THE PENNSYLVANIA STATE UNIVERSITY

SCHREYER HONORS COLLEGE

DEPARTMENT OF ENGINEERING SCIENCE AND MECHANICS

IMPROVED NUMERICAL METHODS FOR CAVITATION MODELING

ASISH T. BALU

Spring 2016

A thesis

submitted in partial fulfillment

of the requirements

for a baccalaureate degree

in Engineering Science

with honors in Engineering Science

Reviewed and approved* by the following:

Michael P. Kinzel

Research Associate at the Applied Research Laboratory

Thesis Supervisor

Judith A. Todd

P. B. Breneman Department Head Chair

Professor, Department of Engineering Science and Mechanics

* Signatures are on file in the Schreyer Honors College and Engineering Science and Mechanics Office.

ABSTRACT

Cavitation dynamics of nuclei are largely governed by the Rayleigh-Plesset Equation. This research analyzes various approaches to solving the RPE that decrease computing time and power with a minimal loss of accuracy. These approaches are conducted in both the numerical integration of the RPE as well as in the implementation of the RPE into CFD models.

First, a number of singularity-handling algorithms that traverse the Rayleigh-Plesset Equation are explored. In order to maintain a constant time step size while maintaining solution quality, the RPE is put through a momentum conservation test to ensure the solution recovers symmetry across collapse events. This increase in efficiency and accuracy allows the program to provide useful solutions in the field of fluids engineering, particularly in the study underwater explosions, optimization methods, and other such applications.

Next, the RPE is implemented in a computational fluid dynamics solver. This requires an extension of the equation from a Lagrangian framework into an Eulerian framework through the use of a material derivative. In addition, formulas are developed to impose upper and lower bounds for the bubble parameters, thereby allowing the use of larger time steps. Cavitation growth and collapse are analyzed in three different flow situations: a simple square model, bubble formation in a shaken bottle, and flow over a cylinder. Data from the models is extracted and validated against numerical solutions, thus showing that the RPE can be implemented efficiently in CFD within an Eulerian framework. This is especially useful when considering simulations with high bubble density because Eulerian methods are better suited than Lagrangian methods to handle large numbers of particles.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank Dr. Michael Kinzel for providing me with the opportunity to conduct cavitation research and helping me every step of the way. In addition, I would like to acknowledge Dr. Scott Miller for reviewing parts of my paper as well as my parents for getting me interested in science from an early age.

Thank you all!

Table of Contents

List Of Figures

List of Tables

Chapter 1

Introduction – What is Cavitation?

In considering fluid mechanics of liquids, evaporation involving cavitation is often a concern. Cavitation is typically classified as evaporating liquid processes driven by pressure variations that create vapor filled cavities. Clearly other methods of evaporation occur, such as boiling due to temperature increases, combinations of temperature and pressure changes such as explosions, or the slow-evolving, common evaporation processes. The distinguishing feature associated with cavitation is that this type of evaporation manifests itself through local pressure fluctuations that drive the liquid well into the vapor state. Some applications include marine craft (propellers, lifting surfaces), secondary events in underwater explosions (UNDEX), and natural processes such as shrimp debilitating prey (Versluis et al., 2001), sea-cliff erosion (), etc.

Cavitation is often seen as bubbles whose radii grow and collapse with time. However, smaller cavitation on the magnitude of micrometers is also possible, and this is usually not visible without the use of special tools. The figure below shows a relatively common cavitation event – an underwater propeller generating vortex bubble streams. In most cavitation cases, including the one below, the lower pressure region behind the propeller is what initiates bubble growth.

Figure 1 – Cavitation from spinning underwater propeller blades (Source: Wikimedia Commons)

Although small cavitation events rarely cause significant damage to engineering materials, repeated implosion of bubble particles near a surface can result in catastrophic wear and tear. For instance, Figure 2 below shows a large hole caused by bubbles exploding near the inner walls of a deaerator.

Figure 2- Cavitation corrosion on the inside surface of a deaerator feed tank (Source: Roberge)

Note that the material here is steel, thus showing the destructive power of cavitation. A term commonly used to refer to the erosion of a material due to imploding bubbles is "cavitation corrosion." Simulating cavitation events is crucial to detecting regions susceptible to cavitation corrosion, and accurate models can prevent expensive or even deadly failures from occurring in the future.

Cavitation Modeling and Solution Frameworks

Cavitation is comprised of two overall steps. First, bubble nuclei form at contaminants, dissolved gas, or other impurities in the liquid. These nuclei are small sites where cavitation initiates (Brennen, 2013). As nuclei are exposed to local decreases in pressure, they can grow into cavities. Conversely, an increase in pressure will cause them to collapse. Bubble growth and collapse can be modeled by the second order Rayleigh-Plesset equation (RPE) (Rayleigh, 1917; Plesset, 1949) shown below:

$$\frac{P_B(t) - P_\infty(t)}{\rho_L} = R\frac{d^2R}{dt^2} + \frac{3}{2}\left(\frac{dR}{dt}\right)^2 + \frac{4\nu_L}{R}\frac{dR}{dt} + \frac{2S}{\rho_L R} \tag{1}$$

In this equation, R is the radius of the bubble, thus, dR/dt and d^2R/dt^2 are the radial velocity and radial acceleration, respectively. Pressure is defined inside the bubble, i.e., $P_B(t)$, which is a combination of the liquid's vapor pressure as well as the gas within the bubble, as well as in the far-field $P_\infty(t)$. The liquid properties are defined as density, ρ_L, kinematic viscosity, ν_L, and surface tension, S. Application of RPE is typically used to assess the transient physics of cavitation events such as snapping shrimp cavitation events (Versluis et al., 2001), or UNDEX (Miller et al, 2013). However, it is also used in more complex computational fluid dynamics models (CFD), where it is used to approximate cavitation physics (Singhal et al., 2002; Hosengadi & Ahuja, 2005), or directly couple to CFD models (i.e., Chahine, 2004). In these approaches, numerical integration of RPE involves far-field pressure variations from a CFD model to determine R(t).

A goal of this thesis is to explore numerical solution approaches to cavitation events. Thus, I review various methods to numerically solve the Rayleigh Plesset Equation and provide ways to increase their efficiency. Standard solution approaches use ordinary differential equations (ODE) methods with a variable time-step size (VTSS) to produce an accurate solution. However, VTSS approaches may not be ideal for all applications. For example, in the coupling of RPE solutions to CFD models (Chahine, 2004), this may require syncing of the CFD model (which generally use a constant time-step size) to a VTSS RPE solution method. Ideally, the models will be matched in time and the RPE solution method does not dictate the time step size. Therefore, investigating a constant-time-step methodology to the RPE is investigated.

A challenge of CTSS methods is that that during collapse an initial growth of bubbles, the velocities and accelerations are very high, demanding small time step sizes for a stable solution. Thus, CTSS methods require temporal resolution at 10,000-100,000 time steps per a cavity oscillation. However, it is found that this issue can be mitigated by jumping over the collapse and initial expansion of a bubble and implicitly correcting for the physical processes the models skipped. The result is a CTSS RPE model that is of comparable accuracy as VTSS methods, while far more efficient than standard CTSS methods.

The next goal of my thesis is to implement the RPE into CFD models. Implementation of the RPE into CFD models is composed of two procedures. First, I develop a method to stabilize CFD solutions even when using larger time steps. My method involves creating “clip” values for the bubble radius, radial velocity, and radial acceleration. These “clips” are simply the upper and lower bounds for these three parameters. Using equations based on the physics of the bubble as well as the structure of the CFD mesh, I formulate lower and upper bounds for the bubble parameters, preventing them from increasing or decreasing to unreasonable values and becoming unstable. This allows the time step to be increased and the convergence time to be reduced; a solution that previously took days now only takes a few minutes.

In addition to the “clip” values, implementation in CFD requires that the RPE be extended from a Lagrangian framework into an Eulerian framework. A description of these two frameworks is provided below.

Lagrangian methods treat the bubble as a particle moving through a fluid. The CFD solver keeps track of each individual bubble’s current position, radius, and other kinematic properties. Data from Chahine (2010) show that a spherical bubble in compressible medium can be modeled in a Lagrangian framework by the

Gilmore equation, which is a variant on the Rayleigh Plesset Equation that takes into account enthalpy differences as well as wave propagation speed in the medium. The equation is shown below:

$$\left(1-\frac{\dot{R}}{C}\right)R\ddot{R}+\frac{3}{2}\left(1-\frac{\dot{R}}{3C}\right)\dot{R}^2=H\left(1+\frac{\dot{R}}{C}\right)+\frac{R\dot{H}}{C}\left(1-\frac{\dot{R}}{C}\right) \tag{2}$$

In this equation, all parameters are the same as in the original Rayleigh Plesset Equation, except that C represents the speed of sound in the medium and H represents the enthalpy difference between the bubble surface and medium at infinity. Models using this framework produced accurate results in the early stages of bubble expansion and collapse, but failed near the end due to unforeseen gravitational effects (Chahine et al.).

Eulerian methods treat the entire medium as one entity and describe the properties of the cavitating flow at each position. At each point in the flow, the bubble parameters are tracked over time. This allows the CFD solver to display the radius, bubble velocity, and other variables at a certain place on the flow. This will be especially helpful when determining where cavitation will occur. Singhal (2002) implemented the Rayleigh Plesset Equation in the Eulerian framework to analyze pressure distributions and cavitation over a hydrofoil.

In this paper, I will use an Eulerian CFD solver to understand the effect of the singularity handling approach on the overall flow parameters at all points in the flow medium. Using a Lagrangian method would be inefficient, as it would require the solver to not only track each bubble's parameters, but also the motion of each individual bubble in the flow field.

The thesis is split into two subsections and is outlined as follows. The first subsection deals with singularity handling in numerical solutions to the RPE. Initially, I examine the two standard numerical solutions commonly used to solve ODEs in the context of RPE. The cons of each method are discussed. Second, various attempts taken with the present work to solve these problems are described within the context of "singularity handling," which is an approach that implicitly captured and recovers singularities in a way that locally (temporally) modifies the RPE solution approach. Thirdly, RPE solutions are generated versus traditional methods indicating both that the present model produces accurate results with less computing power. Each solution is tested with different pressure profiles, and the error is analyzed for various scales (approximate number of time steps per oscillation). Next, I validate my solution by testing the approach for an UNDEX case based on data from Swift and Decius (1950), similar to the approach used by Miller et al (2013).

The second subsection of the thesis deals with implementation of the RPE into a CFD solver. Here, I extend the RPE into an Eulerian framework through the use of material derivatives. Next, I formulate clip values for radius, radial velocity, and radial acceleration; the clips for the first two parameters are given by Brennan (2013). To calculate the clip value for radial acceleration, two separate methods were used. One method involves calculating all possible values of the RPE given the clip values for radius and radial velocity, and the other method imposes a restriction on the bubble radius given the CFD model's mesh density. The RPE implementation is tested in three cases: a simple square model, a bottle slam, and a flow over a cylinder event. Finally, validation of the CFD model is conducted against a numerical solution using pressure data extracted from the converged solution. I show that the RPE can be used in CFD within an Eulerian framework, and that the solver's efficiency can be increased greatly through the use of clip values.

Chapter 2
Solution Methods

The goal of this chapter is to solve the Rayleigh Plesset Equation numerically and examine methods to decrease solution time as well as prevent crashes. The Rayleigh Plesset Equation can be explicitly solved for radial acceleration, which means a solution for the bubble radius can be determined using a variety of solving schemes. Unfortunately, the RPE can be unstable at some points; these points are known as singularities, and at larger time step sizes the solver will crash once it reaches them. Several solving schemes are analyzed in this section. I discuss different ways of traversing the singularity with respect to these solving schemes so that the solver can continue without crashing.

In general, standard solution approaches to the RPE rely on relatively standard ODE methods. For example, in the Euler method, derivatives are calculated to update the solution once at each time step. This approach can be extended using a Runge-Kutta method, which involves advancing the calculation at multiple stages equally spaced within the interval from the current time step to the next time step. A 4-stage Runge-Kutta approach (RK4) is explored within this work. These approaches can be extended using a variable time step (VTS) approach, which improves Euler and RK4 methods by advancing the solution by a certain radius each VTS. Such an approach remains stable, accurate, and much more efficient. In terms of application, non-VTS methods require very small time steps to remain stable during collapse events. Thus, they require significant computational resources based on RPE behavior that occurs for only a small fraction of the cycle. VTS method fix this issue, however, breaking constant time step sizes may not be desirable for many applications. The objective is to alleviate these drawbacks.

Collapse events tend to dictate time-step size, thus, computational requirements, in RPE solution methods. Such events are prone to numerical singularities, thus, approaches that avoid such singularities, i.e., singularity handling, are investigated. The concept of singularity handling is provided in Figure 3 – Various Methods to Traverse Singularity. The idea is to extend the stability of RPE solution methods by detecting singularities, and moderating them via an implicit boundary condition that roughly conserves momentum by imparting an expansion event that is symmetric to the collapse. First, consider singularity detection. In the present work, such events are detected when $R+\Delta t\, dR/dt < \varepsilon$; here I set ε to 0.0. This means the program will detect a singularity if the next radius value were to be less than or equal to zero. The program will also detect a singularity if the radial speed before the cusp reaches a maximum, or in other words, when the next velocity value is lower in magnitude than the current value. Once detected, an approximation RPE-method is used, which is indicated by the red lines in the detail views in Figure 3 – Various Methods to Traverse Singularity. A number of methods to approximate the RPE at singularities were developed which are described as follows:

Method 1 (Triangle): The triangle approach involves approximating the collapse event as an isosceles triangle (Figure 3). The two sides of the triangle have a slope equal to the velocity of dR/dt at the point just before the instability. The bottom tip of the triangle is assumed to approach a radius equal to zero, which is an assumed smallest possible radius value during an oscillation. Essentially, the method imposes R and dR/dt at the next time step as be the same prior to the singularity handling event (but of opposite sign). Note that the next time level is requires an adaptive time step size that is governed by this triangle model, thus, this method is not explored in the context of this work. Nevertheless, this method is effective in a VTS method.

Method 2 (Parabolic): An improved accuracy would be expected using a higher-order function such as a parabola similar to that indicated in Figure 3. This method proved to be complex and not effective thus was dropped.

Method 3 (Jump): The idea of this method is to "jump" over detected singularities in one time step (Figure 3), and reflect the bubble behavior over the jump by inverting the sign of dR/dt and dR^2/dt^2. Such a method explicitly initiates a symmetric expansion while introducing a slight error in the time direction.

Method 4 (Triangle with back tracing, Triangle-BT): An improved, implicit, triangle model is developed. A diagram of the method is indicated in Figure 3D. The idea is to recover symmetry through a triangular model for the singularity handling, while maintaining a constant time-stepping approach. In order to model symmetry in the cusp occurring at the collapse/expansion event, there exists a point on the opposite side of the cusp with an equivalent R and inverted sign dR/dt value as the current step. Meeting this condition at this "mirror point" is required to maintain symmetry. As indicated by the dashed line in Figure 3D, the triangle model projects the solution to the mirror point. Note that it is not guaranteed that this event occur at the correct time level of interest. The concept of Back Tracing is introduced to recover this characteristic. As indicated by the blue line in Figure 3D, the back tracing occurs which is described as solving the RPE, backwards or forwards in time, to recover symmetry at the Mirror Point that does not correspond to the time of interest.

