POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

EXECUTIVE SUMMARY OF THE THESIS

Mixed Eulerian - Lagrangian Polydispersed 3D Droplet Tracking

LAUREA MAGISTRALE IN AERONAUTICAL ENGINEERING - INGEGNERIA AERONAUTICA

Author: GIUSEPPE ALEXANDER GAARD SIRIANNI

Advisor: PROF. ALBERTO MATTEO ATTILIO GUARDONE

Co-advisors: ENG. TOMMASO BELLOSTA, PROF. BARBARA RE

Academic year: 2020-21

1. Introduction

In-flight icing is one of the major threats posed to air-borne vehicles. Ice accreting on aerodynamic surfaces can deteriorate aerodynamic performance, and if it detaches (because of the deployment of an Ice Protection System or because of aerodynamic forces) it can cause considerable damage to aft components such as the first stages of a compressor or the tail empennage. Computing ice shapes requires the knowledge of the droplet trajectories impinging on aerodynamic surfaces. These are then used to obtain the amount of water being caught, measured using the collection efficiency β. Droplet trajectories can be obtained by tracking each droplet individually (Euler-Lagrange) or following a mixed fluid approach (Euler-Euler). Currently, most of the research effort is aimed at obtaining a better understanding of the behavior of Appendix O droplets (median volumetric diameter, or MVD, greater than $40\mu m$) due to their inherently faster ice accretion rate. In this work an Euler-Euler approach will be used to track droplets in a 2D or 3D unstructured mesh in order to compute the collection efficiency due to the free-stream cloud. The equations being considered are the Berthon et al. [3] relaxed pressureless gas dynamics conservation laws, solved using a Godunov finite volume solver with MUSCL. The collection efficiency is then corrected to account for mass loss due to droplet splashing and rebounding using the model presented by Wright [8]. Following this, the removed mass is re-injected using an Euler-Lagrange approach in order to compute the re-impingement collection efficiency. This approach has been followed for two main reasons: firstly, in an Euler-Euler approach, re-injecting droplets would require that each airfoil surface mesh element be converted one by one from an outlet of droplets to an inlet of re-injected droplets, as done by Bilodeau et al. [4]. This is extremely expensive, and according to Bilodeau et al. it causes a 500% to 900% increase in computational cost even when employing grid restriction techniques. Secondly, the heterogeneous nature of the re-injected droplets is more suited for individual tracking rather than for a mixed fluid approach. This heterogeneousness is further increased by considering poly-dispersity (multi-bin).

The multi-bin Eulerian solver with Lagrangian re-injection developed in this work has been tested against experiments from Papadakis et al. [6, 7], namely a 2D NACA 23012, a 2D three-element airfoil and a 3D horizontal swept tail in a wind tunnel. A simple bin restart strategy has been used to mitigate the inherent computational cost increase of a multi-bin simulation and its effects have been evaluated, together with the cost of the Lagrangian re-injection step.

2. Theory

To compute the Euler-Euler droplet trajectories, the carrier fluid field is obtained by solving the Euler equations, whereas the particles equations are computed from the solution of the relaxed PGD system of Berthon et al. [3] (Eq. (1)). Both set of equations are discretized using the finite volume method with a standard edge-based structure on a dual grid with control volumes constructed using a median-dual vertex-based scheme in SU2 [5]. Convective fluxes are discretized using a limited second order MUSCL with the Venkatakrishnan slope limiter. For the Euler equations, the approximate Riemann solver of Roe is employed, whereas for the particle tracking system the exact Riemann problem is solved at each edge to compute the fluxes. Source terms are approximated at each node using a piecewise-constant reconstruction within each control volume. Gradients are obtained via a weighted least-squares approach. A time-marching approach is used to drive the systems to steady state using an implicit Euler scheme.

$$\begin{cases} \dfrac{\partial \alpha}{\partial t} & + \overline{\nabla} \cdot (\alpha \underline{\boldsymbol{u}}_p) = 0 \\ \dfrac{\partial \alpha \underline{\boldsymbol{u}}_p}{\partial t} & + \overline{\nabla} \cdot \left(\alpha \underline{\boldsymbol{u}}_p \otimes \underline{\boldsymbol{u}}_p + \pi \underline{\underline{\boldsymbol{I}}}\right) = \alpha \underline{\boldsymbol{f}}_{drag} \\ \dfrac{\partial \alpha \pi}{\partial t} & + \overline{\nabla} \cdot \left(\alpha \pi \underline{\boldsymbol{u}}_p + c^2 \underline{\boldsymbol{u}}_p\right) = -\lambda \alpha \pi \end{cases} \tag{1}$$

Lagrangian re-injection is performed using an in-house solver called PoliDrop through the Discrete Parcel Method (DPM), therefore tracking parcels that contain a set of identical droplets instead of each physical droplet. The Lagrangian trajectories are obtained integrating Eq. (2), and more can be found in Bellosta et al. [2].

$$\begin{cases} m_p \dfrac{d\underline{\boldsymbol{u}}_p}{dt} & = \dfrac{\pi}{8} \mu_f d_p Re_d \left(\underline{\boldsymbol{U}} - \underline{\boldsymbol{u}}_p\right) C_D(Re_d) + \\ & + \dfrac{\pi}{6} d_p^3 \underline{\boldsymbol{g}} \left(\rho_p - \rho_f\right) \\ Re_d = \frac{\rho_f (\underline{\boldsymbol{U}} - \underline{\boldsymbol{u}}_p) d_p}{\mu_f} \end{cases} \tag{2}$$

The complete computational procedure is recapped in a flow-chart in Fig. (1).

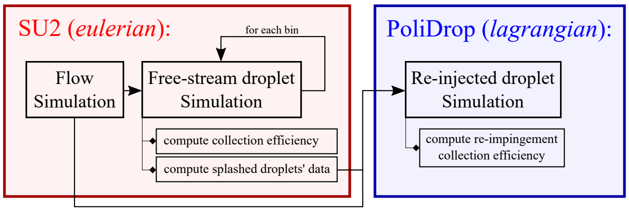

SU2 (*eulerian*):
for each bin
Flow Simulation
Free-stream droplet Simulation
compute collection efficiency
compute splashed droplets' data
PoliDrop (*lagrangian*):
Re-injected droplet Simulation
compute re-impingement collection efficiency

Figure 1: Computational scheme flow chart

Given the steady nature of the computations, all splashing variables are expressed as fluxes of mass per unit time on a surface element. The model used is the one presented by Wright [8] and in Eq. (3) only "flux" modified quantities are reported, since all other quantities are identical to the model.

$$\begin{cases} Splash = K_{L,n} - 200 > 0 \ \text{ splashing occurs} \\ \beta_s = \beta_i \left(1 - \frac{LWC_s |\underline{\boldsymbol{u}}_{p,s}|}{LWC_i |\underline{\boldsymbol{u}}_{p,i}|}\right) \\ L\dot{W}C = LWC \left(\underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}}\right) \\ L\dot{W}C_s = L\dot{W}C_i \cdot 0.7 \left(1 - \sin\theta_i\right) \times \\ \qquad \times \left[1 - e^{-0.0092026 \cdot Splash}\right] \\ \dot{n}_s = \dfrac{mass\ flux_{splashed\ water\ on\ face}}{mass_{single\ splashed\ droplet}} = \\ \quad = \dfrac{L\dot{W}C_s}{\rho_p} \dfrac{3\pi}{4} \left(\dfrac{2}{d_{p,s}}\right)^3 \cdot area \end{cases} \tag{3}$$

To avoid numerical issues due to floating point computations parcels are not re-injected exactly at the surface but with a small displacement from the wall as shown in Fig. (2a). To increase the smoothness of the lagrangian re-impingement collection efficiency, instead of re-injecting one parcel per surface mesh element these can be split up and spaced by the user as in Fig. (2b) (with the consequent increase in computational cost due to the need of tracking more parcels).
The final collection efficiency ($\beta = \beta_{free\text{-}stream} - \beta_{splash} + \beta_{re\text{-}impingement}$) is composed as Eq. (4).

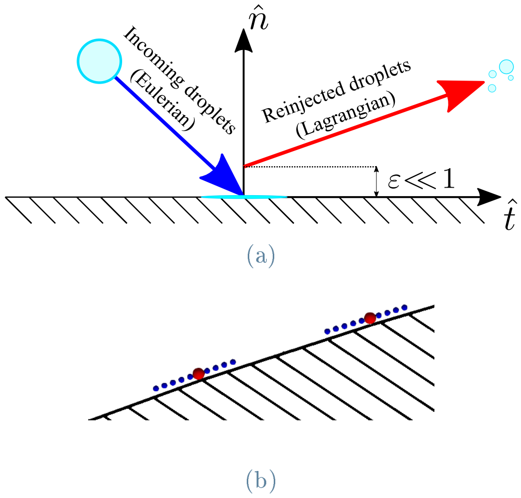

$\hat{n}$
Incoming droplets (Eulerian)
Reinjected droplets (Lagrangian)
$\varepsilon \ll 1$
$\hat{t}$
(a)
(b)

Figure 2: Lagrangian reinjection artificial wall displacement and spacing

$$\beta = \sum_{j=1}^{N_{bins}} \beta_s^j + \frac{\dot{m}}{area \cdot LWC_\infty U_\infty} \tag{4}$$

Lastly, to reduce the computational cost of running a multi-bin simulation, each bin after the first is restarted from the previous bin's converged solution, in an effort to reduce the amount of inner iterations needed to achieve convergence.

3. Results

The solver developed has been tested against experimental data by Papadakis et al. [6, 7] on 2D single and three element airfoils, and a 3D horizontal tail in a wind tunnel with various *MVD* values. Results show that considering both poly-dispersity and re-injection yields the best results on aft surfaces in multi-element configurations such as in Fig. (3). The collection efficiency peak on the main body in Fig. (3c) increases by 12% while on the flap in Fig. (3d) the peak increases by 50%, matching experimental data. Also, the overall solution quality of the flap's collection efficiency is increased, locally by up to 100%. This has been a difficult experimental measure to match until now, as seen from the results shown by many different researchers at the 2021 AIAA's 1st Icing Prediction Workshop [1] .
Looking at the results for the 3D horizontal swept tail in a wind tunnel [6], the re-impingement correction is small (in the order

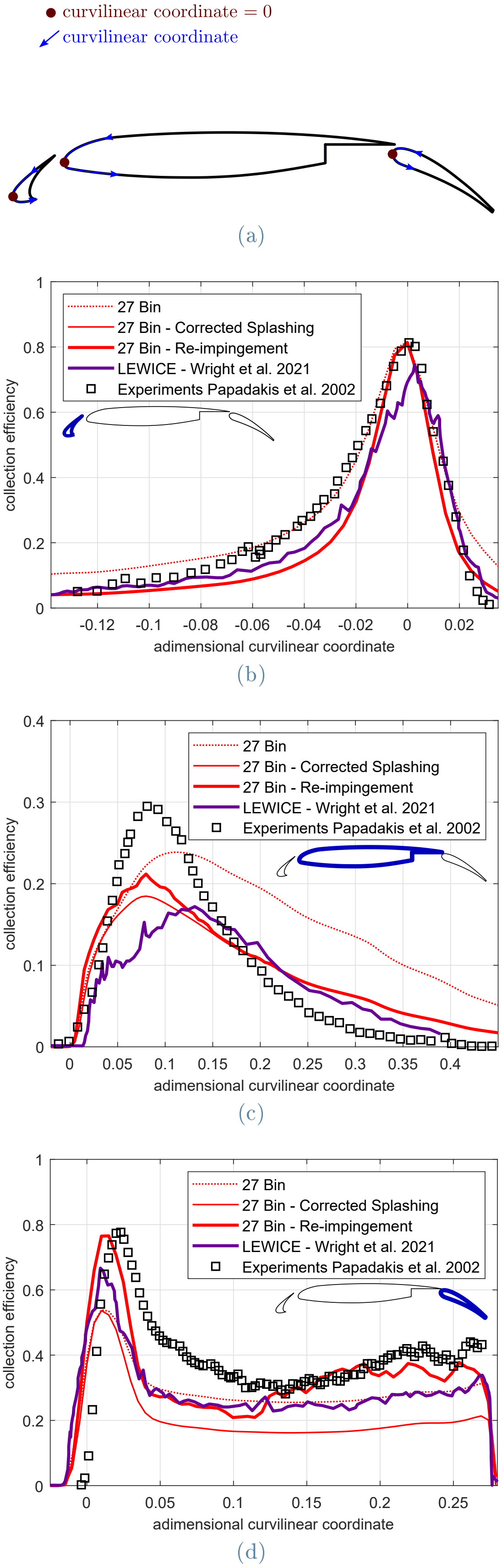
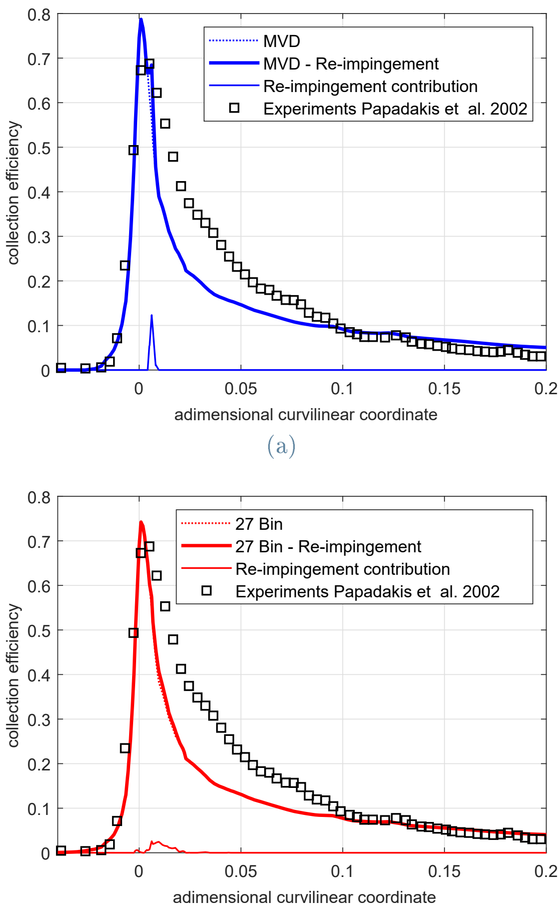


Figure 3: Collection efficiency for $MVD = 92\mu m$ using 27 bins - Three element airfoil [6]

of 4%) as expected given it's a single element geometry, but not zero due to the flow's three dimensionality. The multi-bin simulation in Fig. (4b) is in better agreement with the experiments compared to the MVD simulation in Fig. (4a), with a smoother re-impingement correction due to the heterogeneous nature of the splashed droplet set being re-injected when considering poly-dispersity against not considering it. The complete map of the collection efficiency for the 27bin simulation with re-impingement is reported in Fig. (5).



(a)

(b)

Figure 4: Collection efficiency at $z = 0.914m$ from wind tunnel floor for $MVD = 92\mu m$ - 3D horizontal tail [6]

The bin restart strategy applied to the 3D horizontal swept tail shows that it is possible to obtain a computational cost decrease of up to 65% in Fig. (6) since, starting from a better initial guess, the Euler-Euler solver is able to achieve convergence with a higher CFL number. It is best appreciated looking at the residual across bins and inner iterations in Fig. (7), where a 6 bin simulation ($CFL = 1$) with no bin restart takes more time to converge than a 27 bin simulation ($CFL = 2$) with bin restart.

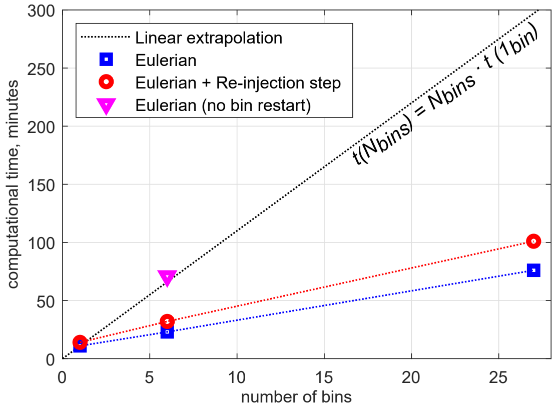


Figure 6: Computational time with and without bin restart - *i7 9750h 6c* @2.5GHz - 3D horizontal tail [6]

4. Conclusions

Considering both poly-dispersity and re-injecting droplets allows for much greater accuracy when computing the collection efficiency in multi-element configurations, with an up to 100% increase in local collection efficiency accuracy on the flap of a three element airfoil in SLD conditions. Employing a bin restart strategy allows for up to 65% less computational time in multi-bin Euler-Euler simulations. Given the computational cost increase reported by Bilodeau et al. [4] when using an Eulerian re-injection procedure, the Lagrangian re-injection proved to be the best compromise, with a cost increase in the most demanding 3D - 27 bin simulation of 32% when using bin restart and of 8% when not using bin restart. All of the results highlight the necessity of tracking re-injected droplets in complex 3D geometries (e.g. a complete aircraft). Given the modest computational cost increase of re-injecting droplets in a Lagrangian framework (compared to an Eulerian one), the solution procedure presented in this work seems to promise a substantial uplift in solution quality with a acceptable cost penalty.

References

[1] AIAA. *1st Icing Prediction Workshop*, 8 2021.

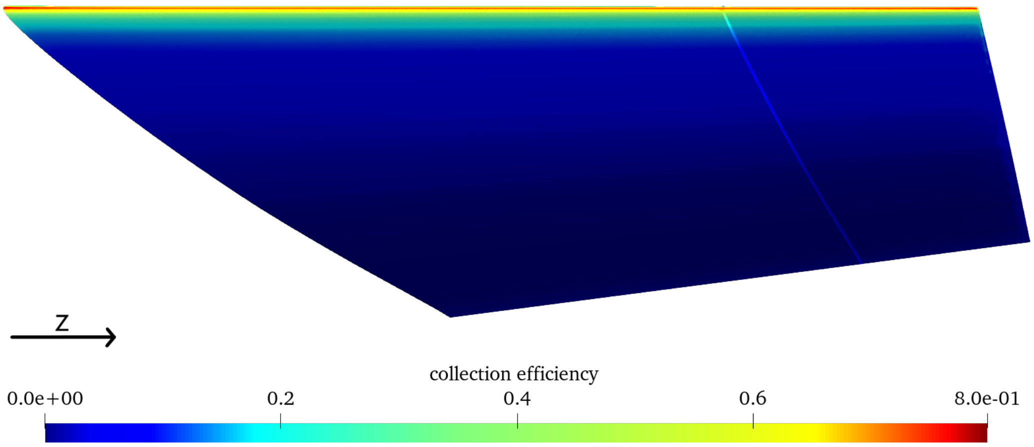


Figure 5: Collection efficiency map for $MVD = 92\mu m$ using 27 bins - 3D horizontal tail [6]

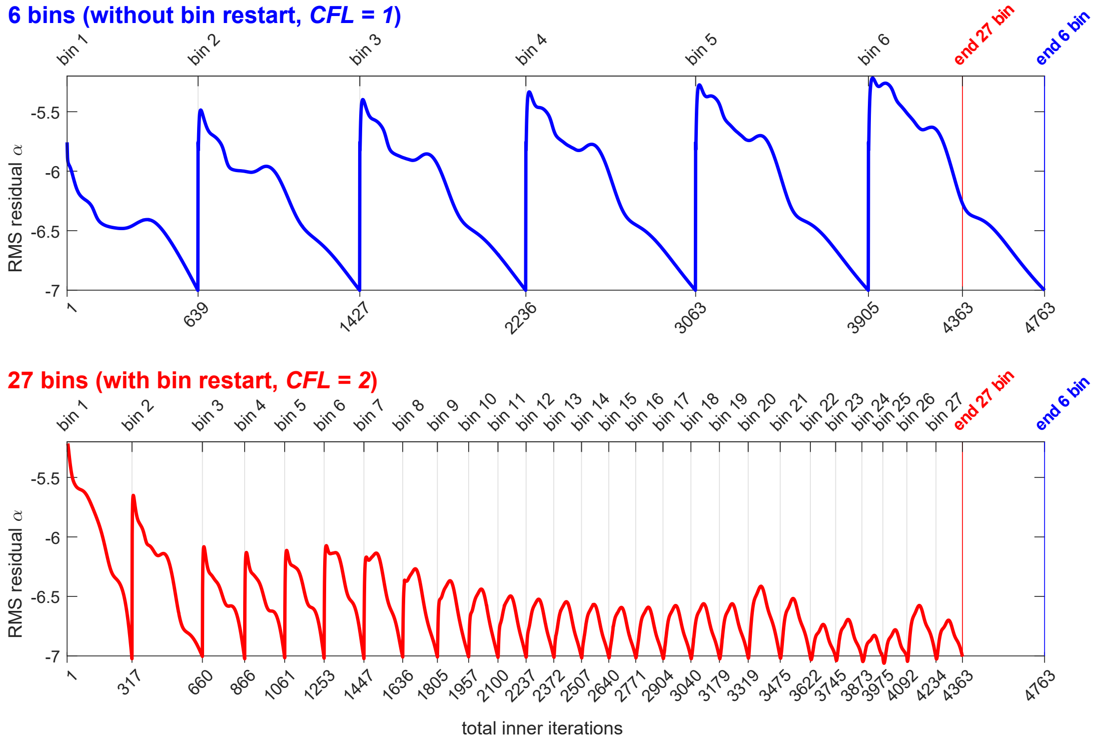


Figure 7: Volume fraction's RMS Residual with and without using bin restart - 3D horizontal tail [6]

[2] Tommaso Bellosta, Gianluca Parma, and Alberto Guardone. A robust 3d particle tracking solver for in-flight ice accretion using arbitrary precision arithmetic. pages 41,50. CIMNE, 2019.

[3] Christophe Berthon, Michael Breuß, and Marc-Olivier Titeux. A relaxation scheme for the approximation of the pressureless euler equations. *Numerical Methods for Partial Differential Equations*, 22(2):484–505, 2006.

[4] David Bilodeau, Wagdi Habashi, M. Fossati, and Guido Baruzzi. Eulerian modeling of supercooled large droplet splashing and bouncing. *Journal of Aircraft*, 52:1611–1624, 03 2015.

[5] Francisco Palacios, Michael Colonno, Aniket Aranake, Alejandro Campos, Sean Copeland, Thomas Economon, Amrita Lonkar, Trent Lukaczyk, Thomas Taylor, and Juan Alonso. Stanford university unstructured (su2): An open-source integrated computational environment for multi-physics simulation and design. 01 2013.

[6] Michael Papadakis, Kuohsing E. Hung, Giao Vu, Hsiung-Wei Yeong, Colin S. Bidwell, Martin D. Breer, and Timothy J. Bencic. Experimental investigation of water droplet impingement on airfoils, finite wings, and an s-duct engine inlet. 2002.

[7] Michael Papadakis, Arief Rachman, See-Cheuk Wong, Hsiung-Wei Yeong, Kuohsing E. Hung, and Colin S. Bidwell. Water impingement experiments on a naca 23012 airfoil with simulated glaze ice shapes. American Institute of Aeronautics and Astronautics, Inc., January 2004.

[8] William Wright. Further refinement of the lewice sld model. 01 2006.

POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Mixed Eulerian - Lagrangian Polydispersed 3D Droplet Tracking

Tesi di Laurea Magistrale in
Aeronautical Engineering - Ingegneria Aeronautica

Author: **Giuseppe Alexander Gaard Sirianni**

Student ID: 945184
Advisor: Prof. Alberto Matteo Attilio Guardone
Co-advisors: Eng, Tommaso Bellosta, Prof. Barbara Re
Academic Year: 2020-21

Abstract

In-flight icing is one of the major threats to air travel safety. It occurs when super cooled droplets freeze upon impact with the airplane's surfaces. To study this phenomena, droplets are tracked within the airflow in order to compute the amount of mass caught by the surfaces. Once this is done, ice accretion is computed. This thesis aims at providing a state of the art three dimensional Euler-Euler 2nd order finite volume tracking tool for polydispersed droplets, with the capability of also tracking splashing and rebounding droplets in a Lagrangian manner. The numerical tools and theory used to implement a 2nd order Godunov solver are presented. A multibin approach is described in order to simulate polydispersed droplets impinging on solid surfaces. A splashing model is presented and a Lagrangian reinjection and reimpigement step is added to track splashed droplets. All of the above are implemented within SU2 (an open source CFD solver) and PoliDrop (Politecnico di Milano's in-house Lagrangian particle tracker). The tools presented are then tested in 2D and 3D against experiments, Lagrangian simulations and industry leading softwares with great agreement. Furthermore, enabling the Lagrangian reimpingement step yields a substantial increase in solution quality for multi-element configurations, and a small increase for three dimensional ones in SLD conditions. Finally, a simple bin restart strategy allows for a computational cost decrease of up to 65%, decreasing the amount of iterations required to achieve convergence.

Keywords: in-flight ice accretion, eulerian droplet tracking, lagrangian re-injection, polydispersity

Sommario

La formazione di ghiaccio in volo è uno dei principali pericoli posti alla sicurezza del trasporto aereo. Accade quando gocce sottoraffreddate congelano all'impatto con le superifici del velivolo. Per studiare questo fenomeno le traiettorie delle gocce vengono calcolate in modo da ottenere la massa di acqua catturata dalle superfici. Successivamente, viene effettuato il calcolo dell'accrescimento di ghiaccio. Questa tesi ha l'obbiettivo di fornire un solutore Euler-Euler ai volumi finiti del 2o ordine per il tracciamento di gocce polidisperse, con la capacità di tracciare in modo Lagrangiano le gocce schizzate e rimbalzate sulle superfici. Verranno presentati gli strumenti teorici e numerici usati per implementare un solutore di Godunov del 2o ordine. Verrà descritto un approccio multibin per simulare gocce polidisperse che impattano sulle superfici. Un modello di splash verrà presentato insieme ad una strategia per reiniettare le particelle in modo Lagrangiano. Tutto questo verrà poi implementato in SU2 (un solver CFD open source) e PoliDrop (il solutore Lagrangiano in-house del Politecnico di Milano). I tool presentati verranno poi confrontati con esperimenti, simulazioni Lagrangiane e software leader dell'industria sia in 2D che 3D, con ottimo accordo. Usando la reiniezione Lagrangiana si ha inoltre un aumento significativo della qualità dei risultati nel caso di configurazioni multi elemento ed uno più modesto per casi tridimensionali in condizioni SLD. Infine, una semplice strategia di inizializzazione del multibin permette di ottenere fino a 65% di risparmio in tempo computazionale, diminuendo il numero di iterazioni necessarie ad ottenere la convergenza.

Parole chiave: accrescimento di ghiaccio, tracciamento gocce euleriano, reiniezione lagrangiana, polidispersità

Contents

1 | Introduction

1.1. In Flight Icing

One of the biggest threats to aircraft safety is posed by in-flight ice accretion. This phenomena occurs when flying through clouds at a low enough temperature, such that droplets impinging on the surfaces (particularly on stagnation points) freeze and ice starts accumulating.

If the ice forms on aerodynamic surfaces, over a sufficiently long time span the shape of these elements will inevitably be changed with severe consequences for aerodynamic performance.

Furthermore, if these ice formations break (naturally or because of the deployment of an Ice Protection System), depending on their position they could hit with considerable inertia the first stages of the engine compressor or aft surfaces (e.g. the tail empennage).

Therefore, understanding where and how ice accretion occurs is fundamental considering that safety is arguably the most important prerogative of air travel.

Because of a series of ideal conditions, clouds can contain so called Supercooled Large Droplets (SLD). These are, as the name suggests, large droplets of water that are still small enough to remain stable in liquid state well below 0°C. Any perturbation to their equilibrium (e.g. an airplane hitting them at 650 km/h) will cause them to instantly freeze, forming ice.

In-flight icing is known to occur with two main "types" of ice (1.1):

1. Rime, (very low temperature and cloud water content): air is trapped into the ice giving it its characteristic opaque white color and brittle texture
2. Glaze, (low temperature and higher cloud water content): water spreads on the

surface forming a film and subsequently freezes with a higher ice density and more transparent appearance

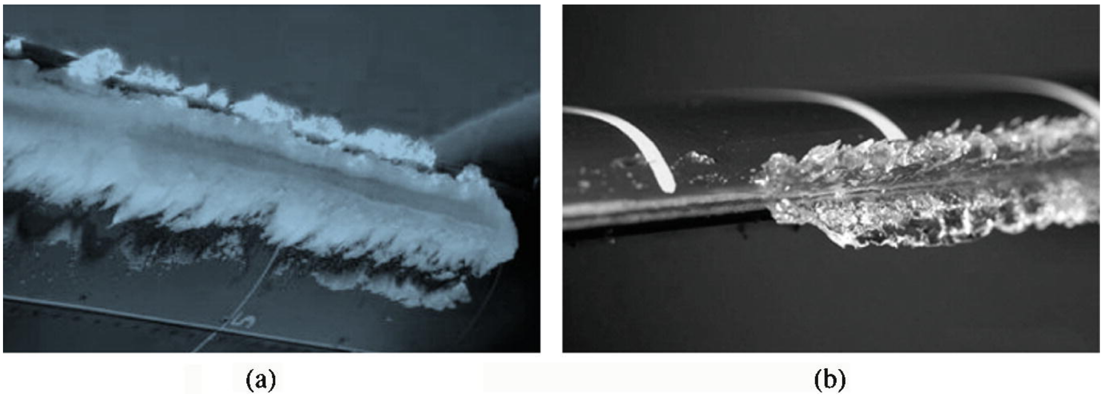


Figure 1.1: Rime ice (a) and Glaze ice (b) from [3]

1.1.1. Important Quantities

A few important notions for in-flight icing that will be used throughout this thesis are defined here. This is by no means a complete list of all concepts and variables used in in-flight icing.

Liquid Water Content (LWC)

Liquid Water Content (LWC) is the measure of how much mass of liquid water is present in a given volume of air. It's measured in $[g/m^3]$ and varies depending on the cloud type. It can go from 0.001 g/m^3 in Cirri to 3 g/m^3 in Cumulonimbi.

Median Volumetric Diameter (MVD)

The Median Volumetric Diameter (MVD) is a measure of how big the droplets in a cloud are (on average). It's an "experimental" property, in the sense that it's meant to be used to characterize a set of real droplets. It's defined as the droplet diameter such that, given a volume of water $V_{droplets}$ in an air-droplets sample, half the volume of water will be composed of droplets with $d_p > MVD$ and the other half with $d_p < MVD$.

Volume Fraction (α)

The Volume Fraction (α) is defined as the ratio of volume occupied by the dispersed phase to the volume occupied by the whole mixture (in this case $volume_{droplets}/volume_{air+droplets}$). This definition is only valid if the volume of measurement considered is big enough to mantain a stationary average.

Collection Efficiency (β)

The Collection Efficiency (β) is defined as the fraction of water that's caught by the surface with respect to the concentration of water in the freestream. It can go from 0 (no water is caught by the surface) to 1 (all the incoming water is caught by the surface).

1.1.2. Current State of the Art

Given the developmental nature of this work, building upon past theses and work done in the PoliMIce research group, this section (1.1.2) is a recap of the introduction presented by Bellosta in his thesis [6]. For a more thorough and general discussion the reader is refferred to his work.

Since the goal of this thesis is to implement, improve and validate an Eulerian alternative to the Lagrangian droplet tracking step (presented in [6]) that is being currently used in the PoliMIce ice accretion software, an overview on the state of the art theory, assumptions and tools will be provided in this section.

The consensus in the research and industrial landscape is to classify the droplet field around the aircraft as a *particle laden flow*. This is a class of flows where a dispersed phase composed of particles of any nature (in this case droplets) is present inside a so called carrier phase (in this case air).

The carrier phase is assumed to be continuum. This can be translated in a condition (1.1) on the particle Knudsen number Kn_p. The condition simply states that the mean free path of the carrier's molecules is much smaller the than the dispersed particle's diameter d_p.

$$Kn_p = \frac{\mu_f}{d_p c_f \rho_f} << 1 \tag{1.1}$$

For sea level P and T (and therefore speed of sound c_f, viscosity μ_f and density ρ_f) this condition is satisfied since $Kn_p \simeq 10^{-3}$.

Another important thing to consider is the coupling between carrier and dispersed phase. Given a set of mass, momentum and energy conservation laws, coupling between these two phases can occur in one way or both, through any of the aforementioned balance equations.

This can occur through phase change (e.g. water droplets evaporating), force exchanges (e.g. aerodynamic forces exchanged between a droplet and the air) or heat transfer (e.g. cold fuel droplets in a hot combustion chamber).

In line with the work from industrial and research entities, the following assumptions regarding phase coupling have been made when developing this thesis in the context of in-flight icing:

1. water content of a cloud (at most $\sim 3\ g/m^3$) is assumed low, such that momentum is one way coupled (air → droplets) only through the drag force exerted by the difference in velocity $\underline{\boldsymbol{U}} - \underline{\boldsymbol{u}}_p$ between airflow and droplets
2. water content of a cloud (at most $\sim 3\ g/m^3$) is assumed low, such that the backwards heat transfer (air ← droplets) can be considered negligible
3. droplets are assumed to be at thermal equilibrium with the air they are in at the freestream, and also after the aircraft has flown through them. This is because the speed at which the aircraft flies is high enough for the thermal changes to be negligible (unless droplets impinge in which case they are stuck on the surface)

The Stokes number of a particle can be defined $St = t_p/\tau_{flow}$ as the ratio between the relaxation time of a particle and a characteristic flow time, with the relaxation time for Stokesian particles $t_p = \frac{\rho_p d_p^2}{18\mu_f}$ being defined as the response time to a step change in the carrier flow velocity $\underline{\boldsymbol{U}}$.

If $St >> 1$ the particles take a long time to adapt in changes in their surrounding flow while if $St << 1$ particles follow every minute change of the smallest flow scale.

The one way momentum coupling assumption that has just been introduced could be disturbed if $St << 1$ and the smallest scales of turbulence (Kolmogorov scales) where considered in it's complete description (e.g. a Direct Numerical Simulation). This would mean particles could follow every variation in even the smallest scales. Fortunately Elgobashi [11] found that for heavy particles (such as water droplets in air) these effects are negligible.

This is true in aeronautical applications, unless the behavior of small droplets inside the boundary layer or wake is of particular interest.

Having arrived at the conclusion that simulating droplets in air for in-flight icing purposes consists in considering a one way coupled (air → droplets) particle laden flow, now the two main simulation frameworks being used in the industrial and research landscape will be highligthed.

Lagrangian Description

The most direct way of simulating droplets is to track every and each one of them singularly. This is what is done in the Euler-Lagrange framework (Eulerian flow, Lagrangian particles) of which PoliDrop [6] is an example. PoliDrop will be one of the two main software tools (together with SU2) used to develop this thesis.

In the Lagrangian framework, particles are not considered as a continuum and the solver only needs to solve the momentum conservation (1.2).

$$m_p \frac{d\underline{\boldsymbol{u}}_p}{dt} = \underline{\boldsymbol{F}}_p \tag{1.2}$$

The forcing term $\underline{\boldsymbol{F}}_p$ is comprised of different contributions:

1. Gravity: $\underline{\boldsymbol{F}}_p^G = -\rho_f \underline{\boldsymbol{g}}\, Vol_p$
2. Drag: $\underline{\boldsymbol{F}}_p^D = \frac{1}{2}\rho_f \pi \left(\frac{d_p}{2}\right)^2 C_D(Re_p) \mid \underline{\boldsymbol{U}} - \underline{\boldsymbol{u}}_p \mid \left(\underline{\boldsymbol{U}} - \underline{\boldsymbol{u}}_p\right)$
3. Neglected contributions from *pressure gradient*, *virtual mass*, *stress gradient*, *lift* and *basset history* forcing terms. These are assumed to be small because of the physics at play in in-flight icing conditions, see [6]

Equation (1.2), combined with the trivial velocity-position differential relation creates the system of ODEs (1.3).

$$\begin{cases} \dfrac{d\underline{\boldsymbol{u}}_p}{dt} = \underline{\boldsymbol{F}}_p\left(\underline{\boldsymbol{x}}_p, \underline{\boldsymbol{u}}_p, y\right) \\ \dfrac{d\underline{\boldsymbol{x}}_p}{dt} = \underline{\boldsymbol{u}}_p \end{cases} \tag{1.3}$$

This can be integrated in time to obtain the particles' trajectories.

In the Lagrangian framework, the collection efficiency is computed as the ratio of the curvilinear distance between two (or more in 3D) impinging droplets and the distance between them at the freestream (Figure (1.2)). Assuming the cloud has equally spaced droplets of equal diameter (or alternatively constant LWC in the cloud) at the freestream a simplification can be done, and on the $i-th$ surface element the collection efficiency becomes $\beta_i = \dfrac{m_i}{A_i LWC_\infty}$.

