University of California, San Diego UCSD-CER-06-02

Time dependant Thermo-mechanical Modeling including Phase Changes in Direct Drive Inertial Fusion Energy Targets

Kurt-Julian Boehm

February 28, 2006

Center for Energy Research
University of California, San Diego
9500 Gilman Drive
La Jolla, CA 92093-0420

UNIVERSITY OF CALIFORNIA, SAN DIEGO

Time dependant Thermo-mechanical Modeling including Phase Changes in Direct Drive Inertial Fusion Energy Targets

A thesis submitted in partial satisfaction of the

requirements for the degree of Master of Science

in

Engineering Sciences (Mechanical Engineering)

by

Kurt-Julian Boehm

Committee in charge:

A. Rene Raffray, Chair
Mark S. Tillack
George R. Tynan

2006

The thesis of Kurt-Julian Boehm is approved:

_____

_____

_____
Chair

University of California, San Diego

2006

Table of Contents

List of Symbols

SYMBOL	MEANING	UNITS
ΔA	Change in Area	m^2
c_p	Specific heat of the corresponding material at constant pressure	$\frac{J}{Kg \cdot K}$
d	Diameter	M
d_{melt}	Melt layer thickness	M
D	Diffusion coefficient	$\frac{m^2}{s}$
D_o	Diffusion constant	$\frac{m^2}{s}$
E	Activation Energy (Chapter 3)	J
E	Young's Modulus (Chapter 7)	Pa
i	Variable representing radial position	No dimension
i_{fg}	Latent heat of vaporization	$\frac{J}{mole}$
Ja	Jacobs Number	No dimension
k	Boltzmann Constant (Chapter 3)	$\frac{J}{K}$
k	Thermal conductivity (Chapter 4,5,6,7)	$\frac{m^2}{s}$
L	Latent heat of Fusion	$\frac{J}{mole}$
M	Molecular mass	$\frac{Kg}{mole}$
$moles_{melt}$	Number of moles melted	$moles$
$1/m$	Poisson's Ratio	No dimension
p_g	Pressure in the gas	Pa
p_f	Pressure in fluid	Pa
$p_{no\ melt}$	Pressure in the liquid before melting occurs (Chapter 7)	Pa
$\dot{q}''$	Incoming heat flux	$\frac{W}{m^2}$
Q	Total heat (Chapter 4) Total heat per unit height (Chapter 5)	W, $\frac{W}{m}$

r	Radius of the sphere / cylinder	m
R	Universal gas constant	m
r^*	Nucleus size	m
r_o	Inner Radius of the Target	m
r_{out}	Outer radius of the target (Chapter 7)	m
r_b	Radius of the bubble	m
Δr	Change in radius	m
$s(t)$	Location of the solid – liquid interface (Chapter 5)	m
t	Thickness of the shell (Chapter 7)	m
t	Time	s
T	Temperature	K
T_{SAT}	Saturation Temperature at pressure in liquid	K
T_{vapor}	Temperature inside the bubble	K
T_w	Temperature of the reactor wall (Chapter 2)	K
Δt	Change in time	s
ΔT_{SAT}	Superheat temperature	K
V	Volume	m^3
V_m	Volume melted	m^3
$v_{DT\ liquid}$	Molecular volume of DT	$\frac{m^3}{mole}$
ΔV	Change in volume	m^3
$\Delta V_{molar,\,phase\ change}$	Change in molar volume due to the phase change solid to liquid at TP	m^3
α	thermal diffusivity	$\frac{m^2}{s}$
κ	Constant used in Chapter 7 to relate the deflection of the plastic and the DT shell	No dimension
λ	Coefficient used in equation 5.1 – 5.4	No dimension
μ	Reflectivity (Chapter 2)	No dimension
ρ	Density	$\frac{Kg}{m^3}$
θ	Angle in the grid	No dimension
$\Delta\theta$	Change in angle	No dimension
σ	Stefan Boltzman Constant (Chapter 2)	$\frac{W}{m^2 K^4}$
σ	Surface tension (Chapter 3-7)	$\frac{J}{m^2}$

SUBSCRIPTS:

		Used on
DT	Referring to a DT property	$V,\ \Delta V$, *t, E, d, 1/m*
i	Radial position in the grid	*T, k, r*
init	Initial temperature (Chapter 5)	*T*
j	Angular position in the grid	T, k, θ
l	Liquid (Chapter 5)	T, α , *k*
melt	Melting temperature (Chapter 5)	*T*
plastic	Referring to a plastic property	$V,\ \Delta V$, *t, E, d, 1/m*
s	Solid (Chapter 5)	T, α , *k*