Method 5 (RPE-Approximation): To improve the singularity handling modeling, an approximation to RPE based on the work of Obreschkow et al. (2012). Such a model should improve the representation of the RPE physics as it is based on an RPE approximation. The approach is similar to the Triangle model, however, it is replaced with the RPE approximation as indicated in Figure 3E and 3F. This approach was functional for large time steps (Figure 3E), where the approximation well represented the approximated physics. However, when reducing the time step size the physics were not well approximated, and yielded a behavior similar to that depicted in Figure 3F. Because of this dependency on temporal resolution, the method was not pursued.

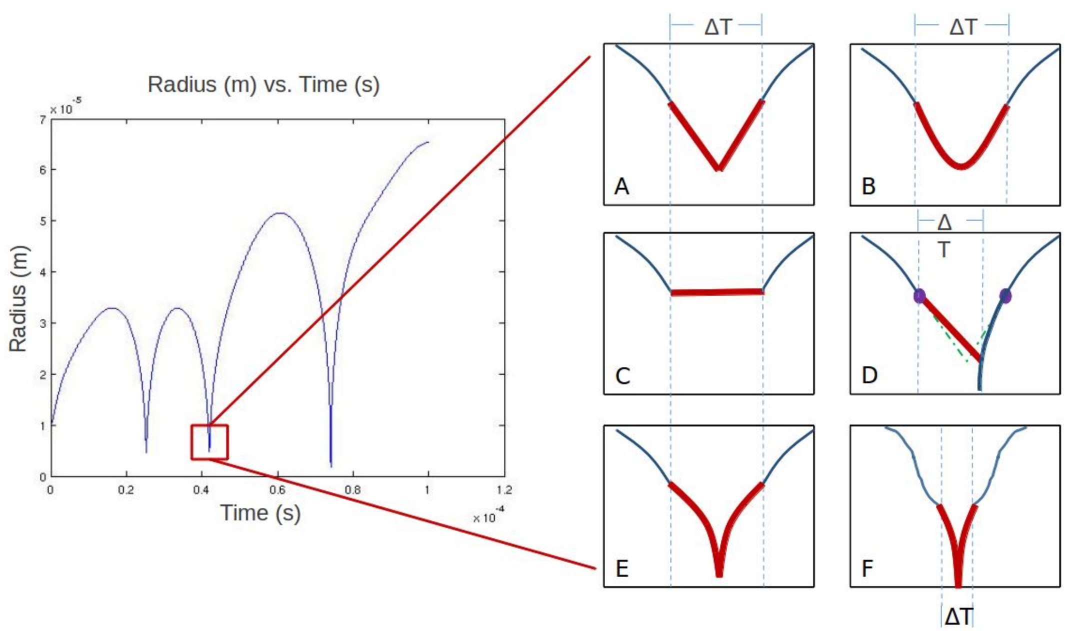


Figure 3 – Various Methods to Traverse Singularity

Assessments of Methods

Test Case Description

Each method is evaluated on the same hypothetical test case. The test case is initiated with an isolated bubble in equilibrium with properties outlined in Table 1. Note that the dynamic viscosity of the liquid is zero by assumption and has little influence in the solution. Similar to previous work by Alehossein & Qin (2007), an absolute pressure described as a piecewise step function oscillating between -1 kPa and 12 kPa is applied to the isolated bubble (see Figure 4). Note that a negative absolute pressure indicates the liquid is in tension, a process drives bubble expansion. The pressure initiates at 12kPa for approximately 5e-6 seconds, drops linearly to -1kPa over 5e-6 seconds, remains at -1kPa for 1e-5 seconds, then linearly increases to 12kPa in the following 2e-5 seconds. The entire 4e-5 s profile is repeated through a specified time interval.

Table 1–Material Properties of Initial Gas Bubble

Property	Value
Density of Liquid	1025 kg/m^3
Vapor Pressure of Liquid	2333 Pa
Initial Gas Pressure Within Bubble	1.31e8 Pa
Surface Tension	0.072 N/m
Dynamic Viscosity	0 Pa*s
Ratio of Heat Capacities	1.25
Initial Radius	1.e-5 m

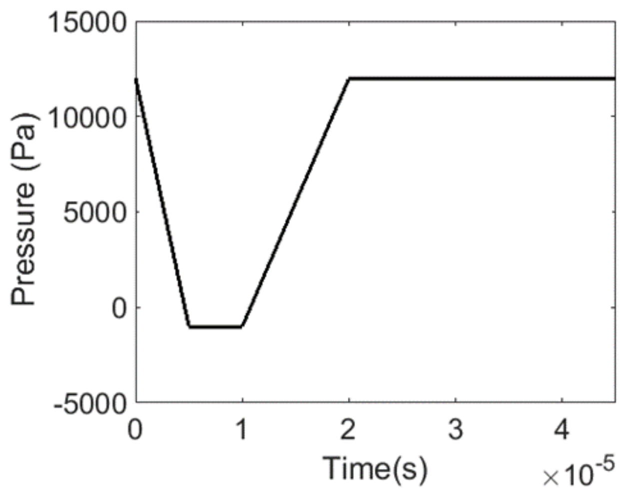


Figure 4 - Pressure profile applied to isolated bubble for RPE evaluations.

Comparisons of Different Methods

Figure 5 displays a comparison of the three most suitable methods applied to the same condition for a very large time step size. In addition, a "real solution" is provided for reference to indicate the time-asymptotic solution (using black X's). The three most suitable methods were Forward Euler, Runge Kutta 4 Triangle Backtrace, and Runge Kutta 4 Jump. Only the Euler and RK4 differential equation solving schemes were used in this comparison because they are the most widely used in the field of computational fluid dynamics. In addition, they are efficient and provide significant accuracy.

The number of time steps for the RK4-Triangle-BT and RK4-Jump methods are 100 steps per oscillation. The Euler Triangle-BT method was graphed at a npp (number of time-steps per oscillation) value of 256 time steps per oscillation, which is roughly the same computational time yielding a fair comparison. At these settings, all the methods closely approximate the solution. Up to the first collapse event, all three methods accurately calculate the solution as the only difference is Euler versus RK4 (approx. 2.5×10^{-5}s). Following the collapse event differences occur due to the procedure. The RK4-Jump method skips the singularity over a single time step that is shorter than the collapse event, causing a lag in the prediction of next oscillation. In the Triangle-BT methods, the handling of the singularity is not plotted and when plotted appears as a cropped off diameter through the jump. The Euler-Triangle-BT jumps too far, due to inaccuracies predicted in the collapse phase using the Euler method. The result leads the solution lead the “real solution” for the next oscillation. The RK4-Triangle-BT method displays similar features (to the Euler-Triangle-BT method), but the error is much less as it better predicted the slope in the cusp occurring at the collapse event. With the best accuracy, the RK4-Triangle-BT method is clearly the best solution to handle the singularity problem.

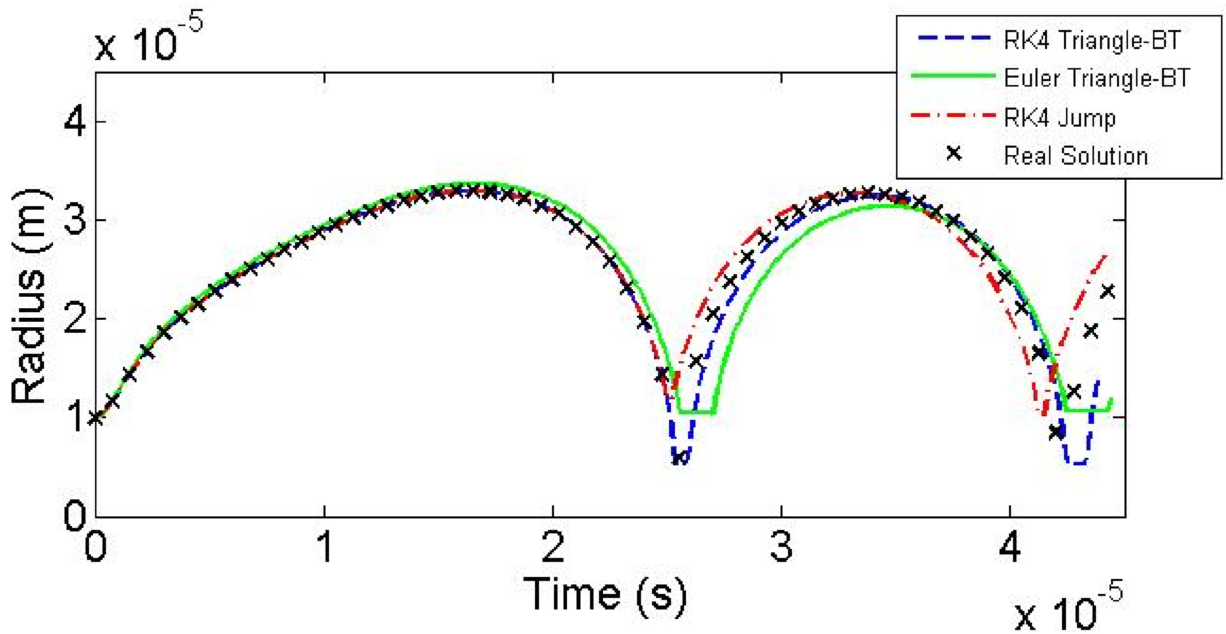


Figure 5 - Comparison of Three Best Methods to "Real Solution"

Performance

The previous results for the singularity handling are promising, and quantification of improvements is necessary and is the subject of this section. The three methods compared are RK4-Jump, RK4 Triangle-BT, and Euler Triangle-BT. Error is evaluated with respect to a solution with a small time step (1e-9 s), which is considered a time-asymptotic baseline representative of an analytical solution. Error is considered in terms of the mean error in the radius and time to reach the crest of the second oscillation. The mean error in the radius indicates the quality of the bubble radius. These results are summarized below.

The mean error in bubble radius versus increasing temporal resolution (npp) is plotted in Figure 6 (A). The term npp refers to the number of time steps in the first oscillation of the bubble. First, consider the behavior of the Jump results. For npp values of less than 300, this method has the least error and outperforms both the Euler-Triangle-BT and the RK4 triangle BT method. For larger npp, it performed better than the Euler method but just as well as the RK4 method. The Euler-Triangle-BT method suffers accuracy issues for small npp values and reduction of this error is clearly addressed by reverting to either the higher-order accurate, RK4-Triangle-BT method or the jump method. The RK4 triangle back-trace, although not as accurate as the Jump method, maintained low and relatively stable error percentages for various npp values. The solution without singularity handling fails for npp values less than 300. Thus, the proposed singularity handling approaches are effective to enable numerical stability in the regime where npp is below 300, while retaining reasonable errors for npp>50.

In general, maintaining a mean error well below ten percent is achievable for vary large time step sizes. Note the small dip and rise in the error around an npp value of 100. This error corresponds to the character highlighted in Figure 2a, indicating a moderate rise in the underlying assumption with decreased time step size. Nevertheless, the results are promising and the method enables a potentially faster, more robust numerical scheme.

A comparison of the computational time for the various methods is provided in Figure 5 (b). The fact that the RK4 curves in Figure 5 (a) and (b) remain on top of each other, indicates that singularity handling has little impact on the computational time. However, the disparity in error between the Euler is minimal when considering computational time. For this reason, Euler's method becomes more attractive. The general conclusions from computational performance are that singularity handling is computationally cheap, enables larger time steps which favor RK4 methods, but, Euler methods may remain attractive is larger errors are acceptable.

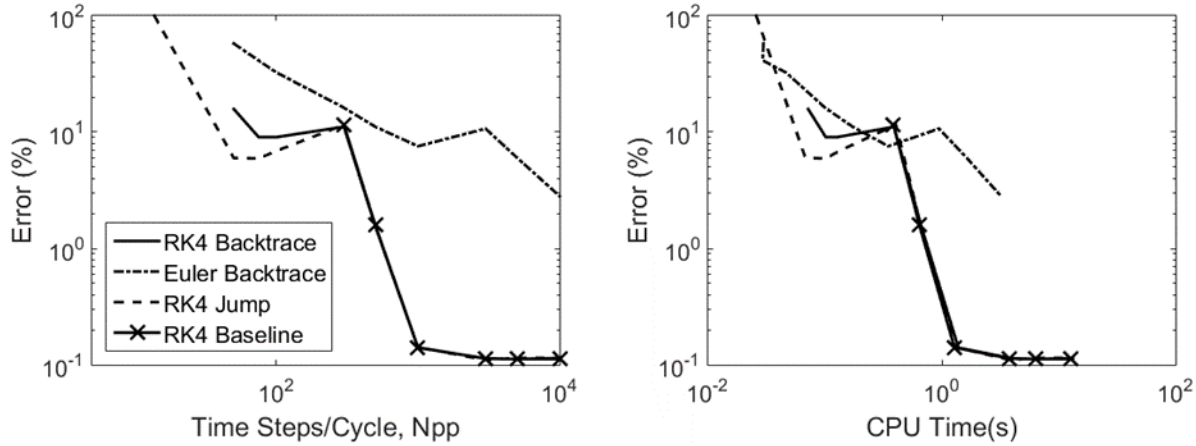


Figure 6: Comparison of Various Numerical Methods. (A) Mean error in radius versus the npp value for various methods. (B) Mean error in radius versus simulation wall time.

After conducting the aforementioned analyses examining error and CPU time for these three potential solutions, it is clear that the RK-4, jump method is the most effective constant step algorithm amongst those studied for solving the RPE. Such model proved quite accurate with significant alleviation of the CPU requirement. Consider the Jump method at an npp value of 50 and the No Singularity Handling method at an npp value of 500. The Jump method took 0.0681 s and it resulted in an error of approximately 6% from the real solution. The Baseline method took 1.28 s and resulted in an error of 0.141%. This means at the cost of only 6% deviation from the real solution, the singularity handling approach took 95% less time than the No Singularity Handling approach. Such an increase in solver speed is vital in more complex cavitation fields, where the accurate and quick simulation of hundreds, thousands, or even millions of bubbles is of great concern.

Despite this recommendation, the jump method may not suit all applications and the other methods indicated low error and low CPU times. Some instances where the jump method may not work properly are when the frequency of oscillations is very high. Since the jump method introduces a time lag every time it is used, a larger number of jumps would result in a larger error. In these cases the RK4 Triangle BT method may be the most effective.

Overview of Singularity Handling Approaches

The aforementioned methods were all implemented and evaluated in the context of RPE solution methods. A summary of the methods is briefly described in Table 2. There are some clear advantages and disadvantages to the various methods, which and may influence the decision process of choosing one of these methods. It was immediately found that methods 2 and 5 are not recommended.