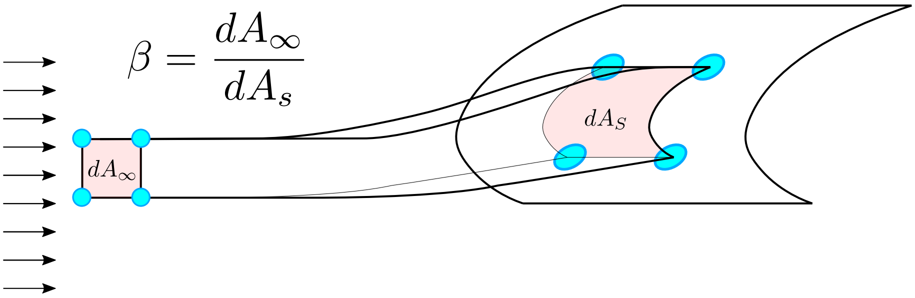


Figure 1.2: Collection efficiency geometrical definition in 3D

Eulerian Description

This thesis develops and expands (with a Lagrangian reinjection step) a new Euler-Euler solver based on the pressureless gas dynamics equations (2.2) in order to compute droplet trajectories in steady conditions. This solver is meant to be an alternative to PoliDrop in the PoliMIce in-flight icing software.

Droplets are now assumed as a continuum, therefore the LWC cannot be too small or this assumption would fail when taking the limits of variables in space to define local quantities (e.g. the volume fraction α).

Averages of the dispersed phase's properties are taken in mesh elements (therefore in space volume).

The size L of the averaging sample volume must be much bigger than the average distance between particles for this average to become stationary. But, at the same time, L must be much smaller than the characteristic length of the case at hand (e.g. the chord of an airfoil) to obtain a smooth enough solution in space.

Varaksin [26] found an expression for the ratio of the sampling volume's cube side L to particle diameter d_p as a function of the volume fraction α. Fixing a number of particles $N_p = 100$ one can plot the function in Figure (1.3).

$$\frac{L}{d_p} = \sqrt[3]{\frac{N_p}{6}\frac{N_p}{\alpha}} \tag{1.4}$$

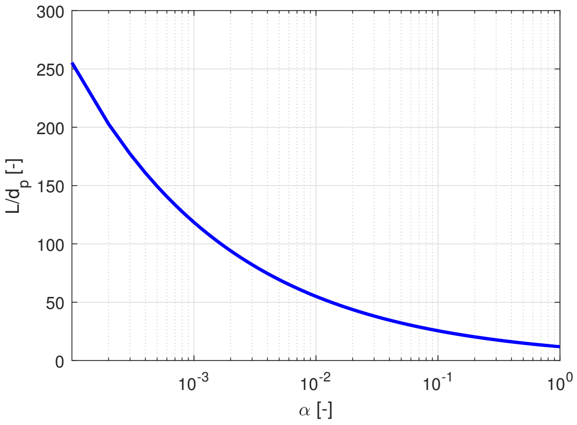


Figure 1.3: Varaksin [26] dependency of volume fraction to particle diameter with $N_P = 100$

This shows how having a higher value of the volume fraction α allows for the possibility of taking the limit in space ($L \to 0$) and therefore leads to being able to define the local variables needed to create an Eulerian description of the dispersed phase without using elements that are too big for the solution to be smooth in space.

Considering negligible mass and energy transfer in in-flight icing conditions, as discussed in (1.1.2), the dispersed phase can be described as a second fluid, completely mixed into the carrier phase but not in equilibrium with it. This is called the *separated fluid* approach and it allows for the evaluation of the droplet behavior, since the main forcing contribution acting on droplets is the drag force generated by the slip velocity $\underline{\boldsymbol{U}} - \underline{\boldsymbol{u}}_p$.

For a comprehensive derivation and description of Eulerian conservation laws for many different particle laden flows the reader is highly encouraged to read [10].

The collection efficiency in the Eulerian framework is much easier to compute, as it's simply found with (1.5) so the knowledge of the distance between droplets at the freestream is not required.

$$\beta = \alpha \underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}} \tag{1.5}$$

1.1.3. PoliMIce

PoliMIce is an ice accretion simulation framework developed at the department of Aerospace Science and Technology of Politecnico di Milano. It takes advantage of open source CFD tools (such as SU2) to compute the air flow and uses this data, combined with in house software (such as PoliDrop) to evaluate ice accretion in a multi step fashion.

The solution procedure is as follows:

1. the clean mesh is generated
2. the flow field is computed in a CFD solver
3. the droplet trajectories are computed with PoliDrop in a Lagrangian approach
4. the ice accretion parameters are computed
5. ice accretion is computed and the mesh in morphed to take this into account
6. repeat from step 2.

PoliMIce can simulate both rime and glaze ice accretion by solving [14] a modified Messinger model for aircraft icing [22].

Ice accumulates perpendicularly to the surface and the solution is iterated over a set exposure time by morphing the mesh to account for the variation in geometry caused by the ice (hence the *multi step* approach).

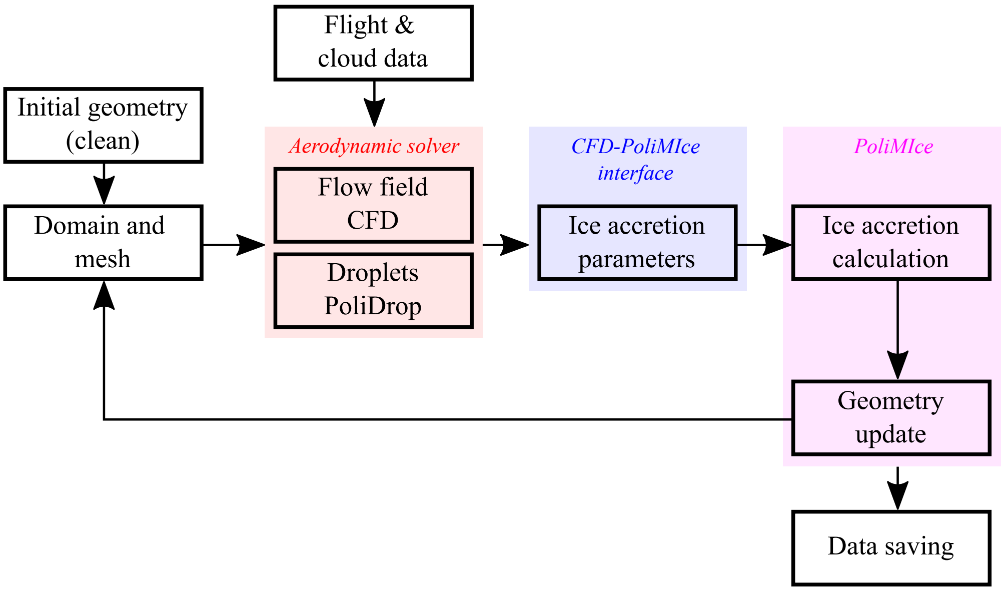


Figure 1.4: PoliMIce ice accretion multi-step computational procedure

The data required by PoliMIce from the droplet trajectories to compute the ice accretion step is simply the collection efficiency on the surfaces.

This is where the work presented throughout this thesis will be concentrated on: providing the fastest and most accurate way of computing the collection efficiency β on the aerodynamic surfaces using an Euler-Euler approach, while also adding a Lagrangian reinjection step to account for splashed & rebound droplets.

1.2. Thesis Goal & Structure

The goal of this thesis is to provide a complete, state of the art Euler-Euler particle tracking tool for in-flight icing purposes with the following features:

1. three dimensional, unstructured mesh
2. 2nd order accuracy
3. poly-dispersity through automatic or user defined droplet size distribution (for collection efficiency computation)
4. Lagrangian tracking of splashed droplets

The overall structure of the contents is, as follows:

1. the theoretical tools required to implement an Euler-Euler finite volume exact

Riemann solver with MUSCL in SU2 will be provided (2.1)

2. the equations used to model the dispersed phase will be analyzed and a relaxation model will be added in an effort to obtain a strictly hyperbolic system of conservation laws (2.2)
3. an automatic multibin implementation will be outlined for test cases where bin data is not provided or available (2.3)
4. a state of the art splashing and rebound model will be presented together with a strategy to simulate the splashed droplets in a separate Lagrangian framework (2.4)

After being theoretically outlined, all of the above features will be implemented within SU2 and PoliDrop and the resulting tool-set will be validated against experimental collection efficiency data (3) for both 2D and 3D cases.

An additional Appendix (F) on a completely new Baer-Nunziato [4] multi-dimensional unstructured solver called PoliPhase is added with some preliminary results. The work on this solver has been started after the core of this thesis had been completed, and is just a starting reference for future work.

2 | Theory Overview

2.1. Numerical Tools & Approach

SU2 has been chosen as the environment to develop and implement the Euler-Euler droplet tracking (from now on reffered to as Eulerian) given it is open source and it has easily expandable code.

A small note on notation:

Primitive variables	$\underline{\boldsymbol{p}} = \left[\alpha, \underline{\boldsymbol{u}}_p, \pi\right]$
Conservative variables	$\underline{\boldsymbol{q}} = \left[\alpha, \alpha\underline{\boldsymbol{u}}_p, \alpha\pi\right]$
Cell averages (overline)	$\overline{\underline{\boldsymbol{p}}}$, $\overline{\underline{\boldsymbol{q}}}$
Finite Volume elements and their shared interface	C_i , C_j , $\partial C_{i,j}$
Volume and surface integration elements	$d\varOmega$, $d\varSigma$
Eigenvalues and eigenvectors	λ_i , $\underline{\boldsymbol{r}}$
Fluid velocity and droplet velocity	$\underline{\boldsymbol{U}} = [U, V, W]$, $\underline{\boldsymbol{u}}_p = \left[u_p, v_p, w_p\right]$

2.1.1. CFD Solution Procedure

CFD simulations are performed using the Finite Volume method, with PDEs being cast in differential form as in (2.1).

$$\frac{\partial \underline{\boldsymbol{q}}}{\partial t} + \overline{\nabla} \cdot \underline{\boldsymbol{F}}_{conv}(\underline{\boldsymbol{q}}) - \overline{\nabla} \cdot \underline{\boldsymbol{F}}_{visc}(\underline{\boldsymbol{q}}) = \underline{\boldsymbol{S}}(\underline{\boldsymbol{q}}) \tag{2.1}$$

Where $\underline{\boldsymbol{q}}$ is the vector of conservative variables, $\underline{\boldsymbol{F}}_{conv}(\underline{\boldsymbol{q}})$ and $\underline{\boldsymbol{F}}_{visc}(\underline{\boldsymbol{q}})$ the vectors of convective and viscous fluxes and $\underline{\boldsymbol{S}}(\underline{\boldsymbol{q}})$ the vector of source terms.

This is then solved in terms of the residual $\underline{\boldsymbol{R}}(\underline{\boldsymbol{q}})$ as written in (2.2).

$$\underline{\boldsymbol{R}}(\underline{\boldsymbol{q}}) = \frac{\partial}{\partial t}\int_{\Omega} \underline{\boldsymbol{q}} d\Omega + \oint_{\partial\Omega} \left[\underline{\boldsymbol{F}}_{conv}(\underline{\boldsymbol{q}}) - \underline{\boldsymbol{F}}_{visc}(\underline{\boldsymbol{q}})\right] \cdot \hat{\boldsymbol{n}} d\Sigma - \int_{\Omega} \underline{\boldsymbol{S}}(\underline{\boldsymbol{q}}) d\Omega = \underline{\boldsymbol{0}} \tag{2.2}$$

If an implicit scheme is being used, the solution is then obtained through the Newton-Raphson method (2.3) until the convergence criteria set on the root mean square of the residual $\underline{\boldsymbol{R}}(\underline{\boldsymbol{q}})$ has been achieved.

To this end the computation of the jacobian $\underline{\underline{\boldsymbol{J}}}_{NR} = \dfrac{\partial R_i}{\partial q_j}$ is required.

$$\begin{cases} \underline{\boldsymbol{q}}_{k+1} = \underline{\boldsymbol{q}}_k + \Delta \underline{\boldsymbol{q}}_k \\ \\ \underline{\underline{\boldsymbol{J}}}_{NR} \Delta \underline{\boldsymbol{q}}_k = -\underline{\boldsymbol{R}}(\underline{\boldsymbol{q}}_k) \end{cases} \tag{2.3}$$

This same approach, together with the many tools used by the already existing solvers, will be employed in this work to implement a pressureless gas dynamics (2.2) solver within SU2.

2.1.2. Finite Volume Method

The equations presented in (2.1.1) still need to be spatially discretized and this is where the finite volume method comes into play.

The domain is subdivided into mesh elements, namely cells and nodes. Then, the discrete solution can be stored in two ways:

1. Nodes (node centered FVM)
2. Cells (cell centered FVM)

SU2 uses the first approach, storing the solution in the nodes of the mesh and constructing the finite volumes around them as shown in Figure (2.1)

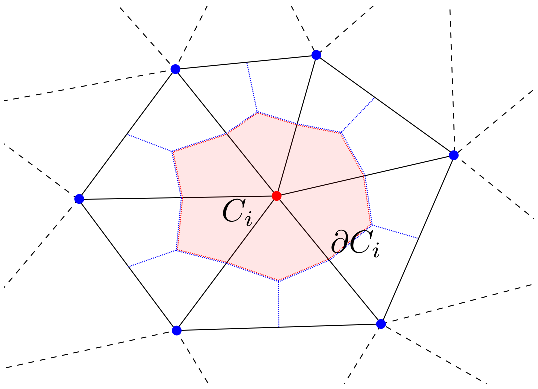


Figure 2.1: Node centered 2D finite volume (in red) sketch

The problem is solved for the average value within these control volumes. The fluxes are evaluated at the interfaces of all neighbouring finite volumes.

Equation (2.2) therefore becomes, in it's discretized form, equation (2.4).

$$\underline{\boldsymbol{R}}_i(t) = \mid C_i \mid \cdot \frac{\overline{\underline{\boldsymbol{q}}}_i^{n+1} - \overline{\underline{\boldsymbol{q}}}_i^{n}}{\Delta t} + \sum_{j \in \partial C_i} \underline{\boldsymbol{F}}_{i,j} \cdot \underline{\boldsymbol{\nu}}_{i,j} + \underline{\boldsymbol{F}}(\underline{\boldsymbol{q}}_{BC}) \cdot \underline{\boldsymbol{\nu}}_{i,j} - \int_{C_i} \underline{\boldsymbol{S}}(\bar{t}) d\Omega = \underline{\boldsymbol{0}} \tag{2.4}$$

Where:

$$\begin{cases} \underline{\boldsymbol{\nu}}_{i,j} = \displaystyle\int_{\partial C_{i,j}} \hat{\boldsymbol{n}}_i d\Sigma \\ \\ \overline{\underline{\boldsymbol{q}}}_i(\bar{t}) = \displaystyle\int_{C_i} \underline{\boldsymbol{q}}(\bar{t}) d\Omega \\ \\ \mid C_i \mid = \displaystyle\int_{C_i} 1 \cdot d\Omega \end{cases} \tag{2.5}$$

2.1.3. Godunov Method

To obtain a result, the discretized equation (2.4) needs to be evaluate control volume by control volume to construct the jacobian and residual and solve the Newton-Raphson iteration (2.3).

There are multiple possibilities when deciding how to evaluate the fluxes $\underline{\boldsymbol{F}}_{i,j}$.

Having assumed that the values inside a finite volume are constant, what happens at interfaces between elements is of great importance.

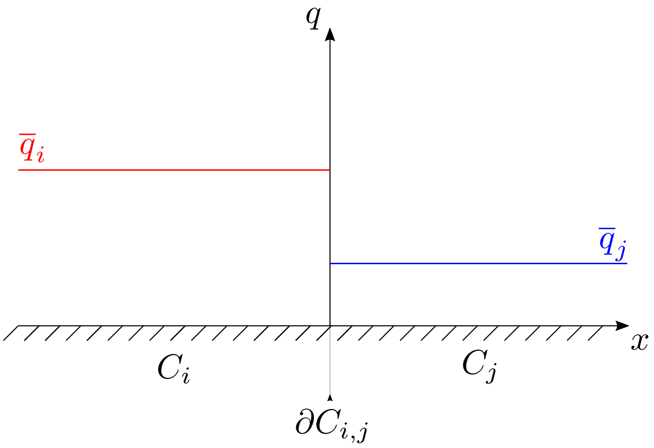


Figure 2.2: 1D Riemann problem sketch at node interfaces

Unfortunately $\underline{\boldsymbol{q}}(\bar{t})$ is undefined at the interface $\partial C_{i,j}$ between the elements C_i and C_j. In the case of hyperbolic conservation laws (eigenvalues $\lambda_i \in \mathbb{R} \quad \forall i$) with piecewise constant initial data (such as this case as depicted in Figure (2.2)) this problem is called the Riemann Problem.

If a simple analytic solution $\underline{\tilde{\boldsymbol{q}}}(\bar{t} + dt)$ of this problem is available at a time $\bar{t} + dt$ at the interface $\partial C_{i,j}$, the fluxes can be evaluated using this value. This is the way fluxes are evaluated in Riemann solvers.

If the exact Riemann problem is solved exactly, then the method is called a *Exact Riemann solver.*

If, to reduce computational cost, the Riemann problem is approximated and then solved exactly, the method is called an *Approximate Riemann solver.*

In this work the equations modeling the droplets will be solved by implementing a *exact Riemann Solver*, therefore an exact solution of the Riemann problem will be required.

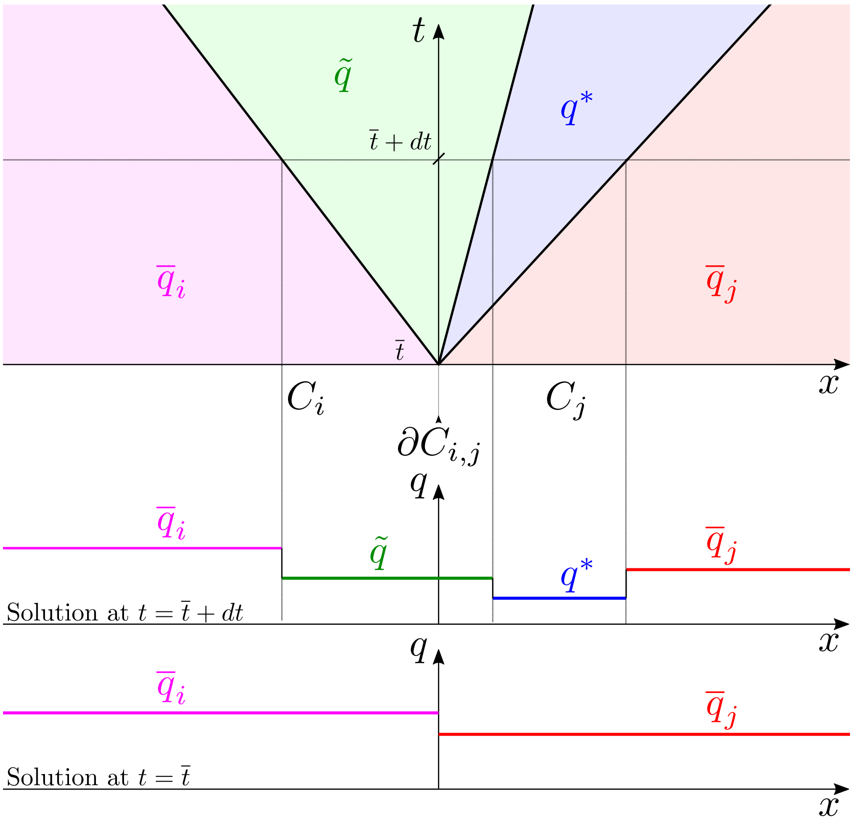


Figure 2.3: 1D Riemann problem solution sketch at interfaces

Figure (2.3) shows an example of a Riemann problem solution. It is composed of three waves, across which $\underline{\boldsymbol{q}}$ changes. These waves travel in time, but the solution in each "slice" of the domain remains constant unless a wave passes through it at a certain time. This means that at $t=\bar{t}+dt$ the value of $\underline{\tilde{\boldsymbol{q}}}$ can be easily found as long as the "slice" in which the interface $\partial C_{i,j}$ lies is known, and the value of $\underline{\boldsymbol{q}}$ is known inside each slice.

MUSCL Reconstruction

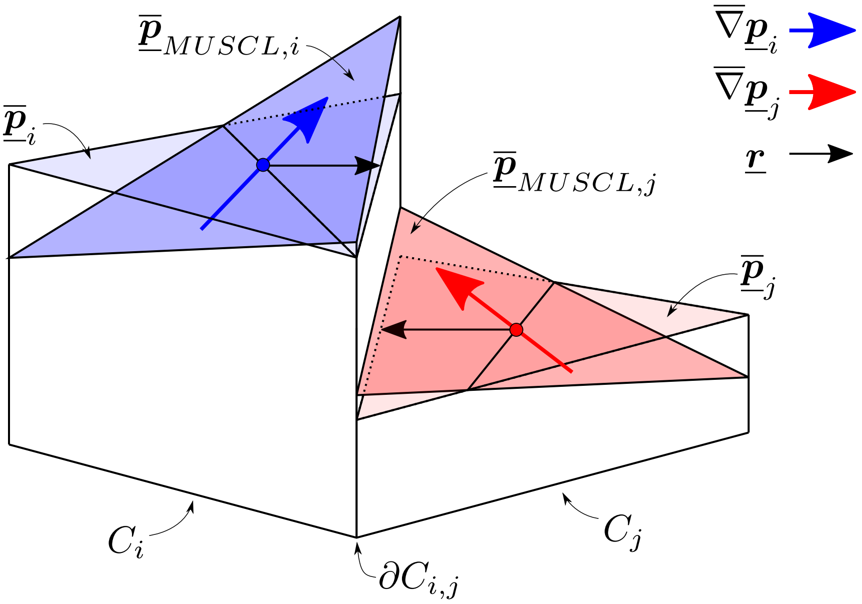

$\underline{\overline{\boldsymbol{p}}}_{MUSCL,i}$
$\underline{\overline{\boldsymbol{p}}}_i$
$\overline{\nabla}\underline{\boldsymbol{p}}_i$
$\overline{\nabla}\underline{\boldsymbol{p}}_j$
$\underline{\boldsymbol{r}}$
$\underline{\overline{\boldsymbol{p}}}_{MUSCL,j}$
$\underline{\overline{\boldsymbol{p}}}_j$
C_i
C_j
$\partial C_{i,j}$

Figure 2.4: 2D MUSCL reconstruction sketch. $\underline{\overline{\boldsymbol{p}}}_{i,j}$ are the cell averages of the primitive variables while $\underline{\overline{\boldsymbol{p}}}_{MUSCL,i,j}$ are the linear approximations obtained using the cell gradients $\overline{\nabla}\underline{\boldsymbol{p}}_{i,j}$

A MUSCL reconstruction has been implemented to (ideally) obtain a second order method (i.e. the error goes down as $e \propto K \cdot h^2$ with h being the mesh size) and therefore be able to obtain mesh convergence with a significantly smaller mesh (and the obvious computational cost decrease associated with it). Actual second order spatial convergence cannot be exactly achieved with complex geometries (such as the ones found in the in-flight icing world), but a faster mesh convergence can be achieved nonetheless.

The primitive variable $\underline{\boldsymbol{p}}$ are reconstructed at control volume interfaces using their gradients inside the cells as (2.6).

$$\underline{\overline{\boldsymbol{p}}}_{MUSCL} = \underline{\overline{\boldsymbol{p}}} + \overline{\nabla}\underline{\boldsymbol{p}} \cdot \underline{\boldsymbol{r}} \tag{2.6}$$

$\underline{\boldsymbol{r}}$ is evaluated at the cell interface for both cells C_i and C_j obtaining the left and right interpolated interface values that will then be used to construct the Riemann problem that needs to be solved in the exact Riemann solver.

The gradient is computed within SU2, using the already available tools, with the possibility to choose between *Green-Gauss* or *Weighted-Least-Squares* techniques.

Multidimensional Approach

The Eulerian solver developed in this work must work in multidimensional unstructured grids. This means the exact Riemann solver must solve the multidimensional Riemann problem on interfaces between elements.

It is important to note that the only equation and variables that change when moving to multidimensional problem are *momentum conservation* and *velocity.* It can also be proven that the eigenvalue given by the normal velocity $\underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}}$ has multiplicity equal to the number of dimension (akin to what happens in the Euler equations for inviscid flows). This is at the core of the multidimensional approach used in this work:

1. the 1D exact Riemann solver is outlined by solving the Riemann problem analytically in Appendix (A)
2. the multidimensional problem is solved by projecting the velocity on the interface normal direction, and then solving the resulting 1D problem. The solution is then casted back into multidimensional form using the normal vector $\hat{\boldsymbol{n}}$

2.2. Pressureless Gas Dynamics

Droplet trajectories can be computed using a Lagrangian approach, therefore tracking each and every parcel by integrating in time (1.3) as shown in (1.1.2). This approach is preferred when the LWC is low, both in terms of computational power and fidelity in representing the physics of the problem.

When the LWC is higher, the Euler-Euler approach becomes a viable (if not preferrable) alternative. Droplets are not longer tracked one by one but are treated as a continuum. This allows for the exploitation of many decades of acquired knowledge in the field of Finite Volume methods for fluid simulations.

The Pressureless Gas Dynamics conservation laws (2.2.1) are used to model the one way coupled particle laden flow. This choice can be easily explained considering that droplet-droplet collisions can be neglected given their dispersed nature.

Removing pressure from the Euler equations allows for the description of shadow zones (of great importance in particle laden flows) but also introduces a sizable set of drawbacks which will be discussed in (2.2.2).

To remove some of the possible numerical issues, the equations are solved in non dimensional form, since otherwise $o\left(\alpha\right) << o\left(\mid \underline{\boldsymbol{u}}_p \mid\right)$.

2.2.1. Conservation Laws

The Pressureless Gas Dyamics conservation laws upon which this work is based are (2.7) from [18].

$$\begin{cases} \dfrac{\partial \alpha}{\partial t} & + \overline{\nabla} \cdot \left(\alpha \underline{\boldsymbol{u}}_p\right) = 0 \\ \dfrac{\partial \alpha \underline{\boldsymbol{u}}_p}{\partial t} & + \overline{\nabla} \cdot \left(\alpha \underline{\boldsymbol{u}}_p \otimes \underline{\boldsymbol{u}}_p\right) = \dfrac{\alpha C_D Re_d}{24K}\left(\underline{\boldsymbol{U}} - \underline{\boldsymbol{u}}_p\right) + \cancel{\dfrac{\alpha}{Fr^2}\left(1 - \dfrac{\rho_f}{\rho_p}\right)\underline{\boldsymbol{g}}} \end{cases} \tag{2.7}$$

1. The gravitational momentum source term is neglected.
2. K is the inertia parameter, $K = \dfrac{\rho_p d_p^2 U_\infty}{18 \mu_f} = t_p U_\infty$ where t_p is the particle relaxation time. If t_p is small particle trajectories mostly follow the flow's streamlines, if it's large their inertia is too large and they behave in a "ballistic" manner.
3. C_D is the droplet's drag coefficient.

As it will be discussed in (2.2.4), the Riemann problem's solution will be constructed using the assumption of equilibrium on left and right states (therefore inside mesh elements). This will translate in the removal of the additional pressure equation in (2.7).

The only source of (one way) coupling between flow and particles is the drag momentum source term $\underline{\boldsymbol{S}}_D = \dfrac{\alpha C_D Re_d}{24K}\left(\underline{\boldsymbol{U}} - \underline{\boldsymbol{u}}_p\right)$. The drag coefficient is modeled by first assuming it to be a sphere, using Morrison's approximation [21] and Clift's et. al. approximation [9] as (2.8).

$$C_D^{sphere}\left(Re_d\right) = \begin{cases} C_{D,Morrison}^{sphere} & Re_d < 10^6 \\ C_{D,Clift\ et.\ al.}^{sphere} & Re_d > 10^6 \end{cases} \tag{2.8}$$

Where the two drag coefficients are (2.9).

$$\begin{aligned} C_{D,Morrison}^{sphere} &= \frac{24}{Re_d} + 2.6\frac{\frac{Re_d}{5}}{1+\left(\frac{Re_d}{5}\right)^{1.52}} + 0.411\frac{\left(\frac{Re_d}{2.63\cdot10^5}\right)^{-7.94}}{1+\left(\frac{Re_d}{2.63\cdot10^5}\right)^{-8}} + \frac{Re_d^{0.8}}{4.61\cdot10^5} \\ C_{D,Clift\ et.\ al.}^{sphere} &= 0.19 - \frac{80000}{Re_d} + \delta \end{aligned} \tag{2.9}$$

To be able to account for the eccentricity brought in by the deformation in the Supercool Large Droplet (SLD) regime, the Weber breakup number [16] is introduced in (2.10).

$$We_b = d_p \mid \underline{\boldsymbol{u}}_p \mid^2 \frac{\rho_f}{\sigma_p} \tag{2.10}$$

This is then used to compute the eccentricity f as [16] (2.11).

$$f = \left[\frac{1}{1+0.07\sqrt{We_b}}\right]^6 \tag{2.11}$$

The eccentricity is finally used to linearly interpolate the C_D between the sphere's drag coefficient C_D^{sphere} and the disk's drag coefficient C_D^{disk} as in (2.12).

$$C_D\left(Re_d\right) = \begin{cases} (1-f)C_D^{sphere} + fC_D^{disk} & We_b \leq 12 \\ C_D^{disk} & We_b > 12 \end{cases} \tag{2.12}$$

Where the disk's drag coefficient is taken from Clift et. al. [9] in (2.13). After $We_b > 12$ the droplet should breakup, but in this work this will not be considered, hence the assumption of $C_D = C_D^{disk}$ beyond $We_b > 12$.

$$C_D^{disk} = \begin{cases} \frac{64}{\pi Re_d}\left(1+\frac{Re_d}{2\pi}\right) & Re_d \leq 0.01 \\ \frac{64}{\pi Re_d}\left(1+10^{-0.883+0.906\log_{10}(Re_d)-0.025\left(\log_{10}(Re)^2\right)}\right) & Re_d \in (0.01, 1.5] \\ \frac{64}{2\pi}\left(1+0.138Re_d^{0.792}\right) & Re_d \in (1.5, 133] \\ 1.17 & Re_d > 133 \end{cases} \tag{2.13}$$

A Note on δ Shocks & Vacuum States

The pressureless gas dynamics equations have been studied extensively in fields such as *relativistic Euler equations* and astrophysics modeling of large scale universal structure [2].

It is well documented that the Riemann solution can contain δ shocks and vacuum states [29]:

1. δ shocks are discontinuities with a overimposed Dirac's delta on α. They are therefore singular solutions with little physical meaning, but they are important to construct a solution of the PGD's Riemann problem
2. vacuum states are solutions with $\alpha = 0$, therefore in this work this state corresponds to the absence of droplets

Capturing vacuum states is very important in simulating droplets since it is evident how a body flying through a cloud does indeed create a *shadow zone* where there are no droplets. The vacuum will not be retrieved exactly since a relaxation model is employed (2.2.2) for numerical purposes.

If no characteristics (characteristics are parallel to $\underline{\boldsymbol{u}}_p$) enter the airfoil in what should have been a shadow zone ($\alpha = 0$) then the solution will be considered satisfying for in-flight icing purposes since no non-physical collection efficiency will be computed.

On the other hand, δ shocks bear no physical meaning in the context of droplet impingement. As the pseudo pressure tends to zero, the three waves arising from the initial discontinuity tend to coalesce into a single δ shock [29].

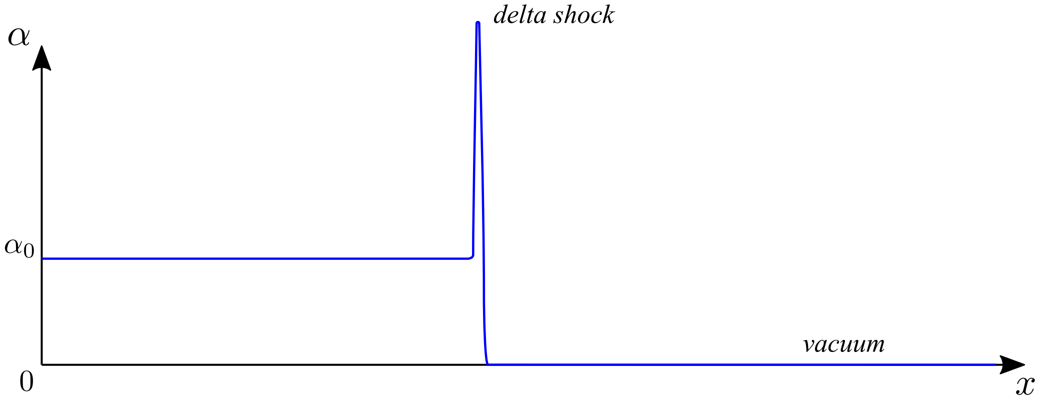


Figure 2.5: δ shock and vacuum zone sketch

2.2.2. Relaxation

Having removed pressure from the Euler equations in (2.7) is a major source of numerical troubles, one of which is that the problem is no longer strictly hyperbolic.

To this end a numerical relaxation method proposed and developed by Berthon et. al. [7] is employed. The relaxation model will be thoroughly described in this section since it's of vital importance to implement a exact Riemann solver for this problem.

The relaxed PGD equations are (2.14)

$$\begin{cases} \dfrac{\partial \alpha}{\partial t} & +\overline{\nabla} \cdot \left(\alpha \underline{\boldsymbol{u}}_p\right) = 0 \\ \dfrac{\partial \alpha \underline{\boldsymbol{u}}_p}{\partial t} & +\overline{\nabla} \cdot \left(\alpha \underline{\boldsymbol{u}}_p \otimes \underline{\boldsymbol{u}}_p + \pi \underline{\underline{\boldsymbol{I}}}\right) = \dfrac{\alpha C_D Re_d}{24K} \left(\underline{\boldsymbol{U}} - \underline{\boldsymbol{u}}_p\right) \\ \dfrac{\partial \alpha \pi}{\partial t} & +\overline{\nabla} \cdot \left(\alpha \pi \underline{\boldsymbol{u}}_p + c^2 \underline{\boldsymbol{u}}_p\right) = -\lambda \alpha \pi \end{cases} \tag{2.14}$$

Where c is the relaxation constant. The addition of a pseudo pressure π allows for the retrieval of a strictly hyperbolic problem and this will be proved in (2.2.4). The relaxation constant c is computed following Berthon's et. al. [7] work as (2.15)

$$c = \begin{cases} \max\left[0,\ \ \max\left(\alpha^i,\ \ \alpha^j\right)\left(u_p^i - u_p^j\right)\right] & \text{if} \quad u_p^i \geq u_p^j \\ \min\left[\varepsilon,\ \ \dfrac{1}{2}\min\left(\alpha^i,\ \ \alpha^j\right)\left(C + u_p^i - u_p^j\right)\right] & \text{if} \quad u_p^i < u_p^j \end{cases} \tag{2.15}$$

Where $C > \max(0, u_p^j - u_p^i)$ and $\varepsilon > 0$.

2.2.3. Boundary Conditions

The solution procedure is, for all intents and purposes, identical to the one described in (2.1.1) with the need to specify the equations being solved (therefore convective fluxes, source terms and conservative variables) and the boundary conditions being enforced. Since the conservations equations have been already discussed in (2.2), and the solution procedure in (2.1.1), only the wall boundary conditions still need to be defined.

In in-flight icing, the collection efficiency β of aerodynamic surfaces is of paramount importance when simulating ice accretion. Being the collection efficiency a measure of how much air impinges on a solid surface, it's easy to understand the need to properly define the boundary condition at the wall.

Bourgault et. al. explored in [8] the boundary conditions required to obtain stability and unicity when simulating the pressureless gas dynamics conservation laws.

Noting that characteristics have the same direction as particle velocity $\underline{\boldsymbol{u}}_p$, three different types of boundaries can be defined:

1. Inflow boundaries, where characteristics are coming into the domain
2. Outflow boundaries, where characteristics are going out of the domain
3. Slip boundaries, where characteristics are parallel to the boundary

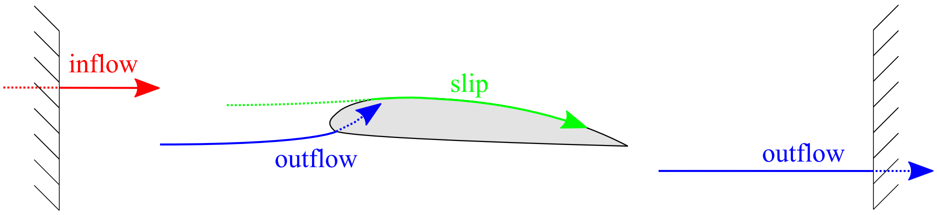


Figure 2.6: Types of boundaries for PGD

At inflow boundaries, boundary conditions on α_∞ and $\underline{\boldsymbol{u}}_p$ must be defined and enforced. This means all variables have to be be prescribed.

At ouflow and slip boundaries, no boundary condition must be applied according to [8] et al. Having changed the character of the problem through the relaxation, boundary conditions must be enforced here too.

The numerical approach used to apply the boundary condition is the one described by Guardone et. al. [15] which uses the eigenvalues of the jacobian to discriminate between boundary conditions (in characteristic variables) that must be enforced and those that must not be enforced.