SUPERSCRPITS:

n	Time step	*T*

List of Figures

List of Tables

ABSTRACT OF THE THESIS

Time dependant Thermo-mechanical Modeling including Phase Changes in Direct Drive Inertial Fusion Energy Targets

by

Kurt-Julian Boehm

Master of Science in Engineering Sciences (Mechanical Engineering)

University of California, San Diego, 2006

A. Rene Raffray, Chair

A two dimensional bubble nucleation mode was added to the previously presented thermo-mechanical model used to define the design margin for direct drive (DD) inertial fusion energy (IFE) targets. Tested on analytical solutions, the new model successfully simulates heating experiments on DT targets conducted at LANL. The ^{3}He present in the DT due to the tritium decay gets trapped in lattice sites evolving into nuclei big enough to serve as nucleation sites for heterogeneous bubble nucleation. Depending on the size of these nuclei, a certain liquid superheat temperature is required to nucleate bubbles. The liquid superheat temperature further increases as the pressure within the target induced by the plastic shell rises. The previous requirement for target survival was for the temperature of the DT to remain below triple point of DT (19.79K). If the existence of a melt layer does not

violate the symmetry requirements on the target for successful implosion, while the existence of a bubble does, the previous restriction can be lifted allowing for a melt layer to grow as long as the occurrence of bubble nucleation can be avoided. This study shows that melting and bubble nucleation can be timely separated. Depending on the ^{3}He nucleus size, the pressure in the target and the initial temperature of the target, the maximum allowable heat flux for a given survival time can be increased, allowing a less restricted design margin for the chamber design.

1. Introduction

Where do we stand?

This study extends the research work on direct-drive (DD) inertial fusion energy (IFE) target survival started at UCSD with the work of Brian Christiansen under supervision of Dr. Rene Raffray [1], [2], and [3]. The key basis of this research is the strict symmetry requirements imposed by target physics for compression and ignition of the DT fuel pellets using multiple laser beams. Wall radiation and energy exchange from the chamber gas can significantly impact these symmetry requirements in particular if phase change occurs. The phase change behavior of DT targets is quite complex (e.g. the presence of ^{3}He from tritium decay can influence the onset of nucleation) and needs to be better characterized through a combination of numerical modeling and experimental work.

The main focus of the present work is the expansion of the previous one-dimensional thermo-mechanical model into a 2-D version in order to more accurately simulate bubble nucleation and growth, and predict its effects. A more detailed estimate of the impact of phase change (solid to liquid and also nucleation) on the target symmetry would provide a better basis to determine whether the previously applied conservative restriction of maintaining the temperature of the target below the DT triple point (19.79 K) could be relaxed while still satisfying the target physics requirements. Allowing a higher outside temperature in the DT would justify the assumption of the target to be able to accommodate higher heat fluxes, and to be more

thermally robust to changes in the chamber environment during injection. In parallel to the numerical simulation of bubble growth, the heat loads on the target need also to be characterized for different chamber designs. Further, the effects of the tritium decay into ^{3}He need to be assessed, as it is suspected that the ^{3}He nuclei might enhance DT bubble formation in the first place.

Since the behavior of cryogenic DT under heat loading has generally been unexplored in the past, it is important that numerical modeling of phase changes as well as thermal and mechanical responses of the target is compared to experimental results to establish accuracy and reliability of the model. The present work includes a comparison of numerical results to analytical solutions for controlled cases first, followed by a comparison of numerical results to experimental results from Los Alamos National Laboratory (LANL) on DT heating. These experiments use cylindrical targets: so, the simulation code was first written in cylindrical coordinates, and then transformed to spherical coordinates to model IFE targets during injection in a chamber.

With this work, we intend to better define the design range for IFE targets and give guidelines as to what pellet design will be required depending on the injection velocity, background gas pressure and wall temperature.