In terms of Method 3, i.e., the jump method, results fared well at low frequency because jump did not significantly alter the radius and velocity. This meant that the error was dominated by a temporal lag introduced into the RPE solution. However, the maximum bubble radius and oscillation behavior was replicated quite well. It is clear that Method 3 is favorable depending on the information needed in the RPE solution.

Method 1, triangle approximation, displayed an improvement for variable time step models, however, did not work for constant time step models. In a variable time step model, the time can adapt to meet the symmetric condition at the collapse event. However, in adapting the time step, integration of the RPE diverges from the triangle model, leading to non-symmetric conditions, leading to a compounding error. For this reason, the triangular approximation (and similar variants) are best when the time step size can adapt. For this reason, Method 4 was developed and cured the issue in the context of a constant time step model. Lastly, Method 4 was found to be very effective at the cost of a slight increase in model complexity.

In addition, as Singularity Handling enables larger time step sizes, the ODE integration approach becomes more important. There are clear improvements with RK4 schemes over an Euler scheme due to increased numerical accuracy of the method. Thus, RK4 schemes are strongly recommended.

Table 2 - Summary of the pros and cons of the singularity approaches

Method	Pros	Cons	Suitable Application

1: Triangle	Simple	Requires variable time step or does not recover symmetry.	Variable time step
2: Parabolic	Complex	Not functional	None
3: Jump	Simple	Introduces phase lag, compounding error with high frequency	Constant time step and have interest in bulk bubble behavior
4: Triangle-BT	Accurate	More complex singularity handling	Constant time step and have interest in time-resolved oscillations
5:RPE-Approximaton	Consistent with theory	Requires non-local information, does not represent detailed collapse (displays different behavior with different temporal resolutions)	Not recommended

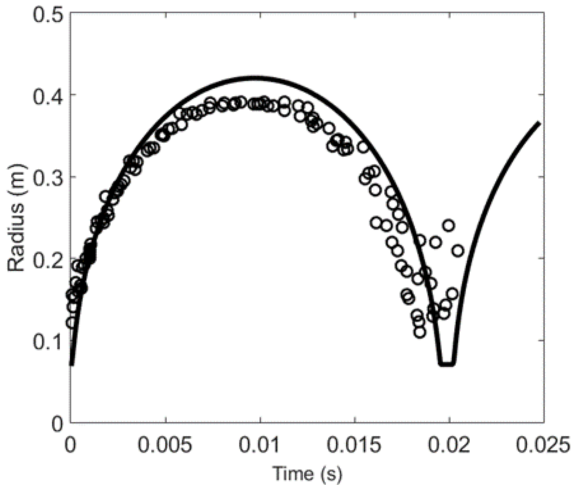
Validation with a Real-World UNDEX Problem

In this section, a validation step is performed against an UNDEX (underwater explosion) experiment. This is similar to the validation efforts of Miller et al. (2013). The initial bubble during the explosion is indicated via the properties in Table 3. These properties are specified identically to Miller et al. (2013), which is based on the explosive properties of the detonation.

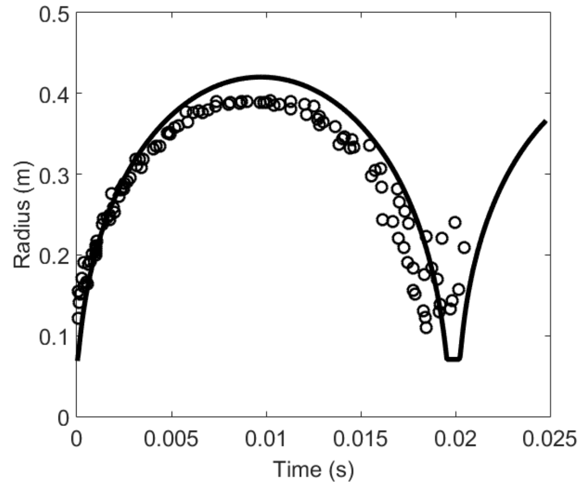
Comparisons to the experiment, using various prediction approaches, are provided in Figure 7. The overall results indicate that the singularity handling did not adversely affect the comparison with the experiment. In this particular case, uncertainty due to the initial velocity, indicated in the lower-right subfigure, is far greater than the impact of various Singularity Handling methods and ODE integration methods. The lower right figure accurately models the data when the initial velocity is known. This means the singularity handling methods provide an accurate and faster way of providing a solution.

Table 3 - Material properties for sample UNDEX model

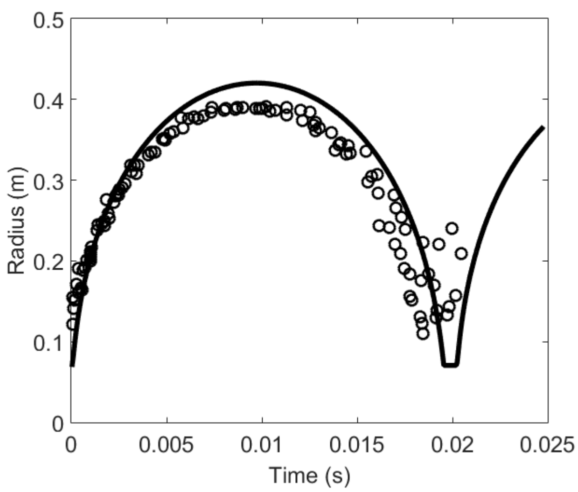
Property	Value
Density of Liquid	1025 kg/m^3
Vapor Pressure of Liquid	2333 Pa
Initial Gas Pressure Within Bubble	1.31e8 Pa
Surface Tension	0.072 N/m
Dynamic Viscosity	0 Pa*s
Ratio of Heat Capacities	1.25
Initial Radius	0.0667815 m



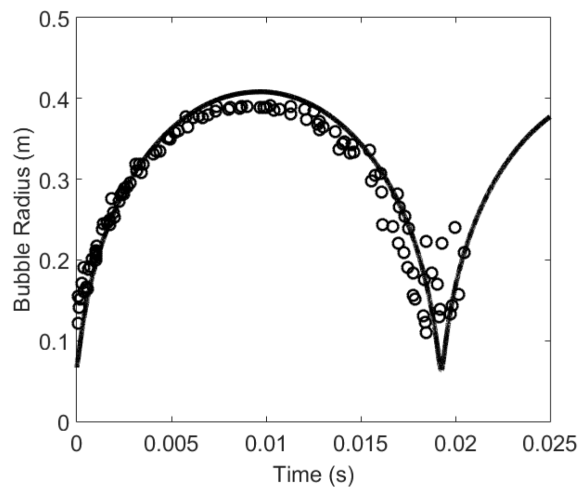
(a) RK4 Backtrace Approach



(b) Euler Approach



(c) Jump Approach



(d) Variable Time Step Approach

Figure 7 - Comparisons of various solution approaches to the UNDEX problem. (a) RK4 Backtrace (b) Euler (c) Jump (d) Variable TIme Step

Conservation of Momentum – Is the approximation valid?

An important consideration is whether or not the mirror point assumption is correct; that is, does the velocity after the cusp closely match the velocity before the cusp? If momentum is conserved, then for each radial speed before the cusp, there must exist a point on the other side of the cusp where the velocity is equal in magnitude. This validates the mirror point assumption as well as the jump method.

Through testing against UNDEX data, it is clear that conservation of momentum holds true for either side of a singularity. In my experiments, mass is constant, therefore conservation of momentum is equivalent to conservation of velocity before and after the singularity. When the Rayleigh Plesset Equation was solved using a variable time step solver (which implemented extremely small time steps), the velocities were equivalent in magnitude on either side of the cusp. shows after-cusp-velocity plotted against before-cusp-velocity for various radius values. The graph is redrawn several times for various values of pressure, surface tension, and viscosity. The blue line is the unity reference line; it represents where the velocity magnitudes were equal. The blue arrow represents the portion of the curve close the cusp and points in the direction of increasing radius. Likewise, the green arrow denotes the portion of the curve away from the cusp and points in the direction of increasing radius as well. At the end of the blue arrow curve and the beginning of the green arrow curve exists a point of maximum velocity. This point is where the radius is decreasing at the fastest rate. More importantly, however, this is where modified Rayleigh Plesset approximations start the "jumping" process. Notice that the curves are almost identical, with most of the differences occurring before the maximum velocity point. At radius values after this point, the velocity before and after converge to the same values. The graph below (Figure 8-Bottom Left) shows the cut portion where the radius lies outside the singularity.

Here, every single curve, regardless of pressure, surface tension, and viscosity, converges to the unity reference line, meaning momentum is conserved. Since momentum is being conserved without any modification of the RP equation, it is a valid constraint to impose on the problem when the modifications are added (ex. Jumping, triangle backtrace, etc.).

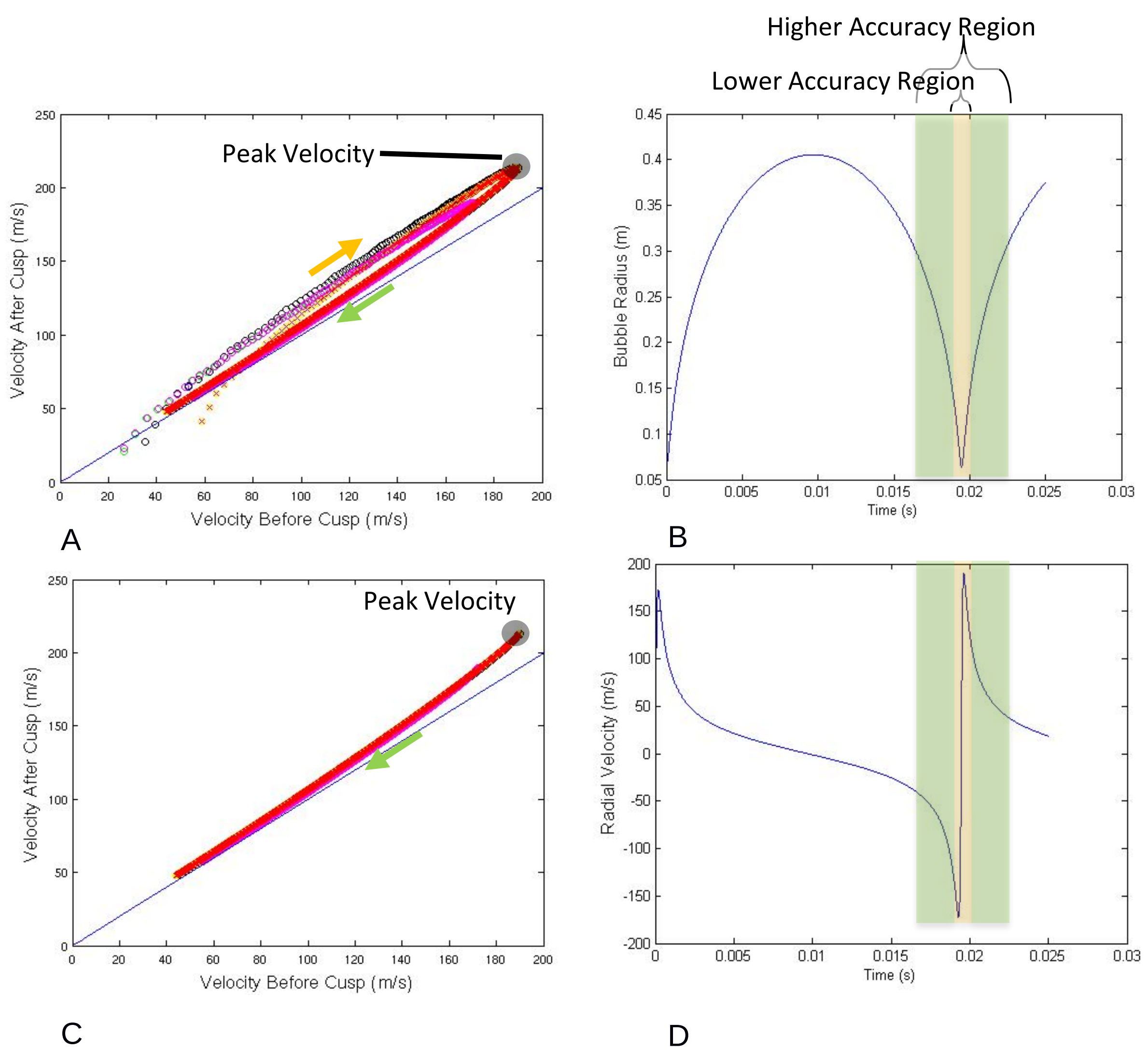


Figure 8 - Assessment of the validity of momentum conservation through a collapses. In part (A), is the velocity after the singularity versus velocity before the singularity at the same radii. Note that alignment with the line indicates excellent correlation. In part (C) a similar plot is shown, however, within the constraints of the singularity handling (that removes the inaccurate region). In part (B), radius versus time is plotted and in part (D), the radial velocity versus time is plotted. The vertical lines indicate the accurate and inaccurate regions.

Figure 8-Top Right and Figure 8-Bottom Right are the graphs of radius vs. time and radial velocity vs. time for the bubble. The solver is inaccurate for time values within the yellow region. This region corresponds to the yellow arrow in the top left graph. In this region, the velocity after the cusp is not equal to the velocity

before the cusp, so my modified Rayleigh Plesset Equation does not produce accurate results. The green region is the accurate region, and it corresponds to the green arrow on the left graphs. The velocity before vs. velocity after curve approaches the reference unity line, which means the velocity after the cusp is nearly equivalent to the velocity before the cusp. Here, the modified Rayleigh Plesset approximation is a valid solution.

Conclusions

The concept of handling collapse events in the solution to the RPE by implicitly considering a symmetric response through the event was presented and evaluated. Results indicate that the approach is possible and appears to recover the solution to the RPE. In addition, the ability to increase the time-step size was also displayed. As compared to traditional RPE solution approaches, singularity handling increased the stable time-step-size through a collapse event by 2-3 orders of magnitude. In addition, it was displayed that this modification retains accuracy of the RPE solution within a few percent. The overall result is a 2-3 order of magnitude reduction in computational time with respect to traditional constant time-stepping algorithm, however, with respect to variable time-step algorithms, the computational savings are more modest (25%).

Improved implicit collapse conditions could be developed. Despite the generally good agreement, improvement in the functional relations is still possible. In particular, the triangle model chosen within this work does not consider acceleration, i.e., d^2R/dt^2. Improved functions that can maintain symmetry in accelerations can likely improve the prediction method with no additional cost. It is expected that introducing such higher order functions require non-local data, which may not be desirable. In addition, the minimum radius equal to zero assumption, whereas the value is finite in reality. Brennen (2013) provides guidance on setting such values, which would require a small modification. Lastly, alternative forms of the RPE could also be considered.

Chapter 3
Modeling in CFD

This section describes the process of implementing the modified Rayleigh Plesset equation (with singularity handling) into a current generation computational fluid dynamics solver. In this case, I used StarCCM+. Created by CD-Adapco, StarCCM+ is an engineering simulation package designed to solve problems requiring even the most complex of geometries. Its fluid flow solvers in particular are extremely robust and are used in this paper. Before implementing the RPE into CFD, it was extended into an Eulerian framework.