The convective fluxes at a boundary control volume are evaluated at using the value $\underline{\boldsymbol{q}}_{BC}$ computed using (2.16).

$$\underline{\boldsymbol{q}}_{BC} = \underline{\boldsymbol{q}}_{domain} + \underline{\underline{\boldsymbol{R}}}\,\mathcal{SN}\left\{\underline{\underline{\boldsymbol{L}}}\left[\underline{\boldsymbol{q}}_{ext} - \underline{\boldsymbol{q}}_{domain}\right]\right\} \tag{2.16}$$

Where:

1. $\underline{\boldsymbol{q}}_{ext}$ is the vector of far-field values of the boundary conservative variables
2. $\underline{\boldsymbol{q}}_{domain}$ is the vector of conservative variables in the boundary control volume
3. $\underline{\underline{\boldsymbol{L}}}$ is the matrix where the $i-th$ row corresponds to the $i-th$ left eigenvector of the jacobian (evaluated in $\underline{\boldsymbol{q}}_{domain}$)
4. $\underline{\underline{\boldsymbol{R}}}$ is the matrix where the $j-th$ column corresponds to the $j-th$ right eigenvector of the jacobian (evaluated in $\underline{\boldsymbol{q}}_{domain}$)
5. $\underline{\underline{\boldsymbol{L}}}$ and $\underline{\underline{\boldsymbol{R}}}$ are adimensionalized as $\underline{\underline{\boldsymbol{R}}}\,\underline{\underline{\boldsymbol{L}}} = \underline{\underline{\boldsymbol{I}}}$
6. $\mathcal{SN}\left\{\underline{\boldsymbol{x}}\right\}$ is an operator defined as (2.17)

$$\underline{\boldsymbol{y}} = \mathcal{SN}\left\{\underline{\boldsymbol{x}}\right\} \quad \rightarrow \quad y_i = \begin{cases} x_i & \text{if} \quad \lambda_i < 0 \\ 0 & \text{if} \quad \lambda_i > 0 \end{cases} \tag{2.17}$$

This approach allows for the automatic enforcement of boundary conditions in a hyperbolic non linear system of conservation laws, making use of the characteristic variables $\underline{\boldsymbol{v}} = \underline{\underline{\boldsymbol{L}}}\underline{\boldsymbol{q}}$.

In the multidimensional (3D) case at hand, the eigenvalues are (2.20), the right eigenvector column matrix is (2.21), the left eigenvector row matrix is (2.22), with $\hat{\boldsymbol{n}}$ being the normal pointing outwards of the boundary face (control volume → exterior).

Also, for ease of notation, the projected velocity is renamed $u_p^{\hat{\boldsymbol{n}}} = \left(\underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}}\right)$.

All of these are computed by projecting the convective fluxes along $\hat{\boldsymbol{n}}$ and computing the projected jacobian $\underline{\underline{\boldsymbol{J}}}^{\hat{\boldsymbol{n}}} = \dfrac{\partial F_i^{\hat{\boldsymbol{n}}}}{\partial q_j}$ remembering it will be evaluated in the domain control volume, where $\pi_{domain} = 0$.

$$\underline{\boldsymbol{F}}^{\hat{\boldsymbol{n}}} = \begin{bmatrix} \alpha \underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}} \\ \\ \dfrac{\alpha \underline{\boldsymbol{u}}_p \otimes \alpha \underline{\boldsymbol{u}}_p + \alpha \pi \underline{\underline{\boldsymbol{I}}}}{\alpha} \cdot \hat{\boldsymbol{n}} \\ \\ \dfrac{\alpha \pi \cdot \alpha \underline{\boldsymbol{u}}_p + c^2 \cdot \alpha \underline{\boldsymbol{u}}_p}{\alpha} \cdot \hat{\boldsymbol{n}} \end{bmatrix} \tag{2.18}$$

$$\underline{\underline{\boldsymbol{J}}}^{\hat{\boldsymbol{n}}} = \begin{bmatrix} 0 & n_x & n_y & n_z & 0 \\ \\ -u_p^{\hat{\boldsymbol{n}}} u_p - \dfrac{\pi n_x}{\alpha} & u_p^{\hat{\boldsymbol{n}}} + u_p n_x & u_p n_y & u_p n_z & \dfrac{n_x}{\alpha} \\ -u_p^{\hat{\boldsymbol{n}}} v_p - \dfrac{\pi n_y}{\alpha} & v_p n_x & u_p^{\hat{\boldsymbol{n}}} + v_p n_y & v_p n_z & \dfrac{n_y}{\alpha} \\ -u_p^{\hat{\boldsymbol{n}}} w_p - \dfrac{\pi n_z}{\alpha} & w_p n_x & w_p n_y & u_p^{\hat{\boldsymbol{n}}} + w_p n_z & \dfrac{n_z}{\alpha} \\ -u_p^{\hat{\boldsymbol{n}}} \left[\pi + \dfrac{c^2}{\alpha}\right] & \left(\pi + \dfrac{c^2}{\alpha}\right) n_x & \left(\pi + \dfrac{c^2}{\alpha}\right) n_y & \left(\pi + \dfrac{c^2}{\alpha}\right) n_z & u_p^{\hat{\boldsymbol{n}}} \end{bmatrix} \tag{2.19}$$

$$\underline{\boldsymbol{\lambda}} = \begin{bmatrix} u_p^{\hat{\boldsymbol{n}}} - \dfrac{c}{\alpha}\,, & u_p^{\hat{\boldsymbol{n}}}\,, & u_p^{\hat{\boldsymbol{n}}}\,, & u_p^{\hat{\boldsymbol{n}}}\,, & u_p^{\hat{\boldsymbol{n}}} + \dfrac{c}{\alpha} \end{bmatrix} \tag{2.20}$$

$$\underline{\underline{\boldsymbol{R}}} = \begin{bmatrix} \dfrac{\alpha}{c^2} & \dfrac{n_x}{u_p^{\hat{\boldsymbol{n}}}} & \dfrac{n_y}{u_p^{\hat{\boldsymbol{n}}}} & \dfrac{n_z}{u_p^{\hat{\boldsymbol{n}}}} & \dfrac{\alpha}{c^2} \\ \dfrac{\alpha u_p - c n_x}{c^2} & 1 & 0 & 0 & \dfrac{\alpha u_p + c n_x}{c^2} \\ \dfrac{\alpha v_p - c n_y}{c^2} & 0 & 1 & 0 & \dfrac{\alpha v_p + c n_y}{c^2} \\ \dfrac{\alpha w_p - c n_z}{c^2} & 0 & 0 & 1 & \dfrac{\alpha w_p + c n_z}{c^2} \\ \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.21}$$

$$\underline{\underline{\boldsymbol{L}}} = \begin{bmatrix} \dfrac{u_p^{\hat{\boldsymbol{n}}} c}{2} & -\dfrac{c n_x}{2} & -\dfrac{c n_y}{2} & -\dfrac{c n_z}{2} & \dfrac{1}{2} \\ u_p^{\hat{\boldsymbol{n}}} n_x & n_z^2 + n_y^2 & -n_x n_y & -n_x n_z & -\dfrac{\alpha u_p}{c^2} \\ u_p^{\hat{\boldsymbol{n}}} n_y & -n_x n_y & 1 - n_y^2 & -n_y n_z & -\dfrac{\alpha v_p}{c^2} \\ u_p^{\hat{\boldsymbol{n}}} n_z & -n_x n_z & -n_y n_z & 1 - n_z^2 & -\dfrac{\alpha w_p}{c^2} \\ -\dfrac{u_p^{\hat{\boldsymbol{n}}} c}{2} & \dfrac{c n_x}{2} & \dfrac{c n_y}{2} & \dfrac{c n_z}{2} & \dfrac{1}{2} \end{bmatrix} \tag{2.22}$$

Far Field Boundaries

In the case of far field boundaries, $\underline{\boldsymbol{q}}_{\mathbf{ext}} = \underline{\boldsymbol{q}}_{ext}$ is the vector of user defined freestream conservative variables, and the method previously described automatically enforces the required boundary condition (inflow or outflow).

Euler Wall Boundaries

In the case of solid boundaries, a mixed *no penetration-outflow* condition is employed. This allows for the computation of the collection efficiency, by allowing mass to "escape" the domain from the walls when the droplets impinge on them.

To implement this, $\underline{\boldsymbol{q}}_{ext}$ needs to be properly defined.

1. $\underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}} > 0$ therefore the droplet is going out of the domain, $\underline{\boldsymbol{q}}_{ext} = \underline{\boldsymbol{q}}_{domain}$
2. $\underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}} < 0$ therefore the droplet is entering from a wall (unphysical), the velocity is reflected to enforce a no penetration condition (2.23)

$$\underline{\boldsymbol{q}}_{ext} = \begin{bmatrix} \alpha_{domain} \\ \underline{\boldsymbol{u}}_p - \left(2\underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}}\right) \hat{\boldsymbol{n}} \\ \pi_{domain} \end{bmatrix} \tag{2.23}$$

2.2.4. 1D Riemann Problem Solution

The solution to this 1D Riemann problem is exactly the same as the one found in Berthon et. al. [7]. The derivation was missing therefore it has been performed from scratch step by step in Appendix (A), using [19] as reference on the procedure. The final solution is (2.24)

$$\begin{array}{c|c|c|c}
\underline{\boldsymbol{p}}^L & \underline{\boldsymbol{p}}^{*L} & \underline{\boldsymbol{p}}^{*R} & \underline{\boldsymbol{p}}^R \\
\hline
\alpha^L & \dfrac{2\alpha^L c}{2c-\alpha^L\left(u_p^L-u_p^R\right)} & \dfrac{2\alpha^R c}{2c-\alpha^R\left(u_p^L-u_p^R\right)} & \alpha^R \\
u_p^L & \dfrac{u_p^L+u_p^R}{2} & \dfrac{u_p^L+u_p^R}{2} & u_p^R \\
0 & c\dfrac{u_p^L-u_p^R}{2} & c\dfrac{u_p^L-u_p^R}{2} & 0
\end{array} \tag{2.24}$$

It's important to note that (2.24) is a valid solution to the Riemann problem under the assumption that $c > \max\left(0, \alpha^L\dfrac{u^L-u^R}{2}, \alpha^R\dfrac{u^L-u^R}{2}\right)$.

The knowledge of the exact solution of the Riemann problem allows for a straightforward implementation of an exact Riemann solver.

2.3. Multibin (Polydispersity)

The classical approach is to define a Mean Volumetric Diameter *MVD* and assume the cloud is homogeneous, therefore all droplets have the same diameter.

In reality, clouds are heterogeneous, being composed of many droplets of many diameters. This means that taking into account the droplet size distribution will yield a more accurate representation of the actual droplet field. From now on, considering a polydispersed droplet field will be referred to as running a *multibin* simulation. This is done by sampling the droplet diameter distribution into bins. Each bin has its own water content percentage and droplet diameter.

In an Eulerian solver this translates in the computation of one droplet field for each value of the diameter d_p, with the obvious increase in computational cost.

The droplet size distribution will be referred to as *bin distribution* from now on, with the $i-th$ bin being characterized by its diameter d_p^i and its LWC fraction $\frac{LWC^i}{LWC^{tot}}$.

	d_p^i	$\frac{LWC^i}{LWC^{tot}}$
Bin 1	$1\mu m$	10%
Bin 2	$4\mu m$	12%
Bin 3	$10\mu m$	15%
Bin 4	$14\mu m$	30%
Bin 5	$21\mu m$	13%
Bin 6	$28\mu m$	11%
Bin 7	$37\mu m$	9%

Table 2.1: Example of a bin distribution

2.3.1. Collection Efficiency

This work is deeply rooted in in-flight icing, where the only parameter required from the droplet field is the collection efficiency β on the body of the aircraft.

In Lagrangian solvers, the computation of collection efficiency does not change whether the droplets are mono or polydispersed since only the mass collected at the surface matter and each droplet carries all the information required to compute it.

In the Eulerian solver the collection efficiencies obtained for each $i-th$ simulation of the $i-th$ bin needs to be combined to obtain the total collection efficiency β_{tot} using (2.25).

$$\beta_{tot} = \sum_{i=1}^{N_{bins}} \left(\frac{LWC^i}{LWC^{tot}} \cdot \beta^i \right) \tag{2.25}$$

2.3.2. Langmuir D Bin Distribution

Usually experimental reports on in-flight icing tests provide a bin distribution that best represents that experimental setup. This is great when validating a new ice accretion tool since it provides a way to minimize the uncertainty in the inputs and verify that numerical results agree with the physics.

Airplanes on the other hand fly in many different conditions, through clouds of varying LWC, MVD and bin distribution. This means that in order to run a multibin simulation in a test case that is not replicating an experiment (e.g. a new three element airfoil setup during the conceptual design phase) the bin distribution needs to be generated automatically from a given MVD.

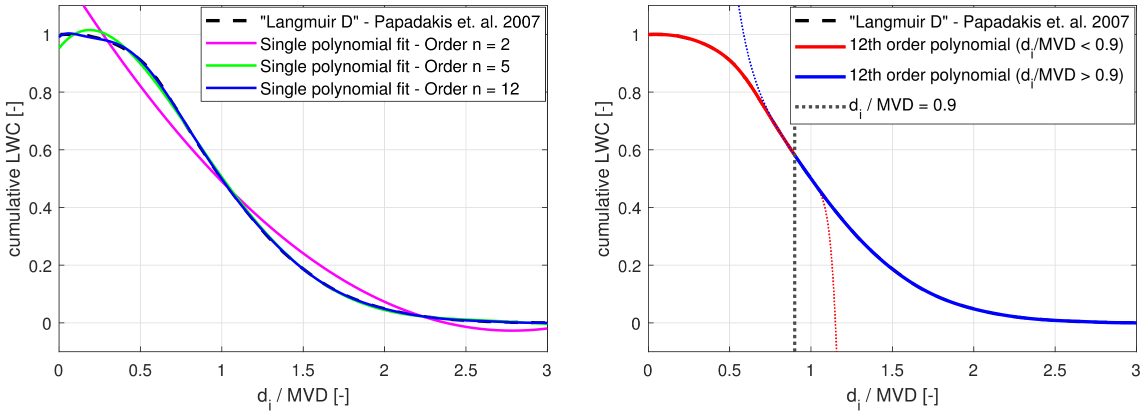
Unfortunately studies on the distribution of d_p and LWC in clouds are not many, with [17] being the notable exception. Langmuir in 1946 published a report [17] commissioned by the Army Air Forces and General Electric on water droplet trajectories and it contains various experimental measurements of droplet sizes in clouds. There is an overall consensus on the fact that the "Langmuir D" distribution contained in this work is the most accurate depiction of actual clouds' droplet size distribution.

The distribution is given as LWC percentage and size ratio $\frac{d_p^i}{MVD}$ in only 7 bins, limiting its direct usage possibilities. A continuous version of Langmuir D is found in [20]. The analytic version (and/or the approach used to obtain it) is unknown, therefore it has been sampled in a large number of points to obtain a sizable dataset from which to create a polynomial approximation.

The use of a single polynomial fit presented in Figure (2.7a) for the whole distribution is not acceptable since, no matter the order of the polynomial, at $\frac{d_p^i}{MVD} \simeq 0$ and $\frac{d_p^i}{MVD} \simeq 3$ the cumulative LWC is not monotone.

Monotonicty of $LWC_{cumulative}$ is fundamental to avoid negative values of LWC^i when computing it using (2.26).

To overcome the monotonicity problem two different polynomial fits have been performed for the data using $\frac{d_p^i}{MVD} = 0.9$ as the switching point from the first and second polynomial. The fit has been performed with a slight overlap to obtain a smooth approximation about the switching point.



(a) Single polynomial fit of Langmuir D

(b) Double polynomial fit of Langmuir D

Figure 2.7: Single vs double polynomial fit of Langmuir D

The final approximation therefore is used as follows:

1. Left polynomial fit (red in (2.7b)) if $\frac{d_p^i}{MVD} \in (0.0, 0.9]$
2. Right polynomial fit (blue in (2.7b)) if $\frac{d_p^i}{MVD} \in (0.9, 3.0]$

This allows for multibin simulations even without prior knowledge of the size distribution of droplets in the case being considered.

If a series of d_p^i has been chosen such that $d_p^{i+1} > d_p^i \quad \forall i$, a bin's LWC^i can be computed as (2.26) from the cumulative LWC curve.

$$LWC^i = LWC_{cumulative}(d_p^{i-1}) - LWC_{cumulative}(d_p^i) \qquad i = 1, ..., N_{bins} \tag{2.26}$$

2.3.3. Computational Cost Mitigation

To mitigate the computational cost of a multibin simulation, each bin after the first is restarted from the converged solution of the previous bin. This can decrease computational cost significantly in high bin number simulations since each bin is very close in droplet diameter (and therefore behavior) to its previous one.

In three dimensional simulations this will prove essential, as Figure (3.28) will show in the final part of this work.

2.4. Lagrangian Reinjection & Reimpingement

Not all droplets striking the surface stick to it. Bai and Gosman [5] found that there are four principal ways a droplet may interact with a wall:

1. **stick**, low $\underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}}$, droplet maintains its shape
2. **spread**, high $\underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}}$, droplet forms a film on the surface
3. **rebound**, medium $\underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}}$, the complete droplet bounces off the surface
4. **splash**, very high $\underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}}$, part of the droplet spreads and part is reinjected in the air as many smaller droplets

In this work, a model to correct the collection efficiency for splashing and rebound effects is implemented. The model that will be use is the one found in LEWICE and described by Wright in [27].

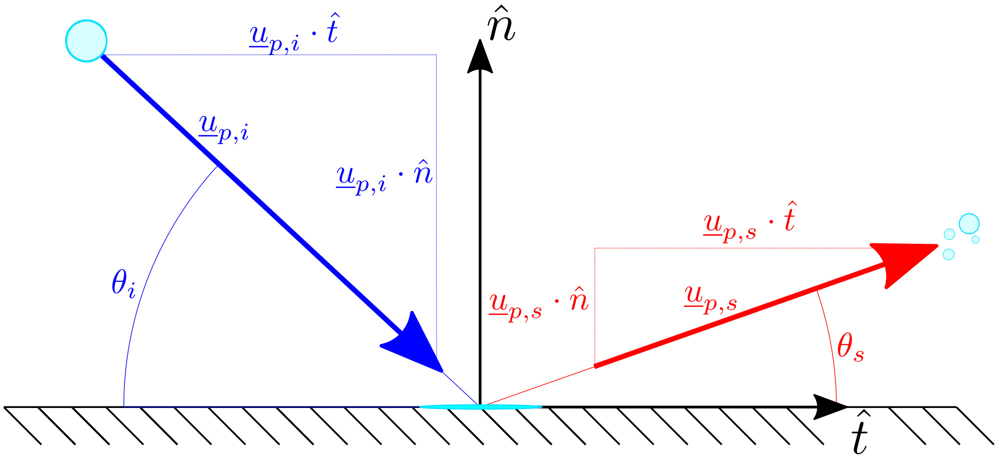
This model will be used to correct the collection efficiency, by removing the mass that rebounds or splashes. The mass that has been removed will then be reinjected using PoliDrop (a Lagrangian solver) and tracked to compute how much of it reimpinges on aft surfaces. The mixed Eulerian-Lagrangian approach is dictated by the impossibility of considering boundaries both as inlets (of splashed/rebounded droplets) and outlets (impinging and reimpinging droplets) in the Eulerian framework.

The splashing model [27] has been obtained by taking the collection efficiency results of many experimental results, and then tuning the parameters in such a way that it matched the largest amount possible of cases.

Denoting $(\cdot)_i$ all incoming properties and $(\cdot)_s$ all properties after splashing, the model requires the computation at the surface of many different parameters:

$$\begin{aligned}
&\text{Droplet Ohnesorge number:} && Oh = \frac{\mu_p}{\sqrt{\rho_p \sigma_p d_p}} \\
&\text{Droplet Reynolds number:} && Re_p = \frac{\rho_p u_p d_p}{\mu_p} \\
&\text{Mundo splashing parameter:} && K = Oh \cdot Re_p^{\frac{5}{4}} \\
&\text{LEWICE splashing parameter:} && K_L = K^{0.859} \left(\frac{\rho}{LWC} \right)^{0.125} \\
&\text{Normal LEWICE splashing parameter:} && K_{L,n} = \frac{K_L}{(sin(\theta_i))^{1.25}}
\end{aligned} \tag{2.27}$$

Where θ_i is the incoming droplet angle measured from the surface tangent (in degrees) towards the velocity as shown in Figure (2.8)



$\underline{u}_{p,i} \cdot \hat{t}$
$\hat{n}$
$\underline{u}_{p,i}$
$\underline{u}_{p,i} \cdot \hat{n}$
θ_i
$\underline{u}_{p,s} \cdot \hat{t}$
$\underline{u}_{p,s} \cdot \hat{n}$
$\underline{u}_{p,s}$
θ_s
$\hat{t}$

Figure 2.8: Splashing velocity components sketch

According to the model, splashing occurs if the splashing discriminator S_{disc} (2.28) is greater than 0.

$$\begin{aligned} &S_{disc} = K_{L,n} - 200 > 0 \quad \text{splashing occurs} \\ &S_{disc} = K_{L,n} - 200 \leq 0 \quad \text{splashing does not occur} \end{aligned} \tag{2.28}$$

After splashing, the model also yields the properties of splashed droplets (2.29). These are needed to start a simulation of the droplets being reinjected into the domain. Given the steady nature of the simulations considered in this work, one must reason in terms of water flux on surfaces and not mass. Note that $L\dot{W}C = LWC\underline{\boldsymbol{u}}_p \cdot \hat{\boldsymbol{n}}$.

$$\begin{aligned}
&\text{Liquid Water Content Flux:} && L\dot{W}C_s = L\dot{W}C_i \cdot 0.7\,(1 - \sin\theta_i) \times \\
& && \quad \times \left[1 - e^{-0.0092026 \cdot S_{disc}}\right] \\
&\text{Droplet Diameter:} && d_{p,s} = d_{p,i} \cdot 8.72 e^{0.0281K} \qquad 0.05 \leq \frac{d_{p,s}}{d_{p,i}} \leq 1 \\
&\text{Normal Velocity:} && \left[\underline{\boldsymbol{u}}_{p,s} \cdot \hat{\boldsymbol{n}}\right] = \left[\underline{\boldsymbol{u}}_{p,i} \cdot \hat{\boldsymbol{n}}\right](0.3 - 0.002\theta_i) \\
&\text{Tangential Velocity:} && \left[\underline{\boldsymbol{u}}_{p,s} \cdot \hat{\boldsymbol{t}}\right] = \left[\underline{\boldsymbol{u}}_{p,i} \cdot \hat{\boldsymbol{t}}\right](1.075 - 0.0025\theta_i) \\
&\text{Collection Efficiency:} && \beta_{corr} = \beta\left(1 - \frac{LWC_s \mid \underline{\boldsymbol{u}}_{p,s} \mid}{LWC_i \mid \underline{\boldsymbol{u}}_{p,i} \mid}\right)
\end{aligned} \tag{2.29}$$

Its worth pointing out that β_{corr} is the collection efficiency corrected for the **removal** of splashed droplets. These can later impinge on aft surfaces, and this is where the

Lagrangian step of this work comes in, by tracking these so called *secondary* droplets individually and adding the collection efficiency.

In the Eulerian framework the area of the surface mesh element is also required to obtain the number per unit time $\dot{n}_s$ of droplets that have been generated by splashing.

The number of droplets per unit second can be computed from $L\dot{W}C_s$ simply as (2.30), where *area* is the wall surface element's area. In the case of a 2D simulation the area is assumed as $area = length \cdot 1\ [m]$.

$$\dot{n}_s = \frac{mass\ flux_{splashed\ water\ on\ face}}{mass_{single\ splashed\ droplet}} = \frac{L\dot{W}C_s}{\rho_p}\frac{3\pi}{4}\left(\frac{2}{d_{p,s}}\right)^3 \cdot area \tag{2.30}$$

Finally, the droplets are reinjected in PoliDrop in the position of the node where the splashing occured. To avoid numerical issues due to floating point computations the splashed droplets are not placed exactly at the node, but they are displaced by a small $\varepsilon << 1$ in the normal direction to the surface as shown in Figure (2.9).

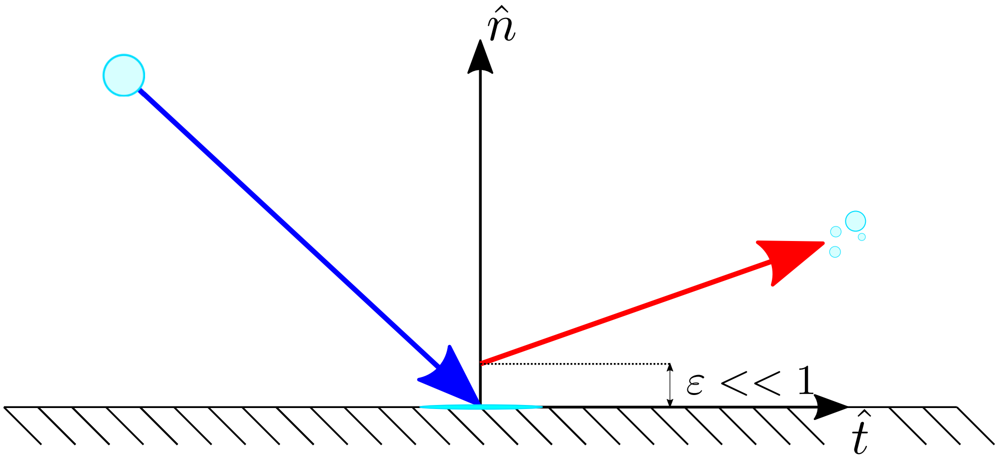


Figure 2.9: Small artificial displacement of splashed droplets from wall

Therefore all information needed by PoliDrop to start a Lagrangian simulation of splashed droplets are available.

Velocity	$\underline{\boldsymbol{u}}_{p,s}$
Number of Droplets per unit time	$\dot{n}_s$
Diameter of Droplets	$d_{p,s}$
Position of Reinjection	$[x_s, y_s, z_s]$

Table 2.2: Initial conditions required by PoliDrop to start splashed simulation

Finally, when the Lagrangian simulation of the splashed droplets is run, the additional collection efficiency gets computed as (2.31) where $\dot{m}_i$ is actually now not the mass caught by the surface element, but the mass per unit time caught by the surface element.

$$\beta_{tot} = \beta_{corr} + \frac{\dot{m}_i}{A_i LWC_\infty U_\infty} \tag{2.31}$$

Another feature implemented is the spacing of the Lagrangian parcels tangentially to the face. Instead of injecting one parcel per surface mesh element, the user has the possibility of splitting the droplets in more parcels (with an increase in computational cost and solution smoothness) as will be seen in (3.5.1). An example of no spacing and a 10 element subdivision spacing is in (2.10).

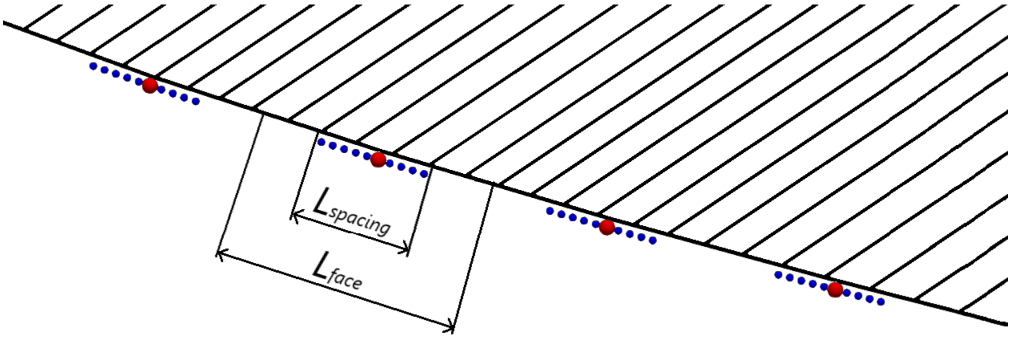


Figure 2.10: Reimpingement spacing example. In red no spacing (1 parcel per surface mesh element), in blue 10 element subdivision spacing (10 parcels per surface mesh element)

As Figure (2.10) shows, the spacing is not done on the whole surfacial element, but it only covers a fraction of it.
This fraction $F_{spacing} = \frac{L_{spacing}}{L_{face}} \leq 1$ is also a user modifiable parameter. This is where an adaptive technique, or a smarter spacing would help increase the robustness of this Lagrangian reimpingement step.

2.5. Complete Solution Procedure & Implementation Strategy

SU2 has one main way of running a simulation, shown in Figure (2.11).

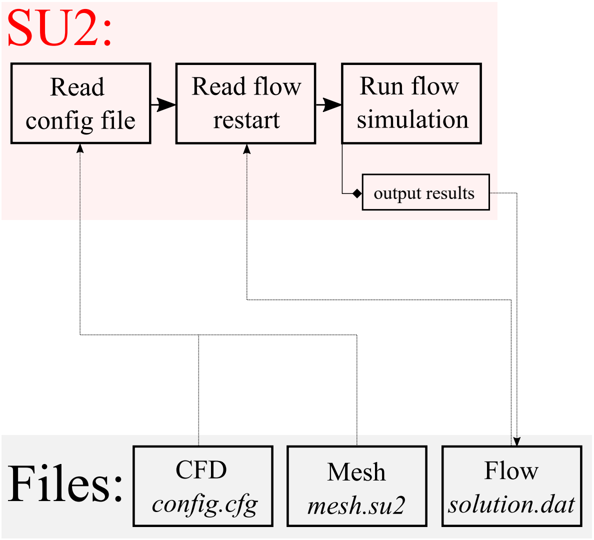


Figure 2.11: SU2 base computational procedure scheme

This approach is used to solve one (or more) set of conservation equations concurrently.

In an effort aimed at reducing the computational cost that would be added by solving all conservation equations (flow and droplets) simultaneously, the approach that will be used in this work is:

1. Compute the (Eulerian) flow solution (once)
2. Compute the (Eulerian) droplets' flow field (for each bin if multibin)

    Compute the collection efficiency β (for each bin if multibin)

    Compute and write the splashed droplets' data (for each bin if multibin)

3. Compute the (Lagrangian) droplets' trajectory from splashed data

    Compute the reimpinged collection efficiency $\beta_{reimpinge}$

This split in the solution procedure between air flow and droplet field is only possible because the motion of droplets through air is assumed to have negligible effects on the air flow itself (one way coupling flow → droplets) as discussed in (1.1.2).

To be able to use SU2 as the "home" of this solution procedure, the base SU2 framework (Figure (2.11)) needed to be expanded to run consecutive simulations of different types, while retaining the old one in memory (the flow velocity $\underline{\boldsymbol{U}}$ appears in the momentum drag source term of the PGD equations in (2.7)).

Finally, a routine has been implemented in PoliDrop to load the data in `splashed_data.dat` and run the simulation, together with the required modifications to compute the collection efficiency in terms of mass fluxes.

The file is structured as a comma separated value file, with the structure shown in Table(2.3).

$\mathbf{x_s}[m]$	$\mathbf{y_s}[m]$	$\mathbf{z_s}[m]$	$\mathbf{u_s}[m/s]$	$\mathbf{v_s}[m/s]$	$\mathbf{w_s}[m/s]$	$\mathbf{d_{p,s}}[m]$	$\mathbf{n_s}[-]$
-0.263567	-0.0382377	0	30.5969	-11.0613	0	4.6193e-06	46
-0.258612	-0.0389092	0	31.9818	-11.212	0	4.6193e-06	60
-0.253654	-0.0395755	0	33.2008	-11.2786	0	4.6193e-06	73
-0.248689	-0.040229	0	34.3232	-11.2379	0	4.6193e-06	87
-0.243717	-0.0408604	0	35.3548	-11.1075	0	4.6193e-06	100
-0.238741	-0.0414657	0	36.2769	-10.9248	0	4.6193e-06	112
-0.233763	-0.0420447	0	37.0796	-10.7226	0	4.6193e-06	124
-0.228783	-0.0426002	0	37.7697	-10.5191	0	4.6193e-06	135
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.

Table 2.3: Example of splashed_data.dat file

A complete description of the simulation procedure, with all the files being shared between SU2 and PoliDrop, is present in Figure (2.12).

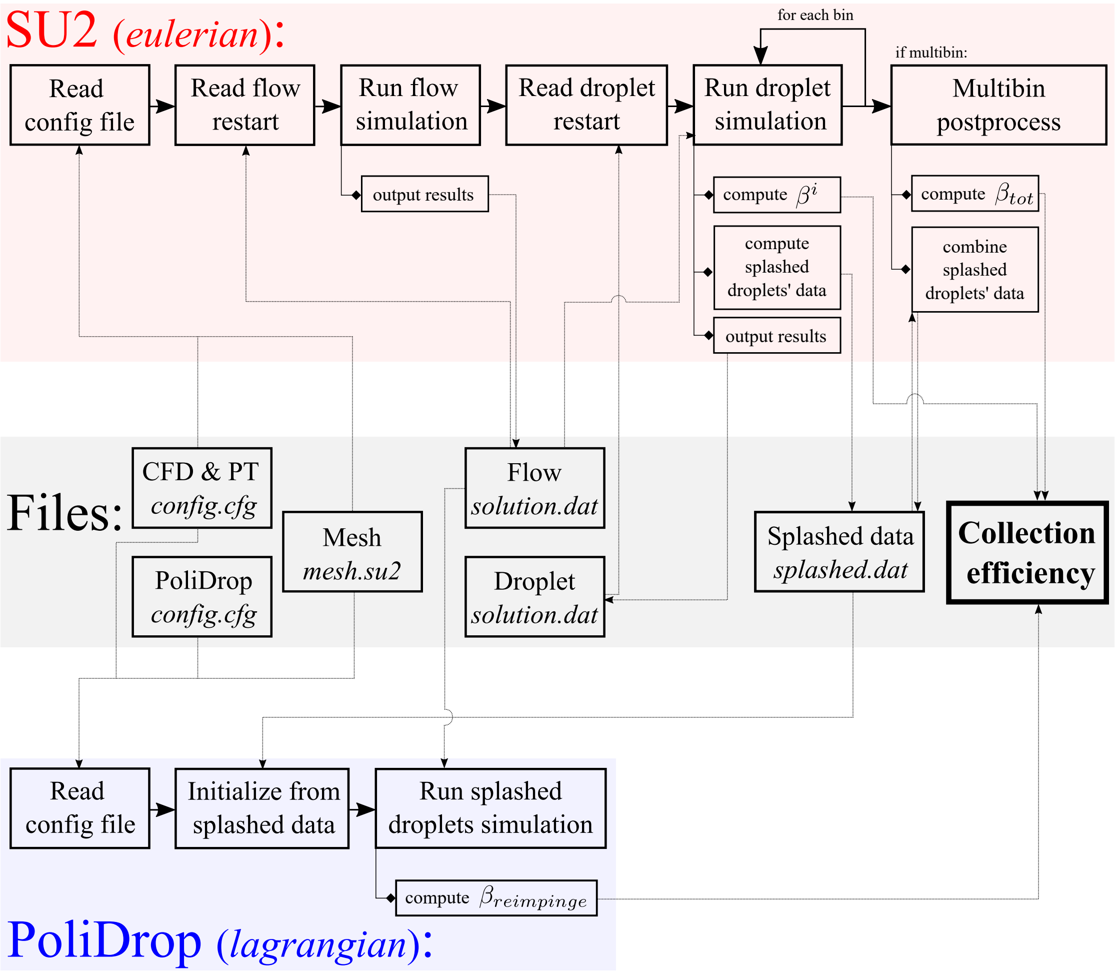


Figure 2.12: Complete computational procedure scheme

3 | Results & Code Validation

3.1. MUSCL Validation

In the first part of this results section, the MUSCL reconstruction will be assessed to ensure mesh convergence is indeed achieved with coarser meshes (i.e. faster spatial convergence).

This will be done by simulating the well known experiments conducted by Papadakis et. al. in 2004 [24], where the collection efficiency for various MVD's has been measured experimentally. All important values needed to run these simulations are in Table (3.1).

