase c represents the matrix.

As described in [3] on Pages 237 to 239, the apparent viscosity is fundamentally defined by the (rate of) momentum transfer between particles. Thus, when determining the apparent viscosity for suspended particles alone, it becomes very dependent on the distance between particles, i.e. on the amount of matrix β_c present in the overall suspension. This means that η_d depends at least on β_c, meaning $\eta_\mathrm{d} = \eta_\mathrm{d}(\beta_\mathrm{c})$. Also, since the quantity of suspended particles will influence the local shear rate $\dot{\gamma}_\mathrm{c}$ in the matrix (see Fig. 4.5 in [3]), which could influence the effect of η_c, then at least $\eta_\mathrm{c} = \eta_\mathrm{c}(\beta_\mathrm{d})$. In addition to this, the matrix usually consists of a non-Newtonian fluid which complicates the behavior in Eq. (5.2) still further. There will also be an additional nonlinear behavior present in η_1 because of the dense solid concentration β_d used in the mixture. Instead of trying to resolve Eq. (5.2) (i.e. using a superpositioning between η_c and η_d), a more empirical

[24] See also the text below Eq. (2.53) on Page 24.

approach is traditionally used, namely by modeling η_1 directly as a function of β_d with other materials parameters and flow properties, e.g. $\eta_1 = \eta_1(\dot{\gamma}, \beta_d, \ldots)$.

In the following text of this chapter, when discussing the solid concentration of the suspended particles (see also Footnote 2, Page 2), the term φ is used instead of β_d or α_d. That is, the term φ can be equal to α_d or β_d, depending on user preference. Whichever is applied, the modification is made in the source codes `correctViscosity.H`, `gravitySegregation.H` and `particleMigration.H`. Note, making the same choice in all the three codes is not necessary. Again, this depends on user preference and knowledge of the material and the overall flow system, which is being investigated.

5.2 Empirical Approach

5.2.1 Linear Weight Function

Normalizing Relative to $w_f(0) = 1$

For the more traditional suspension of monosized spheres submerged in a Newtonian liquid with viscosity of η_0, the apparent viscosity is generally given by Eq. (5.3):

$$\eta_1(\varphi) = w_f(\varphi) \cdot \eta_0 \tag{5.3}$$

Usually, the weight function has the property of $w_f(0) = 1$, meaning that the apparent viscosity of the mixture η_1 approaches that of the matrix η_0 as the solid concentration is decreased. In the case of Einstein's model, this weight function is given by $w_f(\varphi) = 1 + 2.5\,\varphi$, while for the Krieger-Dougherty equation it is $w_f(\varphi) = (1 - \varphi/\varphi_m)^{-[\eta]\varphi_m}$, where $[\eta]$ is the co-called intrinsic viscosity and φ_m is the maximum packing fraction [2, 50, 51].

Normalizing Relative to $w_f(\varphi_0) = 1$, in which $\varphi_0 > 0$

Rather than using a reference point for η_0 that is relative to zero volume fraction ($\varphi = 0$), like initially done in Eq. (5.3), one can use this equation with a reference point relative to a (nominal/initial) homogeneous mixture i.e. when no segregation has occurred (no slip between phases, c.f. Section 2.3.1). At this point, the volume fraction is designated with $\varphi = \varphi_0$, in which "0" symbolizes *initial state of concentration*. The main property of φ_0 is $w_f(\varphi_0) = 1$.

Using the above approach, one can use a linear weight function $w_f(\varphi)$, which is bounded between specific values w_f^{max} and w_f^{min}. Furthermore, limit values can be set for the volume fraction φ, given by a minimum value φ^{min} and a maximum value φ^{max}. An example of a linear weight function that is constrained by these properties is given by:

$$w_f(\varphi) = \frac{w_f^{max} - w_f^{min}}{\varphi^{max} - \varphi^{min}} \cdot (\varphi - \varphi^{min}) + w_f^{min} \tag{5.4}$$

To maintain the important property $w_f(\varphi_0) = 1$, then relative to Eq. (5.4), the term φ_0 must be defined as:

$$\varphi_0 = (1 - w_f^{\min})\frac{\varphi^{\max} - \varphi^{\min}}{w_f^{\max} - w_f^{\min}} + \varphi^{\min} \tag{5.5}$$

With for example, $\varphi^{\min} = 0$, $\varphi^{\max} = 0.4$, $w_f^{\min} = 0.4$ and $w_f^{\max} = 1.6$, the following is obtained:

$$\varphi_0 = (1 - 0.4)\frac{0.4 - 0}{1.6 - 0.4} + 0 = 0.2 \tag{5.6}$$

Thus, $w_f(0.2) = 1$, and if $\varphi < 0.2$ then $w_f(\varphi) < 1$ and if $\varphi > 0.2$ then $w_f(\varphi) > 1$. The approach of Eq. (5.4) is just an example and its implementation can be found in the source code `apparentViscosity.H`:

```
volScalarField weightFunction1
(
    const volScalarField& varPhi,
    const dimensionedScalar& varPhiMIN,
    const dimensionedScalar& varPhiMAX
)
{
    dimensionedScalar WF1_MAX
    (
        "WF1_MAX",
        dimensionSet(0,0,0,0,0,0,0),
        scalar(1.6)
    );

    dimensionedScalar WF1_MIN
    (
        "WF1_MIN",
        dimensionSet(0,0,0,0,0,0,0),
        scalar(0.4)
    );

    dimensionedScalar slope("slope",dimensionSet(0,0,0,0,0,0,0),scalar(1));
    slope = (WF1_MAX - WF1_MIN)/(varPhiMAX - varPhiMIN);

    tmp<volScalarField> weight
    (
        slope*mag(varPhi - varPhiMIN) + WF1_MIN
    );

    return weight();
}
```

The call to Eq. (5.4) is made with the following code available in `apparentViscosity.H`:

```
volScalarField WF1 = weightFunction1(varPhi, varPhiMIN, varPhiMAX);
```

The apparent viscoity of the mixture in homogenous state (i.e. when $\varphi = \varphi_0$ and thus $w_f(\varphi) = 1$ in Eq. (5.3)) can be modelled with the traditional Bingham model [2]

$$\eta_0 = \mu + \frac{\tau_0}{\dot{\gamma}} \tag{5.7}$$

where μ is the plastic viscosity, τ_0 is the yield stress and $\dot{\gamma}$ is the shear rate given by [52, 53, 54]

$$\dot{\gamma} = \sqrt{2\,\dot{\boldsymbol{\varepsilon}} : \dot{\boldsymbol{\varepsilon}}} \tag{5.8}$$

The term $\dot{\boldsymbol{\varepsilon}}$ is the rate–of–deformation tensor as given in Section 3.6 on Page 30.

Because of the nonlinearities in the governing equation and because of the inherent discontinuity in the constitutive equation, a computer simulation of yield stress fluid (i.e. viscoplastic fluid) is difficult. As the yield surface is approached, the presence of shear rate $\dot{\gamma}$ in the denominator of Eq. (5.7) (and later in Eqs. (5.9) and (5.10)) makes the apparent viscosity η_1 unbounded. Furthermore, while simulating the velocity field $\mathbf{U}$, the location of the yield surface is unknown prior to calculation. To overcome these difficulties, a regularized version of the viscoplastic model has been proposed by Bercovier and Engelman [40]. It consists of adding a small regularization parameter δ in the denominator of Eq. (5.7). Bercovier and Engelman used such approach to solve Bingham flow in a closed square cavity subject to a body force [40]. This equation has also been successfully used by Taylor and Wilson to simulate conduit flow of an incompressible Bingham fluid [41]. Furthermore, Burgos et al. used the regularization parameter δ in this manner to simulate antiplane shear flow of a Herschel–Bulkley fluid [42]. They also used other types of regularization approaches for comparison [42]. Hence, with a proper choice of δ, the regularized version of the viscoplastic model can be successfully used to simulate both the yielded region and the unyielded region [40, 41, 42]. The use of the regularization parameter δ in this manner has also been used in [3, 43, 44, 45, 46] for Bingham, modified Bingham as well as thixotropic viscoplastic material models.

