**Secondary atomization of Droplet in a uniform airfow**

(Study of bag-mode droplet breakup)

**Internship Report**

Duration: May 16th, 2023 – July 17th, 2023

Location: IIT Jodhpur, Rajasthan

(Submitted by)

Student:

Parthib Das (21MF10027)

(Submitted to)

Professor:

Sudipto Mukhopadhyay, Department of Mechanical Engineering

**Abstract**

The breakup of droplet is an important phenomena to study, that is used in various daily activities like fuel injection, atomization in combustion engines, raindrop damage to supersonic vehicles.

The droplet breakup is controlled by four independent dimensionless parameters:

1. Weber number
2. Reynold’s number
3. Density ratio
4. Viscosity ratio

Based on these, the breakup can occur in various regimes, namely, bag formation, bag-stamen, multibag and shear breakup. In our simulation, we will focus on two Weber number, 14 and 54. We also wish to see the changes in both the diameters, on different values of Poisson problem tolerance, and the maximum error in f and u.

**Decoding the code**

1. **2D Axisymmetric Simulation -**

**Header Files:**

*axi.h* – It is used when object has rotational axis around z-axis. When considering the cylindrical coordinate system, the longitudinal (z) axis is x and radial coordinate is y. Thus y cannot be negative in axisymmetric. It includes –

1. Refinement of ‘cm’ and face vector ‘fm’ based on Adaptive Mesh Refinement and axisymmetry.
2. Updating ‘cm’ and ‘fm’ based on embedded figures (if present)
3. ‘cm’ takes into account the area and volume scalar fields whereas ‘fm’ is the length of the faces.

*navier-stokes/centered.h* – It computes the values at each grid point by considering the values at neighboring grid points symmetrically around the point of interest. The derivative of a point can be approximated by shape between the values at adjacent grid points. It uses the generic time loop, a CFL limited timestep, the Bell-Collela-Glaz advection scheme and implicit viscosity solver. It incorporates time integration, linearization, boundary conditions, adaptivity and projection. The governing equations are ***Continuity Equation and Navier Stokes Equation.*** The primary variables are

1. Centered pressure field *p*
2. Centered velocity field *u*
3. Centered pressure gradients and acceleration terms in *g*
4. Auxilliary face velocity field *uf*
5. Associated centered pressure field *pf*

Stattistics for multigrid solution of pressure Poisson problem and implicit viscosity are stored in *mgp, mgpf* and *mgu.*

The Boolean *stokes* is set to false so that velocity advection term is not omitted.

In the boundary conditions, the pressure and velocity of top/bottom and front/back are given default 0. The default CFL is 0.8

Pressure is never dumped.

*two-phase.h* – It takes into account multi-phase flows of two immiscible fluids within a computational domain. It incorporates interface tracking, surface tension and phase properties like density, viscosity and interfacial tension. It also has a option of smearing the density and viscosity jumps in the regions of mixed cells. It is generally the arithematic mean of them.

The volume fraction in fluid 1 is f=1 and f=0 in fluid 2. Thus, the densities and viscosity are rho1, rho2, mu1, and mu2 respectively for fluid 1 and 2.

*navier-stokes/conserving.h –* It implements the momentum conserving VOF advection of velocity for two-phase Navier-stokes solver. It gurantees conservation of each component of total momentum and not for each phase. The advection scheme of centered solver is switched off by setting the Boolean stokes to true.

There are two components of momentum:

q1 = fρ1u and q2 = (1-f) ρ2u

Note that navier-stokes/conserving cannot work alone but needs the centered solver before calling this header file.

*tension.h -* It helps in keeping track of surface tension and all the function and macros related to interfacial tension. It takes into account the surface tension forces, interfacial curvature, surface tension model and timestep stability. The governing equation is

Interfacial force density = Φ∇f , where,

f- volume fraction describing interface

Φ = σκ (potential)

σ = surface tension

κ = interface mean curvature

The surface tension scheme used is time-explicit. So the maximum timestep is

*T*=sqrt((*ρm*​Δ*min*^3)/​​​ *πσ*) with   
*ρm = (ρ1 + ρ2)/2* where p1 and p2 are densities on either side of interface.

*View.h* – It contains various commands for graphical representations of Basilisk fields that includes volume of fluid reconstruction, colorscale representation of scalar fields. It takes the help of drawing user functions defined in draw.h

Default format in which output is saved in “ppm”. We can convert it to png,jpg,mp4,gif,ogv etc

**Physical properties:**

Density of liquid = 824 (ρ1)

Density of gas = 1.1856 (ρ2)

Viscosity of liquid = 2.17e-3 (µ1)

Viscosity of gas = 1.84e-5 (µ2)

Diameter of droplet = 1.98e-4 (D)

Velocity of stream = 67.8282 (u)

Acceleration due to gravity = 9.81 (a)

Liquid surface tension = 2e-2 (σ)

Maximum runtime of simulation is made 15 millisecond.