Extending the RPE Into an Eulerian Framework

Traditionally, the Rayleigh Plesset Equation is implemented into CFD in a Lagrangian framework, modeling the dynamics of each bubble (that may represent groups of bubbles) as it moves through the CFD flow field. However, as stated previously, this method can become challenging for a large numbers of bubbles. In this paper I model cavitation in an Eulerian framework and aim to model much higher numbers of bubbles; in this case, it is necessary extend the Lagrangian RPE into an Eulerian framework using the material derivative, which can be utilized to transform reference frames. The material derivative of the bubble radius, for a transported bubble, is given by the following equation:

$$\frac{DR}{Dt} = \frac{\partial R}{\partial t} + u\frac{\partial R}{\partial x} + v\frac{\partial R}{\partial y} + w\frac{\partial R}{\partial z}$$

The left hand side represents the total rate of change of the bubble’s radius with respect to time, where the radius of the bubble is given as R = R(x,y,z,t). Notice how the bubble radius is a function of position as required in an Eulerian framework. The right hand side indicates the components of the rate of change of the bubble with respect to time, which includes both local spatial (or advected) terms. On this side of the equation, the bubble's velocity and rate of change with respect to position denote the use of a Lagrangian framework. In the present form, bubble radius is merely transported, thus, the dynamics are not yet included.

In the context of solving the RPE, I use a process that parallels a second-order ODE solution method. An ODE solution method starts by splitting the second order differential equation into a system of two first order linear differential equations as shown below.

$$y'' = f(t, y, y')$$
$$y' = x_2$$
$$y = x_1$$

$$x_2' = f(t, x_1, x_2)$$
$$x_1' = x_2$$

For a given initial conditions and time-step size, the numerical solver calculates the second derivative and first derivative for the next time step using a specified equation (f). Using the second derivative and the time step value, the solver can calculate the new first derivative using whichever method preferred (Forward Euler, 4-5 Runge Kutta, etc.). Shown below is an example of a forward Euler scheme:

$$y'_{n+1} = y'_n + \Delta t(y''_n)$$

Then the solver calculates the new function value using the old first derivative and time step.

$$y_{n+1} = y_n + \Delta t(y'_n)$$

Afterwards, the solver increments the time by one time step, which can be variable or constant, and continues the process until a specified time has been reached.

In the context of a second-order ODE solution method, applied to the RPE, the following equation represents the second derivative of radius with respect to time as a function of radius and radial velocity.

$$\frac{\partial^2 R}{\partial t^2} = \frac{1}{R}\left(\frac{P_B - P_\infty}{\rho} - \frac{3}{2}\left(\frac{\partial R}{\partial t}\right)^2\right)$$

Note that as this is an ODE method, there are no convective derivative terms. This is introduced in the next section. Once the formula for $\ddot{R}$ is derived, the formula for $\dot{R}$ can be derived through simple integration as follows:

$$\int \frac{\partial^2 R}{\partial t^2} dt = \frac{\partial R}{\partial t} \approx \frac{\partial^2 R}{\partial t^2} \Delta t$$

As stated above, the integration is performed numerically by approximating $\ddot{R}$ to be constant over a small time interval. This is the implementation of the forward euler scheme. In the same way, R can be written as the integral of $\dot{R}$:

$$\int \frac{\partial R}{\partial t} dt = R \approx \frac{\partial R}{\partial t} \Delta t$$

While the above equations form a numerically solvable set of differential equations, they do not take into consideration the convective terms in the flow field. Generally speaking, the ODE approach does not take into account the change in bubble radius due to the bubble's motion. This must be done through a PDE approach. Only then can the Rayleigh Plesset Equation be accurately implemented in an Eulerian-based CFD model.

In the context of an Eulerian CFD model, I define the equations for the second derivative and first derivative of the bubble radius. However, in this case I include the convection terms. This is done below by making use of the material derivative.

I define the RPE in relation to the material derivative with source terms to convert the Lagrangian framework to an Eulerian framework as follows:

$$\frac{D^2R}{Dt^2} = \frac{\partial \dot{R}}{\partial t} + u\frac{\partial \dot{R}}{\partial x} = \frac{1}{R}\left(\frac{P_B - P_\infty}{\rho} - \frac{3}{2}\left(\frac{DR}{Dt}\right)^2\right) = \frac{D\dot{R}}{Dt}$$

The second derivative of radius with respect to time is rewritten as the material derivative of $\dot{R}$ with respect to time. This means that the radial acceleration is not only dependent on radial speed and radius, but also the rate of change of radius with respect to position (aka the convective term).

Now I define the material derivative of the bubble radius in a similar fashion:

$$\frac{DR}{Dt} = \dot{R} + u\frac{\partial R}{\partial x}$$

The RPE is well defined, and the value of the second material derivative of R can be directly calculated once the basic bubble parameters are known. Note that I assumed the surface tension term and viscosity terms were approximately zero. This greatly simplifies the equation and allows for easier implementation in CFD solvers. Such an assumption is valid because the surface tension and viscosity terms do not alter the second material derivative of R very much and are overshadowed by the pressure and velocity-squared terms.

Formulation of Boundary and Clip Values

To increase the convergence rate for the StarCCM+ model as well as to prevent the solver from crashing, boundary and clip values for R, $\dot{R}$, and $\ddot{R}$ had to be implemented. Each of these three parameters had a minimum clip value and a maximum clip value. If the value goes below the minimum clip, the value is set to the minimum clip. If the value goes above the maximum clip, the value is set to the maximum clip.

Clip for R

The clip values for R was given as a direct relationship with the initial radius. The initial radius was calculated with the following equation. The minimum clip value of R is equivalent to the minimum radius of the simulation. Although there is no 100% accurate analytic equation for the minimum radius, Brennan (2013) provides an approximation given by the following equation.

$$R_{min} = R_0 \left(\frac{1}{(k-1)} \frac{p_{G_0}}{(p_\infty^* - p_v - p_{G_0} + \frac{3S}{R_0})} \right)^{\frac{1}{3(k-1)}}$$

Here, R_0 represents the initial radius of the bubble. There is no maximum clip value for R, as this value is constrained numerically by the Rayleigh Plesset Equation. In addition, the radius value has no realistic upper boundary.

Clip for $\dot{R}$

The clip values for $\dot{R}$ can be determined using Brennan's equation below. Note that the pressure of the bubble is a combination of the vapor pressure of water as well as the bubble's initial gas pressure:

$$\dot{R}_{max} = \sqrt{\frac{2}{3}\frac{P_B - P_\infty}{\rho}}$$

This clip value is used for the maximum radial velocity. The minimum radial velocity is simply -1 * $\dot{R}_{max}$.

Clip for $\ddot{R}$

The clip for $\ddot{R}$ is given by the minimum and maximum possible values from the Rayleigh Plesset Equation. This means that the max and min clips are related to R and $\dot{R}$. For the maximum clip value, the value of R is the minimum radius value calculated above, and the value of $\dot{R}$ is dependent on the viscosity, density, and minimum radius. To find a relationship between the value at $\dot{R}$ that gives the maximum value of $\ddot{R}$ and the other three parameters listed above, I used partial derivatives. Taking the partial derivative of $\ddot{R}$ with respect to $\dot{R}$ and setting the result equal to zero gives the value of $\dot{R}$ that maximizes $\ddot{R}$. This value of $\dot{R}$ is denoted with a subscript "U" below.

$$\frac{\partial \ddot{R}}{\partial \dot{R}} = \frac{-4\mu}{\rho R_{min}^2} - \frac{3\dot{R}}{R_{min}} = 0$$

Solving for $\dot{R}_U$ gives:

$$\dot{R}_U = \frac{-4\mu}{3\rho R_{min}}$$

Notice that in an inviscid regime, the value of $\dot{R}_U$ becomes zero, which means the upper bound of $\ddot{R}$ is dependent only on the value of the minimum radius, and that the radial acceleration is greatest when the bubble is not growing. Changing the viscosity is discussed in the next section.

The following equation gives the upper bound of $\ddot{R}$:

$$\ddot{R}_U = \frac{P_v - P(t) + P_{g_0}(\frac{R_0}{R_{min}})^{3\gamma} - \frac{2S}{R_{min}} - \frac{4\mu\dot{R}_U}{R_{min}}}{\rho R_{min}} - \frac{3\dot{R}_U^2}{2R_{min}}$$

The lower clip value for $\ddot{R}$ is found by using the minimum Radius and maximizing $\dot{R}$.This maximum value of $\dot{R}$ comes from the $\dot{R}$ clip equation above. This causes the negative terms in the Rayleigh Plesset equation to increase to their highest possible values, thus minimizing $\ddot{R}$. The bottle-slam validation model described in the next chapter uses this method for the $\ddot{R}$ bounds.

Alternative Method for R and $\ddot{\boldsymbol{R}}$ Bounds

For a more accurate clip calculation, especially when implemented in CFD, the properties of the mesh can be used to determine the maximum radius (as well as bounds on $\ddot{R}$). For a given mesh, the following inequality must always hold:

$$V_{Cell} > N_B V_B$$

Here, V_{cell} represents the volume of each cell in a volume mesh. N_b and V_b represent the number of bubbles per cell and the volume of each bubble, respectively. In other words, this inequality states that the total volume of all the bubbles in a given cell must not exceed the volume of that cell. Since the bubbles are assumed to be spherical, the volume is a function of only the radius. The following equation then relates the volume of the cell to the maximum radius of the bubble:

$$V_{Cell} = N_B \frac{4\pi R_{max}^3}{3}$$

Solving for R_{max} here gives an upper bound for the bubble radius. Next, the bounds for $\ddot{R}$ are formulated. Since the radius of the bubble is a function of time, it can be written as a second order Taylor-series expansion about $t=t_i$ as follows, where t_i is any given time and t_f is the value of time at the next time step:

$$R(t_f) = R(t_i) + \dot{R}(t_i)(t_f - t_i) + \ddot{R}(t_i)\frac{(t_f - t_i)^2}{2}$$

This equation holds for all times, so to find the maximum radial acceleration, I use the R_{max} value from above as $R(t_f)$. To find the minimum radial acceleration, I use the same equation but I set the right hand side equal to zero. Now the goal is to find the effective difference in time between the next time step and the previous time step. This is not exactly equal to the time step because the direction of bubble motion may alter the actual time of the bubble.

To find this value, denoted as t_{eff}, I create a passive scalar for time which is governed by the flow. This passive scalar shows, for a bubble at a certain position in the flow field, how long it took for the bubble to get there. By taking the gradient of this passive scalar and calculating the magnitude of this gradient's projection onto the direction of flow, I determined the effective time step. The following equation models this solution:

$$\Delta t_{eff} = \frac{\vec{V}}{|\vec{V}|} \cdot \nabla t^*$$

After determining the effective time step, the radial acceleration upper bound was calculated as follows:

$$\ddot{R}_{max} = 2\frac{R_{max} - R(t_i) - \dot{R}(t_i)(\Delta t_{eff})}{\Delta t_{eff}^2}$$

The lower bound had a similar form, except R_{max} was replaced by zero:

$$\ddot{R}_{min} = 2\frac{0 - R(t_i) - \dot{R}(t_i)(\Delta t_{eff})}{\Delta t_{eff}^2}$$

This method for formulating clips for $\ddot{R}$ is tougher to implement in CFD code, but it provides a more accurate way of bounding the values of radial acceleration and radius. It is used in the flow-over-cylinder validation case in the next chapter.

Visualizing $\ddot{R}$ at various R and Ṙ Values

The following contour plots display $\ddot{R}$ at regularly spaced values of radius and radial velocity. The goal of this section is to examine both the effects of the boundary clip value for Ṙ and the viscosity value on $\ddot{R}$. Up until this point, all solutions were created using a viscosity of zero. Here I examine a nonzero viscosity. Below are the contour plots of $\ddot{R}$ for various values of R and Ṙ. The graphs on the left were created with a viscosity value of zero, while those on the right have a nonzero viscosity. The viscosity value used is that of water, which is approximately 1.28 m^2/s. The two graphs on the top have no bounds on Ṙ, while those on the bottom have clips given by Brennan's equation in the previous section. The redder regions denote large, positive $\ddot{R}$ values (growth), while the blue regions denote very negative $\ddot{R}$ values.

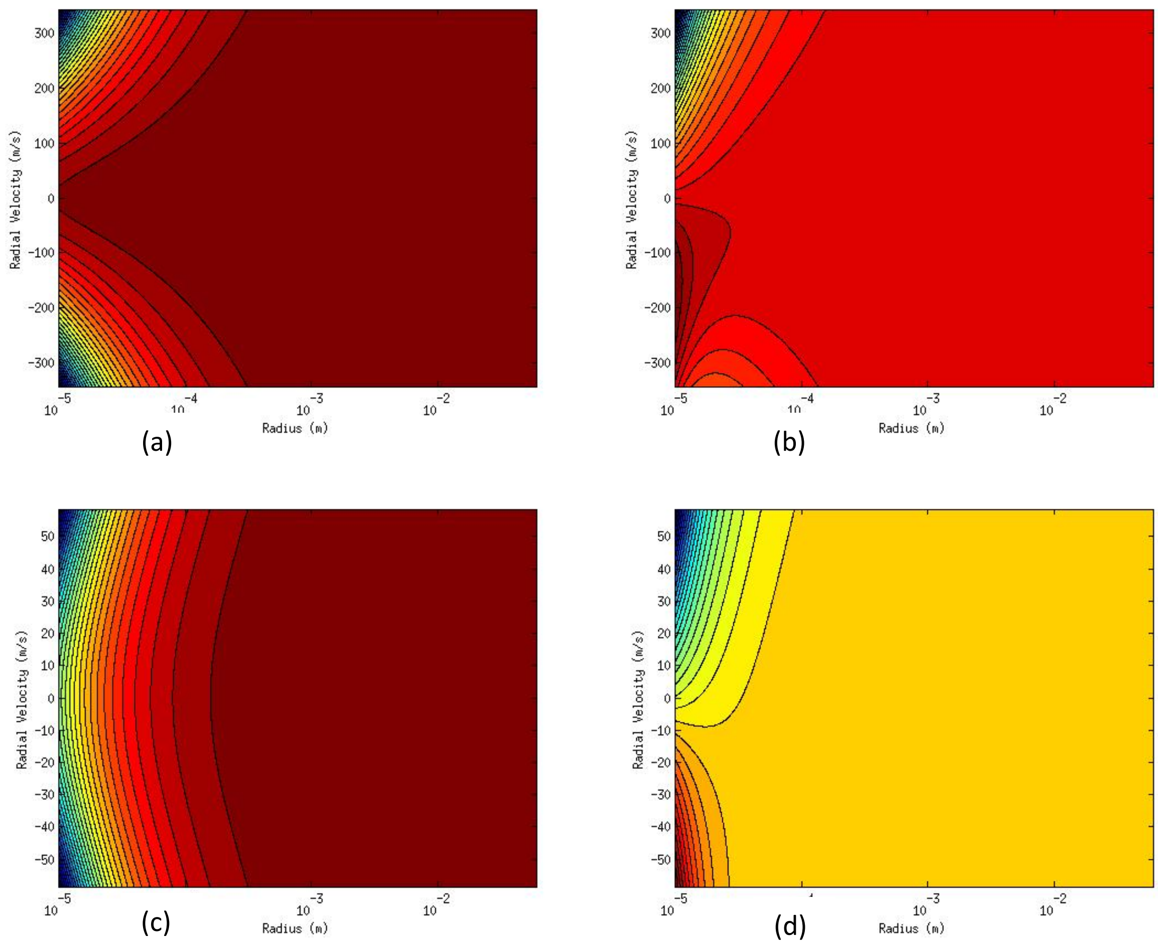


Figure 9 - $\ddot{R}$ contour plot for various R and $\dot{R}$ values: (a) Zero viscosity, unclipped $\dot{R}$ (b) Nonzero viscosity, unclipped $\dot{R}$ (c) Zero Viscosity, clipped $\dot{R}$ (d) Nonzero viscosity, clipped $\dot{R}$

Altering the viscosity from zero to nonzero changed the value of $\dot{R}$ for which $\ddot{R}$ attained its maximum. For zero viscosity, the maximum $\ddot{R}$ occurs at the minimum radius and an $\dot{R}$ value of 0. For a nonzero viscosity, $\ddot{R}$ attains its maximum value when $\dot{R}$ is slightly negative; it is around -170.6 m/s for the unclipped case and -58.5 m/s for the clipped case. $\ddot{R}$ attains its minimum value when the magnitude of $\dot{R}$ is its highest possible value. These results agree with the first method of $\ddot{R}$-clip formulation given in the previous section.