By using the regularization parameter δ (or `delta`), the calls to Eqs. (5.7), (5.3), (5.4) and (5.1) are made with the following code in `apparentViscosity.H`:

```
const dimensionedScalar mu_Bi("mu_Bi", dimPressure*dimTime, scalar(50.0));
const dimensionedScalar tau0_Bi("tau0_Bi", dimPressure, scalar(10.0));

tmp<volScalarField> viscous_2
(
    WF1*mag(alpha1)*
    (
        mu_Bi + tau0_Bi/(shearRate + delta)
    )
    + mag(scalar(1) - alpha1)*eta2
);
```

Other models like the Herschel–Bulkley model Eq. (5.9) [55] can also be applied in Eq. (5.3).

$$\eta_0 = K\dot{\gamma}^{n-1} + \frac{\tau_0}{\dot{\gamma}} \tag{5.9}$$

In the above equation, the term K is the consistency factor and n is the flow index (also, consistency index). The calls to Eqs. (5.9), (5.3), (5.4) and (5.1) are made with the following code in `apparentViscosity.H`:

```
const scalar n_HB = 1.14;
const dimensionedScalar K_HB("K_HB", dimPressure*dimTime, scalar(40.7));
const dimensionedScalar tau0_HB("tau0_HB", dimPressure, scalar(16.5));
const dimensionedScalar tOne("tOne", dimensionSet(0,0,1,0,0,0,0), scalar(1.0));

tmp<volScalarField> viscous_3
(
    WF1*mag(alpha1)*
    (
        K_HB*pow(shearRate*tOne,n_HB-1.0) + tau0_HB/(shearRate + delta)
    )
    + mag(scalar(1) - alpha1)*eta2
);
```

Furthermore, in [45] the modified Bingham model can be a good alternative to either the standard Bingham model or the Herschel–Bulkley model. It is given by

$$\eta_0 = \mu + c\,\dot{\gamma} + \frac{\tau_0}{\dot{\gamma}} \tag{5.10}$$

where c is the so-called second order term. The calls to Eqs. (5.10), (5.3), (5.4) and (5.1) are made with the following code in `apparentViscosity.H`:

```
const dimensionedScalar mu_mBi("mu_mBi", dimPressure*dimTime, scalar(82.6));
const dimensionedScalar c_mBi("c_mBi",dimPressure*dimTime*dimTime,scalar(1.5));
const dimensionedScalar tau0_mBi("tau0_mBi", dimPressure, scalar(23.7));

tmp<volScalarField> viscous_4
(
    WF1*mag(alpha1)*
    (
        mu_mBi + c_mBi*shearRate + tau0_mBi/(shearRate + delta)
    )
    + mag(scalar(1) - alpha1)*eta2
);
```

Regardless of the choice of material model used (Eqs. (5.7), (5.9) or (5.10), or others) for the initial/nominal homogeneous mixture, then through Eq. (5.3) the apparent viscosity η_1 can either increase or decrease relative to η_0 depending on deviation in solid concentration φ from the initial/normal/nominal value φ_0.

The approach presented in this section is an empirical approach but can be quite accurate provided that a good rheometer is available for measuring the material parameters of η_0 (whichever model is used, Eqs. (5.7), (5.9), (5.10) or others not mentioned) as well as measuring the sensitivity of the weight function $w_f(\varphi)$ when changing the solid concentration φ in the mixture (relative to the initial/normal/nominal value φ_0).

5.2.2 Excess and Shortage of φ Relative to φ_0

If the aim is to look at the segregation of coarse aggregates in the fresh concrete (or in other types of mixtures), the distinction between the matrix and suspended particles needs to be defined relative to this process. More precisely, the distinction between the two phases must correlate with the domain of particle sizes that are actually segregating. For example, assuming that aggregate particles larger than 11 mm in diameter are participating in segregation, and all materials of smaller size are not, the distinction between the matrix and the suspended particles must also reflect this division. In this context, it is important to note that the reference viscosity η_0 in Eq. (5.3) and the use of the weight function Eq. (5.4) must be relative to this definition. That is, if η_0 in Eq. (5.3) represents the apparent viscosity of a homogeneous mixture with aggregate range from 0 to 16 mm and the solid concentration of $11-16$ mm aggregates in this homogeneous mixture is $\varphi=\varphi_0$ (e.g. with $\varphi_0=0.2$), the excess of $11-16$ mm aggregates is represented with $\varphi>\varphi_0$ and the shortage with $\varphi<\varphi_0$, resulting in $w_f(\varphi)>1$ and $w_f(\varphi)<1$, respectively.

5.3 Theoretical Approach

Instead of using an empirical approach like mentioned in Section 5.2, it is also possible to apply existing theoretical approach, in which the physical parameters depend on the volume fraction φ. This is the topic of the current section.

5.3.1 Apparent Viscosity

An example of apparent viscosity for the mixture that is explicitly and theoretically dependent on the volume fraction φ is as follows

$$\eta_1=\mu(\varphi)+\frac{\tau_0(\varphi)}{\dot{\gamma}} \tag{5.11}$$

The term $\mu(\varphi)$ can be considered as a plastic viscosity that depends on the volume fraction φ and likewise the term $\tau_0(\varphi)$ as the corresponding yield stress. In spite of the dependency on φ, the above equation can be considered to represent a Bingham model, at least in the limit when $\varphi=$ constant.

In Eq. (5.11), the plastic viscosity $\mu(\varphi)$ can for example be modeled as by Krieger and Dougherty [51]

$$\mu(\varphi)=\mu(0)\left(1-\frac{\varphi}{\varphi_\mathrm{m}}\right)^{-[\eta]\,\varphi_\mathrm{m}} \tag{5.12}$$

while the yield stress can depend on the work by Chateau, Ovarlez and Trung [56]

$$\tau_0(\varphi)=\tau_0(0)\sqrt{(1-\varphi)\left(1-\frac{\varphi}{\varphi_\mathrm{m}}\right)^{-2.5\,\varphi_\mathrm{m}}} \tag{5.13}$$

In the above two equations, the terms $\mu(0)$ and $\tau_0(0)$ are the values of $\mu(\varphi)$ and $\tau_0(\varphi)$ when $\varphi = 0$. Also, the term φ_m represents the maximum packing fraction (i.e. dense packing fraction). As mentioned before, the term $[\eta]$ is known as the intrinsic viscosity and is a measure of the particle shape. More precisely, the intrinsic viscosity is 2.5 for spherical particles and when the particles deviate from spherical shape, this value is higher [2]. For example, for ground gypsum the reported value is $[\eta] = 3.25$ [2]. Nevertheless, the value of $[\eta] = 2.5$ has been used in relation to the fresh concrete [57].

Although the intrinsic viscosity $[\eta]$ and the maximum packing fraction φ_m can vary between materials, it is reported in [2], that the product of the two appears to be a constant, i.e. $[\eta]\,\varphi_\mathrm{m} \approx 2$ (see Table 7.2 in [2]).

Note that in Eq. (5.13), the intrinsic viscosity $[\eta]$ is not used [56]. However, based on Eq. (5.12), one could suggest that the effect of particle shape should be included in this equation and thus the value "2.5" replaced with $[\eta]$, giving

$$\tau_0(\varphi) = \tau_0(0)\sqrt{(1-\varphi)\left(1-\frac{\varphi}{\varphi_\mathrm{m}}\right)^{-[\eta]\,\varphi_\mathrm{m}}}. \tag{5.14}$$

5.3.2 Code Implementation

The calls to Eqs. (5.11), (5.12), (5.13)/(5.14) and (5.1), using $[\eta] = 3.25$ and $\varphi_\mathrm{m} = 0.55$, are made with the following code in `apparentViscosity.H`:

```
// Maximum packing fraction:
const dimensionedScalar varPhiM("varPhiM", dimless, scalar(0.55)); // 0.75

// Intrinsic viscosity:
const dimensionedScalar etaInVi("etaInVi", dimless, scalar(3.25)); // 2.50

tmp<volScalarField> viscous_5
(
    mag(alpha1)*
    (
        mu*pow(mag(scalar(1) - varPhi/varPhiM), -etaInVi*varPhiM)
        +
        tau0*sqrt
        (
            mag(scalar(1) - varPhi)*pow(mag(scalar(1) - varPhi/varPhiM),
           -2.5*varPhiM)
        // -etaInVi*varPhiM)
        )/(shearRate + delta)
    )
    + mag(scalar(1) - alpha1)*eta2
);
```

5.3.3 Distinction between Matrix and Suspended Particles

The fresh concrete consists of particles with a broad range of mass, dimension, shape and surface texture, suspended in a matrix. The distinction between matrix (i.e. the continuous phase) and suspended particles (i.e. the dispersed phase) is a matter of choice, in contrast to the more traditional suspension of spheres submerged in a Newtonian liquid. In [3], the matrix was defined by pure convenience to be the $0-2\,\mathrm{mm}$ mortar inside the fresh concrete. Such an approach is quite common, for example Mørtsell [58] treats the $0-0.125\,\mathrm{mm}$ filler modified cement paste as matrix, instead of the pre-mentioned mortar.

As mentioned in Section 5.2.2, if the aim is to look at the segregation of particles, the above distinction between matrix and suspended particles needs to be redefined relative to this process. More precisely, the distinction between the two phases must correlate with the domain of particle sizes that are actually segregating. Thus, the division must rather be based on observation rather than by pure convenience as given in the above paragraph.

5.3.4 Maximum Packing Fraction φ_{m}

The maximum packing fraction[25] φ_{m} used in Eqs. (5.12) and (5.13) must be relative to the definition between matrix and suspended particles. For example, using the pre-mentioned division at 11 mm in diameter, one cannot use the maximum packing fraction relative to the whole aggregate range $0-16\,\mathrm{mm}$ used in the mixture. For this whole range, the maximum packing fraction has been reported to be up to 0.75 [57].