We define “tsh” as shear time. This value is used while comparing our data with the results of literature.

**Level of refinement:**

Minimum level of refinement we used is 6 (64 cells) and maximum level is 12 (4096 cells).

**Boundary Conditions:**

Boundary conditions are set at the inlet(left) and outlet(right) only for velocity and pressure fields. Rest of the undeclared conditions are set to zero by default as mentioned in N-S/centered header file.

u.n represents normal direction velocity and u.t represents tangential velocity. Dirichlet assigns direct value and neumann assigns derivative of the field.

We assumed fully developed flow at the outlet, thus making

u.n[right] = neumann(0), i.e. no further change in u.n

Pressure at outlet is kept at zero and at inlet it’s derivative is kept zero, because fluid always flows from high pressure to low pressure.

The user can give their own maximum level of refinement through the command line argument. It is parsed in the main function.

**Setting up domain:**

The length of domain is kept as 15D on both X and Y axis.

The origin of the computational domain is given as (-D,0). Thus, the box of the domain ranges {(-D,0), (14D,15D)}

The initial grid is kept at minimum level of refinement i.e. 6

Thus, the initial cells per radius(CPR) is 2 and final is 136

Both the CFL and tolerance for Poisson problem are overwritten. ***Note that default CFL is 0.8 and tolerance is 1e-3***

The physical parameters are set by the initial macros.

**Event init:**

We will first check if there is any “restart” file, that will help in restoring the conditions from a previous simulation. The *restore* function will look for that file whereas *dump* function creates such file.

If there is no such file, the simulation will start by refining the region between two concentric circles center at absolute origin and radius between 0.6D and 0.4D. This will ensure that the grid is refined only in the region of interest. The level of refinement is confined up to maxlevel (12), thus preventing excessive refinement.

The *fraction* function takes the surface interface of the droplet in the form of an equation of a circle with center on absolute origin and radius 0.5D. This is the levelset function Φ, sampled on the vertices of grid. By convention, the Φ > 0 for inside of interface.

The scalar f, defined in two-phase header file, is given default value zero, whenever phi is negative and 1 , whenever phi is positive. For mixed cells (where both liquid and gas exist), appropriate value is assigned to f.

**Event end:**

We end the runtime of simulation as per the macro, defined in the beginning. We will print the number of iteration and simulation time in the *out* file, along with the grid-type used, CPU and real time, and speed of simulation(points per sec)

**Event acceleration:**

In our case, a droplet is supposed to fall through a uniform airflow. Thus, it is necessary to take acceleration due to gravity in the scenario.

We introduce a local face vector variable for a (defined in N-S/centered header file), to compute the effects of acceleration due to gravity more accurately. It will assign the value a, to each faces of the grid cell wall. Next we loop through the x faces only, and add the macro for acceleration to each av.x[] field

**Event adapt:**

The smart adaptive mesh refinement of grid (of Basilisk C) is called in this event. We give the maximum error of u as 1e-3 and that of f as 1e-3. Note that only the maxlevel of refinement is mentioned only. We are not confining the lower limit of grid adaptation.

**Event diameter\_data:**

This event will write the change in diameter at every iteration, in a file *diameter*. We declare two scalar variables *pos\_x[]* and *pos\_y[]* , storing the streamwise diameter and cross-sectional diameter respectively.

We use the pre-defined function *position* for computing the position of the interface. It fills the *pos\_x/y[]* with position defined by f and the desired direction vector. Thus, pos\_y should have y axis as the direction vector and similarly the pos\_x should have x axis. The position is stored using *statsf*() function.

Cross-sectional diameter = 2.\*statsf(pos\_y).max

Streamwise diameter = statsf(pos\_x).max - statsf(pos\_x).min

**Event logfile:**

The following useful data are written in the *log* file:

1. Iteration
2. Timestep
3. Time of simulation
4. No of iteration for multigrid pressure solver
5. No of iteration for multigrid velocity solver
6. No of iteration for multigrid centered pressure field
7. Performance time
8. Performance speed
9. Total no of grids

**Event interface:**

The coordinates of the interface are written at every 1000th interval, saved in a *interface* file. The function *output\_facets()* writes the coordinates of two endpoints of a line segment. These line segements when plotted in *Gnuplot, Matlab, Excel, etc* we can get the interface shape

**Event snapshot:**

Snapshot of the droplet is saved in .ppm format, at every 1000th iteration. The *output\_ppm()* draws the field *f*(vol. of fluid) in a 800 pixels image. The default value of spread 5, is used. The linear interpolation of colorfield is set to *true*. Thus, the image will have a smooth transition of colors.