2. Thermal Loading

In this section the heat load on the target during injection in the chamber is calculated for different scenarios and conditions associated with various chamber designs. Before the effects of the heat flux can be modeled, its magnitude has to be found. As recommended by Christiansen, DS2V, a commercial Monte Carlo gas flow simulation code [4], is used to estimate the energy exchange due to the interaction of the target with the chamber gas [1]. Previous results tended to focus on relatively high chamber gas densities (e.g. corresponding to ~50 mtorr at a ST=300K) required for wall protection in compact chambers (~6 m in radius). Recently, the HAPL program has been looking at the possibility of avoiding the use of a protective chamber gas by considering larger chambers (~10-11 m in radius). The absence of a protective gas relaxes the constraints from target heating and placement, and avoids any potential impact on laser propagation. The chamber environment during injection would then consist of the target burn remnants consisting mostly of He, D and T. The density of these chamber constituents during injection would depend on the vacuum pumping performance, but is estimated as less than ~1-10 mtorr at ST . Table 2.1 summarizes the results for an upper bound density case of 10 mtorr at ST (corresponding to a number density of 3.24×10^{20} m^{-3}) for various constituent gases at temperatures of 1000K and 4000K, respectively, and for injection velocities of 100 m/s and 400 m/s, respectively. Clearly, in the ranges relevant to IFE simulations, the raise in temperature from 1000 K to 4000 K has a higher effect on the heat flux than raising the injection velocity of the target from 100 m/s to 400 m/s. We can also see that a

heavier gas with larger molecules will create a smaller heat flux than a lighter gas with small molecules.

The radiation heat transfer can be found by equation (2.1):

$$q''_{rad} = (1-\mu)\sigma T_w^4 \qquad (2.1)$$

This heat load would be very significant if the reflectivity of the target surface was low. Using the proposed Au-Pd layer, the reflectivity of the target can be increased to as much as 96%. (μ = 0.96) [Appendix A] [1].

Table 2.1: The heat load and the drag force are listed in this table for Helium 4, Tritium, Deuterium and Xenon at temperatures of 1000K and 4000K and a pressure of 10 mTorr at ST. The small effect of the heat load when increasing the speed is shown by giving the heat flux at velocities of 100 m/s and at 400 m/s

Heat Load Chart for Different Gases, Temperatures, and Target Speeds					
Gas	**Temp.**	**Speed**	**Head Load**	**F - Drag**	**Parameter**
He	1000K	100m/s	0.55 W/cm^2	0.083 E-4 *N*	M= 4 g/mol
		400m/s	0.75 W/cm^2	0.35 E-4 *N*	m = 0.664 E-26Kg
	4000K	100m/s	4.3 W/cm^2	0.18 E-4 *N*	D= 1.86 E-10
		400m/s	5.0 W/cm^2	0.66 E-4 *N*	μ = 0.81
T	1000K	100m/s	0.65 W/cm^2	0.073 E-4 *N*	M= 3 g/mol
		400m/s	0.80 W/cm^2	0.30 E-4 *N*	m = 0.498 E-26Kg
	4000K	100m/s	4.9 W/cm^2	0.16 E-4 *N*	D= 1.86 E-10
		400m/s	5.4 W/cm^2	0.59 E-4 *N*	μ = 0.81
D	1000K	100m/s	0.75 W/cm^2	0.058 E-4 *N*	M= 2 g/mol
		400m/s	1.0 W/cm^2	0.24 E-4 *N*	m = 0.332 E-26 Kg
	4000K	100m/s	6.0 W/cm^2	0.11 E-4 *N*	D= 1.86 E-10
		400m/s	6.5 W/cm^2	0.48 E-4 *N*	μ = 0.81
	1000K	25m/s	0.75 W/cm^2		
	4000K	25m/s	4.8 W/cm^2		
Xe	1000K	100m/s	0.14 W/cm^2	0.49 E-4 *N*	M= 131 g/mol
		400m/s	0.6 W/cm^2	2.3 E-4 *N*	m = 0.218 E-24 Kg
	4000K	100m/s	0.9 W/cm^2	0.97 E-4 *N*	D= 3.8 E-10
		400m/s	2.0 W/cm^2	4.01 E-4 *N*	μ = 0.81

The effects on the reflectivity due to cryogenic dust particles are discussed in Appendix A. The concern addressed in Appendix A derives from the high adsorptivity value of cryogenic gas (CO2 and H2O), which might accumulate on the target surface during handling and could lead to a lower reflectivity value of the target surface.

Table 2.2: Different wall temperatures will impose different radiation heat fluxes onto the target. For the two limiting cases of 1000 and 1500 K wall temperature the respective heat flux due to radiation is given. Also, the effect of a reduced reflectivity of the target surface is illustrated.