By using the (multimodal) convention done in [59], the plastic viscosity and the yield stress are given by (see also [56, 60, 61]):

$$\mu(\phi) = \mu_i \left(1-\frac{\phi_{cem}}{\phi_{cem,c}}\right)^{-2.5\,\phi_{cem,c}} \left(1-\frac{\phi_{sand}}{\phi_{sand,c}}\right)^{-2.5\,\phi_{sand,c}} \left(1-\frac{\phi_{gravel}}{\phi_{gravel,c}}\right)^{-2.5\,\phi_{gravel,c}} \tag{5.15}$$

$$\tau_0(\phi) = \tau_i \sqrt{\frac{1-\phi_{cem}}{\left(1-\frac{\phi_{cem}}{\phi_{cem,c}}\right)^{2.5\,\phi_{cem,c}}}} \sqrt{\frac{1-\phi_{sand}}{\left(1-\frac{\phi_{sand}}{\phi_{sand,c}}\right)^{2.5\,\phi_{sand,c}}}} \sqrt{\frac{1-\phi_{gravel}}{\left(1-\frac{\phi_{gravel}}{\phi_{gravel,c}}\right)^{2.5\,\phi_{gravel,c}}}} \tag{5.16}$$

The terms ϕ_{cem}, ϕ_{sand} and ϕ_{gravel} are the volume fractions of the cement particles *in cement paste*, sand particles *in mortar* and gravel particles *in concrete*, respectively. What is important to note is that terms $\phi_{cem,c}$, $\phi_{sand,c}$ and $\phi_{gravel,c}$ are their respective maximum packing fractions. That is, the term $\phi_{gravel,c}$ is only the maximum packing fractions of gravel particles, and NOT the maximum packing fractions of the combined particle system cement, sand and gravel (i.e. not the maximum packing fractions of the $0-16\,\mathrm{mm}$ used in the mixture).

If it is only the largest particles that participate in the segregation/settling, the terms ϕ_{cem} and ϕ_{sand} are constants. With this, the rheological contribution of the smaller particle

[25] Maximum packing fraction is also known as dense packing fraction and eigen–packing, among other terms.

range (i.e. $0 - 11\,\text{mm}$), including the cement paste, is fixed as well and represented with following

$$\mu(0) = \mu_i \left(1 - \frac{\phi_{cem}}{\phi_{cem,c}}\right)^{-2.5\,\phi_{cem,c}} \left(1 - \frac{\phi_{sand}}{\phi_{sand,c}}\right)^{-2.5\,\phi_{sand,c}} = \text{constant} \tag{5.17}$$

$$\tau_0(0) = \tau_i \sqrt{\frac{1 - \phi_{cem}}{\left(1 - \frac{\phi_{cem}}{\phi_{cem,c}}\right)^{2.5\,\phi_{cem,c}}}} \sqrt{\frac{1 - \phi_{sand}}{\left(1 - \frac{\phi_{sand}}{\phi_{sand,c}}\right)^{2.5\,\phi_{sand,c}}}} = \text{constant} \tag{5.18}$$

The two above equations represent the viscous contribution of the matrix phase. Thus, the zero in $\mu(0)$ and $\tau_0(0)$ is relative to the volume fraction of gravel particles ϕ_{gravel}.

Assuming that only the gravel particles are participating in segregation (i.e. ϕ_{cem} and ϕ_{sand} are constants), then $\phi_{gravel} = \varphi$ and thus $\phi_{gravel,c} = \varphi_\text{m}$. With this, Eqs. (5.15) and (5.16) (with Eqs. (5.17) and (5.18) in mind), relapse into Eqs. (5.12) and (5.13).

In [62] it was proposed that the volume fraction φ in which the concrete viscosity approaches infinity[26] could be defined as the maximum packing fraction φ_m. Using this approach in [59], this value was determined to be $\phi_{gravel,c} = \varphi_\text{m} = 0.645$ for $5 - 25\,\text{mm}$ gravel. For a more narrow particle range like $11 - 16\,\text{mm}$, this value is still lower, say 0.55 or even less. Of course, this depends on the properties of the actual aggregates being used.

In [63], an alternative method of determining the maximum packing fraction φ_m is given, which is based on vibration of aggregates in a cylindrical container, under application of pressure. There, it was recognized that the determination of this value is dependent on the method (or process) which is used. Examples of values presented are 0.628 for $8 - 10\,\text{mm}$ rounded aggregates, while 0.572 for crushed aggregates of the same size domain.

5.3.5 Characteristic Particle Diameter D_a

It should be clear that Eqs. (5.12) and (5.13) are valid for monodispersed particle size distribution (i.e. all particles of one size). Thus, if the definition between the matrix and the suspended particles are as described in Section 5.3.4, one has to assume that the suspended particles range $11 - 16\,\text{mm}$ consist of monosized particles with a characteristic particle diameter of D_a. This value is designated as `Da` in the case file `transportProperties`. For this example, the diameter D_a could represent the mass averaged diameter of the whole collection of gravel particles, ranging from 11 mm to 16 mm. Or a slightly different approach could be used to determine D_a. Note that this term is not used in Eqs. (5.12) and (5.13) and thus in that case, its determination is unimportant. However, this value is used in the calculation of settling by gravity $\mathbf{V}_\text{s}^\text{GR}$ as shown in Section 6.3 (see for example Eq. (6.6)) and thus needs to be determined. Furthermore, this value can also be used in the calculation of the shear (rate) induced particle migration $\mathbf{V}_\text{s}^\text{SR}$ as explained in Section 6.4.

[26] I.e. when the overall mixture becomes unflowable and stiff.

Chapter 6

Settling Velocity $\mathbf{V}_{\mathrm{S}}$

6.1 Drift Velocity of the Dispersed Phase $\mathbf{V}_{\mathrm{dj}}$

6.1.1 Continuous and Dispersed Phases

To reiterate from the previous chapters, the fluid mixture (i.e. phase 1) is a suspension that consists of a continuous phase (i.e. a matrix) and a dispersed phase (i.e. suspended particles). Also, the continuous phase is marked with the subscript c, while the dispersed phase with d.

In Eq. (4.20), Page 35, the term $\mathbf{V}_{\mathrm{dj}}$ represents the drift velocity and has already been treated with Eq. (2.27) on Page 21, and is reproduced below:

$$\mathbf{V}_{\mathrm{dj}} = -\beta_{\mathrm{c}}\mathbf{V}_{\mathrm{r}} = \beta_{\mathrm{c}}\left(\mathbf{V}_{\mathrm{d}} - \mathbf{V}_{\mathrm{c}}\right) \tag{6.1}$$

The term $\mathbf{V}_{\mathrm{r}} = \mathbf{V}_{\mathrm{c}} - \mathbf{V}_{\mathrm{d}}$ is the relative velocity between phases, given by Eq. (2.15) on Page 20. The drift velocity $\mathbf{V}_{\mathrm{dj}}$ is the velocity of the dispersed phase relative to the mixture center of volume and is needed to allow slip between phases (see Section 2.3.1). This is required to allow for the mixture (e.g. fresh concrete) to segregate, either by gravitational settling and/or by other means, like settling by the shear (rate) induced particle migration.

Below are two examples given, to better understand the physical meaning of the drift velocity $\mathbf{V}_{\mathrm{dj}}$, at least relative to the segregation of high viscous mixture like the fresh concrete: Fig. 6.1 shows two different cases of suspensions, one is diluted (a) and the other is concentrated (b). In both cases, the particles and liquid together represents a closed system (no mass is flowing in or out of the system). The settlement of the particle is represented with the velocity of the dispersed phase, namely $\mathbf{V}_{\mathrm{d}}$. The observed settlement is also represented with the settling velocity $\mathbf{V}_{\mathrm{s}}$ (that is, $\mathbf{V}_{\mathrm{d}} = \mathbf{V}_{\mathrm{s}}$).

It should be clear that the settling/segregation in Fig. 6.1 may be induced by gravity (i.e. by difference in densities), and/or by shear (rate) induced particle migration, and/or by other means. The relevant settling phenomenon depends of course on the mixture and the overall flow system that is being investigated.