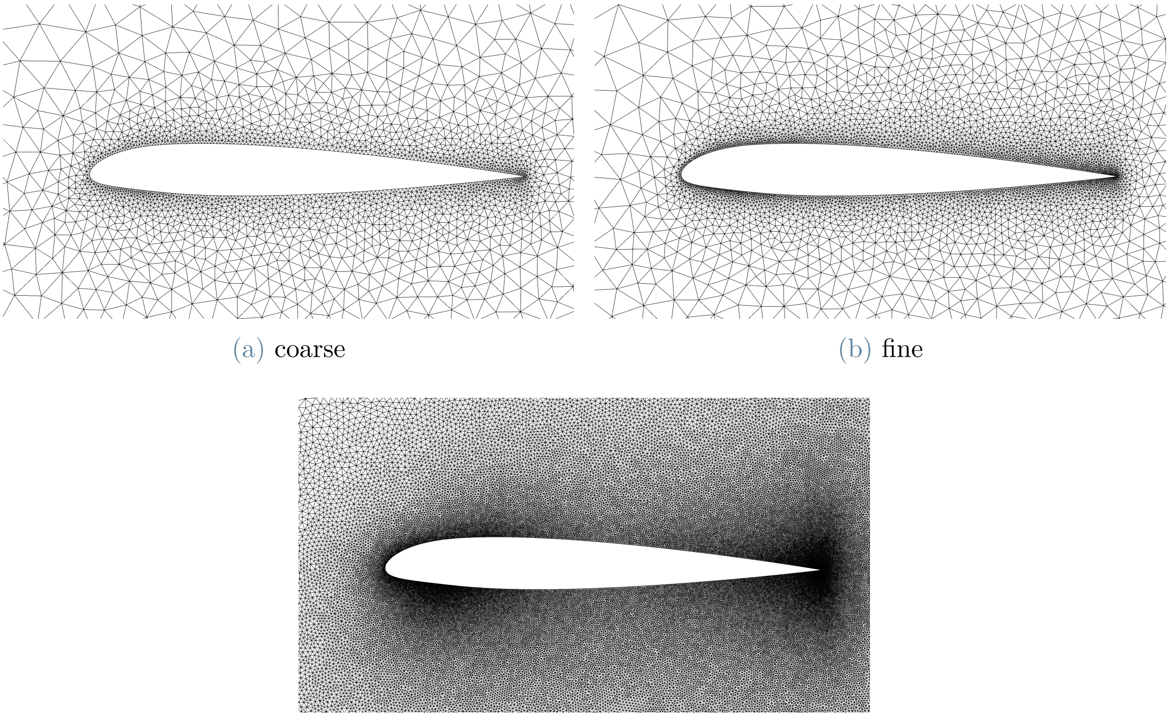
Airfoil	NACA23012
	$chord = 0.914\ m$
	$AoA = 2.5°$
Cloud & Droplets	$LWC = 0.5\ g/m^3$
	$MVD = 20,\ 52,\ 111,\ 154,\ 236\ \mu m$
	$\mu_p = 0.0011208\ Pa/s$
	$\rho_p = 1000\ kg/m^3$
	$\sigma_p = 0.074\ N/m$
Airflow	EULER
	$M_\infty = 0.22937$
	$P_\infty = 94802.914\ Pa$
	$T_\infty = 288.705\ K$

Table 3.1: Experimental setup data - NACA 23012 Papadakis et. al. [24]

To do so three meshes have been generated with Gmsh [13] (Table (3.2) and Figure (3.1)), where the *coarse* and *fine* ones are used to check how the collection efficiency changes by doubling the elements. Meanwhile the *reference* mesh is over refined on purpose and is used as a second check.

Mesh name	Number of nodes
coarse	3842
fine	6381
reference	191621

Table 3.2: Mesh names for mesh convergence assessment


(a) coarse

(b) fine

(c) reference

Figure 3.1: Meshes NACA 23012

The flow-field used for all droplet simulations (both Eulerian and lagrangian) is depicted in Figure (3.2).

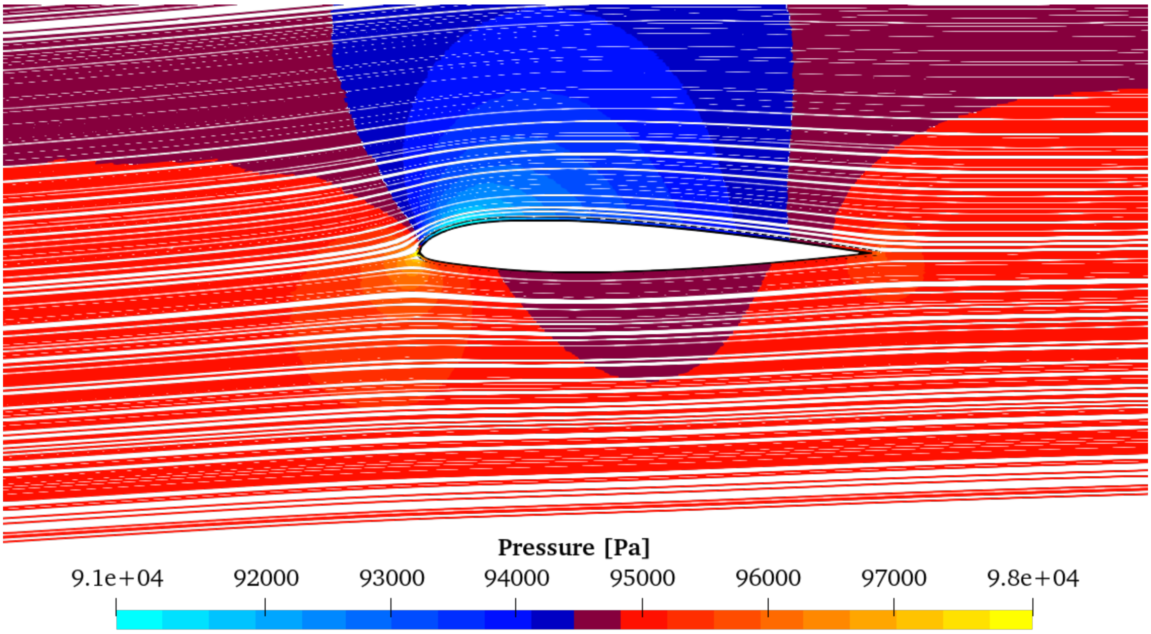
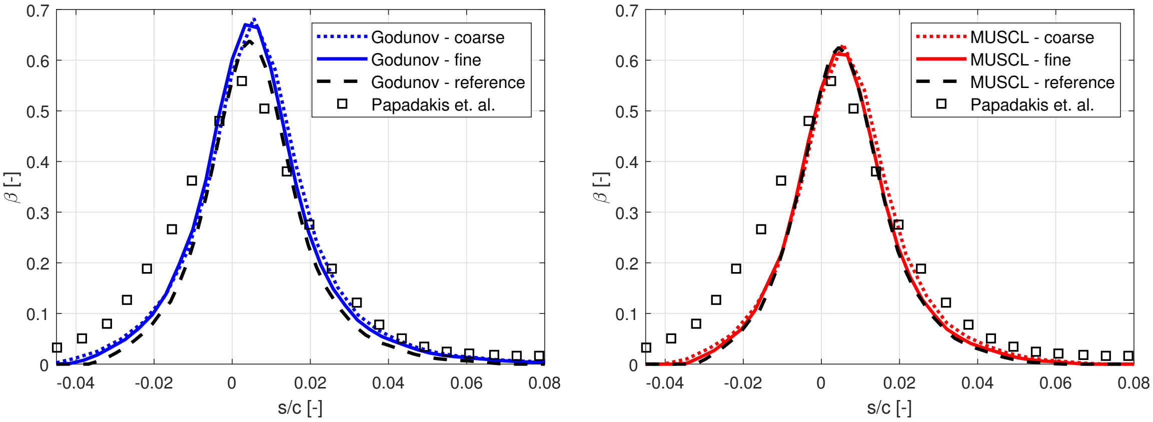


Figure 3.2: Euler flow-field for Papadakis et. al. [24] NACA23012 simulations

3.1.1. Mesh Convergence

The collection efficiency results for the *coarse*, *fine* and *reference* meshes are reported here, and compared to the experimental results for $MVD = 20\mu m$ in Figure (3.3) and $MVD = 154\mu m$ in Figure (3.4).

It's easy to see that for $MVD = 20\mu m$ in Figure (3.3) the MUSCL reconstruction is indeed providing faster mesh convergence, with the *coarse* mesh already yielding usable results (3.3b) while the *fine* mesh is still a small way away from converging when only using cell averages (3.3a).

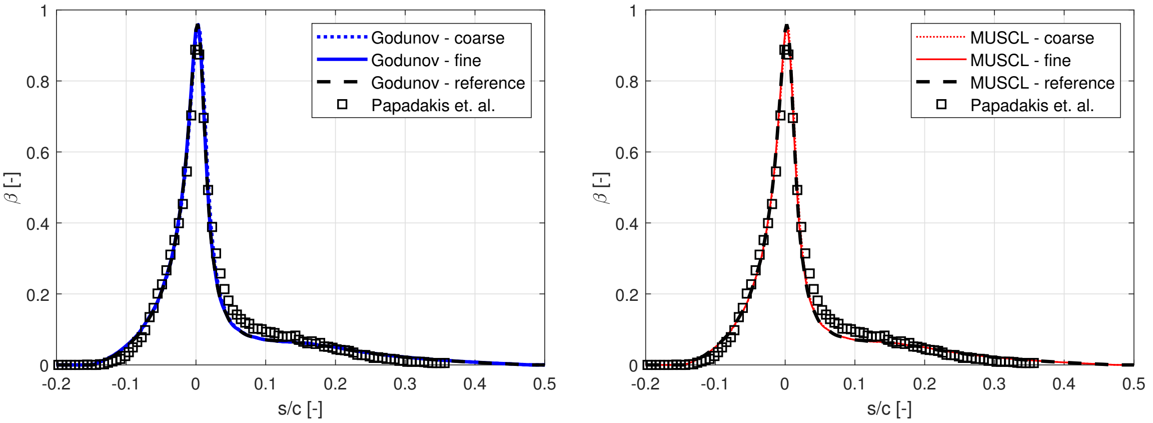


(a) Godunov mesh convergence

(b) Godunov+MUSCL mesh convergence

Figure 3.3: Mesh convergence analysis for $MVD = 20\mu m$ collection efficiency - NACA 23012 [24] - Cell Averages vs MUSCL

When looking at higher values of MVD such as $MVD = 154\mu m$ in Figure (3.4) a coarse mesh with no MUSCL reconstruction is already able to produce a mesh converged result. This is due to the ballistic behavior of droplets, with almost straight trajectory and little to no dependence on the airflow.



(a) Godunov mesh convergence

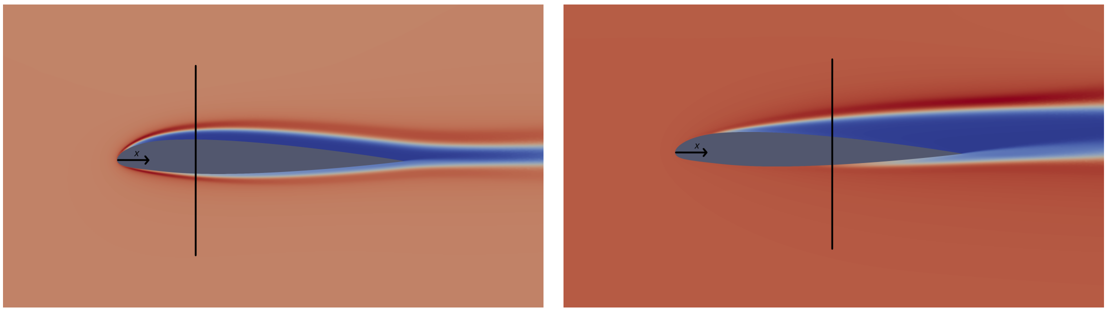
(b) Godunov+MUSCL mesh convergence

Figure 3.4: Mesh convergence analysis for $MVD = 154\mu m$ collection efficiency - NACA 23012 [24] - Cell Averages vs MUSCL

3.1.2. Vacuum Approximation

Another important feature to check when looking at the behavior of higher order methods is the sharpness with which they capture a discontinuity. The main discontinuity occurring in pressure-less gas dynamics is the appearance of shadow zones, in this case when the airfoil shields a portion of the domain from incoming droplets. This would mean, in a ideal world, $\alpha = 0$ and a sharp step between the two areas.

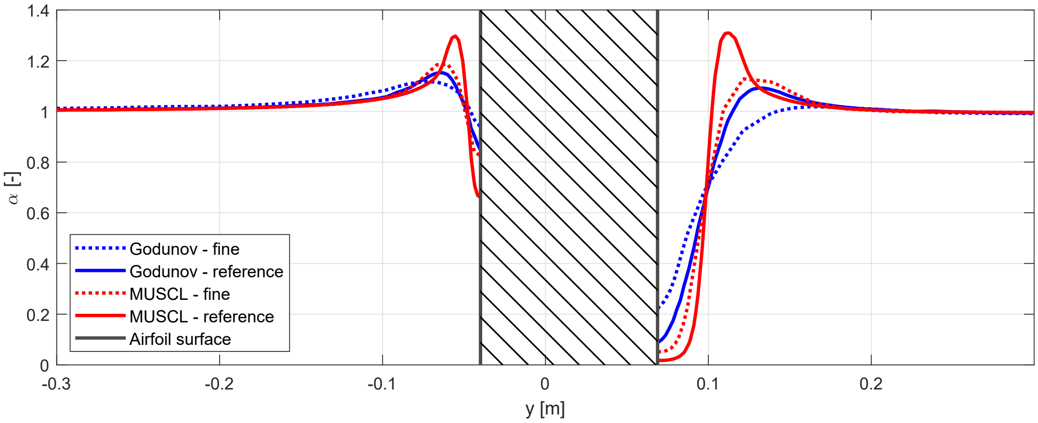
The volume fraction α has been plotted across a vertical section both on the pressure and the suction side of the airfoil. The line sources of these plots are show in Figure (3.5) for both $MVD = 20\mu m$ and $MVD = 154\mu m$. In Figure (3.6) the left hand side of the plots corresponds to the pressure (lower) side of the airfoil and the right hand side corresponds to the suction (upper) side of the airfoil.



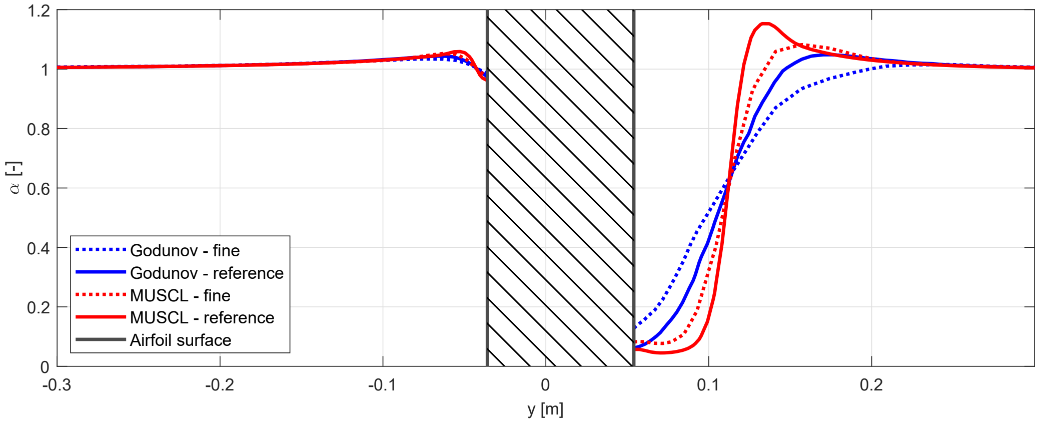
(a) $MVD = 20\mu m$ (b) $MVD = 154\mu m$

Figure 3.5: Line plot position for Figure (3.6) - NACA 23012 [24] - Coloring α

The resulting α profiles in Figures (3.6a) for $MVD = 20\mu m$ and (3.6b) for $MVD = 154\mu m$ show sharper discontinuities with coarser meshes when using the MUSCL reconstruction.



(a) $MVD = 20\mu m$ at $x = 0.25m$



(b) $MVD = 20\mu m$ at $x = 0.5m$

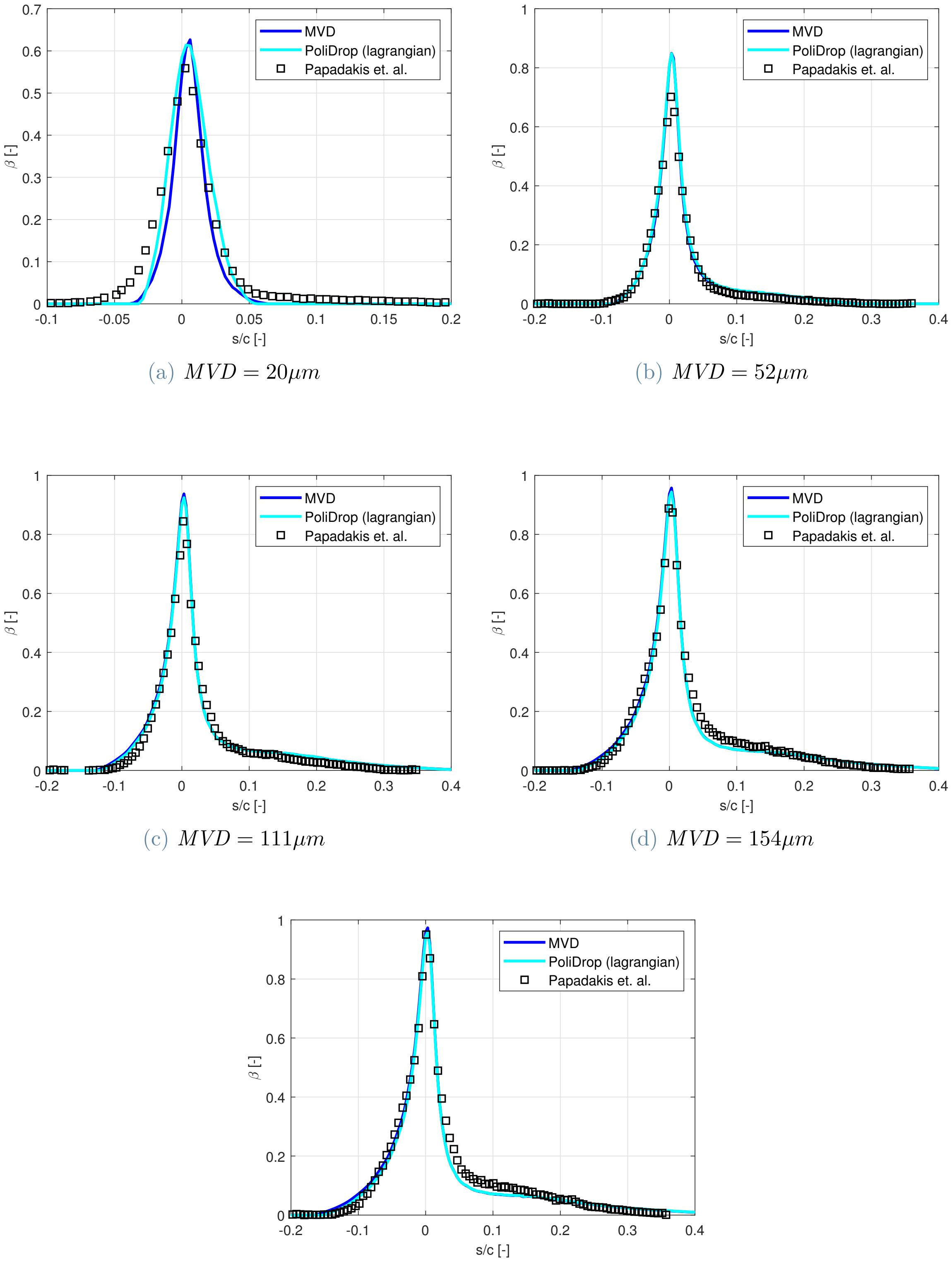
Figure 3.6: Vacuum zone discontinuity for $MVD = 20\mu m$ and $MVD = 154\mu m$ - NACA 23012 [24] - Cell Averages vs MUSCL

Having validated that the MUSCL reconstruction allows for a faster mesh convergence, all results from here on out will be obtained using it.

3.2. Eulerian - Lagrangian Comparison

In this section, the Eulerian solver will be tested against PoliDrop to evaluate how the Euler-Euler approach stacks up against the Euler-Lagrange one.

The same test cases by Papadakis et. al. [24] have been run using the same exact mesh (*fine* from (3.2)) and same exact flow field solution. This way the most apples to apples comparison can be achieved also when comparing computational costs. PoliDrop simulations have been run using an adaptive cloud technique that minimizes the number of droplets needed to obtain converged results.



(a) $MVD = 20\mu m$

(b) $MVD = 52\mu m$

(c) $MVD = 111\mu m$

(d) $MVD = 154\mu m$

(e) $MVD = 236\mu m$

Figure 3.7: Collection efficiency - NACA 23012 [24] - MVD vs PoliDrop (Lagrangian)

The agreement between Lagrangian simulations, Eulerian simulations and experiments is high. There is no way to determine which approach is best in these test cases since they are all within margin of error of the experiments.

Surely an uncertainty quantification study would yield precious insight in quantifying how much space for improvement is actually available when it comes to matching the collection efficiency data of Papadakis et. al. [24].

3.3. Multibin

In this section multibin simulations will be presented and compared to experimental data, MVD simulations and the Lagrangian solver PoliDrop. Also, the Langmuir D bin distribution will be compared to the experimental one in order to verify if the behavior indeed improves (compared to *MVD* simulations) by using the multibin approach even with an artificial bin distribution.

The experimental 10 bin distribution given by Papadakis et. al. [24] is reported in (3.3).

$\boldsymbol{LWC^i}$ [%]	$\boldsymbol{d_p^i}$ [μm]				
5.0	3.850397	6.693706	11.05374	13.88450	16.25037
10.0	9.390637	16.88090	27.48959	44.44510	63.65823
20.0	13.80175	25.44875	56.48542	90.28305	135.4827
30.0	19.60797	59.17969	111.1060	154.1635	298.5197
10.0	30.73474	192.7506	212.7639	284.4519	645.4684
3.0	35.19787	216.5703	235.0038	343.7168	715.8689
1.0	38.32569	224.9867	257.7010	380.2672	747.3936
0.5	40.66701	229.0087	279.5447	400.9252	763.2455
0.5	44.36619	253.9279	312.5901	425.0601	1046.767
$\boldsymbol{MVD}$	**20μm**	**52μm**	**111μm**	**154μm**	**236μm**

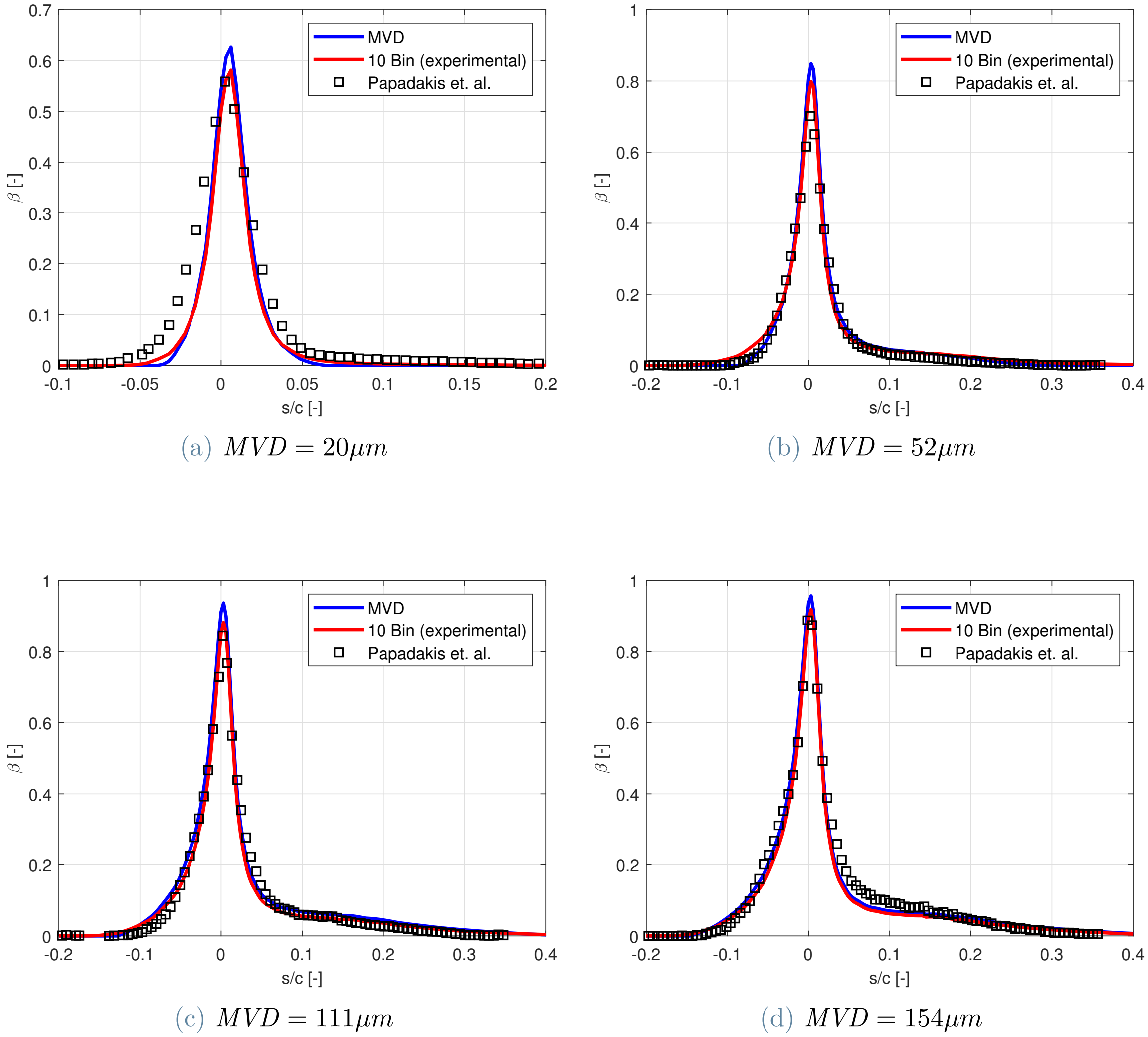
Table 3.3: Experimental 10 bin distribution - Papadakis et. al. [24]

Note: experimental results will be presented with discrete markers and not curves. The

10 bin (experimental) bin distribution mentioned across the following figures and results **is not** an experimental result but a numerical result obtained using the experimental bin distribution.

The flow-field is the same exact one (Figure (3.2)) used prior.

In Figure (3.8) the MVD simulations are compared to the multibin simulations (using the experimental 10 bin distribution) and the experimental results for all MVD values.

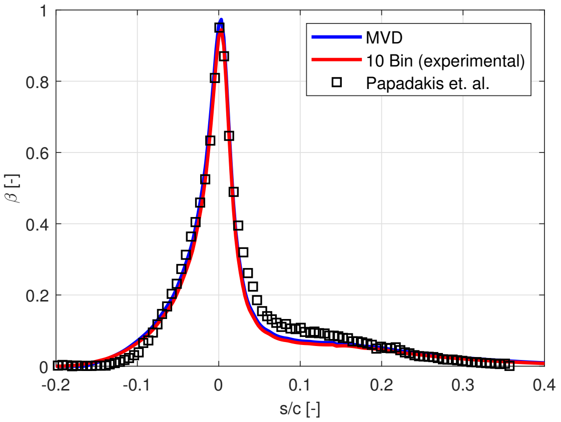


(a) $MVD = 20\mu m$

(b) $MVD = 52\mu m$

(c) $MVD = 111\mu m$

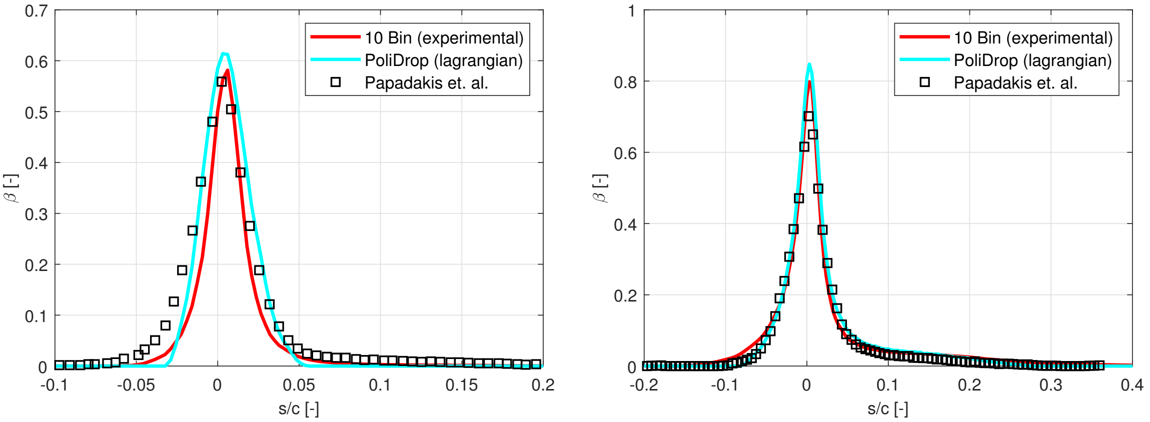
(d) $MVD = 154\mu m$



(e) $MVD = 236\mu m$

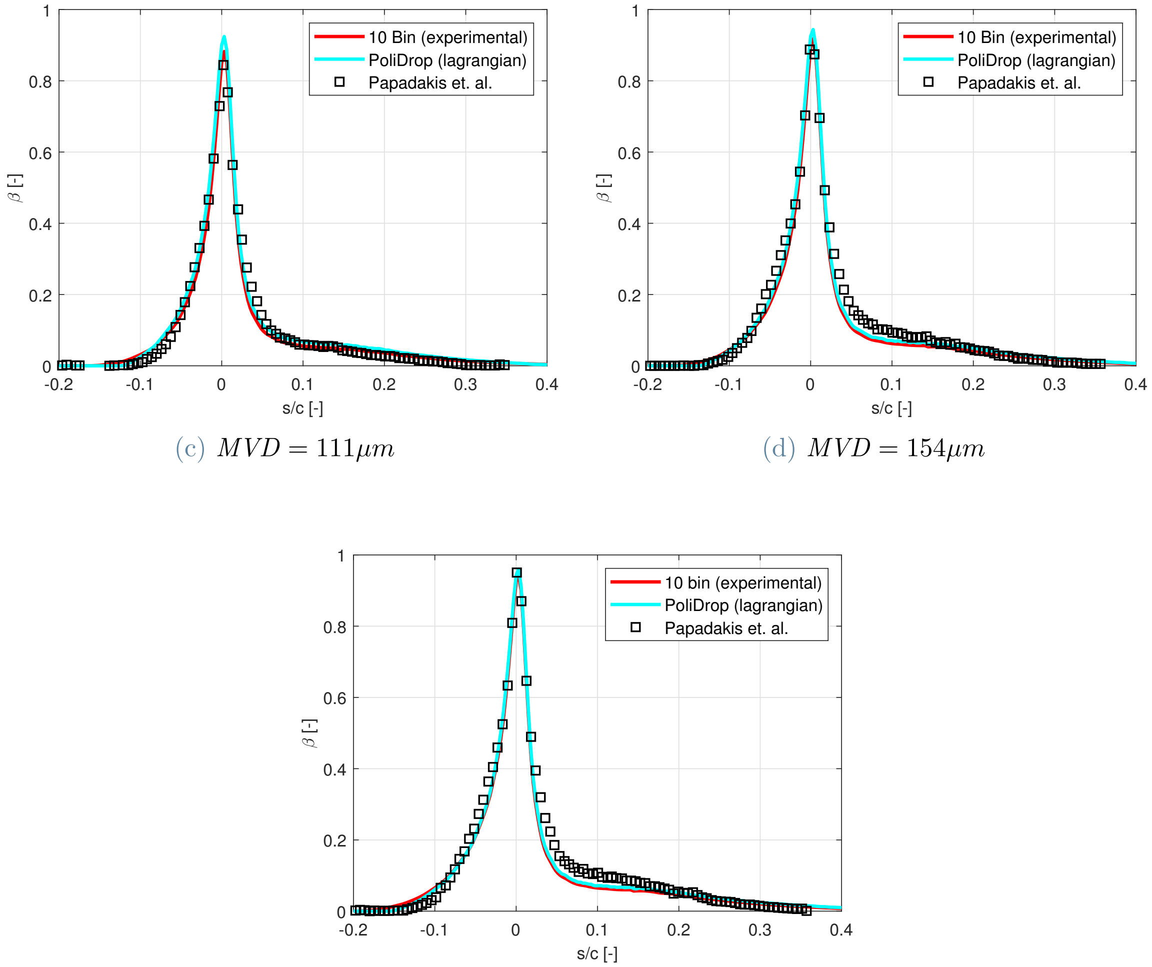
Figure 3.8: Collection efficiency - NACA 23012 [24] - MVD vs 10 bin (experimental)

When the MVD is small, impingement limits are slightly extended for multibin simulations, with a lower peak. For higher MVD values the difference between MVD and multibin simulations decreases, as the droplet behavior becomes more and more ballistic (straight trajectories).



(a) $MVD = 20\mu m$

(b) $MVD = 52\mu m$



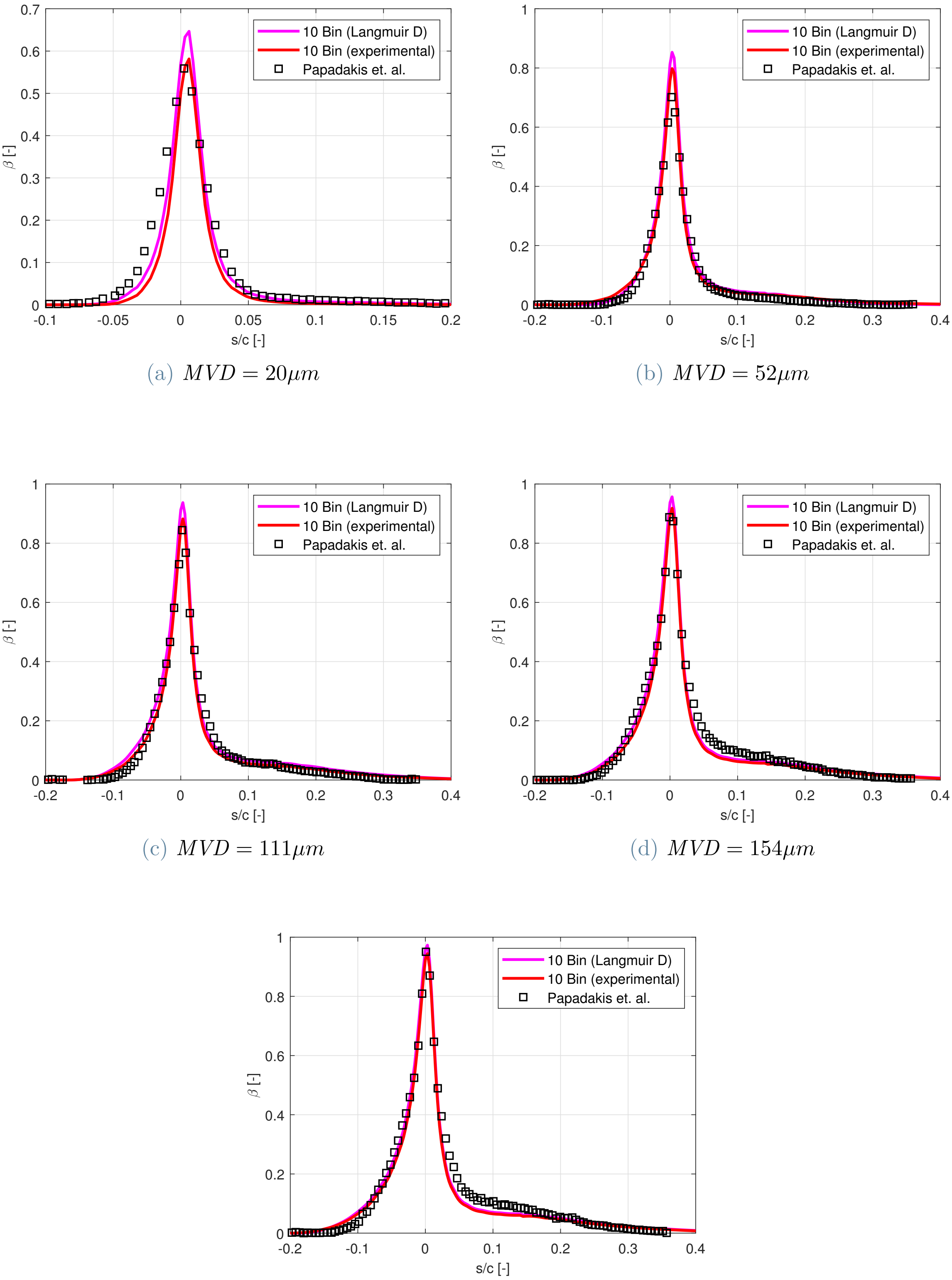
(c) $MVD = 111\mu m$

(d) $MVD = 154\mu m$

(e) $MVD = 236\mu m$

Figure 3.9: Collection efficiency - NACA 23012 [24] - PoliDrop (Lagrangian) vs 10 bin (experimental)

When compared to the Lagrangian solver in Figure (3.9), the multibin simulations yield significantly smoother results for lower MVD values. This is especially true at the impingement limits where PoliDrop goes to $\beta = 0$ sharply while the Eulerian multibin simulations are much closer to the smooth behavior of experimental data.



(a) $MVD = 20\mu m$

(b) $MVD = 52\mu m$

(c) $MVD = 111\mu m$

(d) $MVD = 154\mu m$

(e) $MVD = 236\mu m$

Figure 3.10: Multibin collection efficiency - NACA 23012 [24] - 10 bin (Langmuir D) vs 10 bin (experimental)

Since the experimental distribution is not always available it is useful to test the experimental 10 bin distribution simulations against the automatically computed Langmuir D bin distribution. For this purpose the feature has been enabled with exactly 10 bins in order to have an apples to apples comparison. The results are found in Figure (3.10) with the only notable difference being the slightly higher β peak and overall wider collection efficiency. This means that using a *non exact* bin distribution is still better than just using the MVD.