Reactor wall temperature	Radiation heat flux μ = 0.96	Radiation heat flux μ = 0.92
1000 K	0.2 W/cm^2	0.4 W/cm^2
1500 K	1.2 W/cm^2	2.4 W/cm^2

The heat fluxes from both radiative and convective effects must be added to determine the total heat flux on the target.

The absence of a protective gas might enable the target placement requirements to be met with lower injection velocities, thus allowing for simpler mechanical injection systems [5]. In support of this, a parametric study was done to determine the minimum injection velocity required for the DT to stay below the triple point temperature (19.79 K) for different conditions. The results are described in Appendix B and the key findings summarized below.

Using a chamber pressure of 1 mTorr at ST and a wall temperature of 1000K two limiting cases (background gas temperature of 1000K and 4000K) were investigated to correlate the injection velocity and the initial target temperature required for the target temperature to remain under DT triple point at the end of the flight through a 6.5m

radius chamber. For the lower heat flux case (1000 K background gas) it was found that the injection velocity can be reduced to below 50 m/s, while keeping the initial target temperature reasonably low (16K). Increasing the velocity to values higher than 200 m/s shows little effect on the necessary initial temperature. In the higher heat flux case (4000 K background gas), initial temperatures below 12 K were computed necessary to reduce the injection velocity to 50m/s, which is a very significant reduction compared to the 17.5K required if the target flies at 400 m/s.

Generally, a high temperature background gas will required fast injection speeds while lower background gas temperatures will allow for slower injection speeds. Since the injection velocity has only little influence on the heat flux, the background gas temperature is the most relevant parameter for the heat flux and such for target survival.

3. ^{3}He - Bubble Formation

The effect of Helium in solids, especially in metals has been analyzed by many scientists in the past two decades. The present work focuses on the effect of the ^{3}He, implanted by the Tritium decay, in heating experiments of DT and its relevance to IFE target survival. Observations at the Los Alamos National Laboratories [6] suggest a close relationship between the concentration of ^{3}He in the fuel pellet and the appearance of bubbles during heating experiments. The present chapter explains the physics behind these observations.

3.1 Tritium Decay

Tritium decays into ^{3}He following equation (3.1) with a half life of 12.3 years. As a result of this comparatively low half time, a considerable amount of ^{3}He accumulates in the DT solid lattice within a few hours after layering the targets.

$$^3H_1 \rightarrow {}^3He_2 + electron \qquad (3.1)$$

For a power plant design it is estimated that a time of 4 – 10 hours will be required for target handling between layering and injection into the chamber [7]. During this time the decay of tritium inevitably causes two kinds of defects in the lattice: atomic displacement damage creating a vacancy and/or an interstitial atom, and the creation of a foreign element (^{3}He) which has the tendency to precipitate into bubbles [8]. In the literature, these defects are researched mainly for ^{4}He in different metals; the behavior of ^{3}He in a DT lattice has not been analyzed.

Nevertheless we believe that the physics behind the diffusion mechanisms, the kinetics of bubble nucleation, the changes of mechanical properties and the atomic properties of the ^{4}He in the lattice can be applied to ^{3}He in a DT lattice accordingly. This thesis concentrates on the diffusion mechanisms and nucleus formation because of their relevance to bubble nucleation in heating experiments.

The first physical observation is that with every single tritium atom decaying, one ^{3}He atom is created while the energy released by the decay (= 18.6 keV) would be enough to create an interstitial site. At this point we leave it open whether the net result is the creation of a vacancy and an interstitial or whether the ^{3}He atom only occupies the spot of the decayed tritium atom in the lattice.

3.2 Diffusion

As soon as one ^{3}He atom is present in the lattice, it will start moving around by random jumps to the neighboring lattice sites. These random jumps result in a certain distance that the atom will move in the lattice after a certain time (or a certain number of jumps). This movement is called solid state diffusion [9], [10]; it can happen to an interstitial atom as well as a substitutional atom, and it is highly temperature dependent. It is important to notice that interstitial diffusion is generally very fast compared to the vacancy or substitutional diffusion.