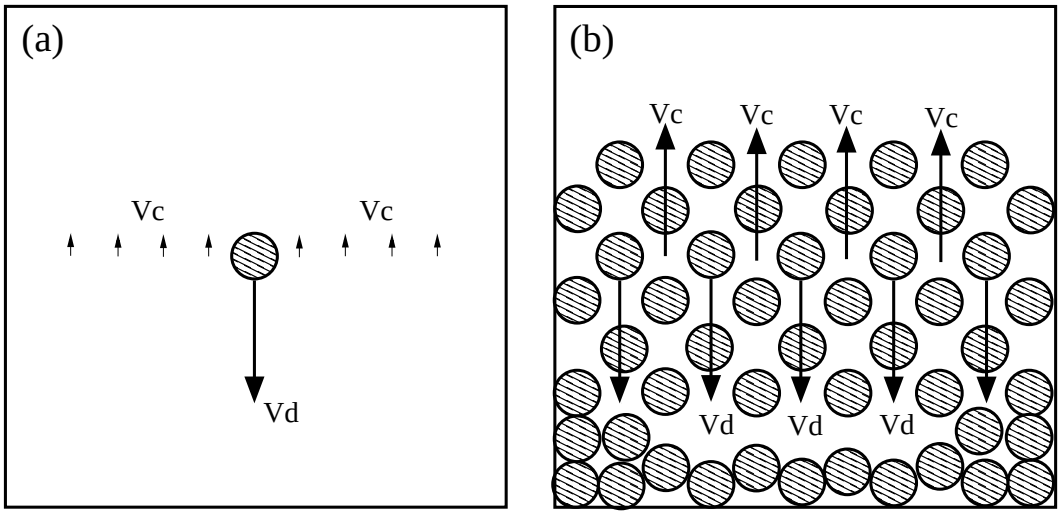


Figure 6.1: Two different cases of suspensions, one is diluted (a) and the other is concentrated (b).

6.1.2 Example 1: Diluted Case

Fig. 6.1a shows an example of settling of a single particle (i.e. of the dispersed phase) in a large liquid medium (the continuous phase). By the conservation of volume in this closed system, the upward velocity of the continuous phase $\mathbf{V}_c$ is initiated by the downward movement of the single particle $\mathbf{V}_d$. That is, as the particle moves downward, an equal volume amount of continuous phase has to move upward. Now, due to the much larger quantity of the continuous phase, i.e. $\beta_c = 1 - \beta_d \approx 1$, and thus a much larger (horizontal) area that the liquid can bypass the particle, the velocity of the continuous phase is more or less zero $\mathbf{V}_c \approx 0$. Thus by Eq. (6.1), the following is obtained:

$$\mathbf{V}_{dj} = \beta_c \left(\mathbf{V}_d - \mathbf{V}_c\right) \approx 1.0 \left(\mathbf{V}_d - 0\right) = \mathbf{V}_d \qquad \text{(Fig. 6.1a)} \tag{6.2}$$

That is, for the above case, the drift velocity $\mathbf{V}_{dj}$ is more or less the same as the velocity of the dispersed phase $\mathbf{V}_d$. Moreover, the latter velocity is the same as the observed settlement, which is represented with the settling velocity $\mathbf{V}_s$ (i.e. here, $\mathbf{V}_d = \mathbf{V}_s$).

6.1.3 Example 2: Concentrated Case

In Fig. 6.1b, the velocity of the continuous phase $\mathbf{V}_c$ is initiated by the downward movement of all the particles. Due to a much larger quantity of particles moving downward, the resulting upward velocity of the continuous phase $\mathbf{V}_c$ is now much larger relative to the previous example. Assuming that the volume fraction of the dispersed phase is $\beta_d = 0.5$ (i.e. half of the system total volume are solids), then by volume conservation principle (i.e. the system is closed and conserved), the upward velocity of the the continuous phase would be similar to that of the particle phase, meaning $\mathbf{V}_c \approx -\mathbf{V}_d$. Thus by Eq. (6.1), the following is obtained:

$$\mathbf{V}_{dj} = \beta_c \left(\mathbf{V}_d - \mathbf{V}_c\right) \approx 0.5 \left(\mathbf{V}_d - (-\mathbf{V}_d)\right) = \mathbf{V}_d \qquad \text{(Fig. 6.1b)} \tag{6.3}$$

That is, for the above case, the drift velocity $\mathbf{V}_{\mathrm{dj}}$ is more or less the same as the velocity of the dispersed phase $\mathbf{V}_{\mathrm{d}}$. As before, the latter velocity is the same as the observed settlement, which is represented with the settling velocity $\mathbf{V}_{\mathrm{s}}$ (i.e. here, $\mathbf{V}_{\mathrm{d}} = \mathbf{V}_{\mathrm{s}}$).

6.1.4 Observed Settling Velocity $\mathbf{V}_{\mathrm{s}}$

From the two above examples, it can be suggested that the drift velocity $\mathbf{V}_{\mathrm{dj}}$ is similar or equal to the observed settling velocity $\mathbf{V}_{\mathrm{s}}$, represented with Eq. (6.4). This approach is for example used in [7], Section 2.4.

$$\mathbf{V}_{\mathrm{dj}} \approx \mathbf{V}_{\mathrm{s}} \tag{6.4}$$

It should be noted that a small dissimilarity between the onset $\mathbf{V}_{\mathrm{dj}}$ (into the source code) and the observed $\mathbf{V}_{\mathrm{s}}$ (from the simulation result) was registered in Section 1.5 (Page 8), giving 5% time difference from the effect of these two velocities.

6.2 Overall Drift Velocity

In order to calculate the mixture flow with settling/segregation (i.e. with slip between phases), the overall velocity of the dispersed phase d relative to the mixture center of volume needs to be calculated, given by

$$\mathbf{V}_{\mathrm{dj}} = \sum_{\mathrm{q}} \mathbf{V}_{\mathrm{dj}}^{\mathrm{q}} = \mathbf{V}_{\mathrm{dj}}^{\mathrm{GR}} + \mathbf{V}_{\mathrm{dj}}^{\mathrm{SR}} + \dots \tag{6.5}$$

where $\mathbf{V}_{\mathrm{dj}}^{\mathrm{GR}}$ is the slip by gravity (Section 6.3) and $\mathbf{V}_{\mathrm{dj}}^{\mathrm{SR}}$ is the slip by shear (rate) induced particle migration (Section 6.4). Other physical processes can be added into Eq. (6.5) as indicated with the dots. As discussed in Section 6.1 and shown with Eq. (6.4), the drift velocity $\mathbf{V}_{\mathrm{dj}}$ is considered to be the observed settling velocity, here designated with $\mathbf{V}_{\mathrm{s}}$ (see also [7]). This is assumed to apply regardless of the physical process responsible for slip between phases, i.e. $\mathbf{V}_{\mathrm{dj}}^{\mathrm{GR}} = \mathbf{V}_{\mathrm{s}}^{\mathrm{GR}}$ and $\mathbf{V}_{\mathrm{dj}}^{\mathrm{SR}} = \mathbf{V}_{\mathrm{s}}^{\mathrm{SR}}$.

6.3 Settling by Gravity

6.3.1 Theory

For a single particle in an infinite medium of Newtonian fluid with the viscosity of μ_{N}, it is relatively straightforward to calculate the settling velocity (see [7, 13]). For a single spherical particle at low Reynolds number, through the equilibrium between weight, buoyancy and drag force, the settling velocity is calculated as (see for example Section 2.3.1 in [13] or Section 2.4.1 in [7])

$$\mathbf{V}_{\mathrm{s}}^{\mathrm{GR}} = \frac{D_{\mathrm{a}}^2\, \mathbf{g}\, (\rho_{\mathrm{d}} - \rho_{\mathrm{c}})}{18\, \mu_{\mathrm{N}}} \tag{6.6}$$

To reiterate, the term μ_{N} is the Newtonian viscosity, $\mathbf{g}$ is gravity, ρ_{c} is the density of the continuous phase, ρ_{d} is the density of the dispersed phase and D_{a} is the particle diameter.

For a single particle submerged in an infinite medium of Bingham viscoplastic fluid with the apparent viscosity of $\eta_1 = \mu + \tau_0/\dot{\gamma}$, some suggestions of settling velocity $\mathbf{V}_{\mathrm{s}}^{\mathrm{GR}}$ has been proposed, based on Eq. (6.6). One suggestion consists of replacing the Newtonian viscosity μ_{N} with the plastic viscosity μ of the Bingham fluid [13]. Another approach consists of using the apparent viscosity η_1 of the Bingham fluid [13, 64, 65]. The third approach is replacing μ_{N} with a so-called tangential viscosity of the Herschel–Bulkley fluid, given by $\mu_{\tan} = n\,K\,\dot{\gamma}^{\mathrm{n}-1}$, where K is the consistency factor and n the consistency index (also, flow index) [13].

In [57], the Newtonian viscosity μ_{N} in Eq. (6.6) is replaced with the local surrounding viscosity of the suspending fluid, modeled as $\mu_{\mathrm{s}} = \eta_1/\lambda$ where $\lambda \geq 1$. As before, η_1 is the mixture[27] apparent viscosity. When λ equals 1, the surrounding fluid has the same behavior as the tested mixture. As the former is more fluid relative to the latter, λ should be higher than 1 [57]. With this approach, Eq. (6.6) is transformed into the following

$$\mathbf{V}_{\mathrm{s}}^{\mathrm{GR}} = \frac{D_{\mathrm{a}}^2\,\mathbf{g}\,(\rho_{\mathrm{d}} - \rho_{\mathrm{c}})}{18\,(\eta_1/\lambda)} \tag{6.7}$$

It should be noted that for hindered settling where the flow field around any one particle is affected by its neighbors, including particle-particle collisions, the settling velocity can also be depended on the solid concentration φ [7, 66]. Thus, Eq. (6.7) may be incomplete. In the solver `vvpfFoam`, a dependency on φ is added by `slowDown2`, which in its current form avoids the continuous filling of a cell with φ if it has reached its designated max capacity φ^{MAX} (for example, equal to 0.4).