3.3.1. Computational Time

As discussed in (3.3) using a multibin approach yields better results than using the MVD. This comes at a computational cost that is linear with the number of bins since each droplet bin needs to be computed independently. The cost is slightly higher for small MVD values, as smaller droplet diameters cause some trouble in the convergence of the solver. Given the comparable results obtained from Eulerian and Lagrangian MVD simulations seen in Figure (3.7), the Eulerian solver has an advantage that grows as droplet size grows. The growth in computational cost for Lagrangian simulations is due to the requirement for more droplets in order to obtain a smooth collection efficiency. The adaptation technique employed in PoliDrop takes care of this automatically. If this adaptation needed to be performed manually, the total computational cost (human + machine) would be much bigger. This highlights both the usefulness of the cloud adaptation technique used in PoliDrop, and the *set it and forget it* nature of the Eulerian solver.

It should be noted that it doesn't matter how the bins have been computed or how they've been constructed, the computational time will be comparable.

All simulations have been performed on an *i7 9750h 6c* $@2.5GHz$ using open MPI [12] for parallelization (for both PoliDrop and the Eulerian solver).

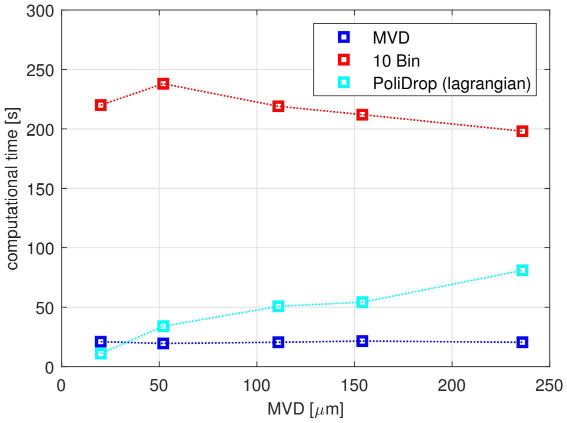


Figure 3.11: Computational time - NACA 23012 [24] - MVD vs 10 bin vs PoliDrop (Lagrangian) - *i7 9750h 6c @2.5GHz*

3.4. Three Element Airfoil

A case of high importance in the in-flight icing field is that of multi element airfoils. This is because having more than one surface when tracking particles in an Eulerian fashion can prove tricky.

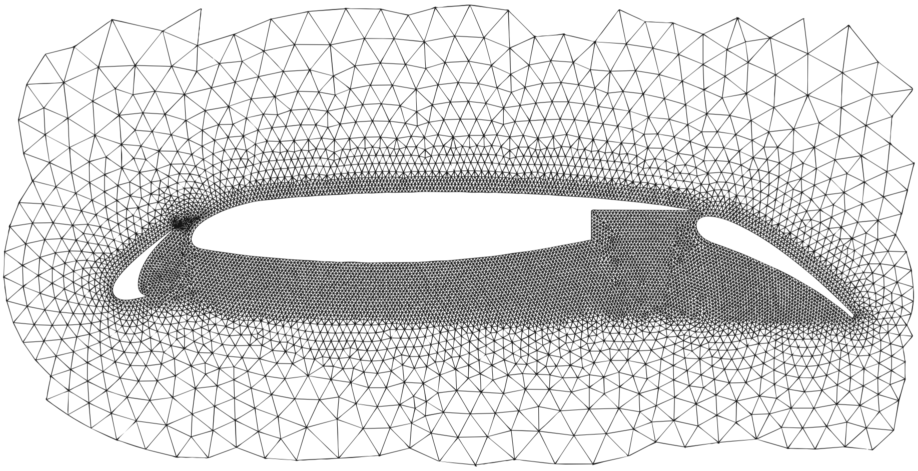
To test the Eulerian solver in this scenario, the cases 121 & 122 from the 2021 AIAA first in-flight icing workshop [1] have been simulated using:

1. MVD
2. multibin 6 bin Langmuir D distribution
3. multibin 27 bin experimental distribution provided by the AIAA [1]

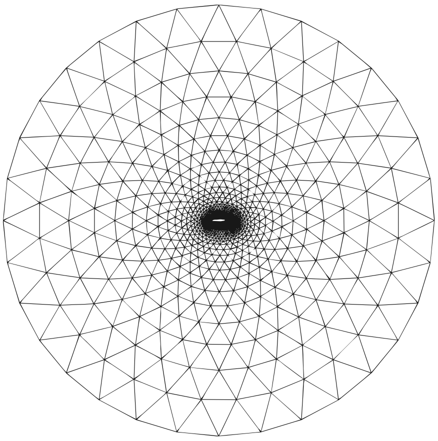
The experimental setup is in (3.4) while the computational mesh can be seen in Figure (3.12).

Airfoil	Three Element Airfoil
	$chord = 0.914\ m$
	$\delta_{slat} = 30°$
	$AoA = 4°$
	$\delta_{flap} = 30°$
Cloud & Droplets	$LWC = 0.5\ g/m^3$
	$MVD = 21\ \mu m \qquad (Case\ 121)$
	$MVD = 92\ \mu m \qquad (Case\ 122)$
	$\mu_p = 0.0011208\ Pa/s$
	$\rho_p = 1000\ kg/m^3$
	$\sigma_p = 0.074\ N/m$
Airflow	EULER
	$M_\infty = 0.233628$
	$P_\infty = 95630\ Pa$
	$T_\infty = 278\ K$

Table 3.4: Experimental setup data - three element airfoil AIAA case 121 & case 122 [1]



(a) Close up



(b) Farfield

Figure 3.12: Meshes - three element airfoil AIAA case 121 & case 122 [1]

The flow-field computed and used for all the following droplet simulations, both Eulerian and Lagrangian, is depicted as reference in Figure (3.13).

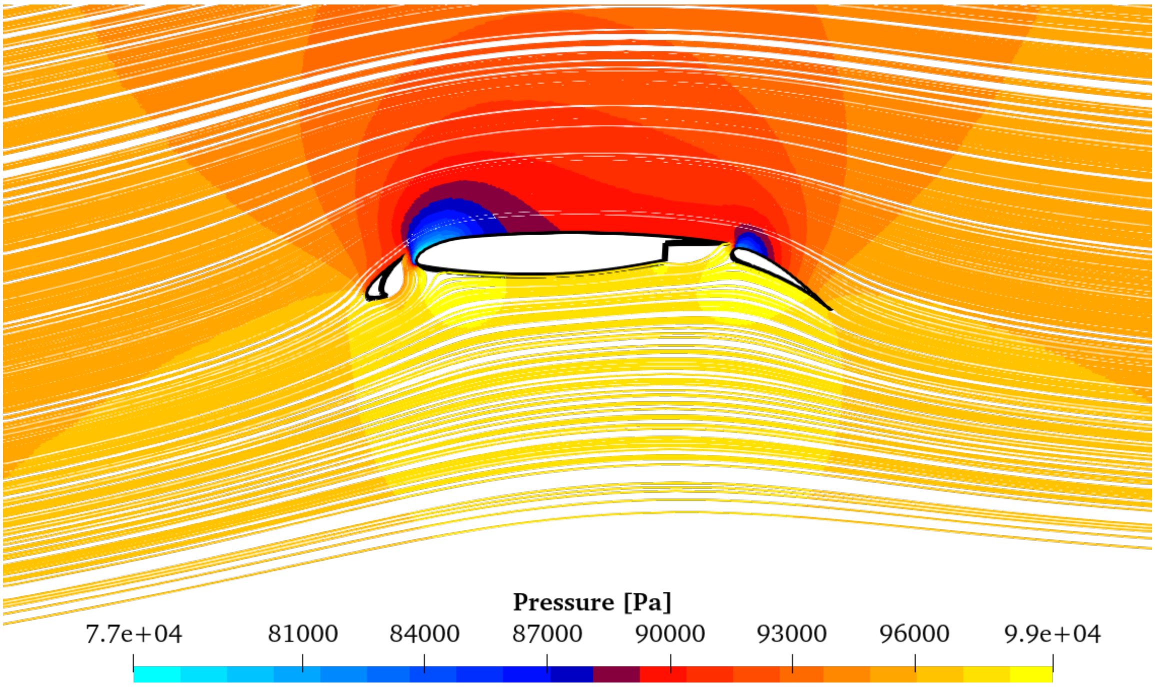
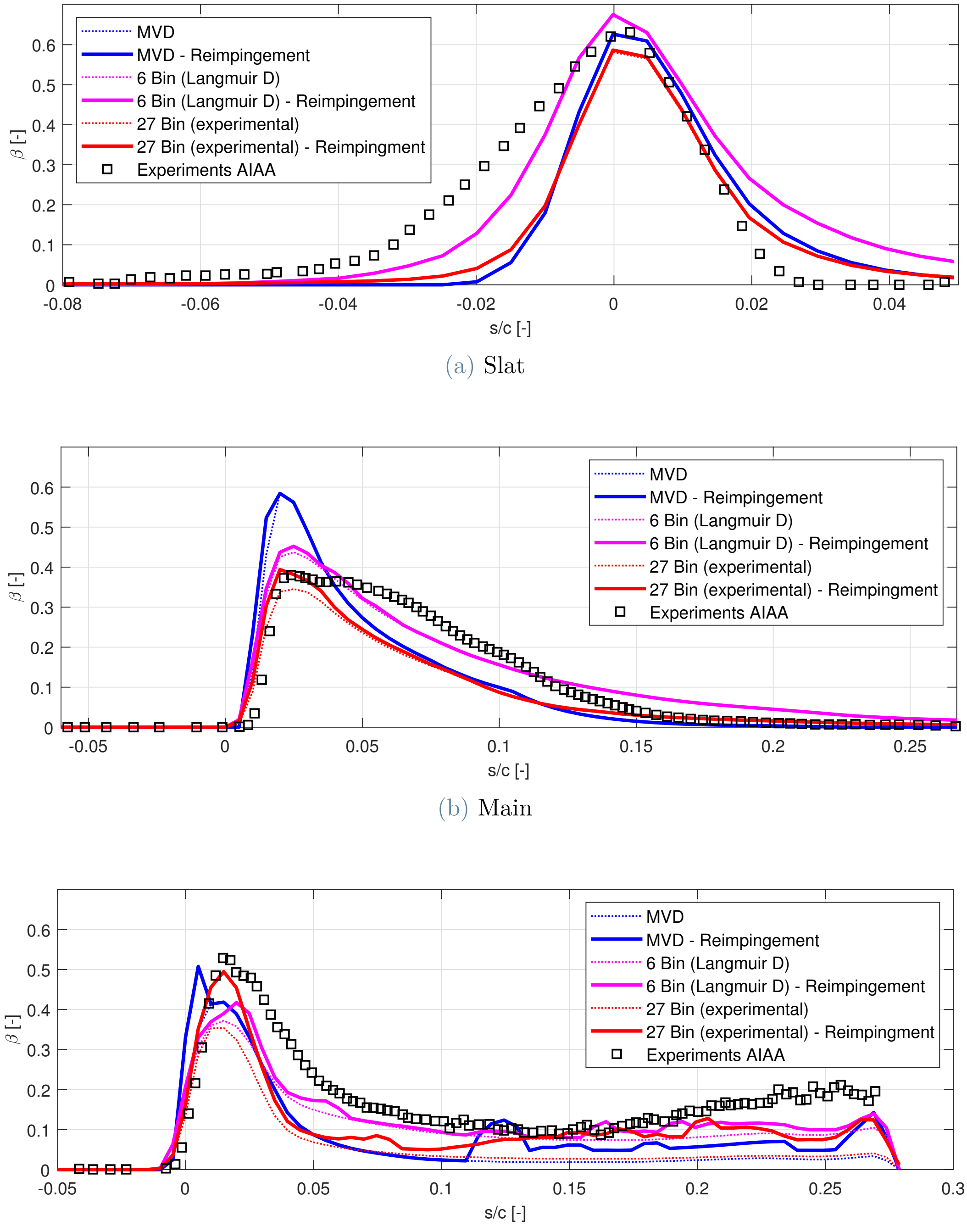


Figure 3.13: Euler flow-field - three element airfoil AIAA case 121 & case 122 [1]

In Figure (3.14) the three Eulerian simulations results (*MVD*, 6 bin Langmuir D and 27 bin experimental) are compared with experimental data from [23] for all three lifting surfaces. It's easy to see that, as before, the multibin approach yields the best results. What's more important is that using a 27 bin *exact* distribution is not necessarily better than computing an approximate 6 bin distribution, if reimpingment is not considered. Once the Lagrangian reimpingement is enabled the 27 bin results become by far the more accurate when compared to the experiments.



(a) Slat

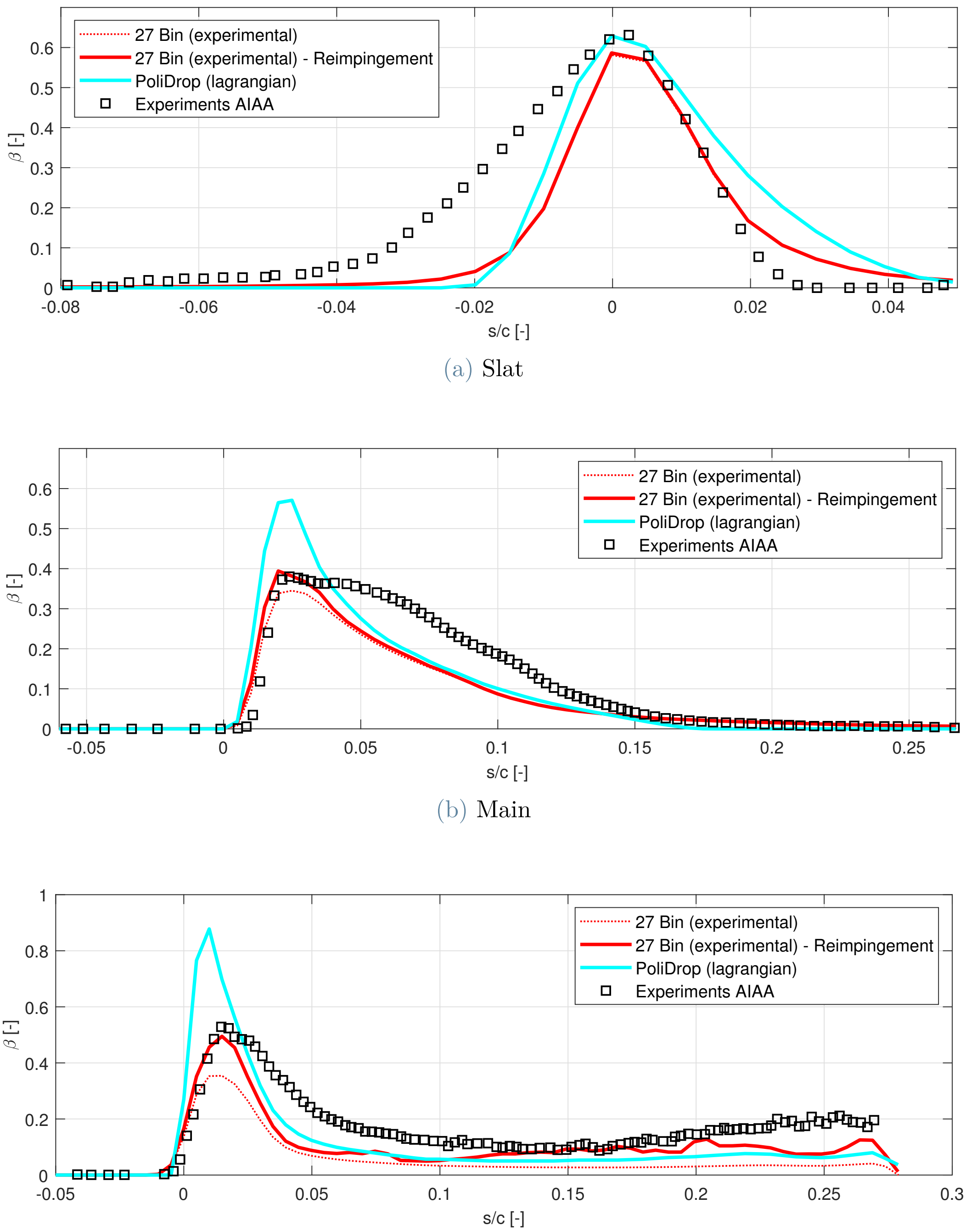
(b) Main

(c) Flap

Figure 3.14: Collection efficiency with and without reimpingement - three element airfoil AIAA case 121 ($MVD = 21\mu m$) [1] - MVD vs 6 bin (Langmuir D) vs 27 bin (experimental)

The slat collection efficiency is badly predicted. This is the case also for PoliDrop in

Figure (3.15a) and many of the participants at the AIAA 1st Icing Prediction Workshop [1]. The reason is not clear, one possibility is that of a bad angle of attack measurement in the setup of the experiments.



(a) Slat

(b) Main

(c) Flap

Figure 3.15: Collection efficiency with reimpingement - three element airfoil AIAA case 121 ($MVD = 21\mu m$) [1] - 27 bin (experimental) vs PoliDrop (Lagrangian)

Comparing the 27 bin distribution Eulerian simulations against a Lagrangian PoliDrop simulation in Figure (3.15) shows how much closer to the experimental results the simulations get by considering both polydispersity and reimpingement. This is most visible on the flap in Figure (3.15c), where the peak grows by $\sim 30\%$ and matches almost perfectly the experimental data. The same is true for the rest of the flap's results. In (3.5) the Lagrangian reimpingement step will be examined more in depth.

3.5. Lagrangian Reimpingement

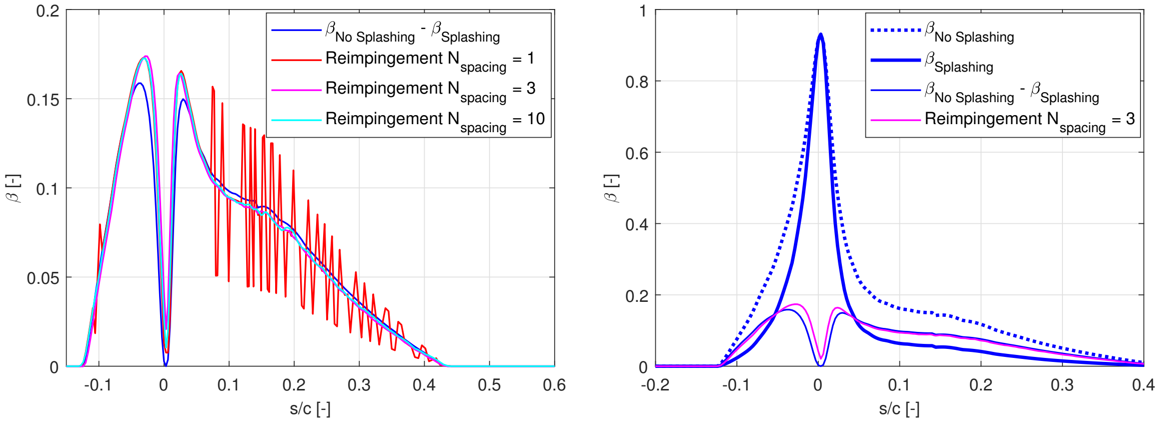
The final step of the Eulerian solver is the Lagrangian tracking of the splashed droplet. This step is extremely dependent on the use of multibin and is important in the SLD regime. This is because a bin distribution yields a much more heterogeneous set of splashed droplets as can be seen in (**??**).

When only using the MVD, droplets are splashed in fewer positions and each splashing position (one per surface mesh element where splashing occurs) only has one set of $d_{p,s}$, $\underline{\boldsymbol{u}}_{p,s}$ and n_s. This means the overall behavior will be "monochromatic".

On the other hand, using 27 bins yields an extremely diversified distribution of splashed droplets with many different droplet diameters. These droplets are also spread out across the whole surface of the airfoil with a much more realistic representation of physical splashing, which is not localized but happens almost everywhere due to polydispersity.

3.5.1. Mass Conservation & Spacing Validation

As a first validation step of the Lagrangian reimpingement, the droplets have been reinjected directly into the airfoil, at the exact location where they splashed. What one would expect is that the reimpingement collection efficiency is exactly the difference between the uncorrected β and the splashing corrected β. This is also a good way to ensure the spacing of the droplets (Figure (2.10)) increases the quality of the collection efficiency.



(a) Reimpingement with different spacings

(b) Reimpingement vs Splashing Correction

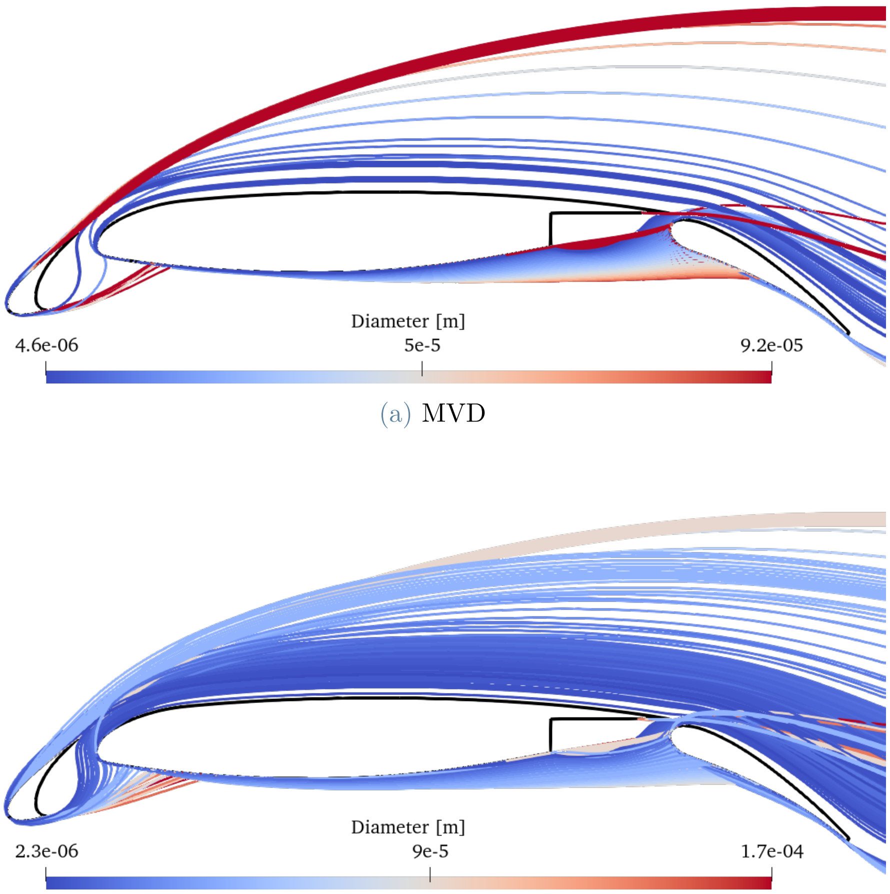
Figure 3.16: Comparison of different Lagrangian reinjection spacings and comparison of splashing collection efficiency correction with reinjected collection efficiency

The slight difference in Figure (3.16) is given by rounding errors in the computation of integer numbers of droplets, therefore mass is (roughly) conserved. It is self evident that reinjecting more than one parcel yields significantly better results. This does not necessarily mean that spacing removes all oscillations in β: sometimes results will be oscillatory due to the formation of actual physical "trains" of droplets that reimpinge in similar locations, and not due to the "sparsity" of Lagrangian parcels.

3.5.2. Three Element Airfoil SLD

To validate the Lagrangian reimpingement step, Case 122 (Table (3.4)) of the 2021 AIAA Icing Workshop [1] has been used. This is an identical case to the case 121, but with a larger droplet size ($MVD = 92\mu m$ against $MVD = 21\mu m$) therefore the reimpingement is expected to be larger.

The more heterogeneous nature of the splashed droplets in a multibin simulation is evident when looking at the Lagrangian trajectories of the reimpinging droplets in Figure (3.17). This diversity is better captured when using a multibin approach since every bin being simulated generates its own splashed droplet set for each surface element.



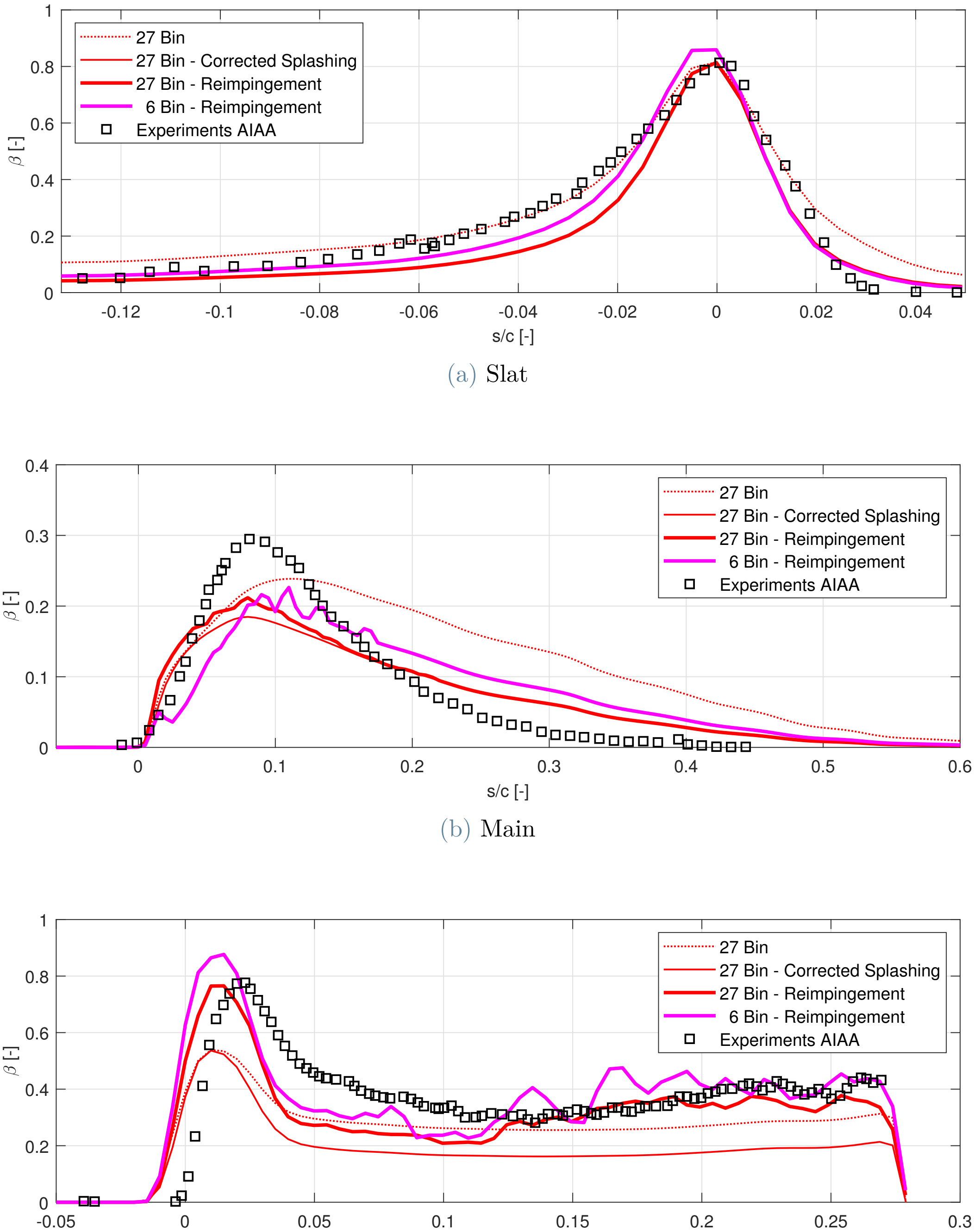
(a) MVD

(b) multibin 6 bin (Langmuir D)

Figure 3.17: Reinjected droplets trajectories - three element airfoil AIAA case 122 ($MVD = 92\mu m$) [1] - MVD vs 6 bin (Langmuir D)

Figure (3.18) highlights the nature of the reimpingement correction, almost solely localized within the flap's pressure side. This correction allows for a greater accuracy in the collection efficiency results on the flap, almost matching the experimental data. The "oscillatory" behavior in the 6 bin results is due to the formation of "trains" of splashed droplets that follow similar trajectories and reimpinge in similar locations. The most diverse set of splashed droplets in the 27 bin simulation allows for a much smoother reimpingement correction in Figure (3.18b) and (3.18c).

The same case has been run with the experimental 27 bin distribution in Figure (3.18), and the collection efficiency prediction quality for the flap suffers. But, if Lagrangian reimpingement is also considered, then the results of the 27 bin simulations are the best obtained so far.



(a) Slat

(b) Main

(c) Flap

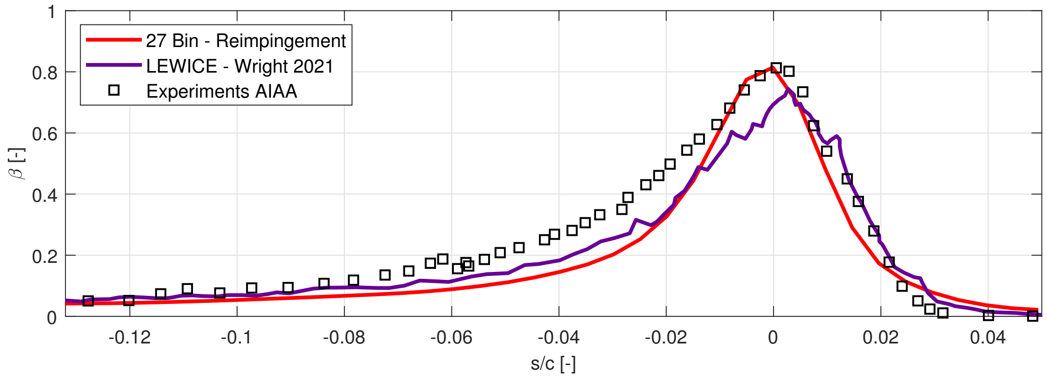
Figure 3.18: Collection efficiency three element airfoil AIAA case 122 ($MVD = 92\mu m$) [1] - complete contribution decomposition (compute impinging, remove splashed, add reimpinged) - 27 bin (experimental) vs 6 bin (Langmuir D)

The biggest variable in the Lagrangian reimpingement step is the splashing model. There is no physically universal splashing model that can be used without care. The Wright and Potapczuk model [27] employed in this work has been created following a "trial and error" approach, searching for coefficients that minimized the difference between the computed and experimental collection efficiency.

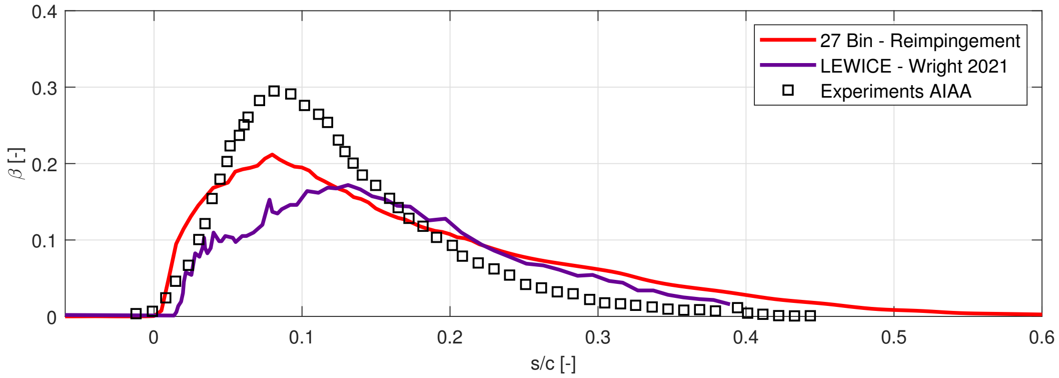
This means it can misbehave when not being used on cases that are extremely similar to the ones against which it was tested, with the risk of actually degrading the quality of the results. When evaluating in-flight icing collection efficiency cases, this model yields great results.

Another possible source of "non robustness" of this step is that the number of splashing droplets is directly linked to the number of mesh elements at the surface (1 droplet set per mesh element, per bin). Therefore, the surface mesh has to be fine enough for the reimpingement collection efficiency to be smooth, and this is a trial and error procedure. Sometimes the spacing technique described in (2.4) helps. A possible solution could be found in the adaptation technique used in PoliDrop. This would need to be re-adapted to the heterogeneous nature of the splashed droplet set.

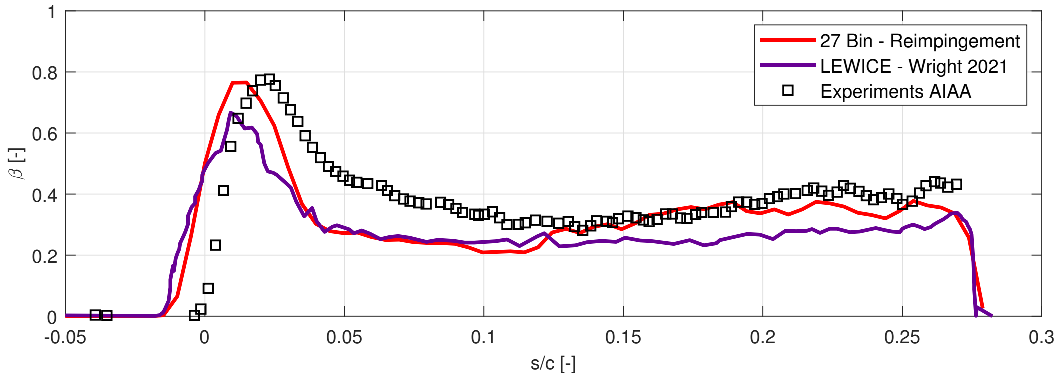
A final comparison to LEWICE collection efficiency results from Wright's presentation [28] at AIAA's 1st Icing Prediction Workshop [1] is present here in Figure (3.19). A general increase in solution quality can be seen across all three surfaces, in particular the flap where the bulk of the reimpingement contribution is found.



(a) Slat



(b) Main



(c) Flap

Figure 3.19: Collection efficiency - three element airfoil AIAA case 122 ($MVD = 92\mu m$) [1] - 27 bin (experimental) with reimpingement vs LEWICE 27 bin experimental [28]

3.5.3. Computational Time

The complex geometry of the three element airfoil case causes the pressure-less gas dynamics exact Riemann solver to struggle with convergence. This means the computational time is no longer a definite advantage of the Eulerian solver since the Lagrangian solver is basically non dependent on the geometry at hand.

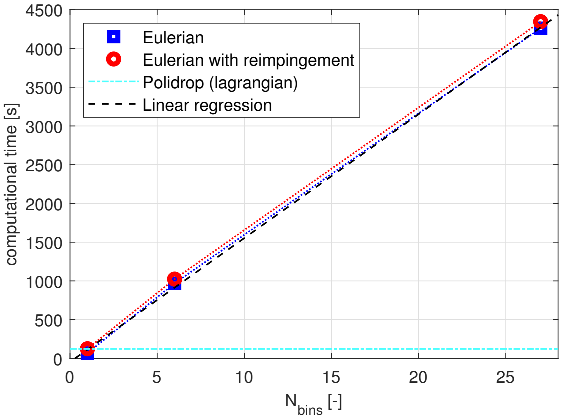


Figure 3.20: Computational time - three element airfoil AIAA case 121 ($MVD = 21\mu m$) [1] - Eulerian (with and without reimpingement) vs PoliDrop (Lagrangian) - *i7 9750h 6c* @2.5GHz

The Lagrangian reimpingement step has a semi-fixed cost. This, combined with the increased diversity in the splashed droplet set, means that the Lagrangian reimpingement step becomes more viable as the number of bins increases.

Different convective flux discretizations could yield better convergence behavior than the exact Riemann solver employed in this work, and this is one important way this solver could be developed in future work.

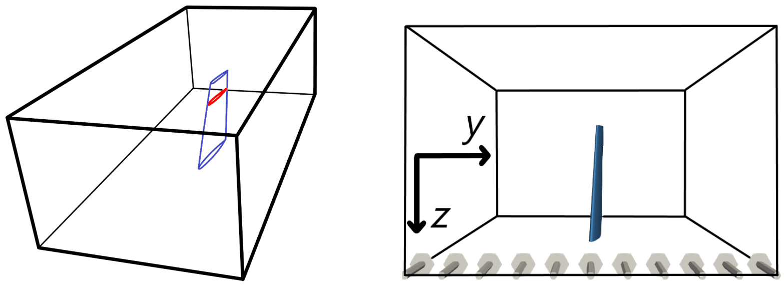
3.6. Swept Tail 3D

For the sake of completeness, two 3D test cases have been run: the case 111 and case 112 from AIAA's first Icing Prediction Workshop [1]. These are three dimensional simulations of a swept horizontal tail (NACA 64A008) [23] in a wind tunnel. The details are found in Table (3.5).