**Event restarting:**

The function *dump* makes a file with the name *restart*. It will store all the list of scalars (namely u.x, u.y, u.z, f, a.x, a.y, a.z). It works together with *restore* function to restart the simulation from certain point.

**Event movie:**

A movie *vof* is made mp4 format. It uses the same function as in snapshot. The special command ppm2mp4 converts every ppm image(produced at every 100th iteration) to a movie. The bounding box for the movie is defined.

Spread is -1. Max and min are set to average of the field plus (resp. minus) *spread* times the standard deviation. The map, for colorscale, is­ *cool\_warm.*

**Event viewing:**

This function produces grid images at every 1000th iteration.

The camera center point is at (-0.5,-0.5). The height and width of image produced is 1024

*clear()* removes all objects previously drawn

*draw\_vof()*  command display the VOF-reconstructed interface of field *f*

*box()* command draws the bounding box of the domain with no tics mark (axis coordinates) present.

*cells()* command displays grid cells

1. **3D Simulation –**

**Header files:**

*grid/octree –* Sets the macro, *DIMENSION* to 3. Thus, making the simulation to run in 3D. This grid is compatible with adaptive mesh refinement, unlike *multigrid/3D*

Rest of the header files remains the same as axisymmetric 2D case, expect the use of *axi*

**Physical parameters:**

Remains the same as axisymmetric 2D case

**Level of refinement:**

Minimum level = 6

Maximum level = 10, since it requires high computational time.

**Boundary Conditions:**

Remains the same as above. The front and back BC are already taken care in the N-S/centered header file

**Setting up domain:**

The length of domain is given 16D in all 3 axes.

The origin of the coordinate is kept at (-3D, -8D, -8D). This will ensure that the droplet is kept in the middle of yz plane.

The initial grid is kept at minimum level of refinement i.e. 6

CFL is 0.5 and tolerance of Poisson problem is kept to 1e-6

The physical parameters are set by the initial macros.

**Event init:**

Same as axisymmetric 2D case.

Since we are dealing with 3D coordinate, we will add the z component in the equation of circles in both *refine* and *fraction* functions.

**Event end:**

Same as axisymmetric 2D case.

**Event diameter\_data:**

Same as axisymmetric 2D case.

**Event logfile:**

Same as axisymmetric 2D case.

**Event interface:**

Same as axisymmetric 2D case.

**Event viewing:**

The camera center point is kept at (-0.29, -0.03, -1.483). The *width* and *height* of the image is 1548 and 970 repsectively. The background is given white color. The field of view is given 30. This gives the angle between the leftmost and rightmost view of the camera. *Near* and *Far* gives the distance between the nearest and farthest distance of the viewing plane. Any object closer than *near* won’t be visible and vice-versa with *far.* The *quat* array gives us the appropriate orientation of the camera. It gives us the angle of rotation around each axes.

***\*Note that these values are taken from BView3D.***

We draw the interface of the droplet using *draw\_vof* function. It draws *f,* and colors it using the magnitude of velocity. The minimum of maximum of the colorfield is given as zero and *USTREAM*. The linear interpolation of color is set to *true*, thus giving smooth transition of color

The image is taken at every 250th iteration and saved in png format

**Event viewing:**

The same values for *view* is given.

Box is drawn with no tics.

Here the grid cells are displayed in the XY plane. Images are taken at every 500th iteration.

**Event snapshot:**

Same as axisymmetric 2D case.

**Observation**

For a low Ohnesorge number (Oh<0.1), the breakup is in the bag-mode formation.

Bag-breakup for We>11 followed by multimode breakup. Formation of multiple bags for 30<We<80. Shear stripping mode for 80<We<350. Beyond We=350, there would be catastrophic breakup of drop. The cause of such breakups is explained using the Rayleigh-Taylor instability (RTI).