6.3.2 Code Implementation

The drift velocity $\mathbf{V}_{\mathrm{dj}}^{\mathrm{GR}}$ is designated with `VdjGR`. Its flux is calculated as $\phi_{\mathrm{dj}}^{\mathrm{GR}} = \mathbf{V}_{\mathrm{dj}}^{\mathrm{GR}} \cdot \mathbf{S}$, or `phiVdjGR = fvc::interpolate(VdjGR) & mesh.Sf();`. The term $\mathbf{S}$ is the face area vector of a cell and ϕ represents the cell face flux [34] (see also Section 4.9). Settling by gravity is implemented in `gravitySegregation.H` and is as follows (with $\lambda = 1.45$):

```
#include "correctViscosity.H"

tmp<volVectorField> VsGR =
    mag(alpha1)*(pow(Da, 2.0)*g*(rhoD - rhoC))/(18.0*(etaEff/1.45));


#ifdef GRAVITY_SEGREGATION
VdjGR =
    slowDown2
    (
        alphaD, // alphaD, or betaD, depending on user preference!
```

[27] For example, by Eq. (5.11), Page 54. Note that in accordance with Eq. (5.1), when exclusively treating the mixture, meaning $\alpha_1 = 1$, then $\eta = \eta_1$. In the source code, η is represented with `etaEff`.

```
        alphaDMIN,
        alphaDMAX
    )*(1.0*VsGR);
#else
VdjGR = zeroVelocity;
#endif


forAll(alpha1.internalField(), celli)
{
    if
    (
        alpha1[celli] > lowerCrit.value()
     && alpha1[celli] < upperCrit.value()
     && alphaD[celli] > criteriaD.value()
    )
    {
        VdjGR[celli] = (1.0 - alpha1[celli])*0.2100*interfaceNormal[celli];
    }
    else if (alpha1[celli] <= lowerCrit.value())
    {
        VdjGR[celli] = (1.0 - alpha1[celli])*0.0306*g.value();
    }
}


forAll(mesh.boundary(), patchi)
{
    VdjGR.boundaryField()[patchi] == vector::zero;

    forAll(alpha1.boundaryField()[patchi], facei)
    {
        if (alpha1.boundaryField()[patchi][facei] < lowerCrit.value())
        {
            VdjGR.boundaryField()[patchi][facei] =
                (1.0 - alpha1.boundaryField()[patchi][facei])*0.0306*g.value();
        }
    }
}


VdjGR.correctBoundaryConditions();

phiVdjGR = fvc::interpolate(VdjGR) & mesh.Sf();
```

6.4 Settling by Shear Induced Particle Migration

6.4.1 Theory

In this work, the term *shear induced particle migration*, will also have the designation *shear* **rate** *induced particle migration*, due to the physical process it originates from (see Section 10.1 in [3]).

Often, the equation for the particle flux is given by [14, 67]

$$\frac{\partial \varphi}{\partial t} + \nabla \cdot (\varphi \mathbf{U}) = -\nabla \cdot \mathbf{N}_t \tag{6.8}$$

The above equation does not account for difference in density between the continuous phase and the dispersed phase and thus is only correct for neutrally buoyant suspension, meaning $\rho_\mathrm{c} = \rho_\mathrm{d} = \rho_1$. The term $\mathbf{N}_t$ represents the particle flux of the dispersed phase with solid concentration of φ and is given by [68] (see also [14, 67])

$$\mathbf{N}_t = \mathbf{N}_c + \mathbf{N}_\eta = -K_c a^2 \varphi \nabla(\dot{\gamma}\varphi) - K_\eta a^2 \dot{\gamma} \varphi^2 \nabla(\ln \eta_1) \tag{6.9}$$

where the terms $K_c = 0.41$ and $K_\eta = 0.62$ are empirical fitted parameters [14, 67, 68]. The term a is the radius of the (characteristic) particle, here equal to $D_\mathrm{a}/2$ (see also Section 5.3.5, Page 57, about the parameter D_a).

The particle flux $\mathbf{N}_t$ accounts for both shear rate induced particle migration as well as compensations by the viscosity gradient induced effects. The latter phenomenon, namely $\mathbf{N}_\eta = -K_\eta a^2 \dot{\gamma} \varphi^2 \nabla(\ln \eta_1)$, accounts for the tendency of particles to migrate away from the high viscous values η_1 to lower values of η_1 [69]. More precisely, as the mixture gets more fluid, the mobility of particles is higher [69].

For Eq. (4.20), Page 35, one can replace the term α_d with φ, c.f. the discussion in the last paragraph of Section 5.1. In this case, the term $\mathbf{U}_\mathrm{r}\, \xi_\mathrm{d}\, (1 - \alpha_1)$ can be excluded since only the mixture (i.e. phase 1) is being considered. Furthermore, since only settling by shear induced particle migration is addressed, then $\mathbf{V}_\mathrm{dj} = \mathbf{V}_\mathrm{dj}^\mathrm{SR}$. Finally, with a neutrally buoyant suspension $\rho_\mathrm{c} = \rho_\mathrm{d} = \rho_1$ (as is assumed in Eq. (6.8)), Eq. (4.20) is transformed into the following

$$\frac{\partial \varphi}{\partial t} + \nabla \cdot (\varphi \mathbf{U}) = -\nabla \cdot (\varphi \mathbf{V}_\mathrm{dj}^\mathrm{SR}) \tag{6.10}$$

Comparing Eqs (6.8) and (6.10), one obtains $\mathbf{N}_t = \varphi \mathbf{V}_\mathrm{dj}^\mathrm{SR}$, meaning

$$\mathbf{V}_\mathrm{dj}^\mathrm{SR} = \frac{\mathbf{N}_t}{\varphi} = -K_c a^2 \nabla(\dot{\gamma}\varphi) - K_\eta a^2 \dot{\gamma} \varphi \nabla(\ln \eta_1) \tag{6.11}$$

With the assistance from the indicial notation and the summation convention [20, 21, 39], as well as using the chain rule, it is possible to calculate $\ln \eta_1$ further

$$\nabla(\ln \eta_1) = \mathbf{i}_q \frac{\partial(\ln \eta_1)}{\partial x_q} = \mathbf{i}_q \frac{\partial(\ln \eta_1)}{\partial \eta_1} \frac{\partial \eta_1}{\partial x_q} = \frac{\partial(\ln \eta_1)}{\partial \eta_1} \nabla \eta_1 = \frac{\nabla \eta_1}{\eta_1} \tag{6.12}$$

Thus, with the above result and $a = D_a/2$, Eq. (6.11) can be rewritten as

$$\mathbf{V}_{dj}^{SR} = -K_c \left(\frac{D_a}{2}\right)^2 \nabla(\dot{\gamma}\varphi) - K_\eta \left(\frac{D_a}{2}\right)^2 \dot{\gamma}\varphi \frac{\nabla\eta_1}{\eta_1} \tag{6.13}$$

or equally

$$\mathbf{V}_{dj}^{SR} = -\frac{D_a^2}{4}\left(K_c \nabla(\dot{\gamma}\varphi) + K_\eta \dot{\gamma}\varphi \frac{\nabla\eta_1}{\eta_1}\right) \tag{6.14}$$

It should be noted that the parameters K_c and K_η may not be constants, but actually depend on φ [70]. Furthermore, Eq. (6.9) is generated from experimentation of small inert neutrally buoyant particles with almost mono-sized particle size distribution [14]. Thus, some modifications of the above might be necessary to accommodate a different mixture type.

In the limited case of $\nabla\eta_1 \approx 0$ and $\varphi \approx$ constant, then Eq. (6.14) becomes

$$\mathbf{V}_{dj}^{SR} \approx -\left(\frac{D_a^2}{4} K_c \varphi\right) \nabla\dot{\gamma} = -k\, \nabla\dot{\gamma} \tag{6.15}$$

For example, with $D_a = 13\,\text{mm}$ and $\varphi \approx 0.2$, then $k = (0.013^2/4)\cdot 0.41 \cdot 0.2 = 0.35 \cdot 10^{-5}\,\text{m}^2$. Eq. (6.15) has been used in relation to self compacting concrete, with $k = 1.4 \cdot 10^{-5}\,\text{m}^2$ [71]. See also Section 10.1 in [3] about the term $-k\,\nabla\dot{\gamma}$ and its physical significance.