Airfoil	NACA 64A008
	$Mean\ Aerodynamic\ Chord = 0.95631\ m$
	$AoA = 6°$
Cloud & Droplets	$LWC = 0.5\ g/m^3$
	$MVD = 21\ \mu m \quad (Case\ 111)$
	$MVD = 92\ \mu m \quad (Case\ 112)$
	$\mu_p = 0.0011208\ Pa/s$
	$\rho_p = 1000\ kg/m^3$
	$\sigma_p = 0.074\ N/m$
Airflow	EULER
	$M_\infty = 0.23$
	$P_\infty = 83025\ Pa$
	$T_\infty = 280\ K$

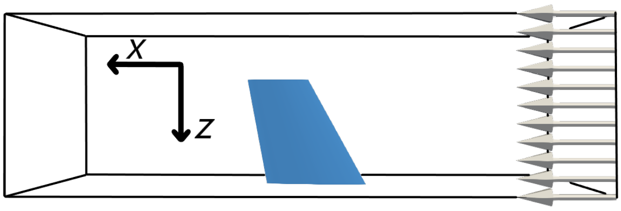
Table 3.5: Experimental setup data - case 111 & 112 [23] AIAA 1st Icing Prediction Workshop [1]

The unstructured mesh made with GMSH [13] is composed of 1749838 tetrahedra and 361409 nodes. The simulations have been run on a consumer laptop CPU *i7 9750h 6c* @3.5GHz with 16GB of RAM.



(a) Collection efficiency measurement location (in red) at $z = 0.9144m$ from the wind tunnel floor

(b) - x-normal domain view



(c) - y-normal domain view

Figure 3.21: Geometry, domain and collection efficiency measurement location - case 111 & 112 [23] AIAA 1st Icing Prediction Workshop [1]

The mesh, and vacuum zone resolution can be appreciated in Figure (3.22). More visualizations are available in appendix (E).

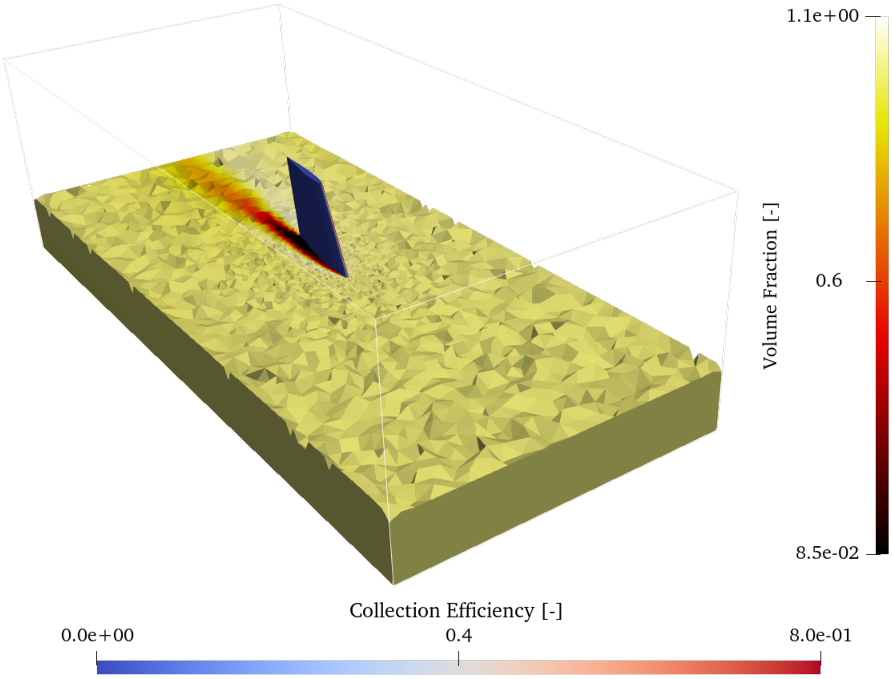


Figure 3.22: Mesh and α slice of case 112 ($MVD = 92\mu m$) [23] AIAA 1st Icing Prediction Workshop [1]

To check the mesh and airflow solution quality the pressure coefficient is compared to experimental data [23] at $z = 1.0922m$ from the wind tunnel floor. There's great agreement with experiments as shown in Figure (3.23).

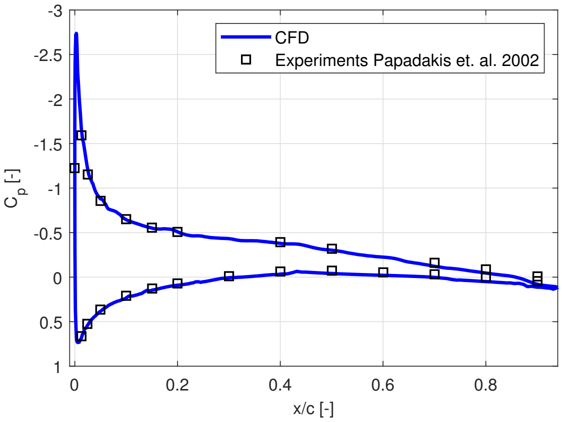
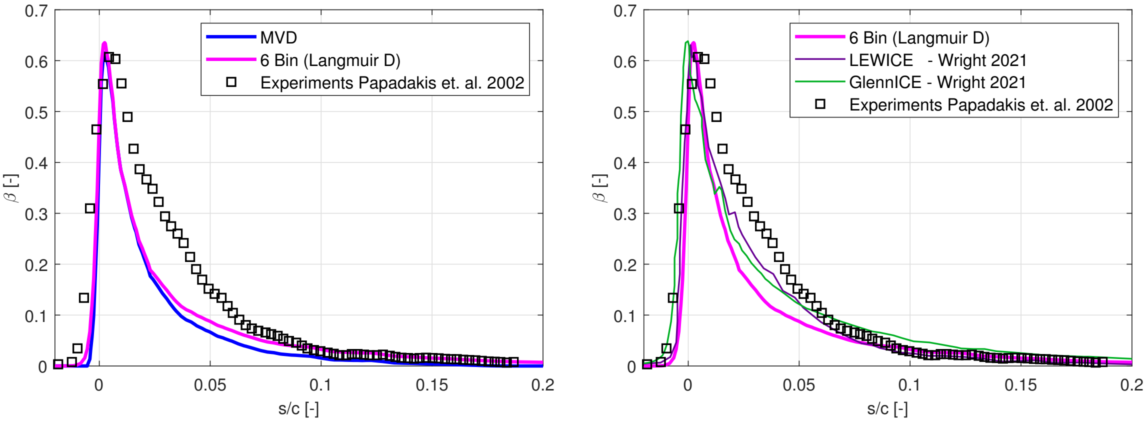


Figure 3.23: Pressure coefficient at $z = 1.0922m$ from wind tunnel floor - case 111 & 112 [23] AIAA 1st Icing Prediction Workshop [1]

The collection efficiency results are compared between an *MVD* simulation and a 6 bin automatically computed Langmuir D droplet size distribution. Additionally some results have been taken from NASA's presentation [28] at the AIAA 1st Icing Prediciton Workshop [1], for both LEWICE and GlennICE from Wright [28] to compare with different software suites in Figure (3.24).



(a) MVD vs 6 Bin

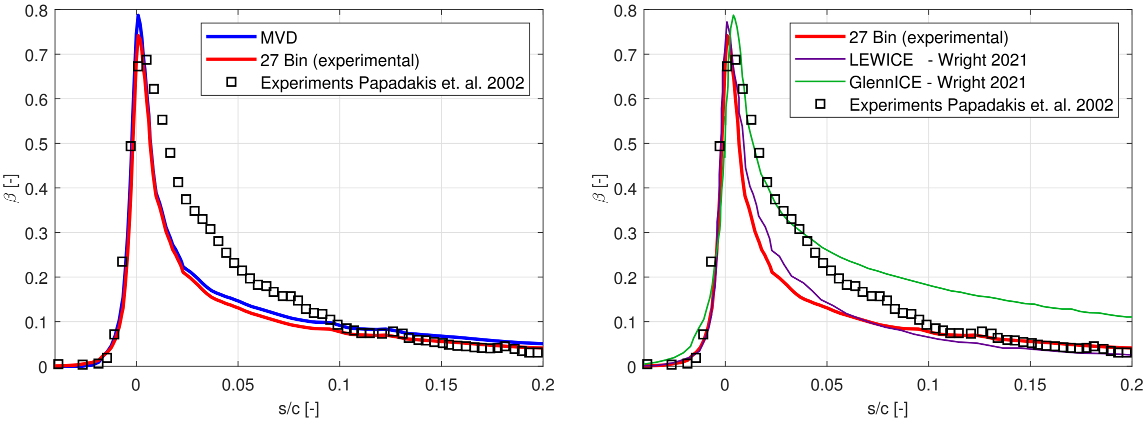
(b) 6 Bin vs LEWICE vs GlennICE

Figure 3.24: Collection efficiency at $z = 0.9144m$ from wind tunnel floor - case 111 ($MVD = 21\mu m$) [23] - MVD vs 6 bin (Langmuir D) vs LEWICE 27 bin experimental [28] vs GlennICE 27 bin experimental [28]

The agreement between all the results is within measurement uncertainty in the low *MVD* case 111. The multibin simulation has a smoother behavior at $s/c \simeq 0$ compared to the sharp *MVD* results, and is overall closer to experiments than the *MVD* results.

The peak is almost in exact agreement between all simulations and other softwares shown in (3.24).

When comparing the SLD case 112 (with $MVD = 92\mu m$) in Figure (3.25) things don't change significantly. LEWICE and the Eulerian solver are in good agreement, while GlennICE achieves greater accuracy close to the peak, while over predicting far from it.

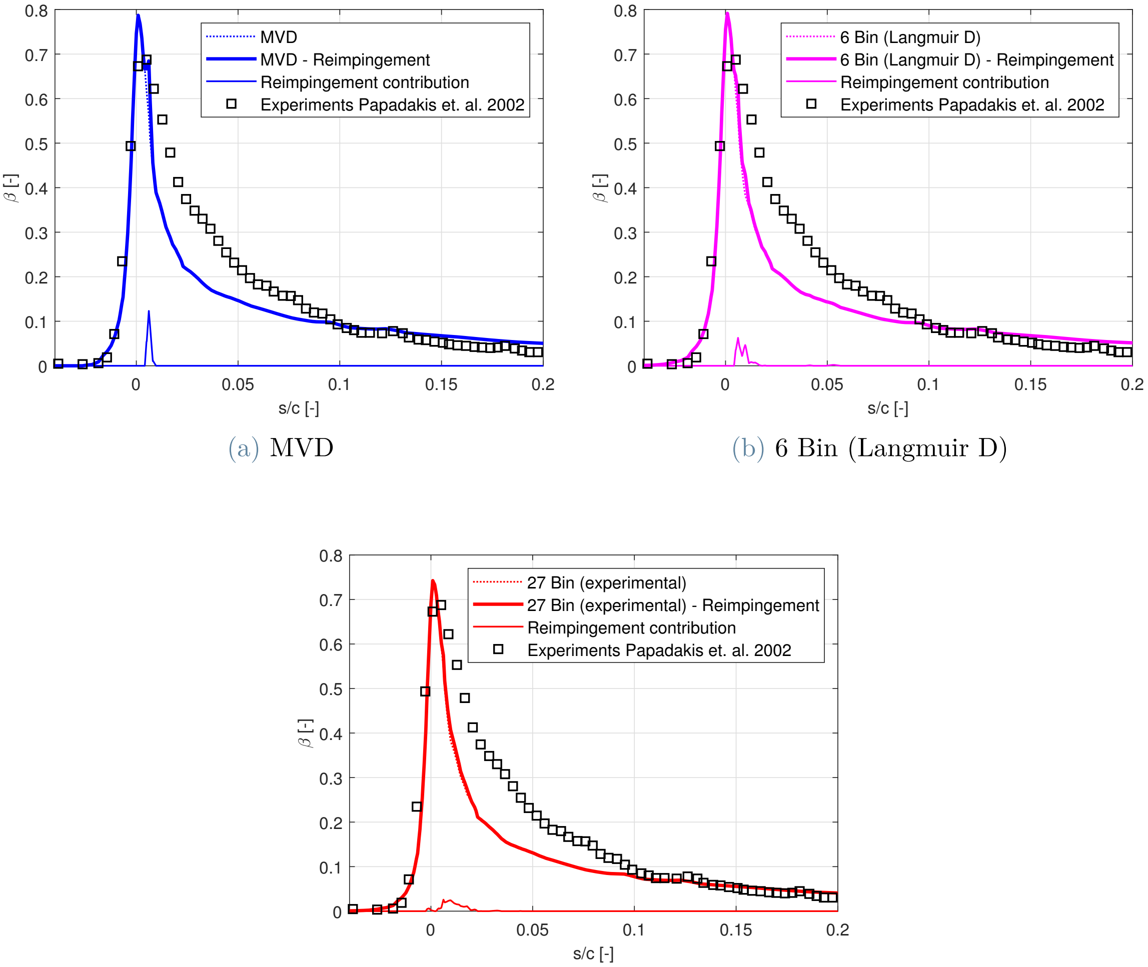


(a) MVD vs 6 Bin

(b) 27 Bin vs LEWICE vs GlennICE

Figure 3.25: Collection efficiency at $z = 0.9144m$ from wind tunnel floor - case 112 ($MVD = 92\mu m$) [23] - MVD vs 27 bin (experimental) vs LEWICE 27 bin experimental [28] vs GlennICE 27 bin experimental [28]

In this single element case, the Lagrangian reimpingement step has been run, with sparse particles reimpinging after being reinjected. The need for a three dimensional spacing approach, or an adaptation technique is required in order to achieve better results in three dimensional simulations. This is even more evident looking at how many droplets are lost in the wake in Figure (3.27), but still need to be tracked one by one even when they are not an icing threat anymore. The computational resources spent tracking these would be better spent adding more parcels where droplets reimpinge on the airfoil.



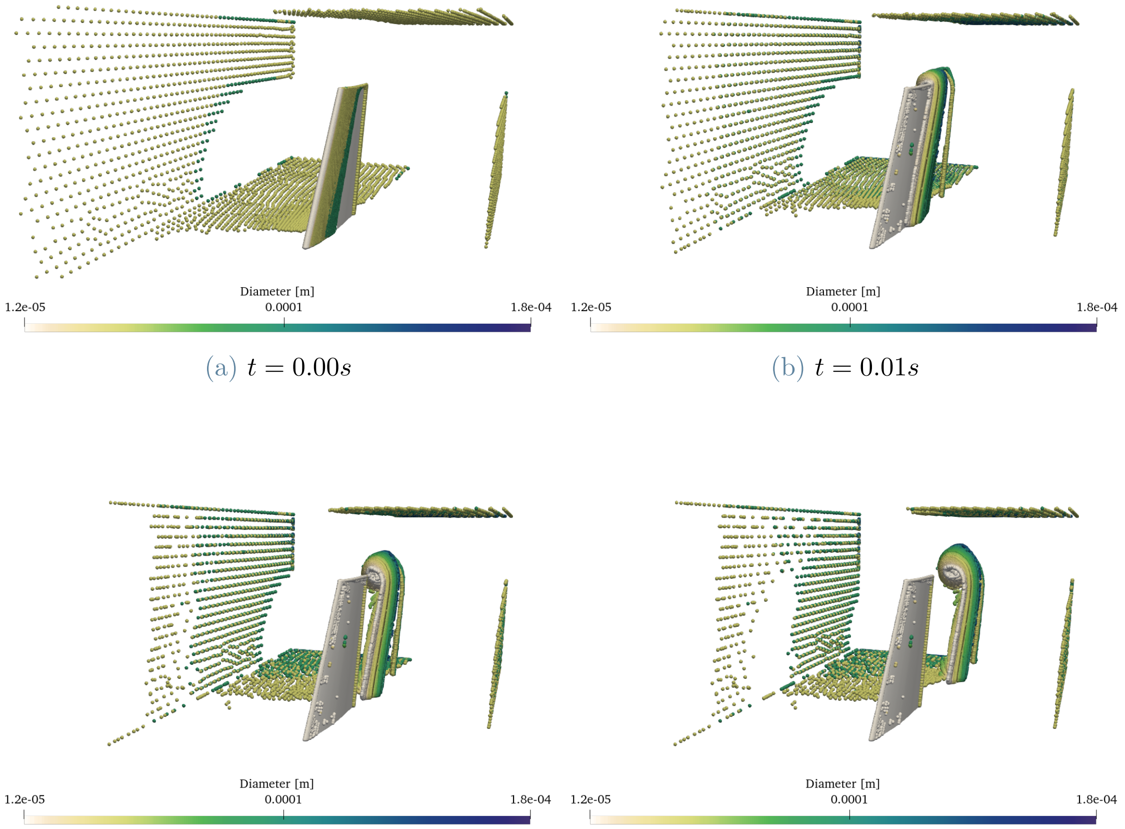
(a) MVD

(b) 6 Bin (Langmuir D)

(c) 27 Bin (experimental)

Figure 3.26: Reimpingement collection efficiency at $z = 0.9144m$ from wind tunnel - floor case 112 ($MVD = 92\mu m$) [23] - MVD vs 6 bin (Langmuir D) vs 27 bin (experimental)

The reimpingement contribution gets smoother when increasing the number of bins. In this single element geometry, the three dimensionality of the flow generates some reimpingement (in 2D none occurs), but it's a small correction. In the 27 bin simulation, the maximum percentage increase in the collection efficiency due to reimpingement is 3.5%, while in the MVD simulation it would probably be best not to include it. The location and total amount of water mass in the reimpingement contribution is roughly the same for all 3 simulations, indicating that a sort of "smoothing" postprocessing of this contribution could yield good results also in the MVD simulation.



(a) $t = 0.00s$

(b) $t = 0.01s$

(c) $t = 0.02s$

(d) $t = 0.03s$

Figure 3.27: Splashed droplets reinjection - case 112 ($MVD = 92\mu m$) [23] - 6 bin (Langmuir D)

3.6.1. Computational Time & Convergence

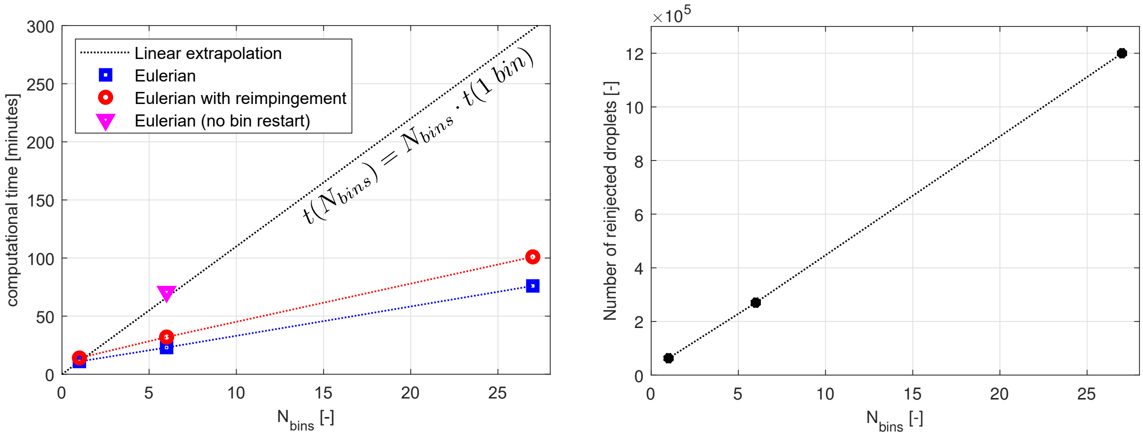
The Eulerian solver proved acceptably stable, since all bins converged using a CFL number equal to 2, in spite of the large difference in the droplet field behavior across different droplet sizes. All bins converged in around 200 iterations to a RMS residual of α of 10^{-7}, with most of its variation being confined in the wake, while the residual is much lower around the airfoil and in the incoming flow. The computational times on 16GB of RAM and a consumer laptop CPU *i7 9750h 6c* @3.5GHz are in Table (3.6) for the Eulerian solver and in Table (3.7) for the Lagrangian reimpingement step.

	MVD	**6 Bin** (Langmuir D)	**27 Bin** (experimental)
Case 111, $21\mu m$	15 *minutes*	32 *minutes*	-
Case 112, $92\mu m$	11 *minutes*	23 *minutes*	76 *minutes*

Table 3.6: Computational times Eulerian solver in 3D - case 111 ($MVD = 21\mu m$) & 112 ($MVD = 92\mu m$) [23] - MVD vs 6 bin vs 27 bin - *i7 9750h 6c* @3.5GHz

	MVD	**6 Bin** (Langmuir D)	**27 Bin** (experimental)
Computational Time	3 *minutes*	9 *minutes*	25 *minutes*
Number of Parcels	$6.3 \cdot 10^4$	$2.7 \cdot 10^5$	$1.2 \cdot 10^6$

Table 3.7: Computational times of Lagrangian reimpingement and number of reinjected droplets with varying number of bins in 3D - case 112 ($MVD = 92\mu m$) [23] - MVD vs 6 bin vs 27 bin - *i7 9750h 6c* @3.5GHz



(a) Computational time

(b) Number of reinjected droplets

Figure 3.28: Computational times and number of reinjected droplets with varying number of bins - case 112 ($MVD = 92\mu m$) [23] - *i7 9750h 6c* @3.5GHz

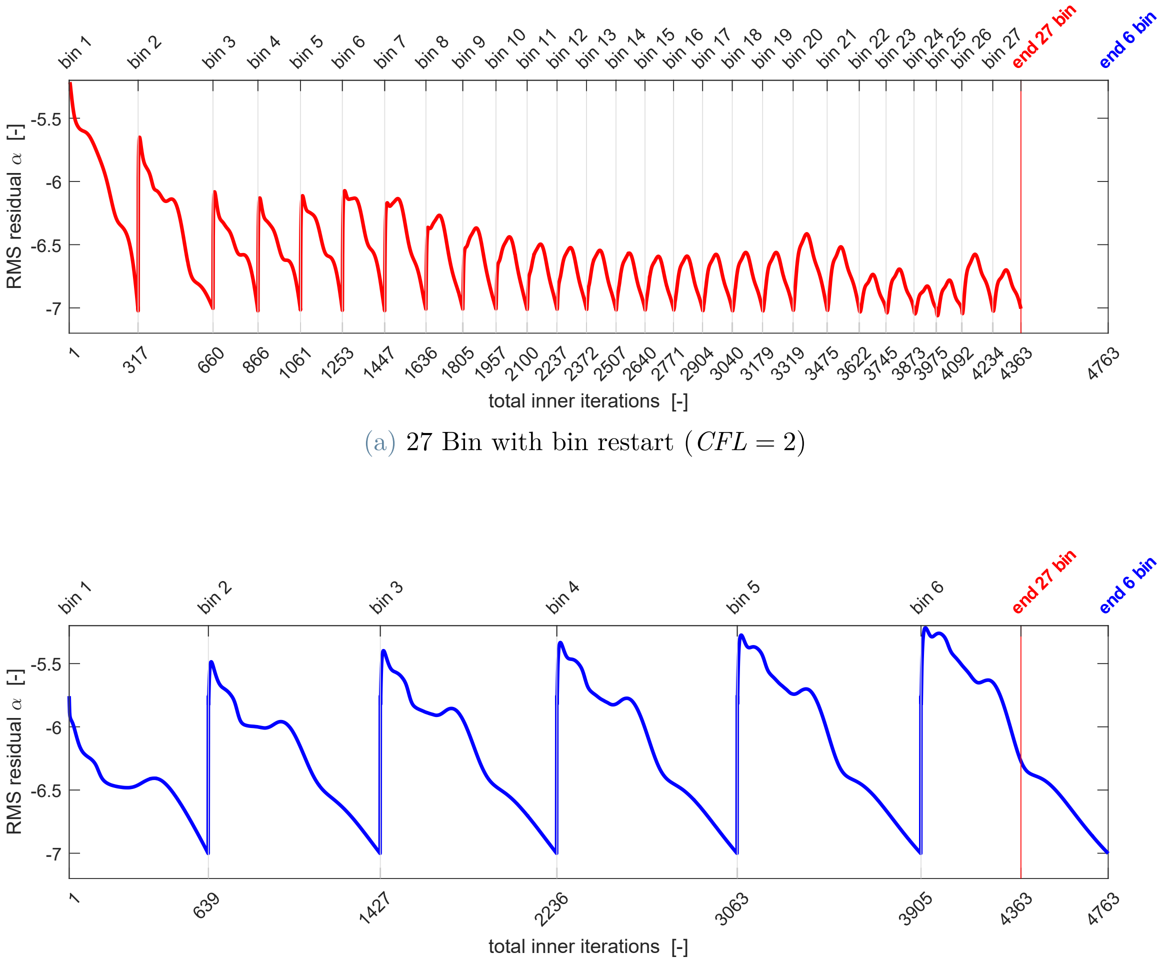
Figure (3.28) shows how using the multibin restarting technique discussed in (2.3.3) has a profound impact on the computational cost of high bin number simulations. The linearity of the computational cost with the number of bins (e.g.

$t_{comp}(N_{bins}) = N_{bins} \cdot t_{comp}(1_{bin})$) does not hold anymore, as Figure (3.28a) shows best. The computational time of the Eulerian solver alone is cut down by 65% with respect to running each bin singularly.

This happens for two main reasons:

1. starting from a better initial guess allows for convergence with higher *CFL* numbers (in this case 2 instead of 1).
2. even if the *CFL* number is identical, all bins after the first have to iterate less to achieve convergence since they start from a solution that is already close to the exact one. This becomes more and more true as subsequent bins become closer in droplet diameter.

It is very clear in Figure (3.29), where a 6 bin simulation without bin restart (Figure (3.29b)) ends up taking more time than a 27 bin one with bin restart (Figure (3.29a)).



(a) 27 Bin with bin restart ($CFL = 2$)

(b) 6 Bin without bin restart ($CFL = 1$)

Figure 3.29: Convergence behavior of multibin simulations with and without bin restart - Root Mean Square residual α - case 112 ($MVD = 92\mu m$) [23]

4 | Conclusions

The goal of this work has been to develop and complete a three dimensional, Euler-Euler droplet tracking solver with MUSCL and a Lagrangian reimpingement step for in-flight icing purposes. To this end, the Pressureless Gas Dynamics conservation laws and their character has been discussed, and then, in a effort to retrieve strict hyperbolicity, a relaxation strategy has been chosen from the literature. The final set of conservation laws has been spatially discretized using a node centered finite volume approach and the convective fluxes have been evaluated through an exact Riemann solver, employing also a MUSCL reconstruction.

A splashing & rebounding model has been used to account for the mass loss in the collection efficiency at solid boundaries, but also, a Lagrangian reimpingement step has been implemented in an effort to better capture the physics of droplet-wall interactions. This Lagrangian reimpingement step consists in the Lagrangian tracking of splashing & rebounding droplets that may reimpinge on aft surfaces.

Furthermore a multibin approach has been outlined to further increase the fidelity of the collection efficiency results since clouds are intrinsically heterogeneous and droplets are polydispersed. Since the cloud droplet size distribution is not known a priori, an automatic bin distribution strategy has been developed.

All of the above have been tested in 2D and 3D, comparing the results against various cases from the literature where experiments were available. The results have also been compared to an in-house Lagrangian solver (PoliDrop).

The findings indicate that the Euler-Euler approach is more robust, always yielding usable results, while the Euler-Lagrange approach requires some throwaway runs to setup correctly. When comparing single droplet size simulations, Euler-Euler and Euler-Lagrange results are very comparable.

The Lagrangian reimpingement step yields an increase in solution quality in multi element and three dimensional cases in SLD conditions. When being used in conjuction with a multibin approach the results improve further, given the more diverse set of splashed droplets being generated. The simple multibin restart implementation highlighted in (2.3.3) proved effective, yielding up to 65% less computational time.

This solver still needs to be plugged into the PoliMIce loop, in order to test the resulting ice shapes since that is the final objective of in-flight icing simulations.

4.1. Future Work

1. The set of splashed droplets being reinjected is, in number, dependent on the mesh, therefore an adaptation technique similar to the one used in PoliDrop should be investigated in order to obtain a more robust solver.
2. Different convective models should be investigated in an effort to achieve better convergence behavior when complex geometries cause discontinuities to interact with other flow features (such as in multi element configurations)
3. A more refined spacing technique than the one shown in (2.4) should be employed when reinjecting droplets
4. Development of a smoothing postprocess for the reimpingement contribution in low bin simulations could be looked into
5. Integrating the Eulerian solver into the PoliMIce loop and compare ice shapes with Lagrangian simulations and industry results

Bibliography

[1] 1st Icing Prediction Workshop, 8 2021. AIAA. Cases 111,112,121,122 used to validate the code.

[2] A. I. Aptekarev and Y. G. Rykov. Detailed description of the evolution mechanism for singularities in the system of pressureless gas dynamics. The Proceedings of the USSR Academy of Sciences, 484(6):655–658, 2019. URL https://journals.eco-vector.com/0869-5652/article/view/12804.

[3] B. Arizmendi, M. Morelli, G. Parma, M. Zocca, G. Quaranta, and A. Guardone. In-flight Icing: Modeling, Prediction, and Uncertainty, pages 455–506. Springer International Publishing, Cham, 2021. ISBN 978-3-030-60166-9. doi: 10.1007/978-3-030-60166-9_15. URL https://doi.org/10.1007/978-3-030-60166-9_15.

[4] M. Baer and J. Nunziato. A two-phase mixture theory for the deflagration-to-detonation transition (ddt) in reactive granular materials. International Journal of Multiphase Flow, 12(6):861–889, 1986. ISSN 0301-9322. doi: https://doi.org/10.1016/0301-9322(86)90033-9. URL https://www.sciencedirect.com/science/article/pii/0301932286900339.

[5] C. Bai and A. D. Gosman. Development of methodology for spray impingement simulation. SAE Transactions, 104:550–568, 1995. ISSN 0096736X, 25771531. URL http://www.jstor.org/stable/44633238.

[6] T. Bellosta. A lagrangian 3d particle tracking solver for in-flight ice accretion. Master's thesis, Politecnico di Milano, Milano, IT, Apr. 2019. URL http://hdl.handle.net/10589/146023.

[7] C. Berthon, M. Breuß, and M.-O. Titeux. A relaxation scheme for the approximation of the pressureless euler equations. Numerical Methods for Partial Differential Equations, 22(2):484–505, 2006. doi: https://doi.org/10.1002/num.20108. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/num.20108.

[8] Y. Bourgault, W. G. Habashi, J. Dompierre, and G. S. Baruzzi. A finite element method study of eulerian droplets impingement models. International

Journal for Numerical Methods in Fluids, 29(4):429–449, 1999. doi: https://doi.org/10.1002/(SICI)1097-0363(19990228)29:4<429::AID-FLD795>3.0.CO;2-F. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-0363%2819990228%2929%3A4%3C429%3A%3AAID-FLD795%3E3.0.CO%3B2-F.

[9] R. Clift, J. Grace, and M. Weber. Bubbles, drops, and particles. Dry. Technol., 11: 263–264, 01 1978.

[10] C. T. Crowe, J. D. Schwarzkopf, M. Sommerfeld, and Y. Tsuji. Multiphase Flows with Droplets and Particles. Routledge, 2011.

[11] S. Elghobashi. On predicting particle-laden turbulent flows. Applied Scientific Research, 52:309–329, 1995. ISSN 0096736X, 25771531. doi: https://doi.org/10.1007/BF00936835.

[12] M. P. Forum. Mpi: A message-passing interface standard. Technical report, USA, 1994.

[13] C. Geuzaine and J.-F. Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING, 2009.

[14] G. Gori, G. Parma, M. . Zocca, and A. Guardone. Local solution to the unsteady stefan problem for in-flight ice accretion modeling. Journal of Aircraft, 52:251–262, 218.

[15] A. Guardone and L. Quartapelle. Spatially factorized galerkin and taylor-galerkin schemes for multidimensional conservation laws. Scientific Report DIA-SR 00-18, Politecnico di Milano, Italy, 2000.

[16] R. Honsek and W. Habashi. Fensap-ice: Eulerian modeling of droplet impingement in the sld regime of aircraft icing. 01 2006. ISBN 978-1-62410-039-0. doi: 10.2514/6.2006-465.

[17] G. E. Irving Langmuir, Katherine B. Blodgett. A mathematical investigation of water droplet trajectories. Technical report, Army Air Forces, 1946.

[18] E. Iuliano, V. Brandi, G. Mingione, C. de Nicola, and R. Tognaccini. Water Impingement Prediction on Multi-Element Airfoils by Means of Eulerian and Lagrangian Approach with Viscous and Inviscid Air Flow, chapter 1, page 4. doi: 10.2514/6.2006-1270. URL https://arc.aiaa.org/doi/abs/10.2514/6.2006-1270.

[19] R. J. Leveque. Numerical Methods for Conservation Laws. Birkhauser Verlag, 1992.

[20] P. Michael, R. A. Wong See-Cheuk, H. Kuohsing, E. V. Giao, T.Bidwell, and C. S. Large and small droplet impingement data on airfoils and two simulated ice shapes. page 50, 2007.

[21] F. A. Morrison. An Introduction to Fluid Mechanics. Cambridge University Press, 2013.

[22] T. G. Myers. Extension to the messinger model for aircraft icing. AIAA Journal, 39: 211–218, 2001.

[23] M. Papadakis, K. E. Hung, G. Vu, H.-W. Yeong, C. S. Bidwell, M. D. Breer, and T. J. Bencic. Experimental investigation of water droplet impingement on airfoils, finite wings, and an s-duct engine inlet. 2002.

[24] M. Papadakis, A. Rachman, S.-C. Wong, H.-W. Yeong, K. E. Hung, and C. S. Bidwell. Water impingement experiments on a naca 23012 airfoil with simulated glaze ice shapes. American Institute of Aeronautics and Astronautics, Inc., Jan. 2004.

[25] B. Re and R. Abgrall. A pressure-based method for weakly compressible two-phase flows under a baer-nunziato type model with generic equations of state and pressure and velocity disequilibrium, 2021.

[26] A. Y. Varaksin. Turbulent Particle-Laden Gas Flows. Springer-Verlag Berlin Heidelberg, 2007.

[27] W. Wright. Further Refinement of the LEWICE SLD Model. doi: 10.2514/6.2006-464. URL https://arc.aiaa.org/doi/abs/10.2514/6.2006-464.

[28] W. Wright, D. Rigby, E. Galloway, and M. Potapczuk. Lewice and glennice results for ice prediction workshop. American Institute of Aeronautics and Astronautics, Inc., 2021.

[29] Q. Zhang, F. He, and Y. Ba. Delta-shock waves and riemann solutions to the generalized pressureless euler equations with a composite source term. Applicable Analysis, 0(0):1–14, 2021. doi: 10.1080/00036811.2021.1959554. URL https://doi.org/10.1080/00036811.2021.1959554.

A | Pressureless Gas Dynamics 1D Riemann Problem Solution

Considering the homogeneous problem rewritten in terms of the vector of conservative variables $\underline{\boldsymbol{q}}$ and the vector of convective fluxes $\underline{\boldsymbol{F}}_{conv}$ one obtains (A.1).

$$\frac{\partial}{\partial t}\begin{bmatrix} \alpha \\ \\ \alpha\underline{\boldsymbol{u}}_p \\ \\ \alpha\pi \end{bmatrix} + \overline{\nabla}\cdot\begin{bmatrix} \alpha\underline{\boldsymbol{u}}_p \\ \\ \dfrac{\alpha\underline{\boldsymbol{u}}_p \otimes \alpha\underline{\boldsymbol{u}}_p + \alpha\pi\underline{\underline{\boldsymbol{I}}}}{\alpha} \\ \\ \dfrac{\alpha\pi\cdot\alpha\underline{\boldsymbol{u}}_p + c^2\cdot\alpha\underline{\boldsymbol{u}}_p}{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \\ 0 \\ \\ 0 \end{bmatrix} \tag{A.1}$$

Or, in a more compact form, (A.2).

$$\frac{\partial}{\partial t}\underline{\boldsymbol{q}} + \overline{\nabla}\cdot\underline{\boldsymbol{F}}_{conv}\left(\underline{\boldsymbol{q}}\right) = \underline{\boldsymbol{0}} \tag{A.2}$$

To obtain the exact Riemann solver that will be later implemented in SU2, the Jacobian is computed in (A.3) assuming a one dimensional problem ($\underline{\boldsymbol{u}}_p = u_p\hat{\imath} + 0\hat{\jmath} + 0\hat{k}$)

$$\underline{\underline{\boldsymbol{J}}} = \frac{\partial F_{conv,i}}{\partial q_j} = \begin{bmatrix} 0 & 1 & 0 \\ -\dfrac{\alpha\pi + \left(\alpha u_p\right)^2}{\alpha^2} & 2\dfrac{\alpha u_p}{\alpha} & \dfrac{1}{\alpha} \\ -\dfrac{c^2 \cdot \alpha u_p + \alpha\pi \cdot \alpha u_p}{\alpha^2} & \dfrac{c^2 + \alpha\pi}{\alpha} & \dfrac{\alpha u_p}{\alpha} \end{bmatrix} \tag{A.3}$$

The corresponding eigenvalues λ_i and eigenvectors $\underline{\boldsymbol{r}}_i$ are (A.4).

$$\lambda_1 = \frac{\alpha u_p - c}{\alpha} \qquad \lambda_2 = \frac{\alpha u_p}{\alpha} \qquad \lambda_3 = \frac{\alpha u_p + c}{\alpha}$$

$$\underline{\boldsymbol{r}}_1 = \begin{bmatrix} \alpha \\ \alpha u_p - c \\ \alpha\pi + c^2 \end{bmatrix} \qquad \underline{\boldsymbol{r}}_2 = \begin{bmatrix} \alpha \\ \alpha u_p \\ \alpha\pi \end{bmatrix} \qquad \underline{\boldsymbol{r}}_3 = \begin{bmatrix} \alpha \\ \alpha u_p + c \\ \alpha\pi + c^2 \end{bmatrix} \tag{A.4}$$

As long as $c > 0$ all eigenvalues are distinct and the eigenvectors span the space of $\underline{\underline{\boldsymbol{J}}}$. This means a solution to the Riemann problem can be constructed and the intermediate states can be computed analytically.