The clip values for radial velocity had a significant effect on where $\ddot{R}$ reached its maximum and minimum. Without the clips, the value for $\dot{R}$ for which $\ddot{R}$ attained the maximum were 0 and -170.6 m/s respectively. With clips, this $\dot{R}$ value stayed at 0 for the inviscid case but changed to -58.5 m/s for the viscous case. In the same way, the value of $\dot{R}$ for which $\ddot{R}$ attained its minimum was reduced from the speed of sound in air (343 m/s) to the upper clip value for $\dot{R}$ (58.5 m/s).

The results from this contour plots show that the clip value for $\dot{R}$ has a very large effect when applied to a viscous case. For an inviscid case, the formulations above can be safely applied to the CFD model to run the simulation at a larger time step and increase the convergence rate. However, applying the clip value formulations to a viscous case could significantly affect the accuracy of the resulting solution, so the effects of viscosity must be taken into serious consideration when constructing the model.

Chapter 4
Validation of CFD Solution with Matlab

Simple Square Model

First, I implement the Rayleigh Plesset Equation into StarCCM+ in a simple model. The body is a square prism with dimensions 0.2 x 0.2 x 0.04 m. The part is filled with water and given the initial conditions listed below. This test case is similar to the UNDEX test case from the previous chapter. The far-field pressure is constant throughout the course of the cavitation event.

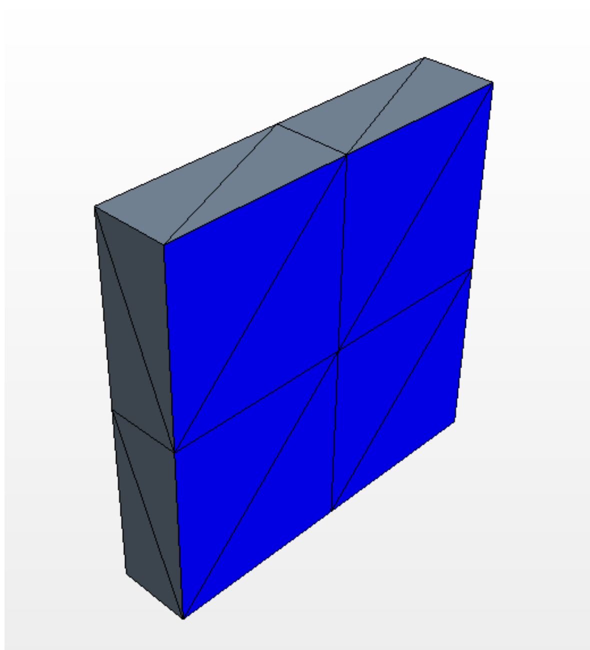


Figure 10 - Mesh for the simple square model.

Table 4 - Fluid Properties for Simple Square Validation Case

Property	**Value**
Pressure (Pa)	1685662
Surface Tension (N/m)	0.072

Dynamic Viscosity (Pa * s)	0
Gamma	1.25
Initial Gas Pressure Within Bubble (Pa)	1.31e8 Pa
Vapor Pressure of Water (Pa)	2300
Density of Water (kg/m^3)	1000
Initial Bubble Radius (m)	0.0667815

I implemented the pressure distribution into StarCCM+ as a new field function called "PressureProfile." Although the pressure distribution used for this test case is a constant value, creating a field function will allow the use of non-constant pressure distributions for other test cases.

In addition, I used the passive scalar feature of Star to create two new variables: R and Ṙ. R represents the radius of the bubble at any given time and Ṙ represents the bubble's growth rate. I also created a field function called $\ddot{R}$ that represents the second derivative of radius with respect to time. This value is calculated at each time step using the Rayleigh Plesset Equation shown below. Since StarCCM+ does not allow commenting of code, parts of the equation were selectively activated for testing purposes.

$$\frac{d^2R}{dt^2} = \frac{P_v - P(t) + Pg_0 \left(\frac{R_0}{R}\right)^{3\gamma} - \frac{2S}{R} - \frac{4\mu\frac{dR}{dT}}{R}}{\rho R} - \frac{3\frac{dR^2}{dt^2}}{2R} \qquad 3$$

Using StarCCM+'s built-in plot and monitor system, I ran the simulation for a physical time of 0.025 seconds and extracted the radius, radial velocity, $\ddot{R}$, and pressure as a function of time. The pressure vs. time distribution (which is constant) was fed into the MATLAB solver and the resulting radius, radial velocity, and $\ddot{R}$ graphs were compared against the ones extracted from StarCCM+. Here are the superimposed graphs to show the StarCCM graph and MATLAB graphs together (MATLAB is blue, Star is green). Below the R vs. time graph is a plot of error versus time.

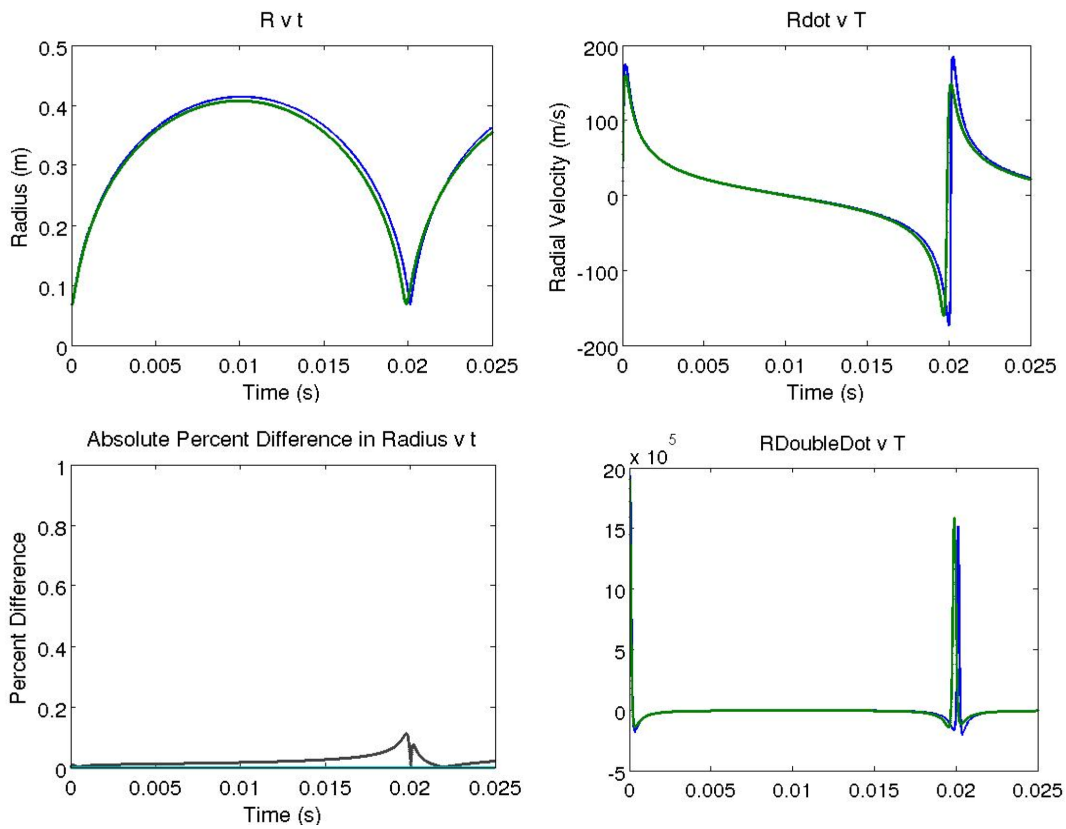


Figure 11 - MATLAB vs. StarCCM+ in a Simple Square Model

Notice how the error is negligible at points far from the singularity. Even at the singularity, the graph of absolute percent difference in radius vs. time shows less than a 20% difference between the StarCCM+ and Matlab solvers. The singularity shows a very large percent difference in $\dot{R}$ and $\ddot{R}$ at the singularity points. However, this error is not an issue because the solution recovers after the singularity. The post-cusp error returns to its pre-cusp state, meaning that the Rayleigh Plesset Equation was successfully implemented into a simple square model in StarCCM+.

Cavitation in a Shaken Bottle

The next goal was to implement the Rayleigh Plesset Equation into a slightly more complicated model. Here, I do not use a pressure profile. Rather, I use the laboratory pressure during a bottle slam event to generate cavitation. A bottle initially filled with water is hit on the top with a large impulsive force, causing cavitation to occur at the bottom. This test will be simulated entirely in StarCCM+ by giving the bottle a sharp vertical velocity for a specified amount of time, causing pressure to drop and bubbles to grow. Then the bottle will abruptly stop, causing the pressure to increase and the bubbles to collapse.

Just as in the previous example, I use StarCCM+'s built in passive scalar feature to model the cavitation bubbles' radius, radial velocity, and radial acceleration. In this case however, the body is not a simple square, and is instead a bottle shaped object shown below. The CAD model was created in StarCCM+ using the built in sketch and revolve CAD features. First, an outline of the outside of the bottle was sketched on the X-Y plane with a Z-coordinate of zero. Then a construction line was created through the center axis of the bottle. I implemented the revolve feature to revolve the sketch around this axis, creating a bottle shape.

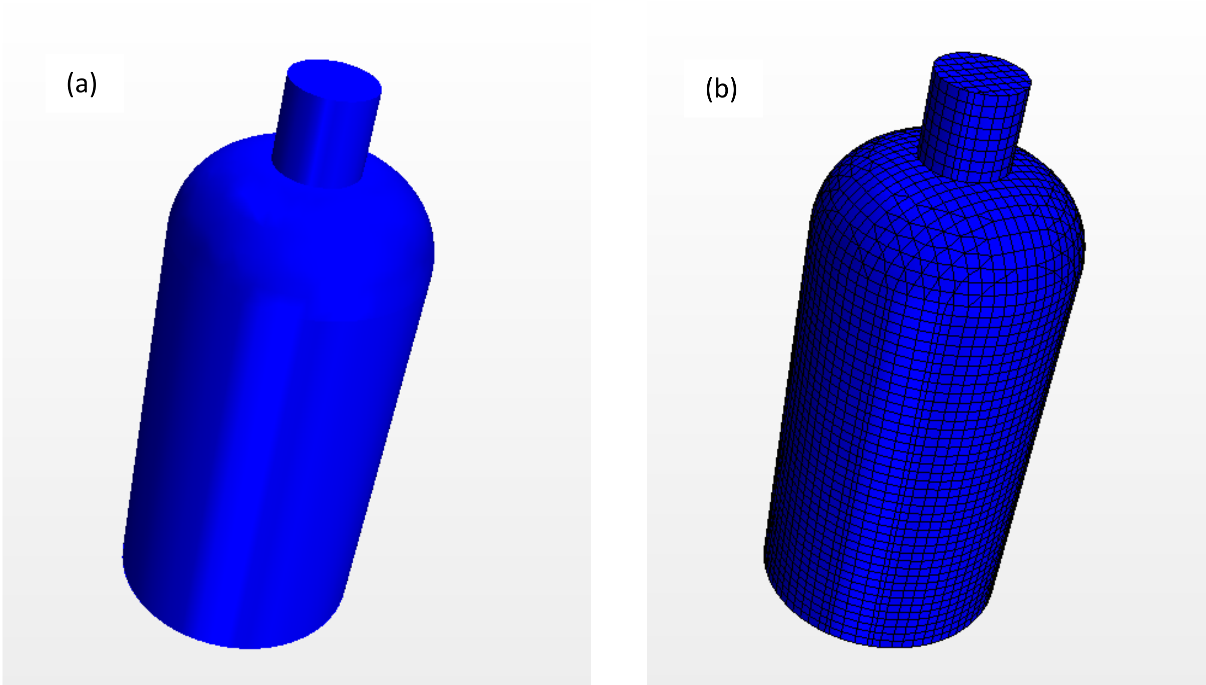


Figure 12 - (a) This is the 3d model used to represent a bottle. (b) This is the volume mesh used for this model. Note that the mesh is relatively uniform throughout.