6.4.2 Code Implementation

The drift velocity $\mathbf{V}_{dj}^{SR}$ is designated with `VdjSR`. Its flux is calculated as $\phi_{dj}^{SR} = \mathbf{V}_{dj}^{SR} \cdot \mathbf{S}$, or `phiVdjGR = fvc::interpolate(VdjSR) & mesh.Sf();`. As before, the term $\mathbf{S}$ is the face area vector. Settling by the shear rate induced particle migration is implemented in `particleMigration.H` and is as follows (with Eq. (6.15) commented out):

```
#include "correctViscosity.H"


volVectorField gradShearRate(fvc::grad(shearRateAlpha1));
// volVectorField gradShearRate(fvc::grad(shearRate));

// ---
// const dimensionedScalar Ksr("Ksr", dimensionSet(0,2,0,0,0,0,0), scalar(0.8e-5));
// tmp<volVectorField> VsSR = -mag(alpha1)*Ksr*gradShearRate;
// ---

const dimensionedScalar Kc("Kc", dimless, scalar(0.41));
const dimensionedScalar Keta("Keta", dimless, scalar(0.62));
const dimensionedScalar a = Da/2.0;

tmp<volVectorField> VsSR =
    -mag(alpha1)*
```

```
    (
        // Kc*pow(a, 2.0)*fvc::grad(shearRate*alphaD)
        // + Keta*pow(a, 2.0)*shearRate*alphaD*(gradEtaEff/(etaEff + deltaEta))
        Kc*pow(a, 2.0)*fvc::grad(shearRateAlpha1*alphaD)
      + Keta*pow(a, 2.0)*shearRateAlpha1*alphaD
      * (gradEtaEff/(etaEff + deltaEta))
    );


#ifdef PARTICLE_MIGRATION
VdjSR =
    slowDown4
    (
        alphaD, // alphaD, or betaD, depending on user preference!
        alphaDMIN,
        alphaDMAX
    )*(1.0*VsSR);
#else
VdjSR = zeroVelocity;
#endif


forAll(alpha1.internalField(), celli)
{
    if
    (
        alpha1[celli] > lowerCrit.value()
     && alpha1[celli] < upperCrit.value()
     && alphaD[celli] > criteriaD.value()
    )
    {
        VdjSR[celli] = (1.0 - alpha1[celli])*0.2100*interfaceNormal[celli];
    }
    else if (alpha1[celli] <= lowerCrit.value())
    {
        VdjSR[celli] = (1.0 - alpha1[celli])*0.0306*g.value();
    }
}


forAll(mesh.boundary(), patchi)
{
    VdjSR.boundaryField()[patchi] == vector::zero;

    /*
    forAll(alpha1.boundaryField()[patchi], facei)
    {
        if (alpha1.boundaryField()[patchi][facei] < lowerCrit.value())
        {
            VdjSR.boundaryField()[patchi][facei] =
                (1.0 - alpha1.boundaryField()[patchi][facei])*0.0306*g.value();
        }
```

```
    }
    */
}


VdjSR.correctBoundaryConditions();

phiVdjSR = fvc::interpolate(VdjSR) & mesh.Sf();
```

Chapter 7

Summary

A multiphase transient simulator, named `vvpfFoam`, has been developed that models the dynamics of multiple fluid phases of a mixture, for example during casting. The development was realized within the OpenFOAM framework and the starting template was the `interFoam` solver. One of the aims with this solver is to simulate operational problems related to uncertainties in casting predictions of fresh concrete. This includes the effect of the settlement of aggregates by gravity (i.e. segregation) as well as by shear (rate) induced particle migration. Although the `vvpfFoam` solver was designed with fresh concrete in mind, it can be used with other high viscous mixtures as well, e.g. aluminum particles submerged in oil. Also, other types of cement-based material can be analyzed with this solver, like the fresh mortar. For the most important analysis of this project, simulations were performed on supercomputers at the Icelandic High Performance Computer Center (ihpc.is). The analysis include explanations of how reinforcement shadows can form on a concrete surface after casting a wall section and how the effect of segregation can be moved further downstream by advection.

The solver has incorporated two theories:

- The first theory is the volume of fluid approach (VOF), which is needed to divide the system between the atmospheric air and the fluid mixture. The fluids do not generally intermix (immiscible) and thus usually have a clear boundary between them.
- The second theory is the implementation of field equation to be able to calculate settling/segregation within the fluid mixture, by the effect of gravity, by the shear (rate) induced particle migration and/or by other means. The fluid phases are usually in an intermixed stated (miscible). The approach used is the Drift Flux Model (DFM).

In addition to the issues mentioned in Section 1.7, there are most certainly other currently unknown problems with this solver. However, as the solver is open and licensed under the GNU General Public License (see Appendix C), as applies for OpenFOAM, the

user has the opportunity to investigate, test and repair it. The user can modify the code, add new capabilities and otherwise enhance (or downgrade) it to the specification needed.

Appendix A

Source Code Overview

This is the overview of the source code files of the solver `vvpfFoam`. At the time of writing, it consists of 27 files. Few of these are more or less unchanged from the template solver (namely the `interFoam` solver).

- `vvpfFoam.C`: The main file, holding everything together.
- `alpha1CourantNo.H`: Courant number calculation.
- `alpha1Eqn.H`: Eq. (4.23) with $\rho_1 = \text{constant}$ (Page 36).
- `alpha1EqnRho.H`: Eq. (4.23) with $\rho_1 \neq \text{constant}$. See also Fig. 4.2, Page 37.
- `alpha1EqnSubCycle.H`: Iteration of Eq. (4.23).
- `alpha1Interface.H`: Interface treatment at boundary between α_1 and α_2.
- `alphaDCourantNo.H`: Courant number calculation.
- `alphaDEqn.H`: Eq. (4.20), Page 35.
- `alphaDEqnSubCycle.H`: Iteration of Eq. (4.20).
- `apparentViscosity.H`: Chapter 5, Page 49.
- `comprsblContErrs.H`: Monitoring of $\nabla \cdot \mathbf{U}$. To active this, `ERROR_ANALYSIS` must be definded in `macroDefinitions.H`.
- `correctPhi.H`: Unchanged from the original `interFoam` solver.
- `correctViscosity.H`: Call to η_1 (see `apparentViscosity.H`).
- `createFields.H`: Creation of the main field variables.
- `createFunctions.H`: Various functions needed to control the drift velocity at boundary as well as in the bulk. See `gravitySegregation.H` and `particleMigration.H`.

- densityContErrs.H: Monitoring of $\partial\rho/\partial t + \nabla \cdot (\rho\,\mathbf{U})$ (see Eq. 4.46, Page 42). To active this, ERROR_ANALYSIS must be definded in macroDefinitions.H.
- driftVelocity.H: Section 6.2, Page 60.
- enableFieldControl.H: Various criteria imposed on some of the field variables. Also, calculation of $\mathbf{U}_{\text{in}} = \mathbf{U} + \boldsymbol{\omega} \times \mathbf{x}$ if SINGLE_REFERENCE_FRAME is defined in macroDefinitions.H (see the text below Eq. (4.43) on Page 41).
- gravitySegregation.H: Section 6.3, Page 60.
- macroDefinitions.H: Conditional compilation with macro definitions.
- particleMigration.H: Section 6.4, Page 63.
- pEqn.H: Calculation of the pressure $p_$rgh either by Eq. (4.50) or by Eq. (4.51), depending on if USE_PRESSURE_RESIDUE is definded in macroDefinitions.H or not. Explicit velocity correction is also done in this file, c.f. Section 4.9.2.
- pEqnResidueErrs.H: Monitoring of $\nabla \cdot \mathbf{R}$ (see Eq. 4.52, Page 43). To active this, ERROR_ANALYSIS must be definded in macroDefinitions.H.
- rho1MaxMinFields.H: Defining max and min of ρ_1 as a field variable and setting the corresponding value.
- setDeltaT.H: Adjustment of the time step Δt, based on alpha1CourantNo.H and alphaDCourantNo.H.
- transportProperties.H: Reads the case file ./constant/transportProperties. Note that this file is relative to return viscous_5() in apparentViscosity.H. For most other rheological models defined in apparentViscosity.H, the material parameters are set in the source file, and thus a recompilation is needed for such usage. A different return viscous_X() must also be set. The user can rewrite transportProperties.H for the particular rheological model needed.
- UEqn.H: Setup of $\mathbf{H}(\mathbf{U})$ either by Eq. (4.42) or by Eq (4.43), depending on if SINGLE_REFERENCE_FRAME is defined in macroDefinitions.H or not. If the case file ./system/fvSolution has momentumPredictor yes, Eq. (4.42)/(4.43) will be solved in order to generate a first guess for the velocity $\mathbf{U}$. Usually, the condition is set as momentumPredictor no.