It can be easily verified that for $c > 0$ the eigenvalues are increasingly ordered ($\lambda_1 < \lambda_2 < \lambda_3$) therefore the solution is always comprised of the three contact discontinuity waves as sketched in (A.1).

The procedure followed to obtain the Riemann solution can be found in more detail in [19].

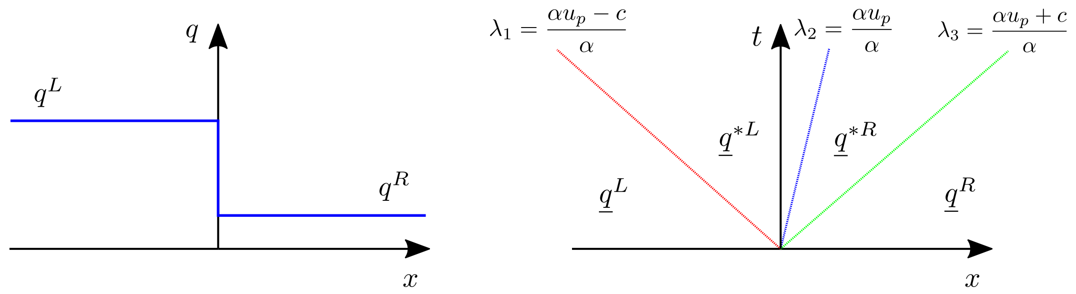

q
q^L
q^R
x
$\lambda_1 = \frac{\alpha u_p - c}{\alpha}$
t
$\lambda_2 = \frac{\alpha u_p}{\alpha}$
$\lambda_3 = \frac{\alpha u_p + c}{\alpha}$
$\underline{q}^{*L}$
$\underline{q}^{*R}$
$\underline{q}^L$
$\underline{q}^R$
x

Figure A.1: Riemann problem wave diagram in one variable

Defining $\underline{\underline{\boldsymbol{R}}} = [\underline{\boldsymbol{r}}_1, |\underline{\boldsymbol{r}}_2, |\underline{\boldsymbol{r}}_3]$ the Riemann problem can be solved by considering a change in variables.

$$\begin{cases} \underline{\gamma} = \underline{\underline{\boldsymbol{R}}}^{-1} \underline{\boldsymbol{q}}_{cons}^{L} \\ \underline{\boldsymbol{\beta}} = \underline{\underline{\boldsymbol{R}}}^{-1} \underline{\boldsymbol{q}}_{cons}^{R} \end{cases} \tag{A.5}$$

This decomposition of the Riemann problem allows the straightforward solution of each intermediate state since the jump across each wave can be computed as (A.6).

$$\underline{\boldsymbol{q}}^{i} = \underline{\boldsymbol{q}}^{L} + \sum_{j=1}^{j<i} \underline{\boldsymbol{r}}_j \cdot \left(\beta_j - \gamma_j\right) \qquad \begin{cases} i = 1 \;\; \text{Left state} \\ \dots \\ i = 4 \;\; \text{Right state} \end{cases} \tag{A.6}$$

Now the solution in all intermediate states $\left[\underline{\boldsymbol{q}}^{L} \mid \underline{\boldsymbol{q}}^{*L} \mid \underline{\boldsymbol{q}}^{*R} \mid \underline{\boldsymbol{q}}^{R}\right]$ is known as function of left and right states $\underline{\boldsymbol{q}}^{L} = \left[\alpha^L, \alpha^L u_p^L, 0\right]$ and $\underline{\boldsymbol{q}}^{R} = \left[\alpha^R, \alpha^R u_p^R, 0\right]$.

Given the non linearity of the conservation laws, these are still functions of the solution in the intermediate states itself, as can be seen in the admittedly difficult to read equation (A.7), where $\underline{\boldsymbol{q}}$ inside each intermediate state has been written in color.

Note the usage of the equilibrium assumption on the pseudo pressure $\pi^L = \pi^R = 0$ inside the control volumes that's been discussed prior. This does not mean that the additional pseudo-pressure relaxation equation has been neglected, since its presence is important and contributes to the intermediate states of the Riemann's problem's solution in a major way. Also, $\pi \neq 0$ in the intermediate states of the Riemann problem's solution in general.

$$\underline{\boldsymbol{q}}^L = \begin{bmatrix} \alpha \\ \\ \alpha u_p \\ \\ \alpha\pi \end{bmatrix} = \begin{bmatrix} \alpha^L \\ \\ \alpha^L u_p^L \\ \\ 0 \end{bmatrix}$$

$$\underline{\boldsymbol{q}}^{*L} = \begin{bmatrix} \alpha \\ \\ \alpha u_p \\ \\ \alpha\pi \end{bmatrix} = \frac{1}{2c^2} \begin{bmatrix} \begin{array}{r} \alpha^L\alpha\pi - \alpha^R\alpha\pi + 2\alpha^L c^2 - \alpha^L\alpha u_p c + \\ +\alpha^R\alpha u_p c + \alpha c\alpha^L u_p^L - \alpha c\alpha^R u_p^R, \\ \\ \dfrac{1}{\alpha}\Big(\alpha^L\alpha\pi\alpha u_p - \alpha^R\alpha\pi\alpha u_p - \alpha^L\alpha\pi c + \\ +\alpha^R\alpha\pi c + \alpha^L\alpha u_p c^2 - \alpha^L{\alpha u_p}^2 c - \\ +\alpha^R\alpha u_p c^2 + \alpha^R{\alpha u_p}^2 c + \alpha\alpha^L u_p^L c^2 + \\ +\alpha\alpha^R u_p^R c^2 + \alpha\alpha^L u_p^L\alpha u_p c - \alpha\alpha^R u_p^R\alpha u_p c\Big), \\ \\ \dfrac{1}{\alpha}\Big(c^2 + \alpha\pi\Big)\Big(\alpha^L\alpha\pi - \alpha^R\alpha\pi - \alpha^L\alpha u_p c + \\ +\alpha^R\alpha u_p c + \alpha\alpha^L u_p^L c - \alpha\alpha^R u_p^R c\Big) \end{array} \end{bmatrix}$$

$$\underline{\boldsymbol{q}}^{*R} = \begin{bmatrix} \alpha \\ \\ \alpha u_p \\ \\ \alpha \pi \end{bmatrix} = \frac{1}{2c^2} \begin{bmatrix} \begin{aligned} \alpha^R \alpha\pi - \alpha^L \alpha\pi + 2\alpha^R c^2 - \alpha^L \alpha u_p c + \\ + \alpha^R \alpha u_p c + \alpha c \alpha^L u_p^L - \alpha c \alpha^R u_p^R , \end{aligned} \\ \\ \begin{aligned} \frac{1}{\alpha} \Big(\alpha^R \alpha\pi \alpha u_p - \alpha^L \alpha\pi \alpha u_p - \alpha^L \alpha\pi c + \\ + \alpha^R \alpha\pi c - \alpha^L \alpha u_p c^2 - \alpha^L {\alpha u_p}^2 c + \\ + \alpha^R \alpha u_p c^2 + \alpha^R {\alpha u_p}^2 c + \alpha \alpha^L u_p^L c^2 + \\ + \alpha \alpha^R u_p^R c^2 + \alpha \alpha^L u_p^L \alpha u_p c - \alpha \alpha^R u_p^R \alpha u_p c \Big) , \end{aligned} \\ \\ \begin{aligned} -\frac{1}{\alpha} \Big(c^2 + \alpha\pi \Big) \Big(\alpha^L \alpha\pi - \alpha^R \alpha\pi + \alpha^L \alpha u_p c - \\ + \alpha^R \alpha u_p c - \alpha \alpha^L u_p^L c + \alpha \alpha^R u_p^R c \Big) \end{aligned} \end{bmatrix} \tag{A.7}$$

$$\underline{\boldsymbol{q}}^{R} = \begin{bmatrix} \alpha \\ \\ \alpha u_p \\ \\ \alpha \pi \end{bmatrix} = \begin{bmatrix} \alpha^R \\ \\ \alpha^R u_p^R \\ \\ 0 \end{bmatrix}$$

To remove this dependency and obtain a completely solved Riemann problem the constancy of Riemann invariants ϕ across each wave is exploited.

These are defined as ϕ_i^j such that they are constant across the $j-th$ wave and their gradient taken with respect to $\underline{\boldsymbol{q}}$ is orthogonal to the $j-th$ eigenvector $\underline{\boldsymbol{r}}_j$ as (A.8).

$$\nabla_{\underline{\boldsymbol{q}}} \phi_i^j \cdot \underline{\boldsymbol{r}}_j = 0 \qquad i = 1, 2 \tag{A.8}$$

The resulting Riemann invariants are (A.9).

$$\begin{aligned}
&\phi_1^1 = u_p - \frac{c}{\alpha} \quad && \phi_2^1 = \pi + c u_p \qquad && \text{across the 1st wave} \\
&\phi_1^2 = u_p && \phi_2^2 = \pi && \text{across the 2nd wave} \\
&\phi_1^3 = u_p + \frac{c}{\alpha} && \phi_2^3 = \pi - c u_p && \text{across the 3rd wave}
\end{aligned} \tag{A.9}$$

Exploiting their continuity across each wave and substituting in (A.7) the final solution can be obtained and is presented in primitive variables for $\underline{\boldsymbol{p}}^i = \left[\alpha, u_p, \pi\right]$ for ease of notation.

$\underline{\boldsymbol{p}}^L$	$\underline{\boldsymbol{p}}^{*L}$	$\underline{\boldsymbol{p}}^{*R}$	$\underline{\boldsymbol{p}}^R$
α^L	$\dfrac{2\alpha^L c}{2c - \alpha^L \left(u_p^L - u_p^R\right)}$	$\dfrac{2\alpha^R c}{2c - \alpha^R \left(u_p^L - u_p^R\right)}$	α^R
u_p^L	$\dfrac{u_p^L + u_p^R}{2}$	$\dfrac{u_p^L + u_p^R}{2}$	u_p^R
0	$c\dfrac{u_p^L - u_p^R}{2}$	$c\dfrac{u_p^L - u_p^R}{2}$	0

(A.10)

It's important to note that (A.10) is a valid solution to the Riemann problem under the assumption that $c > \max\left(0, \alpha^L \dfrac{u^L - u^R}{2}, \alpha^R \dfrac{u^L - u^R}{2}\right)$.

The knowledge of the exact solution of the Riemann problem allows for a straightforward implementation of a exact Riemann solver.

B | Configuration File Samples

To run the eulerian droplet tracking step the SU2 configuration file has been expanded with more options. In the following a couple of samples are reported.

B.1. Main - SU2

Required in it's entirety to run a droplet tracking simulation. Must be added in the CFD configuration file.

```
% --------------------- PARTICLE TRACKING ----------------------%
% -------------------------- GENERAL ---------------------------%
% Enable droplet tracking [YES, NO]
TRACK_PARTICLES=            YES

% Restart or not from solution file of previous PT simulation.
% If multibin, each bin from the second will restart from
% the previous bin's solution [YES, NO]
RESTART_PT=                 YES

% Correct collection efficiency and output splashed data
% for the lagrangian reimpingement step [YES, NO]
SPLASH_PARTICLES=           YES

% Subdivide each splashed parcel into more equispaced parcels
% for the lagrangian reimpingement step (optional)
N_SPACING_SPLASH=           3

% If subdividing the splashed parcels, the fraction of mesh face
% being covered with parcels ( >0, <=1 ) (only in 2D) (optional)
FRACTION_SPACING_SPLASH=    0.5
```

```
% -------------------------- NUMERICS --------------------------%
% Convective fluxes discretization (GODUNOV)
CONV_NUM_METHOD_PT=         GODUNOV

% MUSCL reconstruction for 2nd order [YES, NO]
MUSCL_PT=                   YES

% Limiter for 2nd order reconstruction
SLOPE_LIMITER_PT=           VENKATAKRISHNAN_WANG


% Pseudo time stepping method
% [EULER_EXPLICIT, EULER_IMPLICIT]
TIME_DISCRE_PT=             EULER_IMPLICIT

% Pseudo time stepping CFL number
CFL_NUMBER_PT=              1

% Residual to be achieved for convergence
CONV_RESIDUAL_MINVAL_PT=    -7

% Berthon relaxation coefficients (optional)
RELAXATION_C_PT=            1E6
RELAXATION_EPS_PT=          1


% -------------------- SPLASHING PARAMETERS --------------------%
% [N/m]
DROPLET_SURFACE_TENSION=   0.074

% [kg/m3]
DROPLET_DENSITY=           1000

% [Pa/s]
DROPLET_DYNAMIC_VISCOSITY= 0.0011208
```

```
% [kg/m3]
LWC=                        0.0005

% MVD of droplets [m]
MVD_PT=                     0.000021


% -------------------------- OUTPUT ---------------------------%
SOLUTION_FILENAME_PT=      solution_PT.dat
RESTART_FILENAME_PT=       restart_PT.dat
VOLUME_FILENAME_PT=        vtu_PT
SURFACE_FILENAME_PT=       surface_PT
```

B.2. Multibin - Experimental Distribution - SU2

To be added to run a multibin simulation with known bin distribution.

```
% ------------------------- MULTIBIN --------------------------%
% Enable multibin
PT_MULTIBIN=           YES

% Diameter of droplets in each bin [m]
MULTIBIN_DIAMETER=     (1E-06 ,   2E-06 ,   3E-06 ,   4E-06)

% Percentage of LWC in each bin [%]
MULTIBIN_PERCENTAGE=   (25    ,   50    ,   20    ,   5     )

% Specify a different CFL number for each bin (optional)
MULTIBIN_CFL=          (1     ,   0.5   ,   0.25  ,   0.1   )
```

B.3. Multibin - Langmuir D - SU2

To be added to run a multibin simulation with unknown bin distribution.

```
% ------------------------- MULTIBIN --------------------------%
% Enable multibin
PT_MULTIBIN=      YES

% Number of bins (<60)
N_BINS=           6

% Specify a different CFL number for each bin (optional)
MULTIBIN_CFL=     (1, 1, 1, 0.7, 0.6, 0.5)
```

B.4. Reimpingement - PoliDrop

To be added in PoliDrop's configuration file to run the Lagrangian reimpingement step.

```
% ---------------------- REIMPINGEMENT ------------------------%
% File where splashed data is stored
splashingFile=              ../CFD/splashed/splashed_data.dat

% LWC of freestream incoming cloud [kg/m3]
splashingFreestreamLWC=     0.0005

% Number of droplets below which parcel is ignored
% Use only if strictly necessary
splashingThresholdN=        1
```

C | Flow Euler Equations

All the results presented in this work have been obtained solving the compressible Euler equations for the air flow using the many state of the art tools already available within SU2. Since they have been extensively used throughout the whole thesis, a small mention to the theory of these equations is warranted.

These are an inviscid (and non thermally conductive) approximation of the more complicated Navier-Stokes equations. Furthermore, the equation of state for ideal gases and an expression for the internal energy E are added as shown in (C.1).

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \overline{\nabla} \cdot (\rho \underline{\boldsymbol{U}}) = 0 \\[2ex] \dfrac{\partial \rho \underline{\boldsymbol{U}}}{\partial t} + \overline{\nabla} \cdot (\rho \underline{\boldsymbol{U}} \otimes \underline{\boldsymbol{U}} + P \underline{\underline{\boldsymbol{I}}}) = \underline{\mathbf{0}} \\[2ex] \dfrac{\partial \rho E}{\partial t} + \overline{\nabla} \cdot (\rho E \underline{\boldsymbol{U}} + P \underline{\boldsymbol{U}}) = \underline{\mathbf{0}} \end{cases} \quad + \quad \begin{cases} P = \rho R T \\ P = (\gamma - 1)\rho \left[E - \dfrac{1}{2} \underline{\boldsymbol{U}} \cdot \underline{\boldsymbol{U}} \right] \end{cases} \tag{C.1}$$

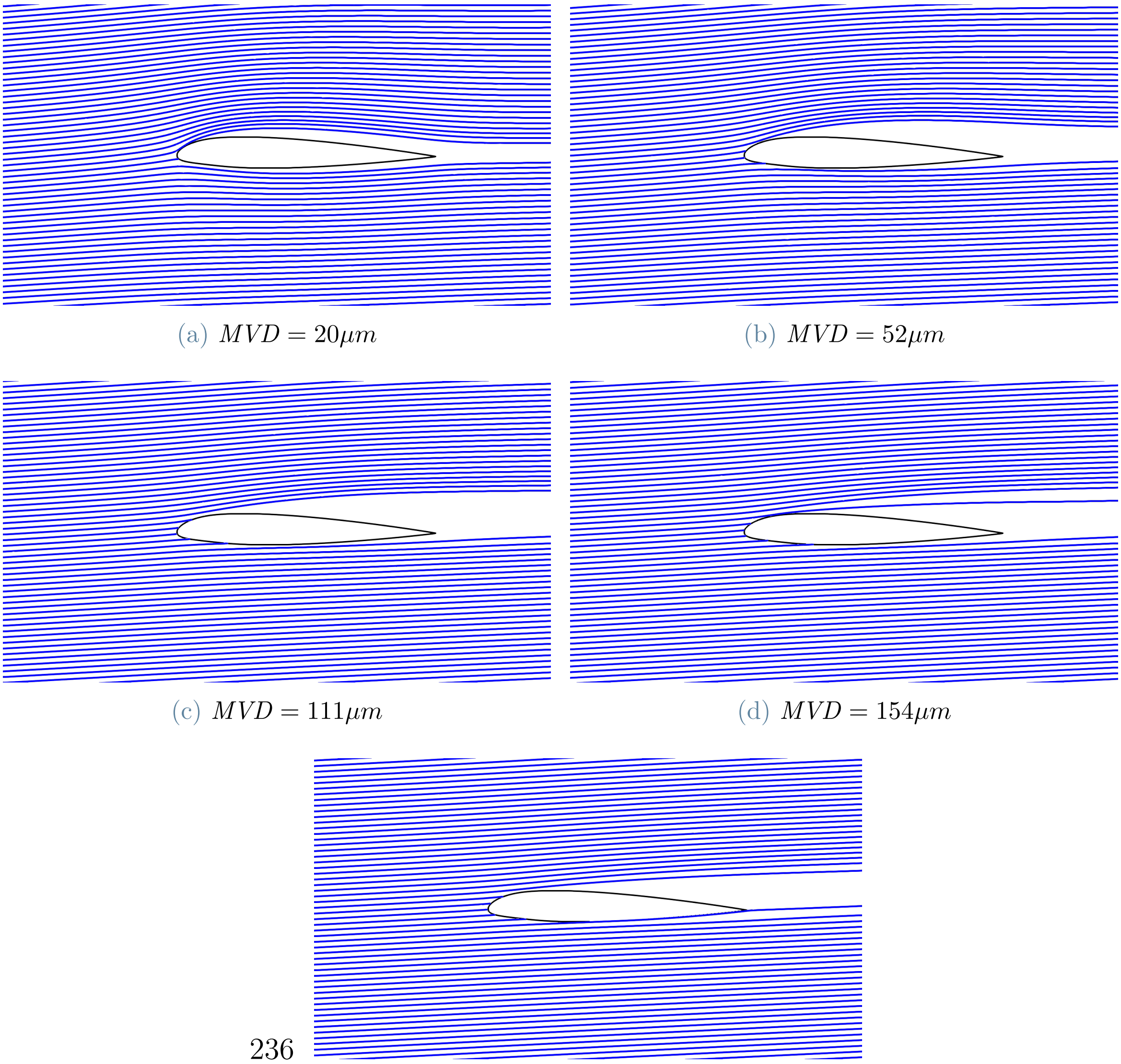
Being an inviscid approximation of Navier-Stokes, fluid particles should not "stick" to solid boundaries. This is why the wall boundary condition employed is a "no penetration" boundary condition rather than a "no slip" one used in RANS simulations for example.

The two boundary conditions can be easily expressed analytically:

1. No penetration BC: $\underline{\boldsymbol{U}} \cdot \hat{\boldsymbol{n}} = 0$
2. No slip BC: $\underline{\boldsymbol{U}} = \underline{\mathbf{0}}$

A drawback of using a no penetration BC is that boundary layers are completely absent. This can cause slight differences in the trajectories of small droplets when close to the body.

D | Droplet Trajectories



(a) $MVD = 20\mu m$

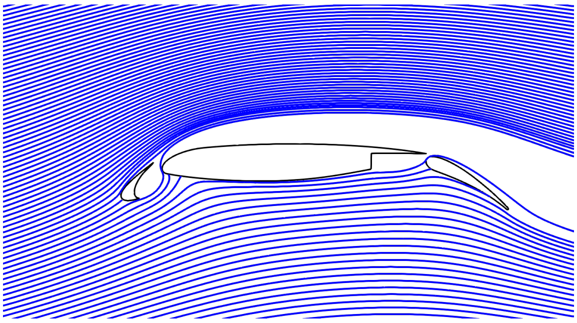
(b) $MVD = 52\mu m$

(c) $MVD = 111\mu m$

(d) $MVD = 154\mu m$

(e) $MVD = 236\mu m$

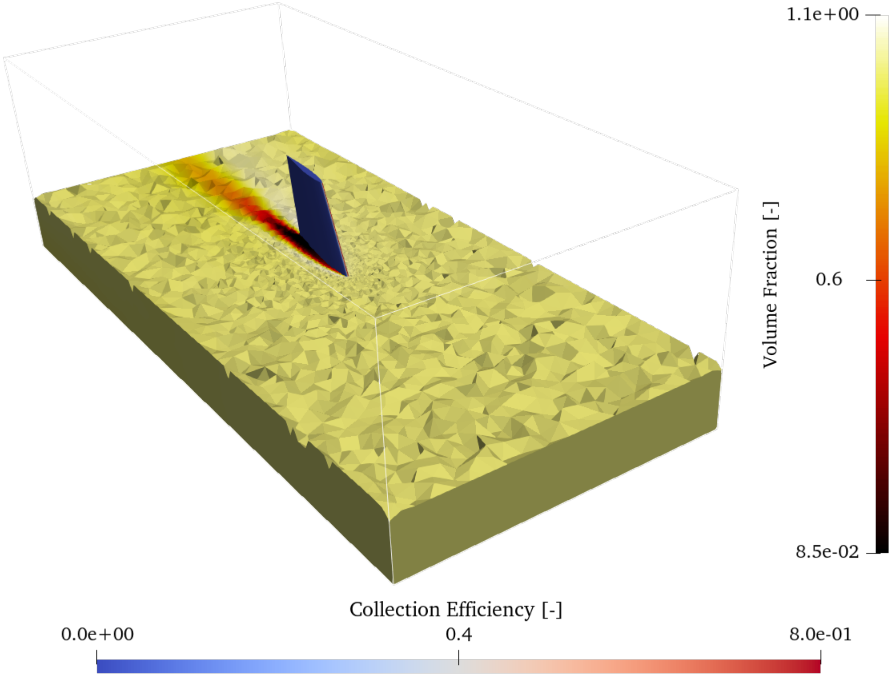
Figure D.1: Droplet trajectories integrated from velocity field NACA 23012 [24]



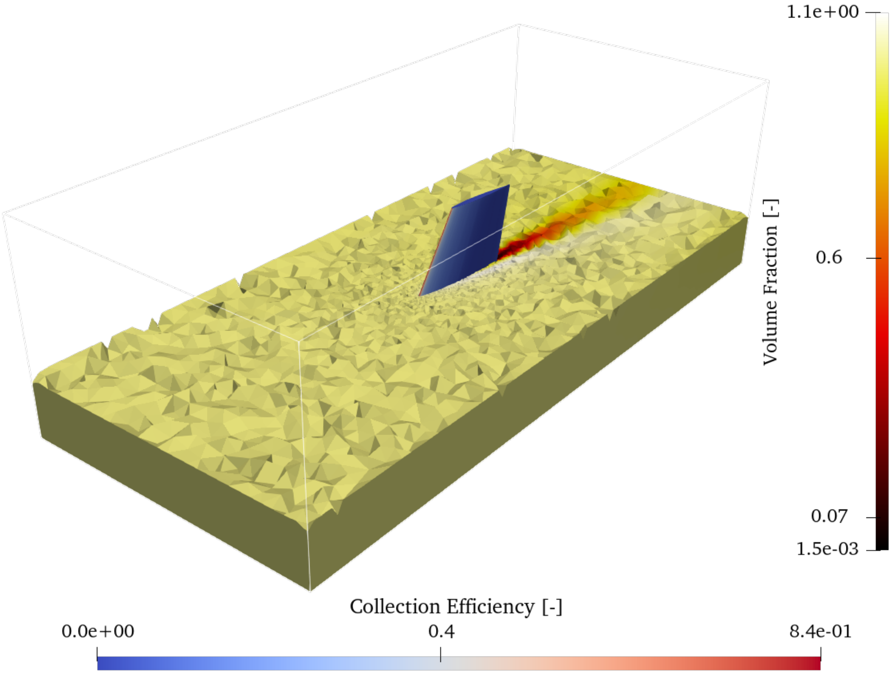
(a) $MVD = 21\mu m$

Figure D.2: Droplet trajectories integrated from velocity field three element airfoil case 121 AIAA 1st Icing Prediction Workshop [1]

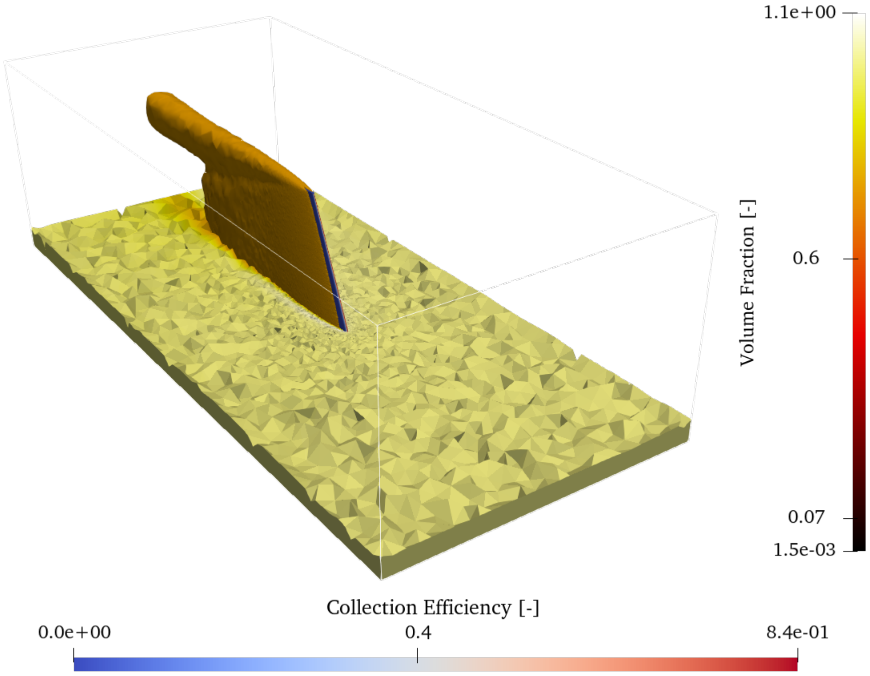
E | Case 111 & 112 Postprocess & Visualization



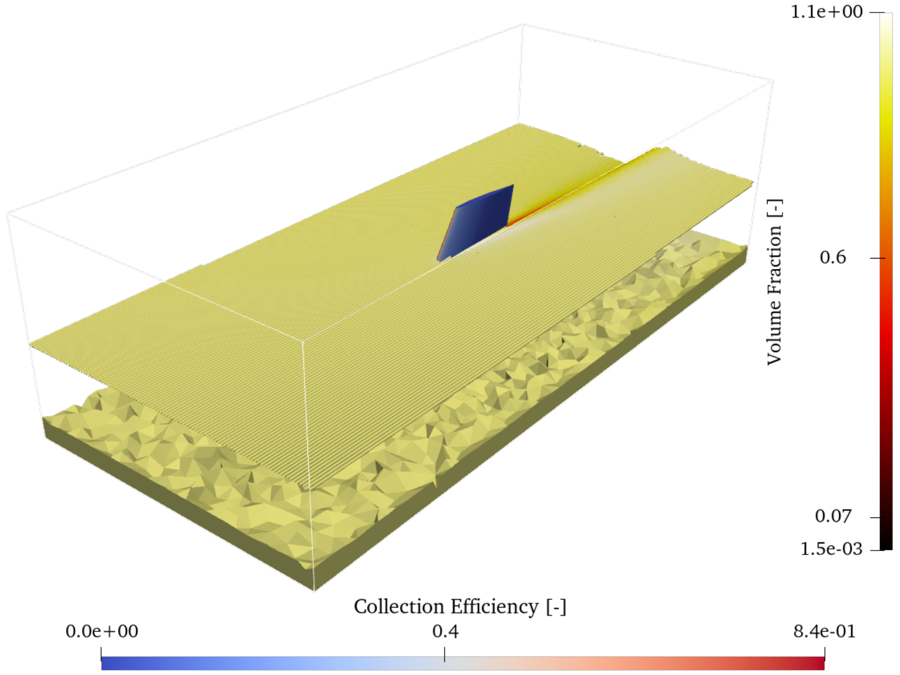
(a)



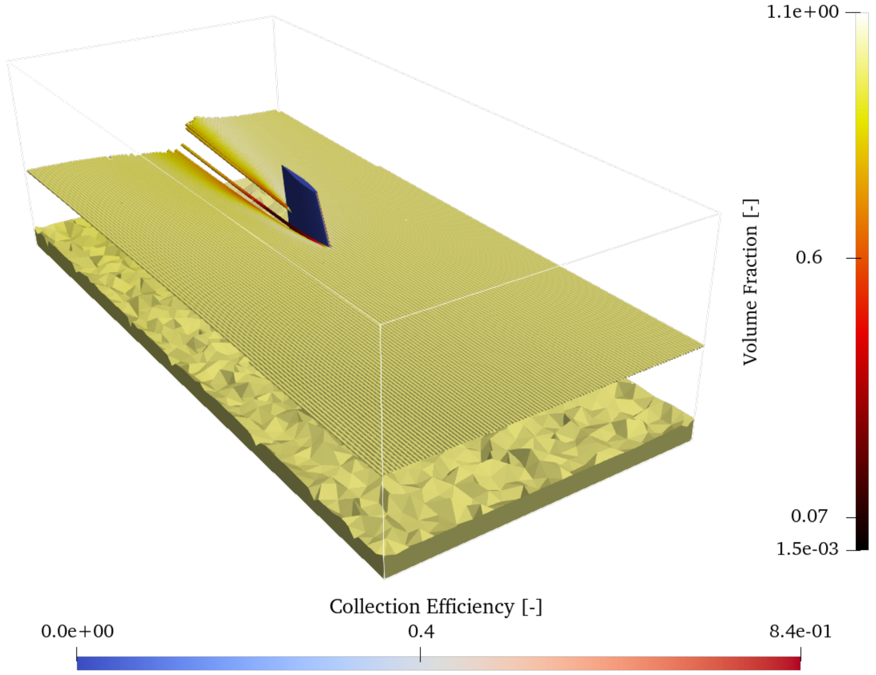
(b)



(c) $\alpha = 0.8$ contour



(d) particle trajectories (forward time integrated)



(e) particle trajectories (forward time integrated)

Figure E.1: Mesh and visualization (streamlines, isosurfaces, collection efficiency) of case 112 [23] AIAA 1st Icing Prediction Workshop [1]

F | PoliPhase - A New Unstructured Baer-Nunziato Solver

After the work presented in the core of this thesis, the development of a new separate Baer-Nunziato [4] solver has been started and it is here outlined. The conservation laws Eq. (F.1) being solved are the multi-dimensional version of the adimensionalized pressure formulation of the Baer-Nunziato CLAWs presented by Re et al. [25]. Note that, to remove one equation for α, one may remember that $\alpha_{N_{phases}} = 1 - \sum_{j=1}^{N_{phases}-1} \alpha_j$.

$$\begin{cases} \dfrac{\partial \alpha_i}{\partial t} & + \underline{\boldsymbol{u}}_I \cdot \overline{\nabla} \alpha_i = 0 \qquad \text{for } i = 1, ..., N_{phases} - 1 \\ \dfrac{\partial \alpha_i \rho_i}{\partial t} & + \overline{\nabla} \cdot \left(\alpha_i \rho_i \underline{\boldsymbol{u}}_i \right) = 0 \\ \dfrac{\partial \alpha_i \rho_i \underline{\boldsymbol{u}}_i}{\partial t} & + \overline{\nabla} \cdot \left(\alpha_i \rho_i \underline{\boldsymbol{u}}_i \otimes \underline{\boldsymbol{u}}_i + \alpha_i P_i \underline{\underline{\boldsymbol{I}}} \right) - P_I \overline{\nabla} \alpha_i = \underline{\boldsymbol{0}} \\ M_r^2 \left[\alpha_i \dfrac{\partial P_i}{\partial t} \right. & \left. + \, \alpha_i \underline{\boldsymbol{u}}_i \cdot \overline{\nabla} P_i + \alpha_i \rho_i c_i^2 \overline{\nabla} \cdot \underline{\boldsymbol{u}}_i - \rho_i c_{I,i}^2 \left(\underline{\boldsymbol{u}}_I - \underline{\boldsymbol{u}}_i \right) \cdot \overline{\nabla} \alpha_i \right] + \\ & + \kappa_i \left[\alpha_i \overline{\nabla} \cdot \underline{\boldsymbol{u}}_i - \left(\underline{\boldsymbol{u}}_I - \underline{\boldsymbol{u}}_i \right) \cdot \overline{\nabla} \alpha_i \right] = 0 \end{cases} \tag{F.1}$$

Where $(\cdot)_i$ indicates the i-th phase, and the interface velocity and pressure are (F.2).

$$\underline{\boldsymbol{u}}_I = \frac{\sum_i \alpha_i \rho_i \underline{\boldsymbol{u}}_i}{\sum_i \alpha_i \rho_i} \qquad P_I = \sum_i \alpha_i P_i \tag{F.2}$$

The thermodynamics are contained in the variables (F.3), and they are general, therefore any equation of state can be implemented as long as one can write $e = e(P, \rho)$.

$$\chi = \left(\frac{\partial P}{\partial \rho} \right)_e \qquad \kappa = \left(\frac{\partial P}{\partial e} \right)_\rho \qquad c_I^2 = \chi + \kappa \frac{P_I + e}{\rho} \qquad c^2 = \chi + \kappa \frac{P + e}{\rho} \tag{F.3}$$

As a starting point, the Stiffened Gas equation of state has been used due to its easy expression and possibility of describing both liquid and gas phases. P_∞ is called the minimal pressure and q_0 the heat of formation; if $q_0 = P_\infty = 0$ the polytropic ideal gas law is retrieved. Inverting $e(P,\rho)$ one gets $P(e,\rho)$ (F.4).

$$P(e,\rho) = (\gamma - 1)\left[e - \rho q_0\right] - \gamma P_\infty \tag{F.4}$$

Therefore, for the Stiffened Gas EOS, (F.5).

$$e = \frac{P + \gamma P_\infty}{(\gamma - 1)} + \rho q_0 \qquad \chi = -(\gamma - 1)\, q_0 \qquad \kappa = (\gamma - 1) \qquad T = \frac{P + P_\infty}{\rho c_v (\gamma - 1)} \tag{F.5}$$

The dimensional variables (with a tilde $\tilde{(\cdot)}$) and their adimensinal counterpart (without a tilde) are in (F.6). The reference values ρ_r , U_r and P_r are user inputs and the reference Mach number is $M_r^2 = \dfrac{\rho_r U_r^2}{P_r}$

$$\begin{cases} P = \dfrac{\tilde{P} - P_r}{\rho_r U_r^2} & \underline{\boldsymbol{u}} = \dfrac{\tilde{\underline{\boldsymbol{u}}}}{U_r} \\ \rho = \dfrac{\tilde{\rho}}{\rho_r} & e = \dfrac{\tilde{e}}{\rho_r U_r^2} \\ \chi = \dfrac{\tilde{\chi}}{U_r^2} & \kappa = \tilde{\kappa} \\ c_I^2 = \dfrac{\tilde{c}_I^2}{U_r^2} - \dfrac{1}{M_r^2}\dfrac{\kappa}{\rho} & c^2 = \dfrac{\tilde{c}^2}{U_r^2} - \dfrac{1}{M_r^2}\dfrac{\kappa}{\rho} \end{cases} \tag{F.6}$$

An important note on thermodynamic quantities: all of the thermodynamic quantities are computed in dimensional form and only at the end (so before going into the residuals in the solver) c^2 , c_I^2 and κ are adimensionalized. Therefore first all dimensional quantities are computed as (F.7).

$$\begin{cases} e & = \dfrac{P + \gamma P_\infty}{(\gamma - 1)} + \rho q_0 \\ \chi & = -\left(\gamma - 1\right) q_0 \\ \kappa & = (\gamma - 1) \\ c_I^2 & = \chi + \kappa \dfrac{P_I + e}{\rho} \\ c^2 & = \chi + \kappa \dfrac{P + e}{\rho} \end{cases} \tag{F.7}$$

And then only the ones needed in the equations directly (κ, c^2 and c_I^2) are adimensionalized (F.8).

$$\begin{cases} \tilde{\kappa} & = \kappa \\ \tilde{c}^2 & = \dfrac{c^2}{U_r^2} - \dfrac{1}{M_r^2} \dfrac{\tilde{\kappa}}{\tilde{\rho}} \\ \tilde{c}_I^2 & = \dfrac{c_I^2}{U_r^2} - \dfrac{1}{M_r^2} \dfrac{\tilde{\kappa}}{\tilde{\rho}} \end{cases} \tag{F.8}$$

The code in PoliPhase has been purpose built to allow for an easy implementation of different thermodynamic models down the line.