To fill the bottle, I altered the physics continuum of the bottle. First I included the Eulerian multiphase model and added two phases: air and water. Each of these phases is treated as compressible. Under the initial conditions tab, the volume fraction was set to composite. The top of the bottle until the center (at the origin) was filled with air. Then the next 0.048 meters were filled with water. Finally, the last 0.002 meters were filled with more air. This air at the bottom of the bottle is required to allow cavitation to begin. The corresponding scalar scene are shown in Figure 13. Note that Figure 13 shows the model with a very fine mesh density for clarity. The volume mesh on which the simulation ran was slightly coarser than the one shown below:

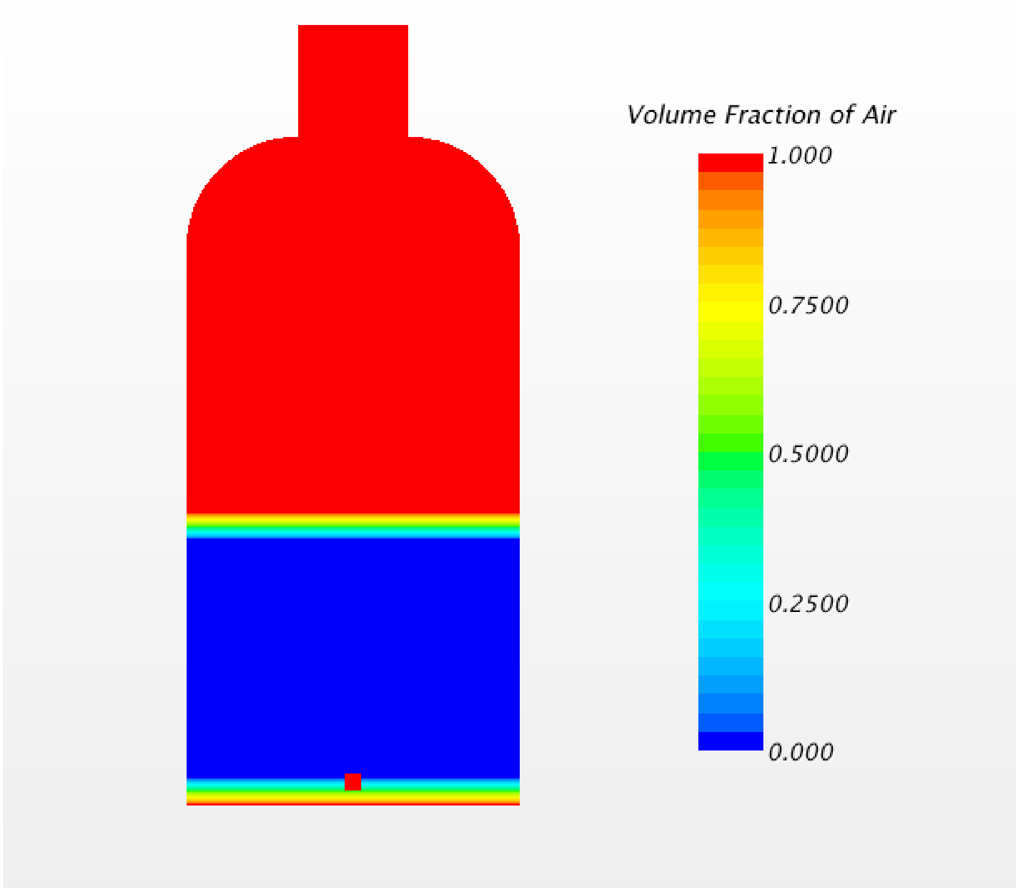


Figure 13 – Initial distribution of air and water in StarCCM+ bottle simulation.

In this case, the bottle has an initial velocity of 10 m/s downwards and abruptly stops. To model this movement, the fluid velocity was initialized to -10 m/s. The dynamics of the event were then resolved.

The simulation ran at a time step size of 1e-7 s to prevent any crashes due to singularities. StarCCM+ does not allow the simulation to go back a few time steps if a crash occurs, so I kept the time step small to prevent such errors preemptively. Just as in the simple square simulation, the absolute pressure, radius, radial velocity, and radial acceleration are extracted from the model using a derived part. Here my derived part was a point probe that traveled with the bottle approximately 0.005 meters from the bottom.

Unfortunately, the numerical requirements of the simulation resulted in a time step size of 1e-10s, and it was reduced even further at the singularities. This meant the entire simulation would take days, if not weeks to complete. To fix this problem, I restarted the simulation and implemented the clip system defined in the previous chapter. If the bubble radius at any point in the bottle fell below 1e-7m, it would be clipped back to 1e-7m. This prevented the radius from becoming too small, which would cause overflow errors in $\ddot{R}$. This also prevented the radius from going below zero, which would cause floating point exception errors. The clip for Ṙ according to Brennan's equation was also introduced.

At a minimum radius of 1e-7m, and with Ṙ clipped, the value of $\ddot{R}$ still had the potential to grow to unreasonably high values. To prevent this problem, I included a clip on $\ddot{R}$ as well. The clip values for $\ddot{R}$ come from the first set of formulations presented in the previous chapter. Here, the upper and lower bounds for the radial acceleration come from using the partial derivatives of the Rayleigh Plesset Equation to find the global maximum and minimum. It was found that the lower boundary for $\ddot{R}$ was about -2e8 m/s^2 and the upper limit was approximately 2e8 m/s^2. After running the new simulation and exporting the data, the pressure vs. time values were inputted into the MATLAB Rayleigh Plesset Solver and compared against the extracted R, Ṙ, and $\ddot{R}$ values. Below are the graphs comparing the StarCCM+ solution to the MATLAB solver. In the bottom right is the pressure vs. time graph.

The MATLAB solution matched StarCCM+ almost perfectly until the end of the second oscillation. At this point, the solver became unstable and did not match the CFD solution. This could be due to MATLAB's inability to take into account spatial terms as well as time terms. In the bottle collapse simulation, the water and air are both moving within the bottle, and since R and Ṙ are convective terms, they will also travel in the model. This differs from the simple square simulation because there is more than just a time term. The

MATLAB code only takes in pressure vs. time data, and this could have caused some dissimilarities.

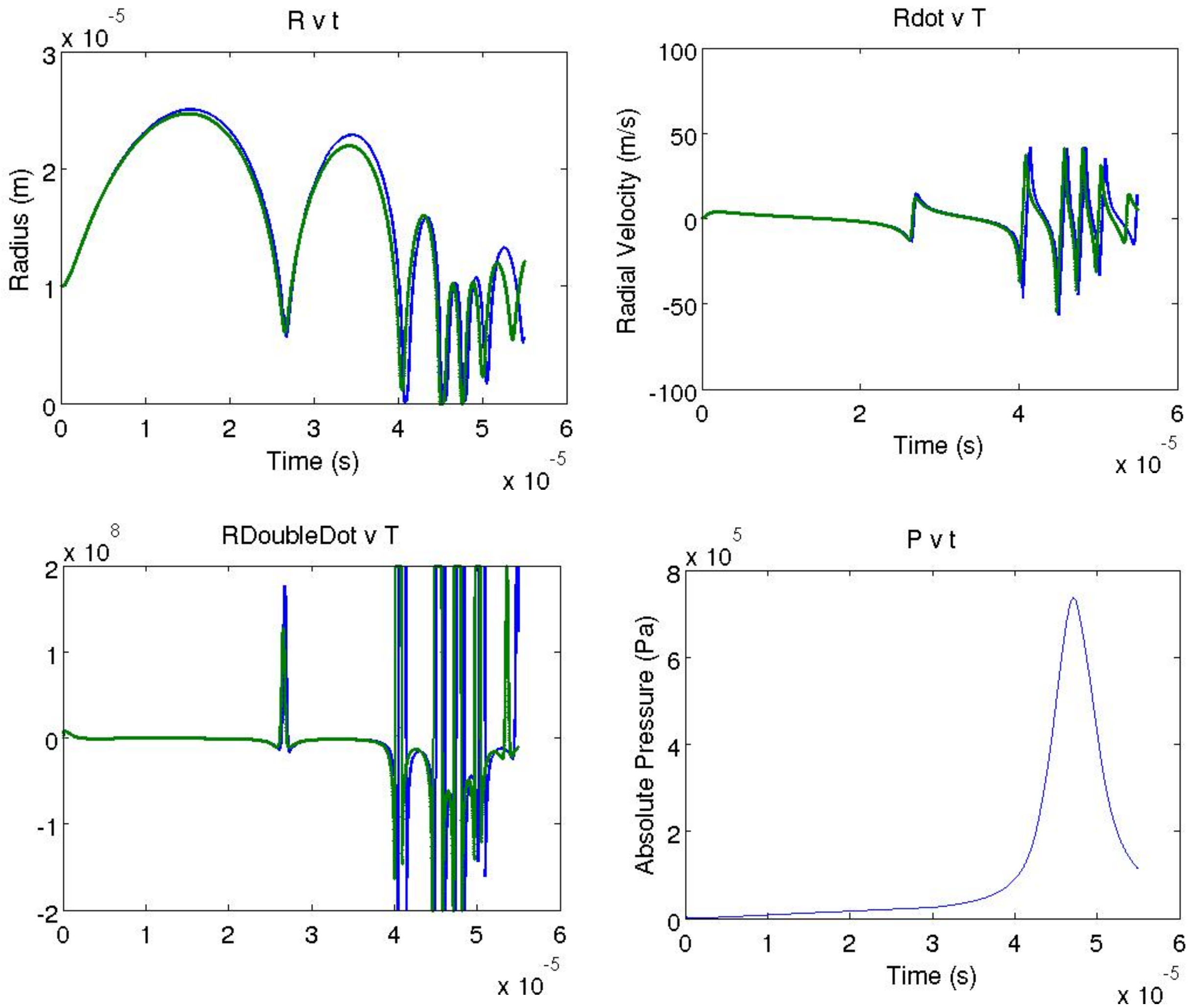
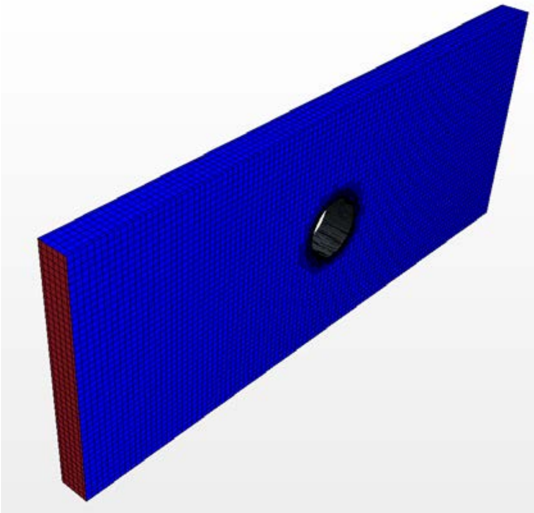


Figure 14 - MATLAB vs. StarCCM+ in the bottle collapse simulation

Flow Over Cylinder

The third simulation involves a flow over a cylinder. Here, a cylindrical hole is cut orthogonally into the center of a 1m x 0.5m x 0.1m channel. Water travels from the left hand side to the right and flows over the hole. (a) Since there is a large amount of space (b) far from the cylinder where the fluid flow is not significantly affected, the mesh was constructed to be finer near the hole and coarser farther away. This mesh is shown on the left in the figure below. The converged solution for bubble radius is shown on the right.



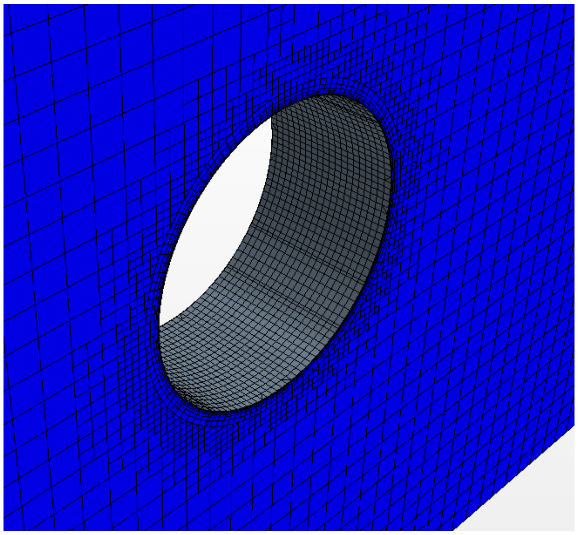


Figure 15 – (a) This is the mesh for the cylinder model. Notice that the mesh is uniform far away from the cylinder, but becomes very fine close to the hole. (b) This figure shows a close up view of the fine mesh around the hole.

The converged solution for the bubble radius, given the mesh above, is shown below:

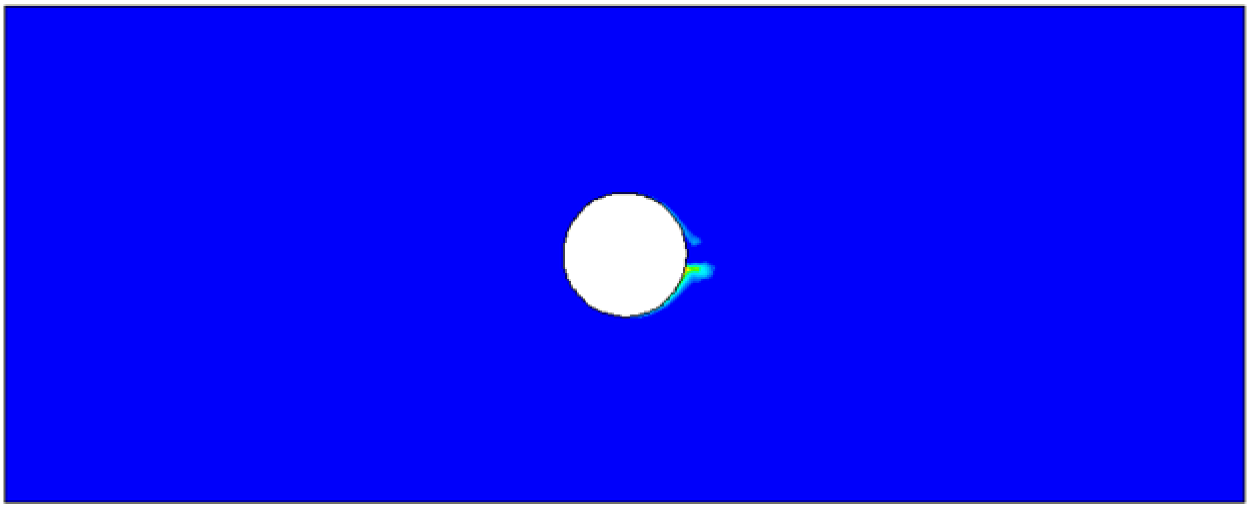


Figure 16 - This is the converged solution for bubble radius in the flow-over-cylinder model.

Due to the nature of the flow, cavitation developed on the far side of the cylinder and created a stream of bubbles. Since the channel through which the fluid flows is not vertically symmetrical, the cavitation region is not directly to the right of the cylinder. Instead, cavitation occurs slightly to the right and below the cylinder surface, due to a higher flow velocity in that area.

This model was more complicated than the previous two because it included convective terms in addition to a time term in the differential equation. I used the same passive scalar model as in the previous two simulations. Here, however, data was extracted from a streamline that started at the channel inlet and made its way to the channel outlet to give a Lagrangian result. This streamline is the path of a Lagrangian bubble flowing through the channel. When extracting data from a streamline, StarCCM+ gives pressure, radius, radial velocity, and radial acceleration values as a function of position. However, to verify my data with MATLAB, I needed these values as a function of time. To calculate the time data, I found the distance between consecutive points along the streamline and divided it by the average velocity of the flow at those two points. This gave a set of time data which could be plotted against radius, radial velocity, radial acceleration, and pressure. The streamline that data was extracted from is shown below. It was important that the streamline travel around the cylinder to accurately model cavitation on the cylinder surface.