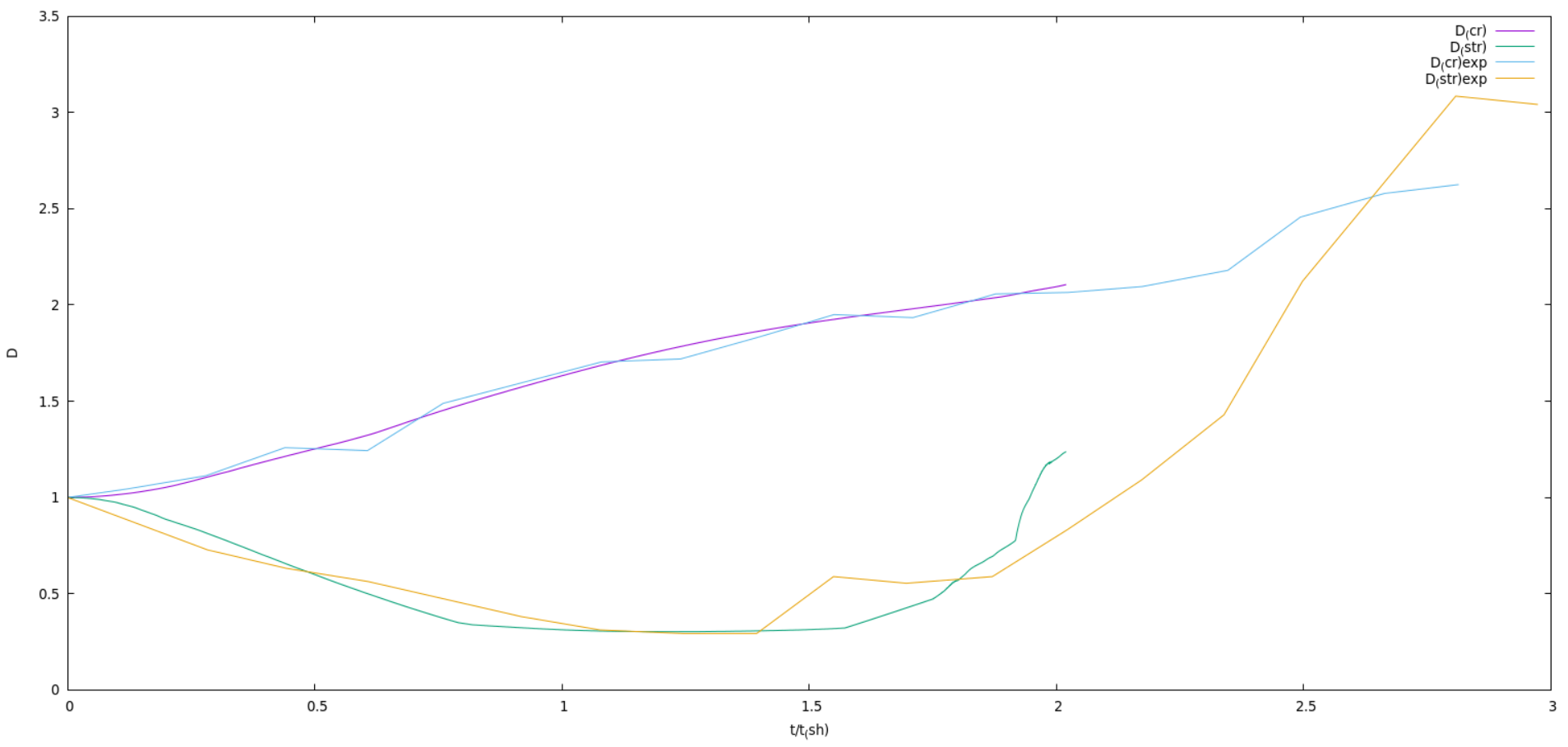
We first use We = 14 and vary different tolerance and error estimate values to check our result with that done in Ansys Fluent.

1. Poisson tolerance = 1e-4

uemax = 1e-2

femax = 1e-2

Along with f and u, another scalar quantity ff[] is used in adaptive mesh refinement, to take take-care of kinetic energy dissipation of the droplet. It’s maximum error in refinement is 1e-2. Below is the graph of diameters comparing the two results:

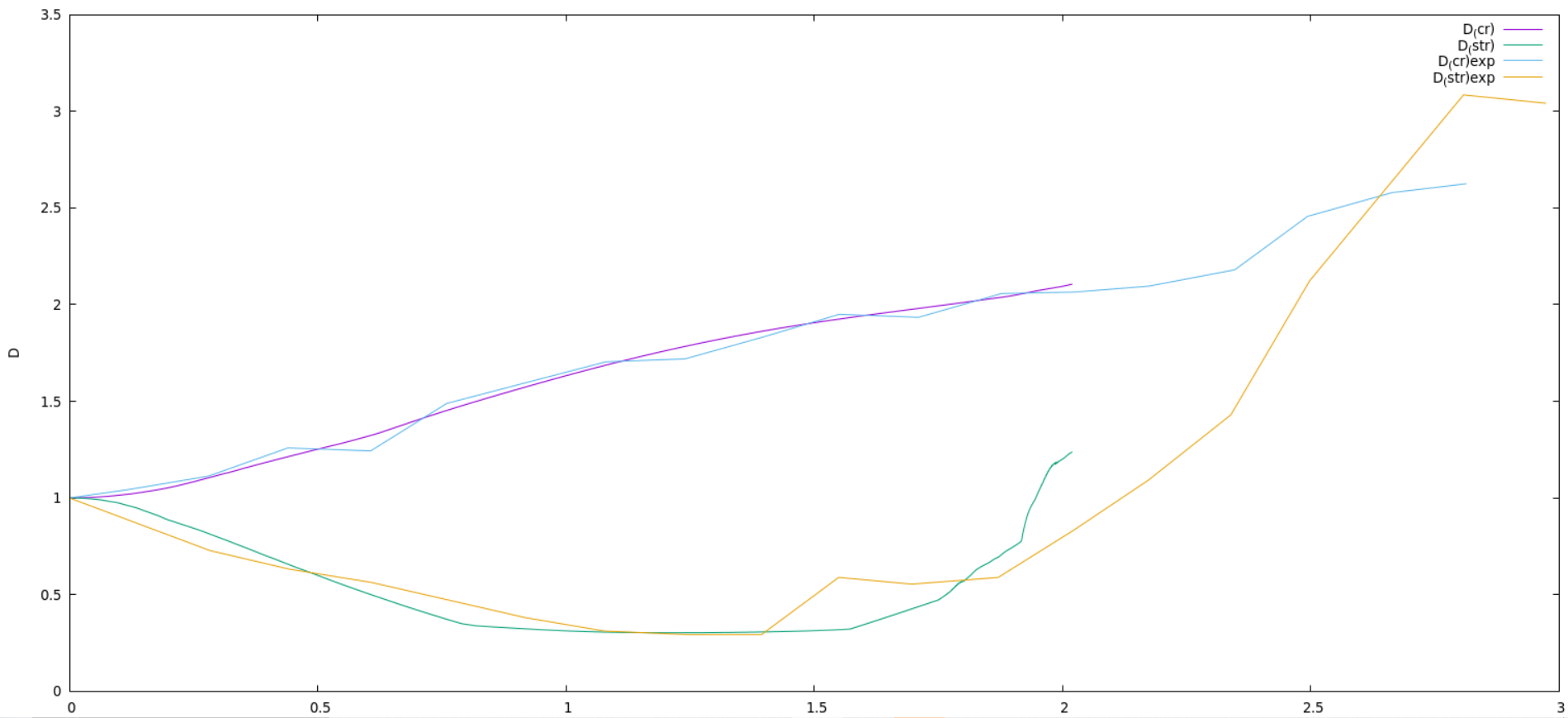


1. Poisson tolerance = 1e-6

Uemax = 1e-2

Femax = 1e-3

Same as above, ff[] is used in AMR, but with maximum tolerance 1e-3. Below is the graph of diameters comparing both the results:

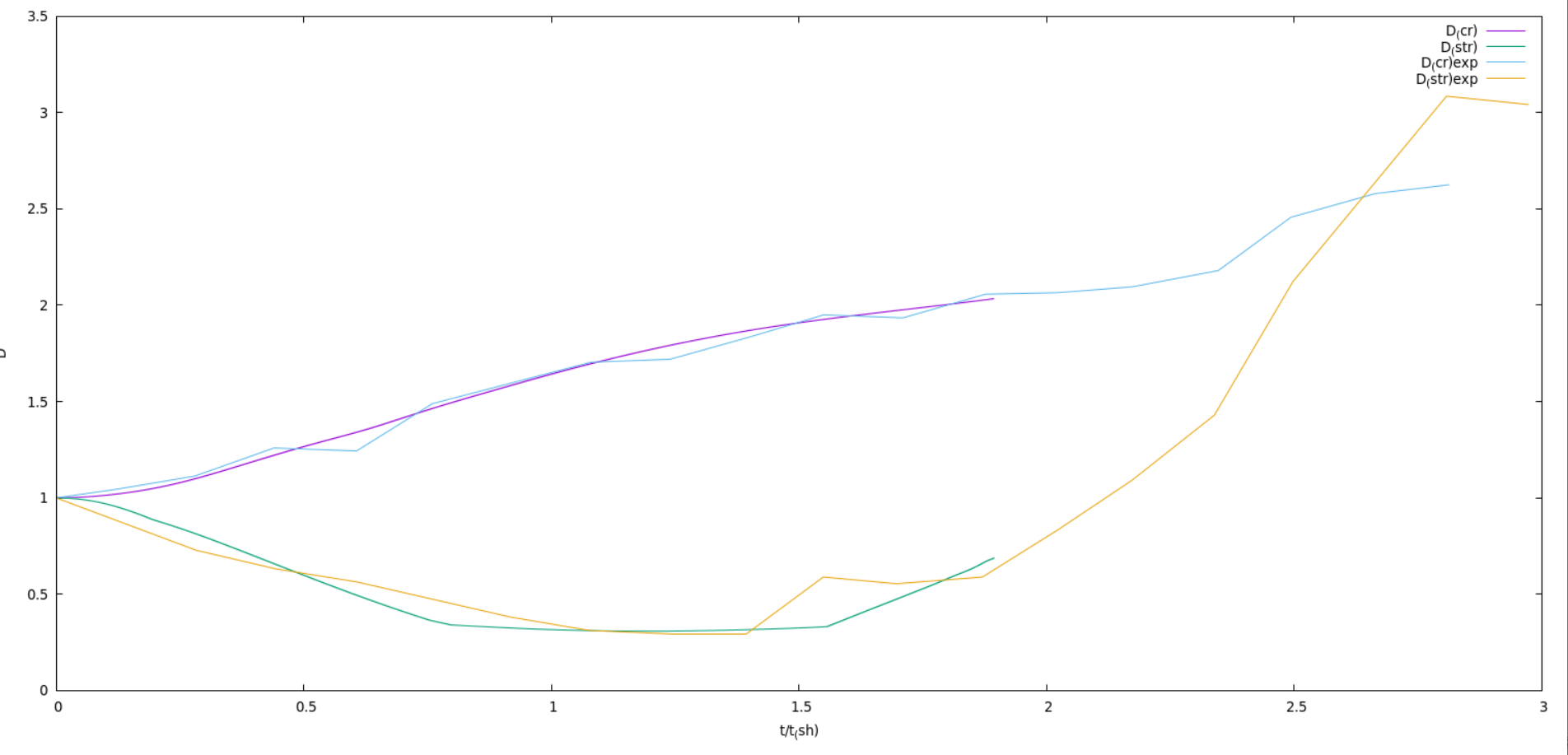


1. Poisson tolerance = 1e-6

Uemax = 1e-3

Femax = 1e-3

Below is the graph of diameters comparing both the results:

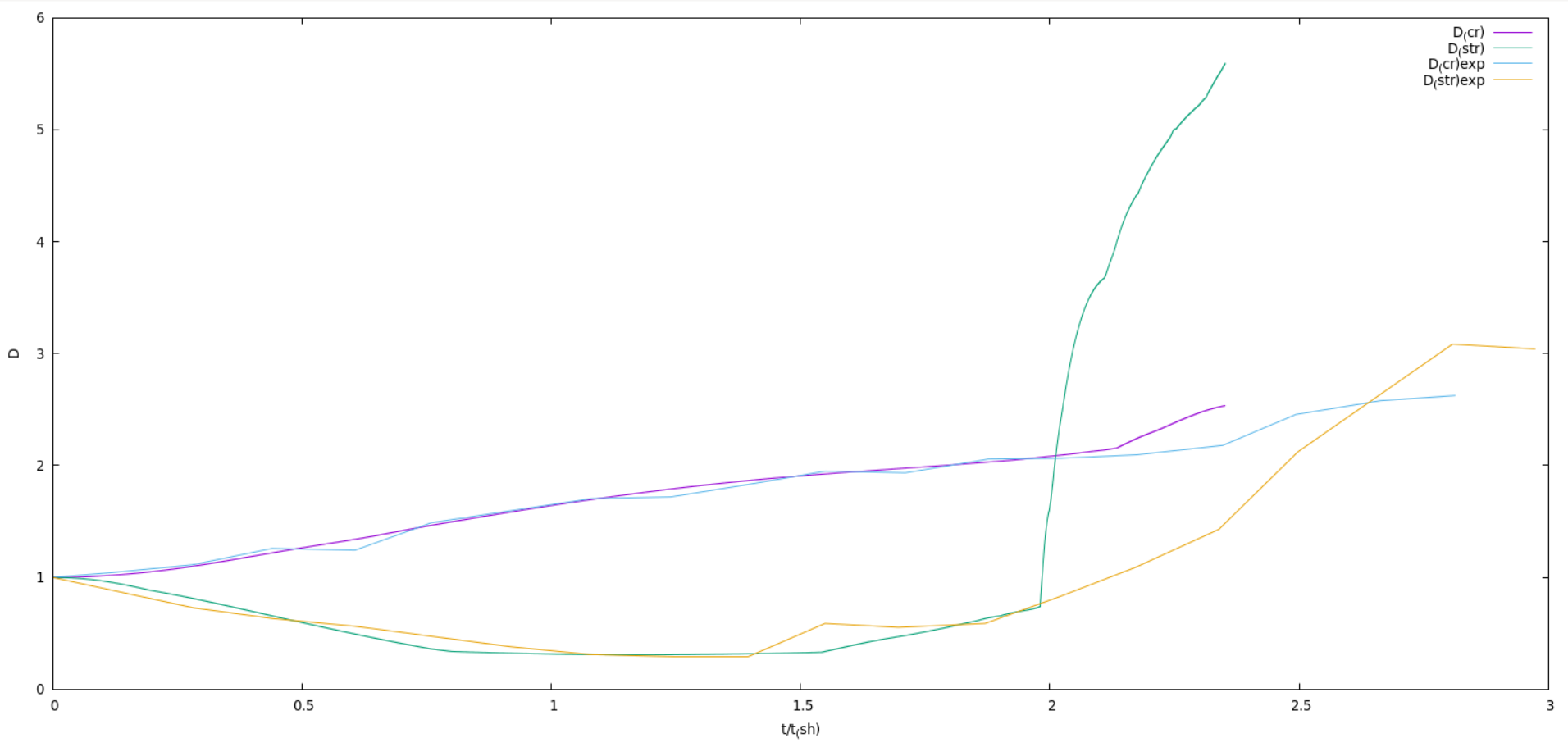


1. Poisson tolerance = 1e-4

Uemax = 1e-2

Femax = 1e-3

Below is the graph of diameters comparing both the results:



We have plotted the interface of the droplet for two different cases at different time intervals.

1. We = 14

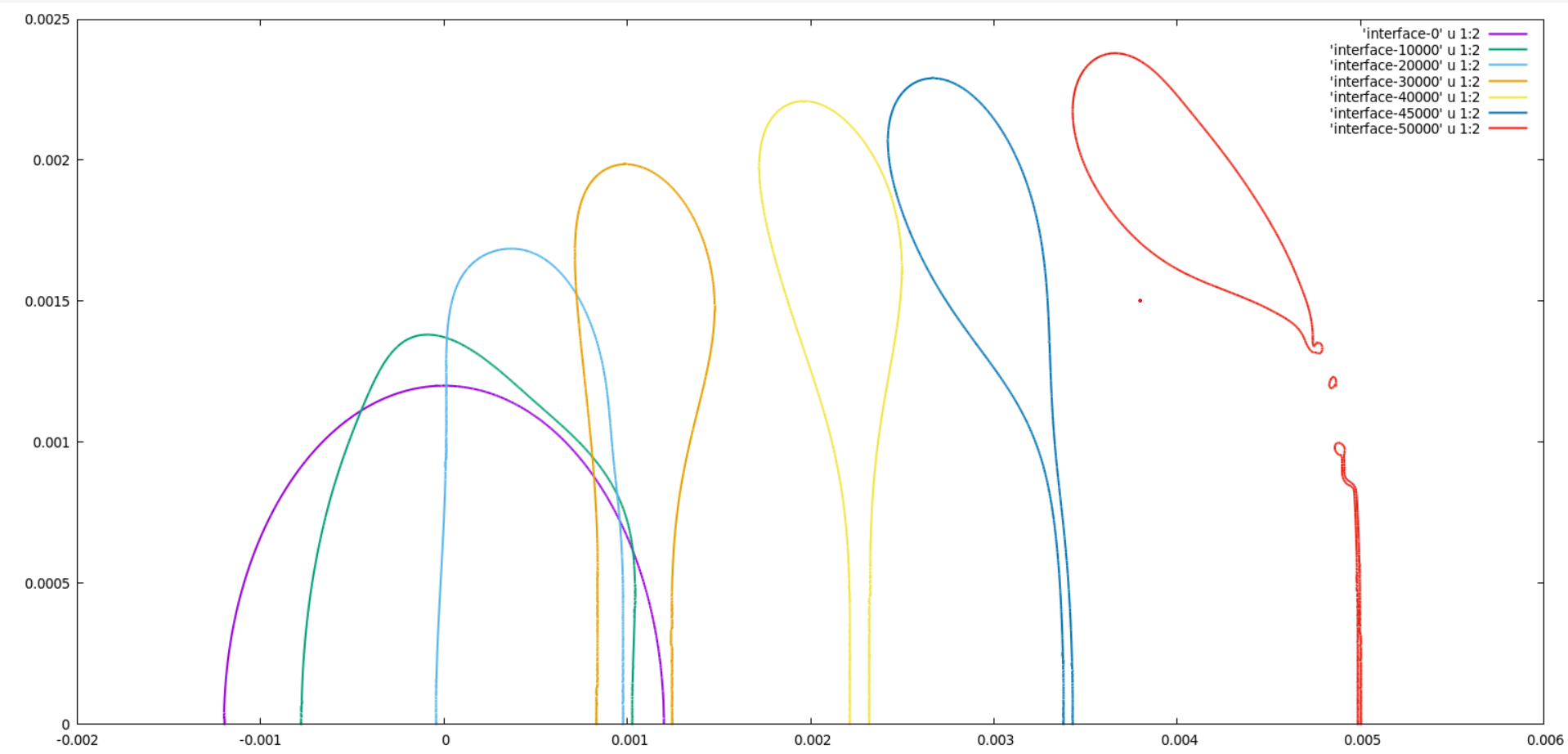
Poisson tolerance = 1e-4

uemax = 1e-2

femax = 1e-2

ffemax = 1e-2

CFL = 0.25



We can clearly see that the droplet deforms into a single disc shape, followed by formation of forward hollow bag attached to a rim. The thin bag breaks up into tiny droplet, owing to Rayleigh-Plateau type instability