F.1. Numerics

The convective terms are discretized using Rusanov fluxes while non conservative terms are treated as sources, except for the one in the volume fraction equation which has been formulated in an effort to preserve homogeneity, that means that if α is homogeneous at $t = t_0$, it should remain so $\forall t \geq t_0$. This is the reason $\underline{\boldsymbol{u}}_I^i$ has been used for the residual in control volume i. To do so, a slight modification to the edge based upwind residual computation procedure in SU2 was required since the residual was originally re-used between nodes i-j and j-i by just changing the sign.

The pressure gradient term in the pressure equation Eq. (F.9), is split into a convective flux and a source term.

$$\underline{\boldsymbol{u}}_i \cdot \overline{\nabla} P_i = \overline{\nabla} \cdot (P_i \underline{\boldsymbol{u}}_i) - P_i \overline{\nabla} \cdot \underline{\boldsymbol{u}}_i \tag{F.9}$$

Highlighting in color each term and their respective discretization in (F.10).

$$\begin{cases} \dfrac{\partial \alpha_i}{\partial t} & + \underline{\boldsymbol{u}}_I \cdot \overline{\nabla} \alpha_i = 0 \\ \dfrac{\partial \alpha_i \rho_i}{\partial t} & + \overline{\nabla} \cdot \left(\alpha_i \rho_i \underline{\boldsymbol{u}}_i \right) = 0 \\ \dfrac{\partial \alpha_i \rho_i \underline{\boldsymbol{u}}_i}{\partial t} & + \overline{\nabla} \cdot \left(\alpha_i \rho_i \underline{\boldsymbol{u}}_i \otimes \underline{\boldsymbol{u}}_i + \alpha_i P_i \underline{\underline{\boldsymbol{I}}} \right) - P_I \overline{\nabla} \alpha_i = \underline{\mathbf{0}} \\ M_r^2 \left[\alpha_i \left(\dfrac{\partial P_i}{\partial t} \right. \right. & \left. + \ \overline{\nabla} \cdot \left(P_i \underline{\boldsymbol{u}}_i \right) \right) - \alpha_i P_i \overline{\nabla} \cdot \underline{\boldsymbol{u}}_i + \alpha_i \rho_i c_i^2 \overline{\nabla} \cdot \underline{\boldsymbol{u}}_i + \\ & \left. - \ \rho_i c_{I,i}^2 \left(\underline{\boldsymbol{u}}_I - \underline{\boldsymbol{u}}_i \right) \cdot \overline{\nabla} \alpha_i \right] + \kappa_i \left[\alpha_i \overline{\nabla} \cdot \underline{\boldsymbol{u}}_i - \left(\underline{\boldsymbol{u}}_I - \underline{\boldsymbol{u}}_i \right) \cdot \overline{\nabla} \alpha_i \right] = 0 \end{cases} \tag{F.10}$$

F.1.1. Convective Terms

The upwind contribution to the discrete residual computed between control volumes $(\cdot)^i$ and $(\cdot)^j$ are here presented. Note that for ease of notation $\hat{\boldsymbol{n}} = A^{i,j} \hat{\boldsymbol{n}}$ already contains the adimensional edge area for the integration across the interface. This area is adimensionalized as $A^{i,j} = \tilde{A}^{i,j} / L_r^2$.

Volume fraction:

$$+\frac{1}{2} \left(\underline{\boldsymbol{u}}_I \cdot \hat{\boldsymbol{n}} \right)^i \left[\alpha^i + \alpha^j \right] \qquad -\frac{1}{2} \mid \underline{\boldsymbol{u}}_I \cdot \hat{\boldsymbol{n}} \mid^i \left[\alpha^j - \alpha^i \right]$$

Mass:

$$+\frac{1}{2} \left[\alpha\rho \underline{\boldsymbol{u}}^i \cdot \hat{\boldsymbol{n}} + \alpha\rho \underline{\boldsymbol{u}}^j \cdot \hat{\boldsymbol{n}} \right] \qquad -\frac{1}{2} \max_{k \in i,j} \mid \underline{\boldsymbol{u}}^k \cdot \hat{\boldsymbol{n}} \mid \left[\alpha\rho^j - \alpha\rho^i \right]$$

Momentum x:

$$+\frac{1}{2} \left[\alpha\rho u \underline{\boldsymbol{u}}^i \cdot \hat{\boldsymbol{n}} + \alpha\rho u \underline{\boldsymbol{u}}^j \cdot \hat{\boldsymbol{n}} \right] \qquad -\frac{1}{2} \max_{k \in i,j} \mid \underline{\boldsymbol{u}}^k \cdot \hat{\boldsymbol{n}} \mid \left[\alpha\rho u^j - \alpha\rho u^i \right] +$$
$$+\frac{1}{2} \left[\alpha P^i + \alpha P^j \right] n_x$$

Momentum y:

$$+\frac{1}{2} \left[\alpha\rho v \underline{\boldsymbol{u}}^i \cdot \hat{\boldsymbol{n}} + \alpha\rho v \underline{\boldsymbol{u}}^j \cdot \hat{\boldsymbol{n}} \right] \qquad -\frac{1}{2} \max_{k \in i,j} \mid \underline{\boldsymbol{u}}^k \cdot \hat{\boldsymbol{n}} \mid \left[\alpha\rho v^j - \alpha\rho v^i \right] +$$
$$+\frac{1}{2} \left[\alpha P^i + \alpha P^j \right] n_y$$

Momentum z:

$$+\frac{1}{2} \left[\alpha\rho w \underline{\boldsymbol{u}}^i \cdot \hat{\boldsymbol{n}} + \alpha\rho w \underline{\boldsymbol{u}}^j \cdot \hat{\boldsymbol{n}} \right] \qquad -\frac{1}{2} \max_{k \in i,j} \mid \underline{\boldsymbol{u}}^k \cdot \hat{\boldsymbol{n}} \mid \left[\alpha\rho w^j - \alpha\rho w^i \right] +$$
$$+\frac{1}{2} \left[\alpha P^i + \alpha P^j \right] n_z$$

Pressure:

$$+M_r^2 \alpha^i \left\{ \frac{1}{2} \left[P \underline{\boldsymbol{u}}^i \cdot \hat{\boldsymbol{n}} + P \underline{\boldsymbol{u}}^j \cdot \hat{\boldsymbol{n}} \right] \qquad -\frac{1}{2} \max_{k \in i,j} \mid \underline{\boldsymbol{u}}^k \cdot \hat{\boldsymbol{n}} \mid \left[P^j - P^i \right] \right\}$$

F.1.2. Source Terms

The source term contribution to the discrete residual in control volume $(\cdot)^i$.

Volume fraction:

$$+0$$

Mass:

$$+0$$

Momentum x:

$$- \mid C^i \mid P_I^i \frac{\partial \alpha^i}{\partial x}$$

Momentum y:

$$- \mid C^i \mid P_I^i \frac{\partial \alpha^i}{\partial y}$$

Momentum z:

$$- \mid C^i \mid P_I^i \frac{\partial \alpha^i}{\partial z}$$

Pressure:

$$+ \mid C^i \mid \alpha^i \left[M_r^2 \left(\rho c^2 - P \right) + \kappa \right]^i \overline{\boldsymbol{\nabla}} \cdot \underline{\boldsymbol{u}}^i +$$

$$- \mid C^i \mid \left(M_r^2 \rho c_I^2 + \kappa \right)^i \left(\underline{\boldsymbol{u}}_I - \underline{\boldsymbol{u}} \right)^i \cdot \overline{\boldsymbol{\nabla}} \alpha^i$$

F.1.3. Dual Time Stepping Source Terms

The dual time stepping source term contribution to the discrete residual in control volume $(\cdot)^i$.

	1st order time stepping	*2nd order time stepping*
Volume Fraction:	$\frac{\alpha^{n+1}-\alpha^{n}}{\mathit{\Delta} t} \mid C^i \mid$	$\frac{4\alpha^{n+1}-3\alpha^{n}+\alpha^{n-1}}{2\mathit{\Delta} t} \mid C^i \mid$
Mass:	$\frac{\alpha\rho^{n+1}-\alpha\rho^{n}}{\mathit{\Delta} t} \mid C^i \mid$	$\frac{4\alpha\rho^{n+1}-3\alpha\rho^{n}+\alpha\rho^{n-1}}{2\mathit{\Delta} t} \mid C^i \mid$
Momentum x:	$\frac{\alpha\rho u^{n+1}-\alpha\rho u^{n}}{\mathit{\Delta} t} \mid C^i \mid$	$\frac{4\alpha\rho u^{n+1}-3\alpha\rho u^{n}+\alpha\rho u^{n-1}}{2\mathit{\Delta} t} \mid C^i \mid$
Momentum y:	$\frac{\alpha\rho v^{n+1}-\alpha\rho v^{n}}{\mathit{\Delta} t} \mid C^i \mid$	$\frac{4\alpha\rho v^{n+1}-3\alpha\rho v^{n}+\alpha\rho v^{n-1}}{2\mathit{\Delta} t} \mid C^i \mid$
Momentum z:	$\frac{\alpha\rho w^{n+1}-\alpha\rho w^{n}}{\mathit{\Delta} t} \mid C^i \mid$	$\frac{4\alpha\rho w^{n+1}-3\alpha\rho w^{n}+\alpha\rho w^{n-1}}{2\mathit{\Delta} t} \mid C^i \mid$
Pressure:	$\alpha^{n+1}M_r^2\frac{P^{n+1}-P^{n}}{\mathit{\Delta} t} \mid C^i \mid$	$\alpha^{n+1}M_r^2\frac{4P^{n+1}-3P^{n}+P^{n-1}}{2\mathit{\Delta} t} \mid C^i \mid$

It should be noted that the 2nd order time stepping has been implemented but not tested. All derivatives and gradients are computed by SU2 using either a *weighted least square* or *Green-Gauss* approach. The physical time step $\mathit{\Delta} t$ is either user defined or computed using the CFL, same as the inner time step (F.11). Both are adimensionalized as $t = \tilde{t} \cdot U_r/L_r$.

$$\delta t = \frac{CFL \cdot \mid C^i \mid}{\max_{phases}\left(\mid \underline{\boldsymbol{u}} \mid + c\right) \overline{A^{i,j}}} \tag{F.11}$$

Up to now, only far-field (F.12) and Euler wall (F.13) boundary conditions have been implemented as for each phase $(\cdot)_i$ and each pair of node $(\cdot)^i$ (domain) and $(\cdot)^j$ (ghost cell).

$$\begin{cases} [\alpha,\rho,u,v,w,P]_i^j = [\alpha,\rho,u,v,w,P]_{BC,i} & \text{if } \underline{\boldsymbol{u}}_i \cdot \hat{\boldsymbol{n}} > 0 \text{ (inflow)} \\ [\alpha,\rho,u,v,w,P]_i^j = [\alpha,\rho,u,v,w,P]_i^i & \text{if } \underline{\boldsymbol{u}}_i \cdot \hat{\boldsymbol{n}} < 0 \text{ (outflow)} \end{cases} \tag{F.12}$$

$$\begin{bmatrix} \alpha \\ \rho \\ u \\ v \\ w \\ P \end{bmatrix}_i^j = \begin{bmatrix} \alpha \\ \rho \\ u - 2un_x \\ v - 2vn_y \\ w - 2wn_z \\ P \end{bmatrix}_i^i \tag{F.13}$$

F.2. Results

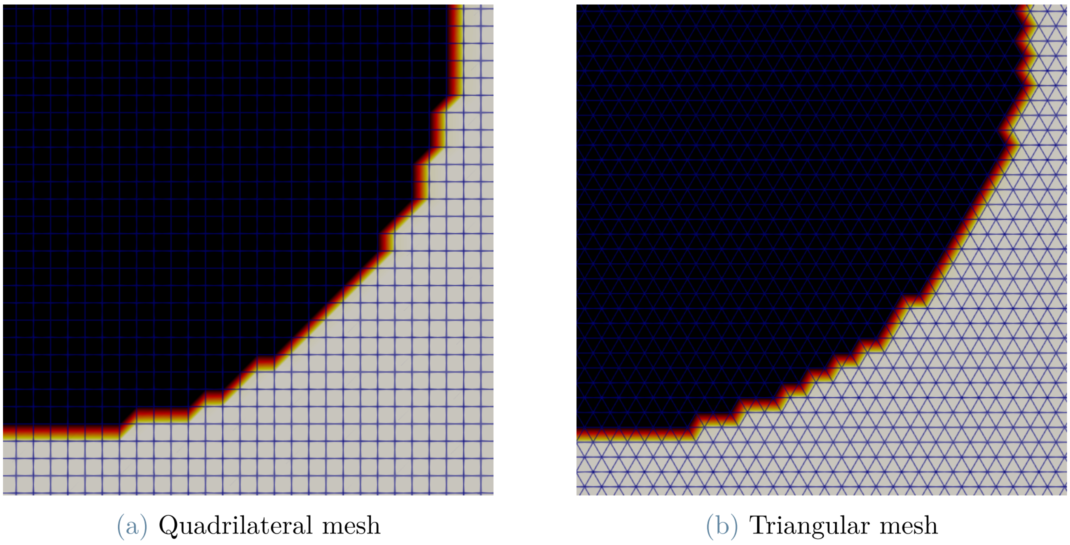
F.2.1. Advection Test

A simple water bubble in air test has been run to validate the code in these first phases. This test consists of a bubble of water inside an air carrier fluid at pressure and velocity equilibrium ($P_{air} = P_{H_2O}$ and $\underline{\boldsymbol{u}}_{air} = \underline{\boldsymbol{u}}_{H_2O}$). The expected result is that of conserved equilibrium, with the bubble being rigidly transported within the carrier fluid.

The test has been run on a regular quadrilateral (but unstructured) mesh and a triangular unstructured mesh with similar mesh size with the following numerical settings.

Numerics	*1st order time stepping*	
	$\Delta t = 10^{-6}\ s$	
	$CFL_{inner}(\mid \underline{\boldsymbol{u}} \mid) = 0.05$	
Meshes	$n_x = 250$	*quadrilateral*
	$h = 0.004\ m$	*triangular*
I.C. *Bubble*	$\alpha_{air} = 0.1$	$\alpha_{H_2O} = 0.9$
	$\rho_{air} = 1\ kg/m^3$	$\rho_{H_2O} = 1000\ kg/m^3$
	$P_{air} = 10^6\ Pa$	$P_{H_2O} = 10^6\ Pa$
	$\underline{\boldsymbol{u}}_{air} = \begin{bmatrix}100 & 0\end{bmatrix}\ m/s$	$\underline{\boldsymbol{u}}_{H_2O} = \begin{bmatrix}100 & 0\end{bmatrix}\ m/s$
I.C. *Free-Stream*	$\alpha_{air} = 0.9$	$\alpha_{H_2O} = 0.1$
	$\rho_{air} = 1\ kg/m^3$	$\rho_{H_2O} = 1000\ kg/m^3$
	$P_{air} = 10^6\ Pa$	$P_{H_2O} = 10^6\ Pa$
	$\underline{\boldsymbol{u}}_{air} = \begin{bmatrix}100 & 0\end{bmatrix}\ m/s$	$\underline{\boldsymbol{u}}_{H_2O} = \begin{bmatrix}100 & 0\end{bmatrix}\ m/s$
Thermodynamics	$\gamma_{air} = 1.4$	$\gamma_{H_2O} = 4.4$
	$c_{v,air} = 717.60\ J/kgK$	$c_{v,H_2O} = 4178\ J/kgK$
	$q_{\infty,air} = 0\ J/kg$	$q_{\infty,H_2O} = 0\ J/kg$
	$P_{\infty,air} = 0\ Pa$	$P_{\infty,H_2O} = 6 \cdot 10^8\ Pa$

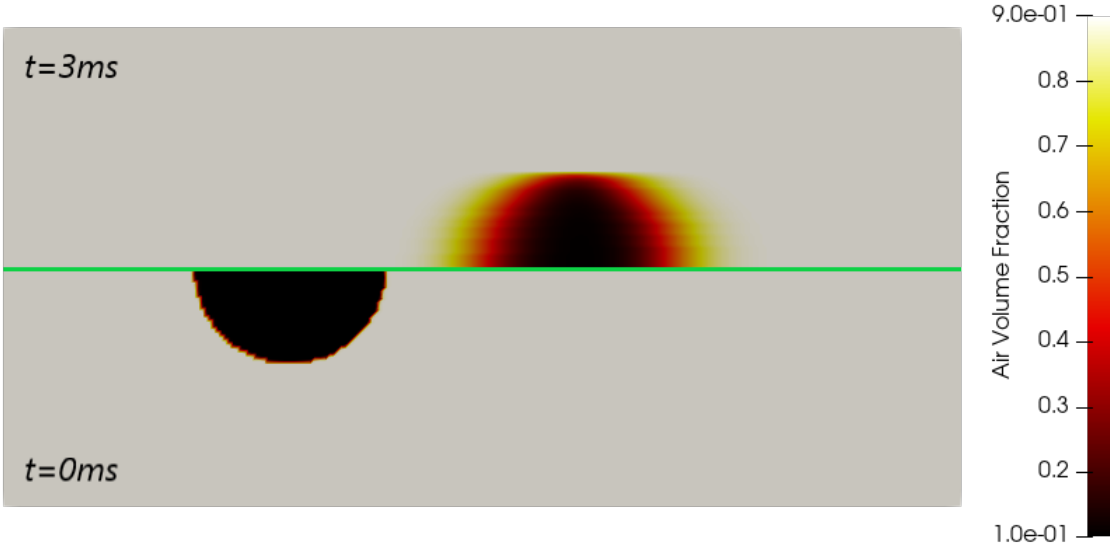
Table F.1: Numerical setup for bubble advection test



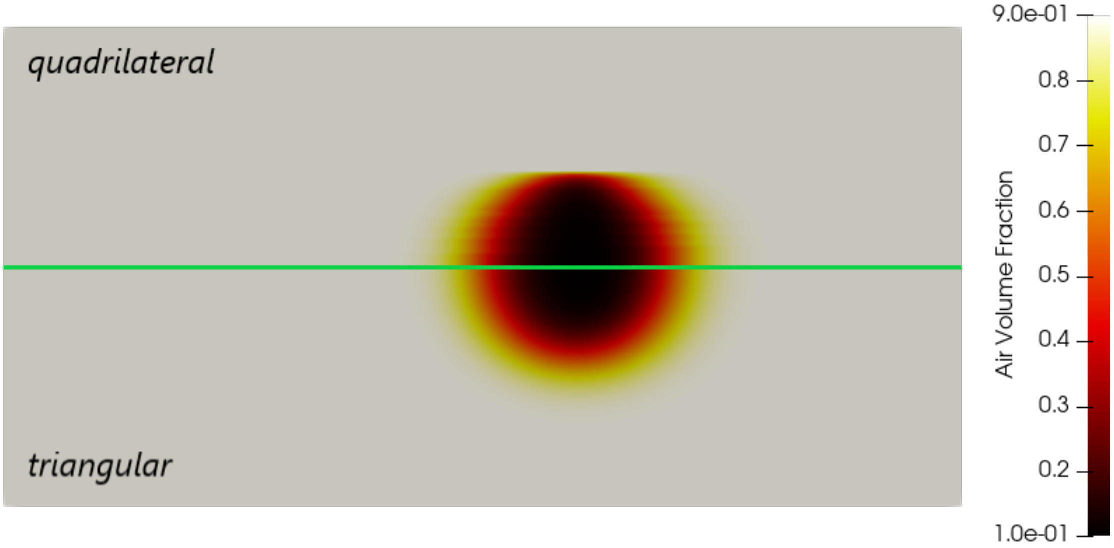
(a) Quadrilateral mesh

(b) Triangular mesh

Figure F.1: Mesh details for bubble advection test



(a) Initial conditions and final solution for quadrilateral



(b) Comparison triangular and quadrilateral

Figure F.2: Volume fraction field for bubble advection test

As Figure (F.3) shows, rigid transport of the bubble occurs at the correct speed (100 m/s) and results compare well to the 1D results shown by Re et al. [25] in their $n_x = 1600$ results. There is more dissipation, as expected given the much coarser mesh used for computational cost reasons. Pressure and velocity remain constant.

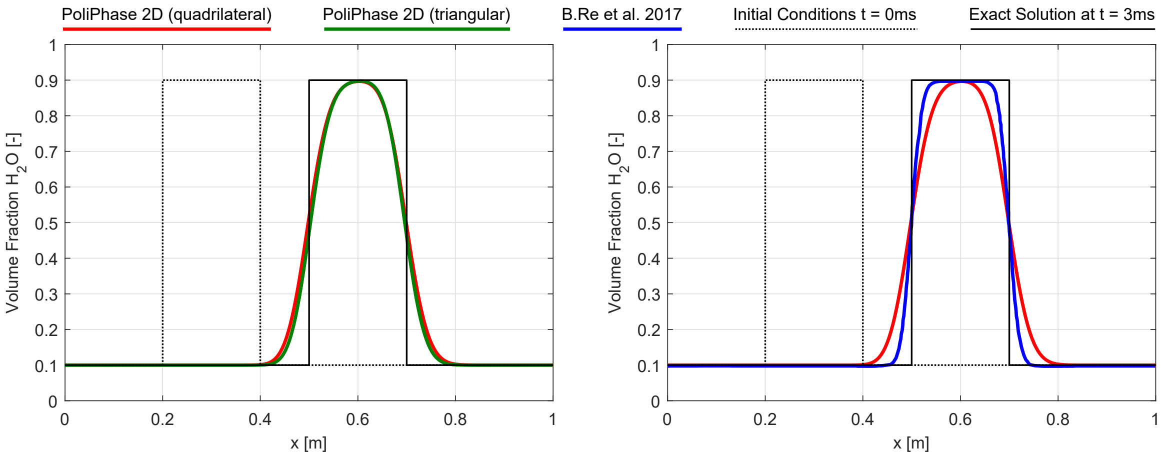


Figure F.3: PoliMIce ice accretion multi-step computational procedure

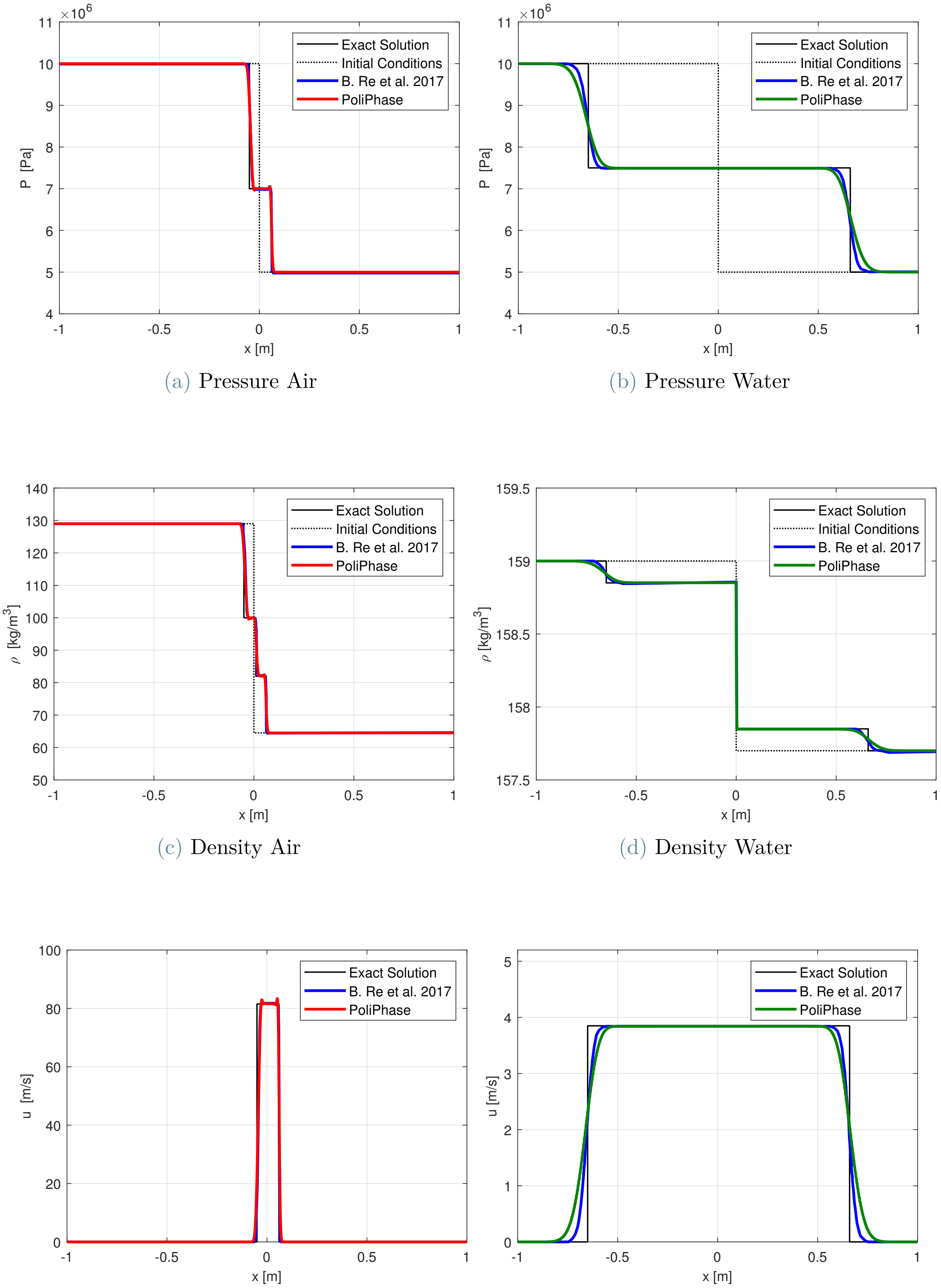
F.2.2. No Mixing Shock Tube Test

Another test performed is that of a $2m$ long shock tube of a 50/50 water/air mixture initially at rest ($u = 0\ m/s$) with a pressure jump at $x = 0\ m$. The whole tube is initially at $T = 270\ K$ and the simulation evolves for 0.16 ms and the initial densities are retrieved from the Stiffened Gas EOS. The simulation is 1D in nature but is performed on a 2D mesh.

Numerics	*1st order time stepping*	
	$\Delta t = 10^{-6}\ s$	
	$CFL_{inner}(\lvert\ \underline{\boldsymbol{u}}\ \rvert + c) = 20$	
Mesh	$n_x = 1000$	*quadrilateral*
I.C. *Left*	$\alpha_{air} = 0.5$	$\alpha_{H_2O} = 0.5$
	$\rho_{air} = 129\ kg/m^3$	$\rho_{H_2O} = 159\ kg/m^3$
	$P_{air} = 10^7\ Pa$	$P_{H_2O} = 10^7\ Pa$
	$\underline{\boldsymbol{u}}_{air} = \begin{bmatrix} 0 & 0 \end{bmatrix}\ m/s$	$\underline{\boldsymbol{u}}_{H_2O} = \begin{bmatrix} 0 & 0 \end{bmatrix}\ m/s$
I.C. *Right*	$\alpha_{air} = 0.5$	$\alpha_{H_2O} = 0.5$
	$\rho_{air} = 64.5\ kg/m^3$	$\rho_{H_2O} = 157.7\ kg/m^3$
	$P_{air} = 5 \cdot 10^6\ Pa$	$P_{H_2O} = 5 \cdot 10^6\ Pa$
	$\underline{\boldsymbol{u}}_{air} = \begin{bmatrix} 0 & 0 \end{bmatrix}\ m/s$	$\underline{\boldsymbol{u}}_{H_2O} = \begin{bmatrix} 0 & 0 \end{bmatrix}\ m/s$
Thermodynamics	$\gamma_{air} = 1.4$	$\gamma_{H_2O} = 4.4$
	$c_{v,air} = 717.60\ J/kgK$	$c_{v,H_2O} = 4178\ J/kgK$
	$q_{\infty,air} = 0\ J/kg$	$q_{\infty,H_2O} = 0\ J/kg$
	$P_{\infty,air} = 0\ Pa$	$P_{\infty,H_2O} = 6 \cdot 10^8\ Pa$

Table F.2: Numerical setup for no mixing shock tube test

The analytical solution is obtained by assuming each phase is governed by the Euler equations as independent gasses. Results are also compared to Re et al. [25] 1D results with 2000 elements (compared to 1000 for PoliPhase) and $\Delta t = 3.1 \cdot 10^{-7}\ s$ (compared to $\Delta t = 10^{-6}\ s$ for PoliPhase).



(a) Pressure Air

(b) Pressure Water

(c) Density Air

(d) Density Water

(e) Velocity Air

(f) Velocity Water

Figure F.4: Results at $t = 0.16ms$ for no mixing shock tube test

In Figure (F.4) results for PoliPhase show good agreement despite the coarser mesh, the bigger timestep and an overall less refined spatial and time discretization compare to Re et al. [25]. The contact discontinuity, shock and rarefaction waves are well tracked, with the second two being relatively smeared in the liquid phase. There are some oscillations in the air's pressure and velocity, this is something that needs to be looked into when developing better discretizations.

In Figure (F.5) the final density fields are reported in the domain for reference.

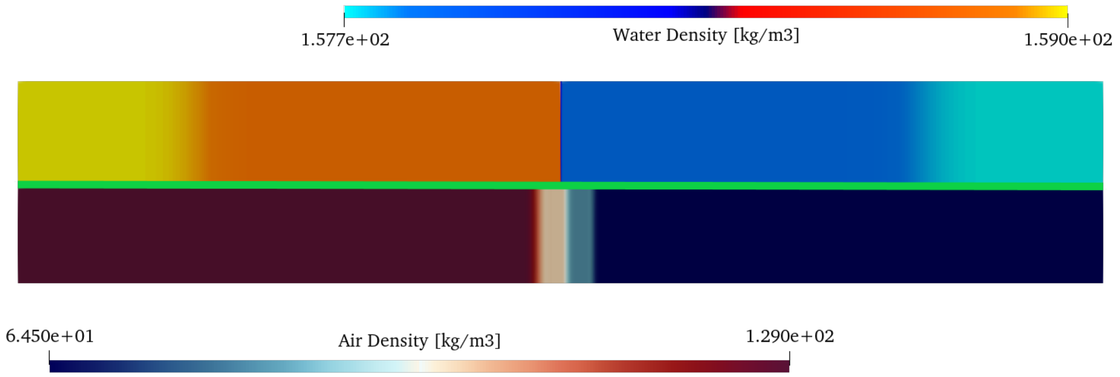


Figure F.5: Density fields at $t = 0.16ms$ for air and water in no mixing shock tube test

The volume fraction remains constant confirming that the discretization of the non conservative term in the α equation preserves homogeneity even with sharp discontinuities.

F.2.3. Further Work

This solver is still in its infancy and will certainly require extensive development to reach state of the art. Work that needs to be done includes, but is not limited to:

1. Implementation of different thermodynamic models
2. Development of a hybrid conservative/non conservative solver for varying Mach number
3. Rhie & Chow interpolation or grid staggering for low Mach checkerboarding
4. Better discretization of the convective fluxes
5. Better discretization of the non conservative terms

List of Figures

List of Tables

List of Symbols

Mathematical Notation

$\underline{\boldsymbol{x}}$	-	vector
$\underline{\underline{\boldsymbol{X}}}$	-	matrix
$\underline{\underline{\boldsymbol{I}}}$	-	identity matrix
$\hat{\boldsymbol{n}}$	-	unit normal vector

Droplet Variables

$\underline{\boldsymbol{u}}_p$	-	droplet velocity
d_p	-	droplet diameter
α	-	droplet volume fraction
MVD	-	mean volumetric diameter
LWC	-	liquid water content
ρ_p	-	droplet density
μ_p	-	droplet dynamic viscosity
σ_p	-	droplet surface tension
β	-	collection efficiency

Other Variables

$\underline{\boldsymbol{U}}$	-	air velocity
ρ_f	-	air density
μ_f	-	air dynamic viscosity
π	-	relaxation pseudo pressure
c	-	relaxation constant
M_∞	-	free stream Mach number
P_∞	-	free stream static pressure
T_∞	-	free stream stati temperature
AoA	-	angle of attack
$\delta_{slat,flap}$	-	relative deployment angle of lift surfaces

Differential Operators

$\overline{\nabla}(\cdot)$	-	gradient
$\overline{\nabla} \cdot (\cdot)$	-	divergence
$\dfrac{df}{dx}$	-	derivative
$\dfrac{\partial f}{\partial x}$	-	partial derivative
$\int_{\Omega}(\cdot)d\Omega$	-	volume integral
$\oint_{\partial\Omega}(\cdot)d\Sigma$	-	closed surface integral
$d\Omega$	-	volume integration element
$d\Sigma$	-	surface integration element

Finite Volume Notation

$\underline{\boldsymbol{q}}$	-	vector of conservative variables
$\underline{\boldsymbol{p}}$	-	vector of primitive variables
$\overline{\underline{\boldsymbol{q}}}_i$	-	averaged vector of conservative variables on element i
$\overline{\underline{\boldsymbol{p}}}_i$	-	averaged vector of primitive variables on element i
$\underline{\boldsymbol{F}}$	-	vector of flux terms
$\underline{\boldsymbol{F}}_{conv}$	-	vector of convective fluxes
$\underline{\boldsymbol{S}}$	-	vector of source terms
$\underline{\boldsymbol{R}}$	-	vector of residuals
$\underline{\underline{\boldsymbol{J}}}$	-	jacobian matrix
$\underline{\boldsymbol{r}}_i$	-	$i-th$ eigenvector of the jacobian matrix
λ_i	-	$i-th$ eigenvalue of the jacobian matrix
C_i	-	finite volume element i
$\partial C_{i,j}$	-	interface between elements i and j
$area$	-	finite volume element area (2D)
vol	-	finite volume element area (3D)
$(\cdot)^L$	-	left value of property in Riemann problem
$(\cdot)^R$	-	right value of property in Riemann problem
ϕ_i^j	-	$i-th$ Riemann invariant across $j-th$ wave

Multibin Notation

$(\cdot)^i$	-	$i-th$ bin property

Splashing Notation

$(\cdot)_{*,i}$ - incident (pre splashing) property

$(\cdot)_{*,s}$ - splashed (post splashing) property

θ - angle between droplet velocity and surface tangent (may be θ_i or θ_s)

Acknowledgements

Code

```cpp
void CGiuSolver::ThankPeople() {
    // Define array of people to thank
    Person *people_array = new Person[nPeople];

    // Populate array of people to thank
    CGiuToolbox::RetrieveHelpfulPeople(people_array);

    // Iterate over array of people to thank and thank them
    for (unsigned short iPerson = 0; iPerson < nPeople; iPerson++) {
        people_array[i]->OutputThankForSupport();
    }

    // Clear array from memory (do not get offended please,
    // memory management is very important)
    if (people_array != nullptr) {
        delete [] people_array;
    } else {
        cout << "You␣lonely␣c**t." << endl;
    }
}
```

Output

```
> Thank you Professor Re!
> Thank you AeroPuladri!
> Thank you Professor Guardone!
> Thank you /*merda*/!
> Thank you Trenord!
> Thank you Oncoematologia Pediatrica di Padova!
> Thank you La famiglia!
> Thank you Pausa Macchinetta / Cafferino?!
> Thank you Samuele Coinquilino!
> Thank you Moto Pizza!
> Thank you BeeP!
> Thank you Staff Mensa Serist!
> Thank you La 54!
> Thank you Ospedale di Vittorio Veneto!
> Thank you Italo!
> Thank you Bellosta Tommaso!
> Thank you Pan Sushi!
> Thank you Il treno delle ’48!
> Thank you The two remaining brain cells in my skull!
> Thank you Caffe’!
> Thank you Music!
> Thank you Computer!
> Thank you Trenitalia!
> Thank you Cooking Time!
> Thank you PoliMIce Progress Meeting!
> Thank you ATM!
> Thank you Citta’ della Vittoria!
> Thank you InDomus!
> Thank you Nicole!
> --More--(2%)
```