Appendix B

Compilation

B.1 ./Make/options

The solver can be compiled within OpenFOAM 2.2.0 to 2.2.2. When compiling the solver `vvpfFoam` in OpenFOAM 2.2.0, the file `./Make/options` must consist of the following:

```
EXE_INC = \
    -I$(LIB_SRC)/transportModels/twoPhaseInterfaceProperties/ [cont. next line]
        alphaContactAngle/alphaContactAngle \
    -I$(LIB_SRC)/transportModels \
    -I$(LIB_SRC)/transportModels/incompressible/lnInclude \
    -I$(LIB_SRC)/transportModels/interfaceProperties/lnInclude \
    -I$(LIB_SRC)/finiteVolume/lnInclude

EXE_LIBS = \
    -ltwoPhaseInterfaceProperties \
    -lfiniteVolume
```

However, when compiling the solver in OpenFOAM 2.2.1 or OpenFOAM 2.2.2, the file `./Make/options` must consist of:

```
EXE_INC = \
    -I$(LIB_SRC)/transportModels/twoPhaseProperties/ [cont. next line]
        alphaContactAngle/alphaContactAngle \
    -I$(LIB_SRC)/transportModels \
    -I$(LIB_SRC)/transportModels/incompressible/lnInclude \
    -I$(LIB_SRC)/transportModels/interfaceProperties/lnInclude \
    -I$(LIB_SRC)/finiteVolume/lnInclude

EXE_LIBS = \
    -ltwoPhaseProperties \
    -lfiniteVolume
```

Examples of each setup is present in `./Make/options` by the names `options_OF220.txt` and `options_OF221_OF222.txt`, respectevly.

B.2 Fedora Linux

The following compile instruction has been tested on Fedora 18. The outcome has also been tarballed and moved to supercomputers[28] using CentOS 7.x, without issues.

When compiling OpenFOAM 2.2.0 to 2.2.2, the steps are as follows:

```
(1) In $HOME/.bashrc, add
alias of220='source $HOME/OpenFOAM/OpenFOAM-2.2.0/etc/bashrc && PS1="[2.2.0][\W] " '
alias of221='source $HOME/OpenFOAM/OpenFOAM-2.2.0/etc/bashrc && PS1="[2.2.1][\W] " '
alias of222='source $HOME/OpenFOAM/OpenFOAM-2.2.0/etc/bashrc && PS1="[2.2.2][\W] " '

(2) New terminal + of220
in $WM_THIRD_PARTY_DIR
tar xzf cmake-2.8.3.tar.gz
./makeCmake

(3) New terminal + of220
 in $WM_THIRD_PARTY_DIR
./makeParaView -qmake $(which qmake-qt4)

(4) New terminal + of220
in $WM_THIRD_PARTY_DIR
./Allwmake

(5) New terminal + of220
foam
export WM_NCOMPPROCS=$(cat /proc/cpuinfo | grep processor | wc -l)
./Allwmake 2>&1 | tee wmake_log_file.txt
```

When finished, make sure that openmpi exists in the third-party-dir:

```
ls $WM_THIRD_PARTY_DIR/platforms/linux64Gcc/openmpi-1.6.3
```

B.3 Ubuntu Linux

OpenFOAM 2.2.2 can be compiled on Ubuntu 18.04, by following the steps provided in

```
https://openfoamwiki.net/index.php/Installation/Linux/OpenFOAM-2.2.2/Ubuntu
```

The approach has been tested on a fresh Ubuntu 18.04 (Gnome) installation without problems. Also, the solver `vvpfFoam` compiles and run without issues. On this note, the latest part of the solver development was done on computer running Xubuntu 18.04, using OpenFOAM binaries generated on a Fedora Linux workstation.

[28] Owned and hosted by the Icelandic High Performance Computing Centre (ihpc.is).

Appendix C

GNU General Public License

GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/> Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions. "This License" refers to version 3 of the GNU General Public License. "Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code. The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions. All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law. No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies. You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions. You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms. You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms. "Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination. You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies. You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients. Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents. A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom. If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License. Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License. The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty. THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16. If the disclaimer of warranty and limitation of liability provided

above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

Bibliography

[1] H.G. Weller, G. Tabor, H. Jasak, C. Fureby, A tensorial approach to computational continuum mechanics using object-oriented techniques, Comput. Phys. 12 (1998) 620 - 631.

[2] H.A. Barnes, J.F. Hutton, K. Walters, An Introduction to Rheology, Elsevier Science B.V., Netherlands, 1989.

[3] J.E. Wallevik, Rheology of Particle Suspensions – Fresh Concrete, Mortar and Cement Paste with Various Types of Lignosulfonates, Ph.D. thesis, Department of Structural Engineering, The Norwegian University of Science and Technology, Trondheim, Norway, 2003, https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/236410.

[4] J. Spangenberg, N. Roussel, J.H. Hattel, H. Stang, J. Skocek, M.R. Geiker, Flow induced particle migration in fresh concrete: Theoretical frame, numerical simulations and experimental results on model fluids, Cem. Concr. Res. 42 (2012) 633 - 641.

[5] C.W. Hirt, B.D. Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries, J. Comput. Phys. 39 (1) (1981) 201 - 225.

[6] M. Ishii, Thermo-Fluid Dynamic Theory of Two-Phase Flow, Eyrolles, 1975.

[7] D. Brennan, The Numerical Simulation of Two-Phase Flows in Settling Tanks, Ph.D. thesis, Department of Mechanical Engineering, Imperial College of Science, London, UK, 2001.

[8] N. Roussel, Computational Modelling of SCC Flow: Reinforcement Network Modelled as a Porous Medium, Proceedings of the 3rd International RILEM Symposium on Rheology of Cement Suspensions such as Fresh Concrete, August 19-21, 2009, Reykjavik, Iceland, RILEM Publications S.A.R.L., ISBN: 978-2-35158-091-2.

[9] G.H. Tattersall, P.F.G. Banfill, The Rheology of Fresh Concrete, Pitman Books Limited, Great Britain, 1983.

[10] A.M. Neville, Properties of Concrete, Addison Wesley Longman Limited, Great Britain, 1995.

[11] J. Spangenberg, C. Tutum, J. Hattel, N. Roussel, M. Geiker, Optimization of Casting Process Parameters for Homogeneous Aggregate Distribution in Self–Compacting Concrete: A Feasibility Study, Proceedings of the 6th IEEE Congress on Evolutionary Computation, 2011.

[12] K. Vasilic, N. Roussel, B. Meng, H-C. Kuhne, Computational Modelling of SCC Flow: Reinforcement Network Modelled as Porous Medium, Proceedings of the 3rd International RILEM Symposium on Rheology of Cement Suspensions such as Fresh Concrete, August 19-21, 2009, Reykjavik, Iceland, RILEM Publications S.A.R.L., ISBN: 978-2-35158-091-2.

[13] J. Spangenberg, Numerical Modelling of Form Filling with Self–Compacting Concrete, Ph.D. thesis, Department of Mechanical Engineering, Technical University of Denmark (DTU), Lyngby, 2012.

[14] Z. Fang, N. Phan-Thien, Numerical simulation of particle migration in concentrated suspensions by a finite volume method, J. Non-Newtonian Fluid Mech. 58 (1995) 67 - 81.

[15] M. Manninen, V. Taivassalo, S. Kallio, On the Mixture Model for Multiphase Flow, Technical Research Centre of Finland, VTT Publications 288, 1996.

[16] S.M. Damian, An Extended Mixture Model for the Simultaneous Treatment of Short and Long Scale Interfaces, Ph.D. thesis, Facultad de Ingenieria y Ciencias Hidricas, Universidad Nacional del Litoral, 2013.

[17] M. Ishii, One-Dimensional Drift-Flux Model and Constitutive Equations for Relative Motion Between Phases in Various Two-Phase Flow Regimes, Report ANL-77-47, Argonne National Laboratory, Illinois, USA, 1977.

[18] Drew, D.A. Mathematical Modeling of Two-Phase Flow, Ann. Rev. Fluid Mech. 15 (1983) 261-291.

[19] H. Rusche, Computational Fluid Dynamics of Dispersed Two-Phase Flows at High Phase Fractions, Ph.D. thesis, Department of Mechanical Engineering, Imperial College of Science, Technology & Medicine, London, 2002.

[20] G.E. Mase, Schaums Outline Series: Theory and Problems of Continuum Mechanics, McGraw-Hill Inc., USA, 1970.

[21] L.E. Malvern, Introduction to the Mechanics of Continuous Medium, Prentice-Hall Inc., New Jersey, 1969.

[22] D. Gerlach, G. Tomar, G. Biswas, F. Durst, Comparison of volume–of–fluid methods for surface tension–dominant two–phase flows, Int. J. Heat. Mass. Tran. 49 (2006) 740 - 754.

[23] E. Berberovic, Investigation of Free-surface Flow Associated with Drop Impact: Numerical Simulations and Theoretical Modeling, Ph.D. thesis, Technical University of Darmstadt, Germany, 2010.

[24] V.R. Gopala, B.G.M. van Wachem, Volume of fluid methods for immiscible–fluid and free–surface flows, Chem. Eng. J. 141 (2008) 204 - 221.

[25] J. Klostermann, K. Schaake, R. Schwarze, Numerical simulation of a single rising bubble by VOF with surface compression, Int. J. Numer. Meth. Fluids 71(8) (2013) 960 - 982.