Trinkhaus [8] points out the strong binding of the helium atoms to vacancies. This means that an interstitial ^{3}He atom will move around freely in the lattice, until it falls into a vacancy. Once the atom occupies the vacancy it will require a much higher

energy to dislodge the atom for further diffusion. The energy required to move the helium atom from the lattice spot into an interstitial again is very large, which suggests that it will remain occupying the lattice site. Furthermore, there is space for more than one ^{3}He atom in one lattice vacancy, which results in the possibility of another interstitial ^{3}He atom falling into the vacancy, if the vacancy lies on the helium atom's random path [11]. This results in the vacancies acting like traps; as a result of these traps, He-3 starts building clusters, as there will be more and more He-3 atoms accumulating in the vacancies [11].

3.3 Nucleus Formation

Now that we have established the fact that there is an increasing concentration of ^{3}He (with time) in the DT region and that the ^{3}He atoms can diffuse into traps, we can assume that the traps will form into nuclei over a sufficiently long time period. According to Trinkhaus [8], the size and number density of these bubble nuclei are dependent on the temperature and the helium production rate. To be more precise, the bubble parameters depend on the diffusion rate and the helium concentration, with bubble formation occurring by concurrent diffusion and clustering of ^{3}He.

It seems reasonable to assume that the higher the overall concentration of ^{3}He, the higher the ^{3}He concentration trapped in clusters [11]. At the same time, the higher the trapped ^{3}He concentration, the fewer the formation of new bubbles. This sort of self limiting mechanism is responsible for the bubble concentration to tend to a constant value over a sufficiently long period of time. Both this characteristic time and the

number density of clusters will depend on the temperature of the medium, the energy levels of different size clusters and other unknown parameters like the strength of the sink imposed by the traps.

In addition, grain boundaries and other larger lattice defects can affect the diffusion and the trap strength. Diffusion along grain boundaries is extremely fast, while large lattice defects will trap gas molecules with a higher trapping energy than that of the lattice vacancies [9].

N.M Ghoniem, S. Sharafat, et al. developed a model including all the different adsorption and emission energies of atom and atom clusters into a nucleation model for helium diffusion in different metals [12]; results for ^{3}He in a DT lattice are not available yet. A model similar to the one developed by Ghoniem and Sharafat for ^{3}He diffusion in DT would go beyond the scope of this thesis. A simplified model has been developed to give a rough idea of the bubble sizes occurring in our case. The assumptions and simplifications are presented in the next section.

3.4 The Nucleus Formation Model

A precise calculation of the number density and nucleus size of ^{3}He gas in a DT lattice would require a time-dependent numerical model with all statistical possibilities, far too extensive to be within the scope of this thesis. Instead, the following model is proposed to provide a rough first estimate. The simplifying assumptions and physical justifications are explained here.

The time-dependent number density of ^{3}He in the domain can be calculated from the decay of the tritium in the domain. The production of the gas within the lattice can be assumed to be uniform, since the decay happens at random.