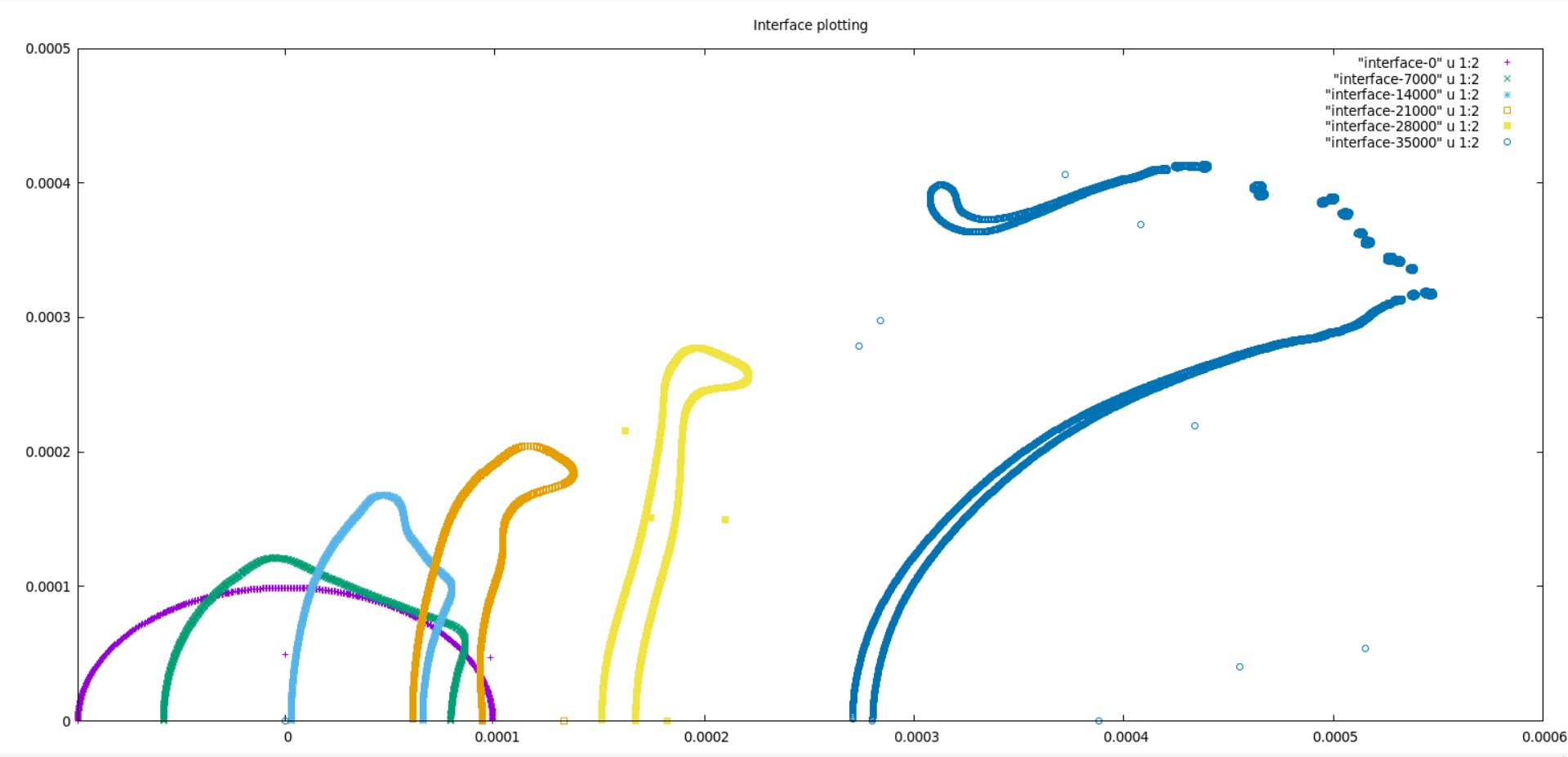
1. We = 54

CFL = 0.25

Poisson tolerance = 1e-6

Uemax = 1e-3

Femax = 1e-3



We can clearly see that the droplet breaks in multimode bag formation. The spherical droplet undergoes a bulge in the centre, leading to a stamen formation. Also, the leeward side of drop shows a depression towards the airflow side. This is in contrast to the above case (with We = 14), where the leeward side is merely stagnates and forms depression in downward direction.