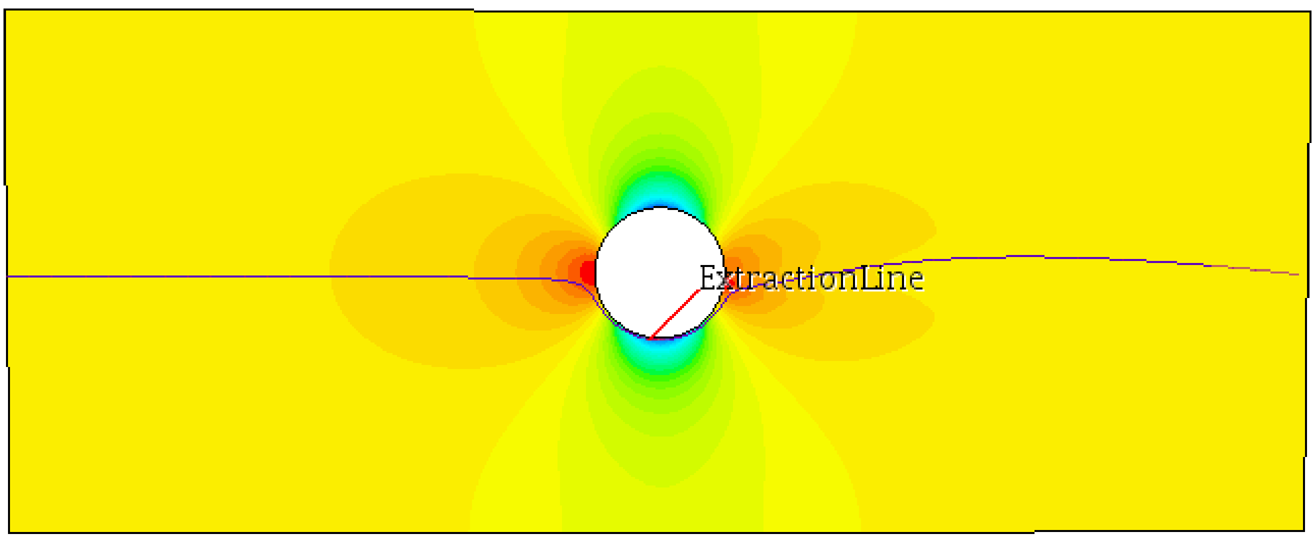


Figure 17 - Extraction streamline and final solution for absolute pressure in flow over cylinder simulation.

The solution above shows the absolute pressure at different points in the flow field. The red areas denote a large absolute pressure, while the blue areas denote very low pressure regions where cavitation is most likely to begin. Note that since the flow is constantly moving, the bubbles will start to grow near the top and bottom of the cylinder, but they will reach their maximum size behind it. Once the bubbles reach the back side of the cylinder, the high pressure region causes them to collapse abruptly.

I switched the model from an implicit unsteady flow to a steady flow to increase the rate of convergence. To prevent the radius of the bubble from falling below zero, I set the minimum clip value on the passive scalar R to 1e-7m. In addition, a maximum clip value for R was set. Clips were added to both $\ddot{R}$ and $\dot{R}$ as well. The clips for R, $\dot{R}$, and $\ddot{R}$ were formulated in the previous section. For the $\ddot{R}$ clip, the second formulation is implemented since it takes into account the varying mesh density around the cylinder.

Below are the graphs of R vs. time extracted from the streamline (green) as well as solved with MATLAB (blue).

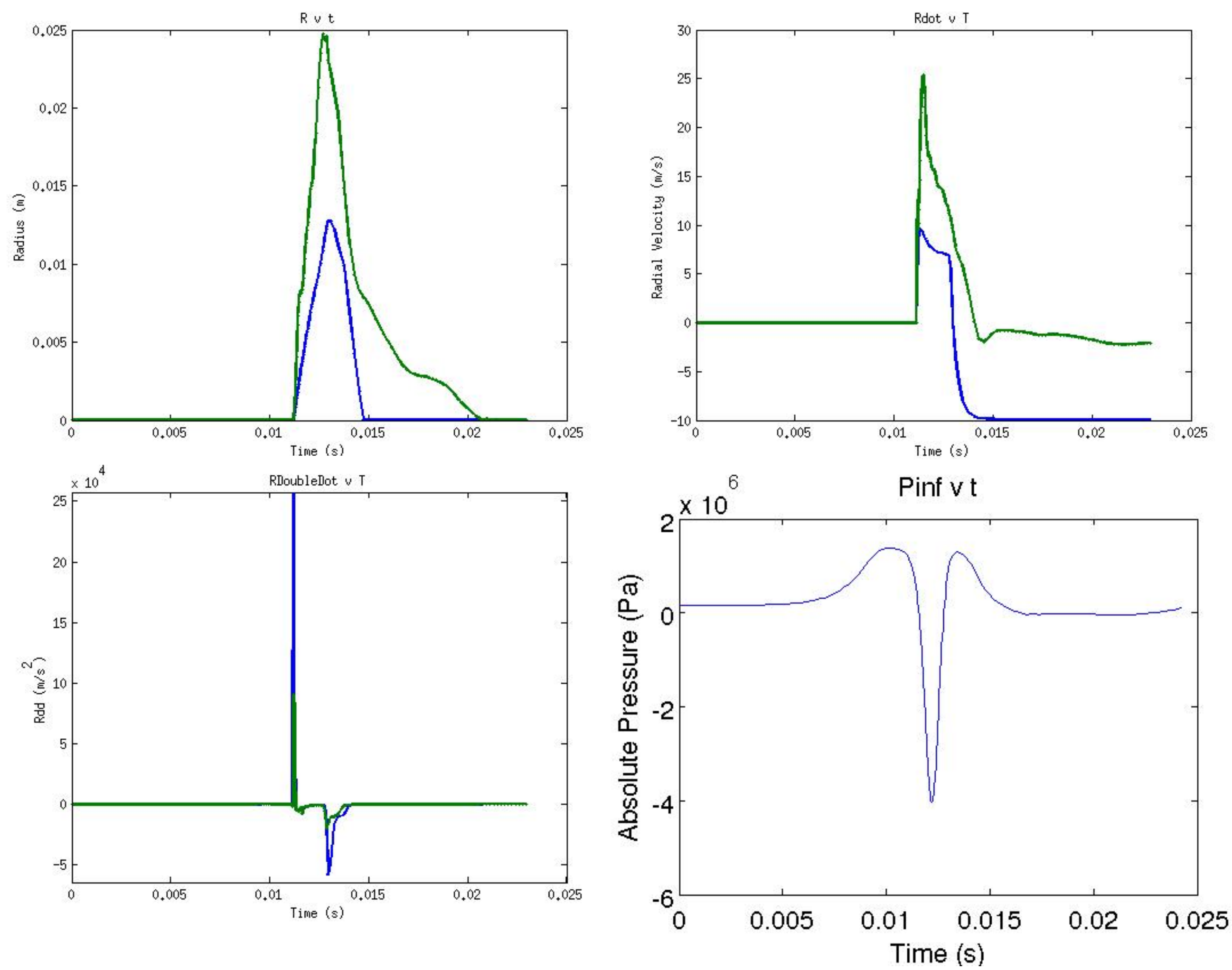


Figure 18 - MATLAB vs. StarCCM+ in flow over cylinder simulation

Although the magnitude of the radius is not exactly the same for the extracted StarCCM+ solution and the MATLAB solution, the general shape of the curves is nearly identical. As predicted from Figure 12, the radius of the bubble is near zero until it reaches the bottom of the cylinder, where it rapidly grows due to the low pressure region. Once it reaches the high pressure region behind the cylinder, the radius of the bubble decreases back to a minimum value. The error between the MATLAB and CFD solutions most likely arise from the convective terms in the latter. The CFD solution takes into account the motion of the bubbles in space as well as time, while the MATLAB solution only takes into account the pressure distribution as a function of time. The MATLAB's solution inability to include convective terms causes it to lose accuracy.

Another potential cause of error is the order of each solution's numerical integration schemes. The CFD solution implements a fourth-order Runge-Kutta scheme while the MATLAB solution implements a first order Euler scheme. Changing the latter to an RK4 scheme might accentuate the bubble radius growth, making it more closely match the CFD solution.

Altering Parameters in the Flow Over Cylinder Problem

After successfully integrating the RPE into a flow-over-cylinder problem, it is important to check the stability of the solution given various parameter changes. In this section, two parameters are altered, "Number of Bubbles per unit Volume" and "Initial Bubble Radius", and the effects that these two parameters have on the solution is analyzed. The goal is to verify that the RPE implementation produces reasonable results regardless of the field parameters, meaning it is reliable and applicable to a variety of flow situations.

First, the number of bubbles per unit volume is altered from 1 bubble/m^3 to 10, 100, 1000, and 10000 bubbles/m^3. Changing this parameter changes the clip value for the maximum radius as well as the clip values for $\ddot{R}$, since the number of bubbles per unit volume directly affects the maximum size of each bubble. Altering this parameter gives useful information as to how the solution changes in different flow situations. For the solutions to be viable in a large variety of problems, it must converge for many different bubble sizes. An UNDEX problem, for example, could have anywhere between roughly 1e2 bubbles/m^3 to 1e6 bubbles/m^3. Knowing how the solution changes based on bubble density is crucial for accurate modeling of real life cavitation events. For the following solutions, the initial bubble radius is kept at 1e-5 m. FIGURE NUMBER HERE shows the solution for R, $\dot{R}$, and $\ddot{R}$ as the number of bubbles changes:

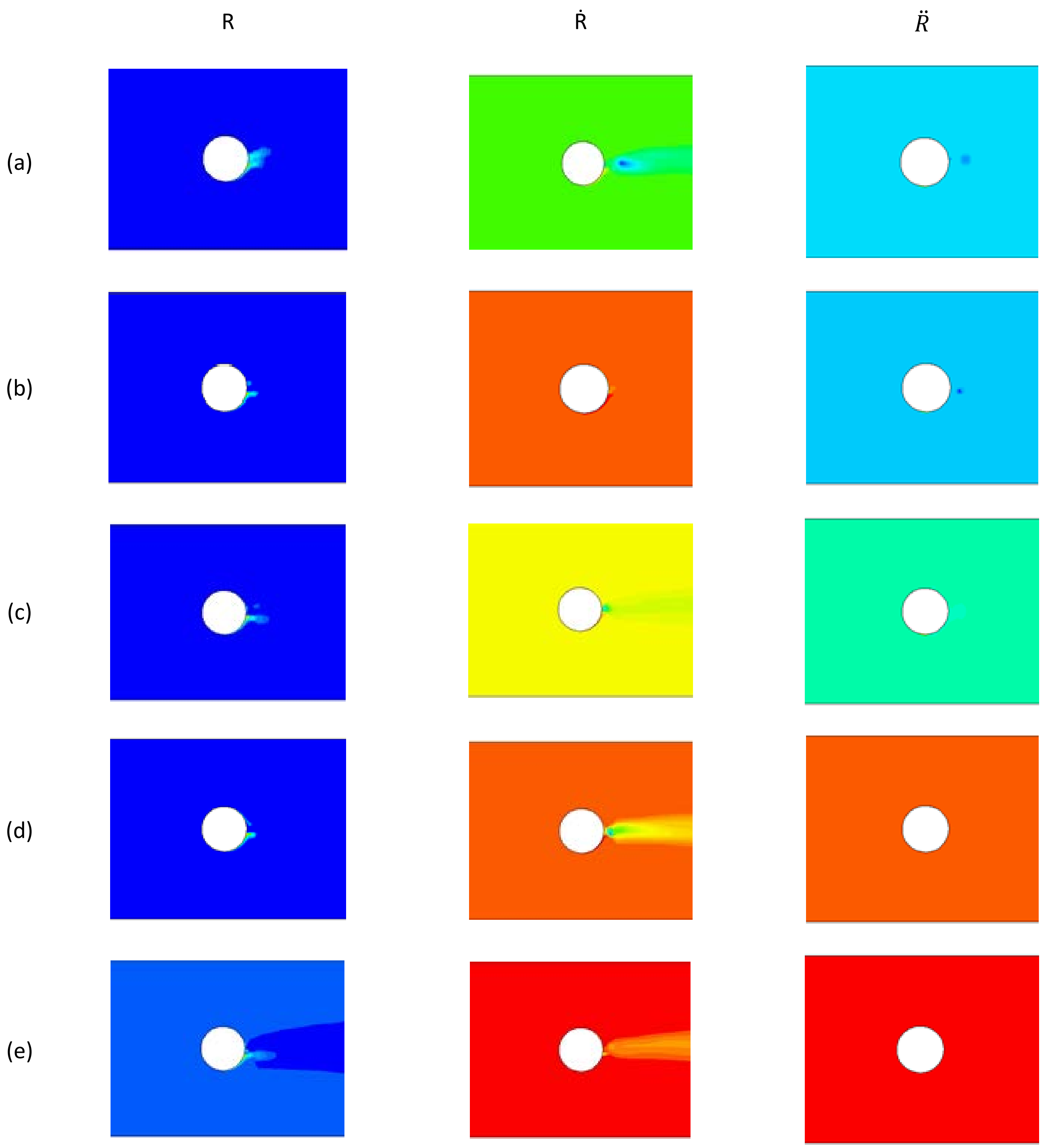


Figure 19 - Comparison of converged solutions for R, $\dot{R}$, and $\ddot{R}$ for the flow over cylinder problem. The parameter changed is "Number of Bubbles per unit Volume" and the units are bubbles/m^3: (a) 1 (b) 10 (c) 100 (d) 1000 (e) 10000

In terms of the bubble radius, the alteration of the number of bubbles per unit volume only affects the maximum bubble radius, not the location of cavitation. Similarly, $\dot{R}$ is not affected greatly either. The graphs for $\dot{R}$ above are shown at different scales, meaning the maximum and minimum values of $\dot{R}$ changed. However, on average, the values of $\dot{R}$ are relatively equal from one solution to the next. Although the magnitude of $\dot{R}$ did not increase or decrease greatly, the region in which $\dot{R}$ was negative was much larger for the higher bubble density cases. When the bubble density is 1 bubble/m^3 or 10 bubbles/m^3, $\dot{R}$ is low only near the back side of the cylinder. However, when the number of bubbles is increased, $\dot{R}$ is low from the back of the cylinder all the way until the end of the channel. This makes sense because $\dot{R}$ is directly dependent on $\ddot{R}$, which experienced a large effect from the change in bubble density. As bubble density increases, the maximum bubble size decreases. Thus, the maximum value of $\ddot{R}$ decreases throughout the channel. This in turn lowers the value of $\dot{R}$ in the channel, especially at points behind the cylinder, causing the low $\dot{R}$ region behind the cylinder to expand.

For very sparse bubble densities, namely 1 and 10, there are points to the right of the cylinder that have very low $\ddot{R}$ values. Here, the radial acceleration is negative, which means the lower bound for $\ddot{R}$ is low. This makes sense, since a sparse bubble population means bubbles will grow very large; a bubble collapsing from a large radius to a small one will have a large negative radial acceleration. Essentially, for lower bubble densities, the bounds on radial acceleration are extended, thus allowing large changes in bubble radius.