[26] K. Kissling, J. Springer, H. Jasak, S. Schutz, K. Urban, M. Piesche, A Coupled Pressure Based Solution Algorithm Based on the Volume–of–Fluid Approach for Two or More Immiscible Fluids, Fifth European Conference on Computational Fluid Dynamics, ECCOMAS CFD 2010, J.C.F. Pereira and A. Sequeira (Eds.), Lisbon, Portugal, 14–17 June 2010.

[27] D.L. Youngs, An Interface Tracking Method for a 3D Eulerian Hydrodynamics Code, Technical Report AWRE/44/92/35, Atomic Weapons Research Establishment, 1987.

[28] W. Noh, P. Woodward, SLIC (simple line interface calculation), Proceedings of the Fifth International Conference on Fluid Dynamics, in: A.I. van de Vooren, P. Zandbergen (Eds.), Lecture Notes in Physics, Vol. 59, Springer-Verlag, Berlin, 1976.

[29] T. Waclawczyk, T. Koronowicz, Comparison of CICSAM and HRIC high-resolution schemes for interface capturing, J. Theor. Appl. Mech. 46(2) (2008) 325 - 345.

[30] O. Ubbink, Numerical Prediction of Two Fluid Systems with Sharp Interfaces, Ph.D. thesis, Imperial College of Science, Technology & Medicine, 1997.

[31] O. Ubbink, R.I. Issa, Method for capturing sharp fluid interfaces on arbitrary meshes, J. Comput. Phys. 153 (1999) 26 - 50.

[32] S. Muzaferija, M. Peric, P. Sames, T. Schelin, A Two–Fluid Navier–Stokes Solver to Simulate Water Entry, Proc. Twenty-Second Symposium on Naval Hydrodynamics, 1998.

[33] Weller, H.G.: Derivation, Modelling and Solution of the Conditionally Averaged two-phase Flow equations, Technical Report TR/HGW/02, OpenCFD Ltd., 2005.

[34] OpenFOAM User Guide, Version 7, The OpenFOAM Foundation, 2019.

[35] J.P. Boris, D.L. Book, Flux-corrected transport, I. SHASTA, a fluid transport algorithm that works, J. Comput. Phys. 11 (1973) 38 - 69.

[36] S.T. Zalesak, Fully multidimensional flux-corrected transport algorithms for fluids, J. Comput. Phys. 31 (1979) 335 - 362.

[37] J.U. Brackbill, D.B. Kothe, and C. Zemach, A continuum method for modelling surface tension, J. Comput. Phys. 100 (1992) 335 - 354.

[38] R.I. Tanner, K. Walters, Rheology: An Historical Perspective, Elsevier Science B.V., Netherlands, 1998.

[39] G.T. Mase, G.E. Mase, Continuum Mechanics for Engineers, 2nd Edition, CRC Press LLC, 1999.

[40] M. Bercovier, M. Engelman, A finite element method for incompressible non–Newtonian flows, J. Comput. Phys. 36 (1980) 313 - 326.

[41] A.J. Taylor, S.D.R. Wilson, Conduit flow of an incompressible, yield–stress fluid, J. Rheol. 41(1) (1997) 93 - 101.

[42] G.R. Burgos, A.N. Alexandrou, V. Entov, On the determination of yield surfaces in Herschel–Bulkley fluids, J. Rheol. 43(3) (1999) 463 - 483.

[43] J.E. Wallevik, Minimizing end–effects in the coaxial cylinders viscometer: Viscoplastic flow inside the ConTec BML Viscometer 3, J. Non-Newtonian Fluid Mech. 155 (2008) 116 - 123.

[44] J.E. Wallevik, Thixotropic investigation on cement paste: experimental and numerical approach, J. Non-Newtonian Fluid Mech. 132 (2005) 86 - 99.

[45] D. Feys, J.E. Wallevik, A. Yahia, K.H. Khayat, O.H. Wallevik, Extension of the Reiner–Riwlin equation to determine modified Bingham parameters measured in coaxial cylinders rheometers, Mater. Struct. 46 (2013) 289 - 311.

[46] J.E. Wallevik, K. Krenzer, J-H. Schwabe, Numerical Errors in CFD and DEM Modeling, in Simulation of Fresh Concrete Flow (N. Roussel and A. Gram, Eds.), Springer Netherlands, 2014.

[47] Ansys Fluent 6.3 User's Guide 2006, ANSYS, Inc., USA, 2006.

[48] M.L. Salby, Fundamentals of Atmospheric Physics, Academic Press, 1996.

[49] F.P. Karrholm, Rhie-Chow Interpolation in OpenFOAM, Department of Applied Mechanics, Chalmers University of Technology, Goteborg, Sweden, 2006.

[50] A. Einstein, Eine neue bestimmung der moleküldimensionen, Ann. Phys. 19 (1906) 289 - 306.

[51] I. Krieger, T. Dougherty, A Mechanism for non-Newtonian flow in suspensions of rigid spheres, Trans. Soc. Rheol. 3 (1959) 137 - 152.

[52] J.E. Wallevik, Effect of the hydrodynamic pressure on shaft torque for a 4-blades vane rheometer, Int. J. Heat Fluid Flow 50 (2014) 95 - 102.

[53] H.P. Langtangen, Computational Partial Differential Equations, Numerical Methods and Diffpack Programming, Lecture Notes in Computational Science and Engineering, Springer-Verlag, Berlin, 1999.

[54] F. Irgens, Continuum Mechanics, Springer–Verlag, Berlin, 2008.

[55] W.H. Herschel, R. Bulkley, Konsistenzmessungen von Gummi-Benzollösungen, Kolloid Zeitschrift, 39 (1926) 291 - 300.

[56] X. Chateau, G. Ovarlez, K.L. Trung, Homogenization approach to the behavior of suspensions of non-colloidal particles in yield stress fluids, J. Rheol. 52 (2008) 489 - 506.

[57] J. Spangenberg, N. Roussel, J.H. Hattel, E.V. Sarmiento, G. Zirgulis, M.R. Geiker, Patterns of gravity induced aggregate migration during casting offluid concretes, Cem. Concr. Res. 42 (2012) 1571 - 1578.

[58] E. Mørtsell, The Effect of the Constituent Materials on the Rheology of Fresh Concrete, The Norwegian University of Science and Technology, Ph.D. thesis (in Norwegian), Trondheim, Norway, 1996.

[59] S.D. Jo, C.K. Park, J.H. Jeong, S.H. Lee, S.H. Kwon, A computational approach to estimating a lubricating layer in concrete pumping, CMC, 27(3) (2012) 189 - 210.

[60] H. Hafid, G. Ovarlez, F. Toussaint, P.H. Jezequel, N. Roussel, Estimating Measurement Artifacts in Concrete Rheometers from MRI Measurement on Model Materials, Proceedings of SCC2010, Motreal, Canada, 2010.

[61] V.N. Nerella, V. Mechtcherine, Virtual Sliding Pipe Rheometer for estimating pumpability of concrete, Constr. Build. Mater. 170 (2018) 366 - 377.

[62] D.M. Liu, Particle packing and rheological property of highly-concentrated ceramic suspensions: Determination and viscosity prediction, J. Mater. Sci. 35 (2000) 5503 - 5507.

[63] F. de Larrard, Concrete Mixture Proportioning – a Scientific Approach, F & FN Spon, USA, 1999.

[64] N. Roussel, A theoretical frame to study stability of fresh concrete, Mater. Struct. 39 (2006) 81 - 91.

[65] L. Shen, L. Struble, D. Lange, Modeling static segregation of self-consolidating concrete, ACI Mat. J. 106(4) (2009) 367 - 374.

[66] A. Bond, Behaviour of Suspensions, Civil Engineering Transactions of the Institution of Engineers, Australia, March 1959.

[67] B.J. Medhi, M.M. Reddy, A. Singh, Particle migration of concentrated suspension flow in bifurcating channels, Adv. Powder Technol. 30 (2019) 1897 - 1909

[68] R.J. Philips, R.C. Armstrong, R.A. Brown, A.L. Graham, J.R. Abbott, A constitutive equation for concentrated suspension that accounts for shear-induced particle migration, Phys. Fluids A, 4 (1992) 30 - 40.

[69] T. Dbouk, Rheology of Concentrated Suspensions and Shear-Induced Particles Migration, Ph.D. thesis, University of Nice-Sophia Antipolis, LPMC UMR - CNRS 7336, Nice, France, 2011.

[70] J.M. Kim, S.G. Lee, C. Kim, Numerical simulations of particle migration in suspension flows: Frame-invariant formulation of curvature-induced migration, J. Non-Newtonian Fluid Mech., 150 (2008) 162 - 176.

[71] J.E. Wallevik, W. Mansour, O.H. Wallevik, Computational Segregation Analysis During Casting of SCC. In: Mechtcherine V., Khayat K., Secrieru E. (eds), Rheology and Processing of Construction Materials, RheoCon 2019, SCC 2019, RILEM Bookseries, Vol. 23, Springer, Cham, 2020.