Altering the initial bubble radius produced similar results as above. Although changing the initial radius did not directly affect the clip values, it affected the maximum and minimum values of R. This in turn changed the upper and lower limits of $\dot{R}$ and $\ddot{R}$. The following charts of images show the solution for R, $\dot{R}$, and $\ddot{R}$ as the initial radius was changed between the values 1e-3, 1e-4, 1e-6, and 1e-7 meters. The number of bubbles is kept constant at 1000 bubbles/m^3:

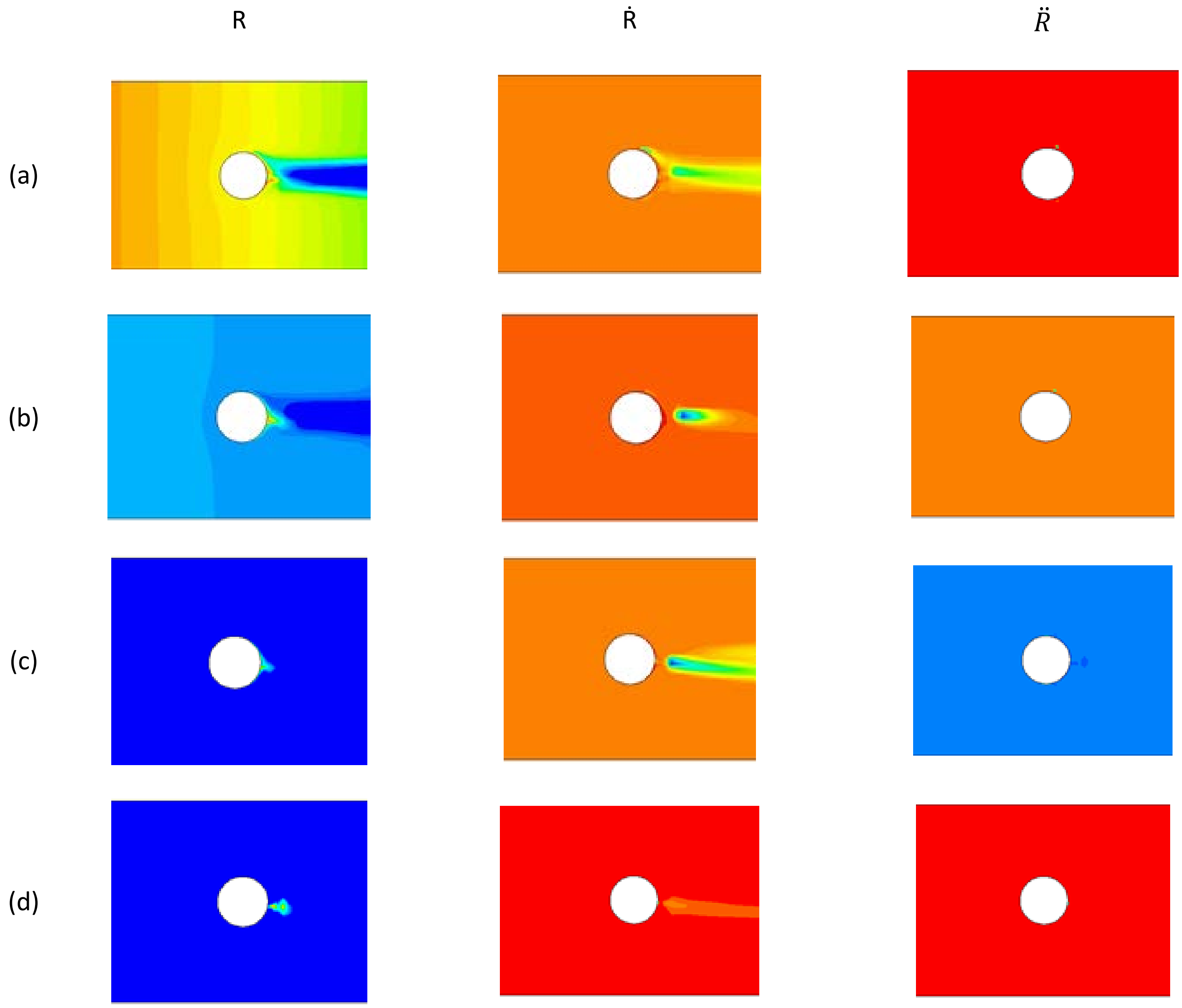


Figure 20- Comparison of converged solutions for R, $\dot{R}$, and $\ddot{R}$ for the flow over cylinder problem. The parameter changed is "Initial Radius" and the units are meters: (a) 1e-3 (b) 1e-4 (c) 1e-6 (d) 1e-7

Chapter 5 Conclusions

Cavitation modeling is of utmost concern to both Naval and commercial institutions. The ability to simulate bubble oscillations allows manufacturers to test the effects of cavitation on their product without wasting resources. Most modeling methods currently in use are reliant on Lagrangian frameworks, and although this makes it easy to track each individual bubble's parameters, it is quite difficult to simulate larger cavitation events. For this reason, an Eulerian method is explored.

The model discussed in this paper provided a singularity handling solution that allowed for solutions over 17 times faster than previously imagined. This solution, known as the "jump method", effectively skipped over singularity points and continued solving the Rayleigh Plesset Equation (RPE) while introducing a minimal error. Additional solutions, namely the RK4 Backtrace and Euler Backtrace methods, were provided as alternatives to the Jump method in specific cases where the latter method did not produce optimal results. I showed that these three singularity handling methods make it possible to solve the RPE in an Eulerian framework.

The three main methods presented were developed following a key assumption. The momentum conservation assumption, also called the "mirror-point" assumption, states that for each point near a cusp, there exists a point on the opposite side of the cusp with the same radius and opposite radial velocity. The jump method requires this assumption since it maintains the radius and multiplies the radial velocity by negative one. The backtrace methods also require it because they start calculating the next radius and radial velocity value backward from the mirror point. The mirror point assumption is validated by analyzing the variable time step solution. I showed that the ratio between the radial speed before the cusp and the radial speed after the cusp approaches one as the solution approaches the singularity.

The RPE was implemented in several different simulations of cavitation ranging from a simple square model to a more complex flow-over-cylinder model. The process involved creating source terms for the radius and radial velocity: the source term for radius was radial velocity and the source term for radial velocity was radial acceleration, given by the RPE.

The simplest case was the “Simple Square” simulation, where a pressure distribution as a function of time was inserted into StarCCM+. The resulting radius and radial velocities were analyzed as a function of time. This case was the starting point of the StarCCM+ validation, and it showed that for an uncoupled simulation without convection, the RPE implementation was successful.

The next case was the “Bottle Slam” simulation. This model represented a quarter-filled bottle of water being thrust downwards and then abruptly stopped. Here I tested the solver’s accuracy and speed when coupled with fluid dynamics and with minimal convection. Instead of inputting my own pressure distribution, I used the absolute pressure of the model in the RPE. Plotting the radius and radial velocity versus time in both StarCCM+ and MATLAB, I successfully validated the CFD model.

The final validation case was the “Flow Over Cylinder” model. This simulation was coupled with fluid dynamics and was dominated very strongly by convection. Again, I used the absolute pressure of the model in the RPE to calculate bubble radius and radial velocity. This simulation reached a steady state, so we extracted radius and radial velocity along a streamline that passed through the top of the cylinder. By using the flow parameters, namely flow velocity, I could calculate radius, radial velocity, radial acceleration, and pressure versus time. Inputting the pressure versus time data into MATLAB and comparing the results to the extracted radius and radial velocity values validated the cylinder model.

Although all the models were stable at small time steps, many required the use of boundary clip values to work with larger time steps. In addition, most simulations required an alteration of initial gas pressure to produce a reasonable suggestion. This suggests that cavitation is not only dependent on farfield pressure, but that it is also highly dependent on the pressure of the gas inside the bubble initially. The results show that if the farfield pressure is held constant, cavitation is still not guaranteed unless the initial gas pressure falls within a certain range.

As mentioned above, most of the simulations required boundary values to produce stable results. To calculate these clips, I assumed that the radius of a given bubble must be greater than the minimum radius. This minimum radius value was calculated using Brennan's equation. There was no upper bound for the radius. The bounds were $\dot{R}$ were formulating using another one of Brennan's equations. Using this range of values for R and $\dot{R}$, the bounds of $\ddot{R}$ could be calculated using the Rayleigh Plesset Equation. These bounds led to reasonable results, but more accurate bounds for R and $\ddot{R}$ were calculated with another method. By taking into account the mesh size and density, I could set upper limits for the radius of bubbles within each given cell. Then, using the second order Taylor series expansion for bubble radius, the upper and lower limits for radial acceleration could be found.

Until this point in the paper, the cases were all inviscid flows. A contour plot analysis of the effect of $\dot{R}$ and viscosity on $\ddot{R}$ values showed that for a viscous case, the formulations for clip values presented above may produce less than optimal results. For this reason, it is vital to study the viscous effects in a fluid flow before applying these boundary conditions. If the flow is relatively inviscid, then the derived clip formulae are more likely to produce accurate solutions.

My work here allows the CFD solvers to run at much smaller time steps while preventing any singularities from crashing the simulation. This is very similar to the singularity handling solution in MATLAB, except that it is based more on the physics of the bubble rather than the shape of the graph.

Further Considerations

In MATLAB and StarCCM+, I have implemented the RPE and formulated solutions to handle singularities, thus allowing the solver to run at a smaller time step. However, the next step in this research is to use the MATLAB singularity handling methods in StarCCM+ and the boundary clip conditions in MATLAB. This not only would validate both classes of methods, but it would also show that these modifications are versatile and applicable to many different environments. In some cases it may be easier to use one method over the other.

My Eulerian based framework needs to be tested against a Lagrangian framework to analyze the error and CPU time differences. This would give further proof that, for cavitation events with a high bubble density, the Eulerian framework outperforms the Lagrangian framework.

Relating to high bubble density, the RPE simulations in this paper should implement a two way coupling of the models to more accurately determine bubble radius and radial velocity. In a two-way coupled model, the particles are not only affected by the surrounding flow. They are also affected by the movement of other nearby particles that can alter the fluid flow. Currently, my models are all one-way coupled, and because I am using an Eulerian framework this means that the fluid does not respond to the bubble growth. Realistically, there are voids between bubbles that cause a significant alteration in flow parameters, thus changing the oscillations of surrounding bubbles. Two way coupling can be included, but it will require further investigation.

Finally, an important consideration is varying the mesh size of the simulations. Currently, my simulations run at similar mesh parameters. However, the validity of the models may change based on the fineness or coarseness of the mesh. It is important to run each simulation at multiple different mesh sizes to confirm that the methods above work no matter the mesh complexity.

BIBLIOGRAPHY

Alehossein, H., & Qin, Z. (2007). Numerical analysis of Rayleigh–Plesset equation for cavitating water jets. *International Journal for Numerical Methods in Engineering*, *72*(7), 780-807.

Brennen, C. E. (2013). *Cavitation and bubble dynamics*. Cambridge University Press.

Hosangadi, A., & Ahuja, V. (2005, June). A New Unsteady Model for Dense, Cloud Cavitation in Cryogenic Fluids. In *Paper No. AIAA-2005-5347, 17th Computational Fluids Dynamic Conference, Toronto, Ontario, CA*.

Hsiao, C. T., & Chahine, G. (2004). Prediction of tip vortex cavitation inception using coupled spherical and nonspherical bubble models and Navier–Stokes computations. *Journal of Marine Science and Technology*, *8*(3), 99-108.

Jayaprakash, Arvind, Sowmitra Singh, and Georges Chahine. "Bubble Dynamics in a Two-Phase Bubbly Mixture." Volume 7: Fluid Flow, Heat Transfer and Thermal Systems, Parts A and B (2010): n. pag. Web. 26 Oct. 2015.

Miller, S. T., Jasak, H., Boger, D. A., Paterson, E. G., & Nedungadi, A. (2013). A pressure-based, compressible, two-phase flow finite volume method for underwater explosions. *Computers & Fluids*, *87*, 132-143.

Obreschkow, D., Bruderer, M., & Farhat, M. (2012). Analytical approximations for the collapse of an empty spherical bubble. *Physical Review E*, *85*(6), 066303.

Plesset, M. S. (1949). The dynamics of cavitation bubbles. *J. Appl. Mech.*, *16*, 277.

Rayleigh, L. (1917). VIII. On the pressure developed in a liquid during the collapse of a spherical cavity. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *34*(200), 94-98.

Roberge, Pierre. "Cavitation and Impingement." *Cavitation Corrosion*. N.p., n.d. Web. 02 Nov. 2015.

Singhal, A. K., Athavale, M. M., Li, H., & Jiang, Y. (2002). Mathematical basis and validation of the full cavitation model. *Journal of fluids engineering*, *124*(3), 617-624.

Swift, E., & Decius, J. C. (1950). Underwater Explosion Research, Office of Naval Research. Underwater explosion research: vol. 2: the gas globe, Tech. rep., Department of the Navy.

Versluis, M., Schmitz, B., von der Heydt, A., & Lohse, D. (2000). How snapping shrimp snap: through cavitating bubbles. *Science*, *289*(5487), 2114-2117.

Academic Vita

Asish T. Balu

500 East College Avenue Apt. 301, State College, PA 16801
asishbalu@gmail.com | Cell (484)-340-2059

Education: **The Pennsylvania State University, University Park, PA**
B.S Engineering Science & B.S. Mathematics

Experience: **Applied Research Laboratory,** University Park, PA **June. 2014 – Present**
Intern – Computational Fluid Dynamics, Cavitation Modeling

- Created a singularity handling approach for numerical solutions to the Rayleigh Plesset Equation
- Simulated cavitation growth and collapse on various models using commercial CFD software
- Presented results at the American Physical Society, Division of Fluid Mechanics
- Drafted a senior thesis, successfully integrating the Rayleigh Plesset Equation into StarCCM+

Unmanned Aerial Systems Club, University Park, PA **July.2015 – Present**
Autopilot Team Lead – Flight Control Software

- Lead software development on the Pixhawk flight controller for the unmanned aerial vehicle
- Coded a QR Code Reader in C++ to accomplish the corresponding task at the AUVSI competition

Robotics Club, University Park, PA **Sept. 2012 – May 2014**
Treasurer – Arduino Developer

- Built and programmed an Arduino based robot to complete a maze autonomously
- Tracked the funds used to purchase and repair new and existing equipment
- Assisted newer members by helping debug their code and improve efficiency

Awards: **1st Place General Motors Competition**, Detroit, MI **Sept. 2014 – Oct. 2014**

- Collaborated with a multidisciplinary team of four to redesign the manufacturing line of a General Motors factory through technical innovation.

1st Place University of Pittsburgh DataPalooza Competition, Pittsburgh, PA **August 2013**

- Designed a Java program that organized AIDS data from 1984-2005. This data was fed into visualization software and then carefully analyzed.

Skills: C++ (and SDL Game Library), MATLAB, StarCCM+, Java (and Android SDK)