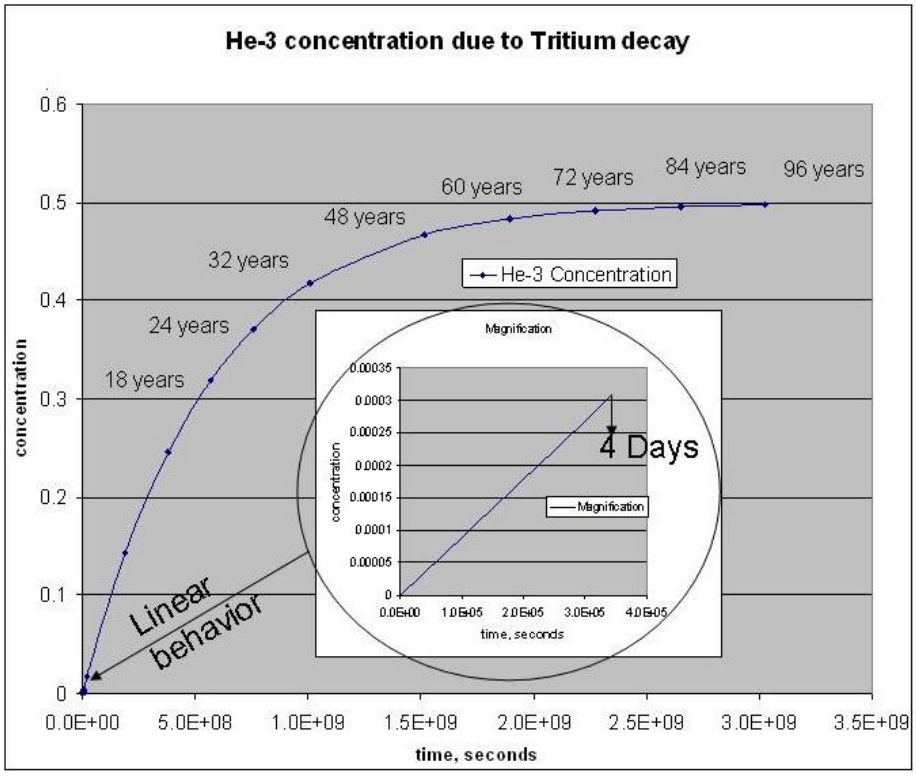


Fig. 3.1: The time-dependent concentration of ^{3}He due to tritium decay is plotted here. Notice that for the time frame we are interested in (up to a few days) the behavior is almost perfectly linear (slope: 1.6069 x 10^{-6} moles/hour).

A newly formed ^{3}He atom will migrate through the solid until it falls into a trap (a vacancy for example). The calculations assume a homogeneous distribution of traps and focuses on the spherical domain surrounding the trapping site.

Since we focus on a small part of the target domain (a spherical portion of a few microns in radius), the shape of the overall domain (in the target case the domain would be a hollow cylinder, the trap would be off- centered) is irrelevant, and we can

apply spherical coordinates to the small portion we are inspecting. The concentration on the innermost point of the domain is set to be zero, since we assume that the atom jumps into the vacancy, and doesn't come back out. The assumption of having a sink or irreversible trap at a random point in the domain can be justified following the ideas presented in the sections above. Ref.[11], [13], justifies the possibility of a continuously growing cluster by reporting that the closest lattice atom gets pushed into an interstitial position, if the number of ^{3}He atoms in the cluster becomes too large. This creates a di – vacancy which provides more room for more ^{3}He atoms. W.D. Wilson and, C.L. Bisson and M.I. Baskes [13] confirm that after the creation of a lattice vacancy 5 -8 atoms can conglomerate in the lattice. As more gas atoms fall into the vacancy trap, adjacent lattice atoms are pushed into an interstitial position providing space for the growing cluster. They also predicted a continuous growth of the nucleus as more and more atoms diffuse into the vacancy and call the trap 'insaturable'.

N. Kawamura et alt. [14] confirm that most to the ^{3}He created by the tritium decay will be trapped in the lattice of the frozen DT which supports the above described idea of insaturable traps.

Ref. [11] also suggests that the assumption of an immobile vacancy capturing interstitial ^{3}He atoms as they diffuse through the lattice can be defended. It is also mentioned by the author, that the trap effectiveness increases as the bubble nucleus continues growing.

Ref. [10] confirms a high mobility of He atoms even at low temperatures (although in that case copper is used as a lattice material).

Based on these statements, it seems reasonable to assume a simple model with a trap in the center of the domain acting as an irreversible sink term.

Once we have an estimate of the number of ^{3}He atoms which have migrated in the trap, an estimate of the radius of the formed ^{3}He nucleus can be made.

A key parameter in estimating the time for ^{3}He to conglomerates in the nucleus is the diffusion coefficient of ^{3}He in the solid DT lattice. In the literature, the following values were published for the diffusion coefficient:

Table 3.1: Different values for the temperature dependant diffusion coefficient

$D = D_0 \exp\left(-\frac{E}{kT}\right)$ (3.2)			
	Silvera [15] (**H2**)	Souers [16] (**H2**)	Souers [16] (**DT**)
$\frac{E}{k}$, [K]	200 ± 10	191	310
D_0 $[\frac{m^2}{s}]$	3E-3	1.4E-3	3E-4
D (@ 18K), $[\frac{m^2}{s}]$	n.a.	n.a	9.95 E-12

For the simple 1-D model, zero concentration is assumed as boundary condition in the center (trap location r=0) and zero concentration gradient as assumed at the outer boundary assuming equidistant traps.

Once the model returns the concentration profiles at each time step, we can compute the number of ^{3}He atoms in the trap by integrating the concentration in the whole domain.

3.5 Results from the Nucleus Formation Model

Following the simplifications presented above, the 1-D solid state diffusion code was implemented. Figure 3.2 shows the ^{3}He concentration profile as it changes every 5 minutes. The overall concentration of ^{3}He increases according to the tritium decay, while the trap effectively pulls a number of ^{3}He atoms into the void. By integrating the green shaded area and multiplying it with the corresponding volume, the number of ^{3}He atoms in the trap can be computed.

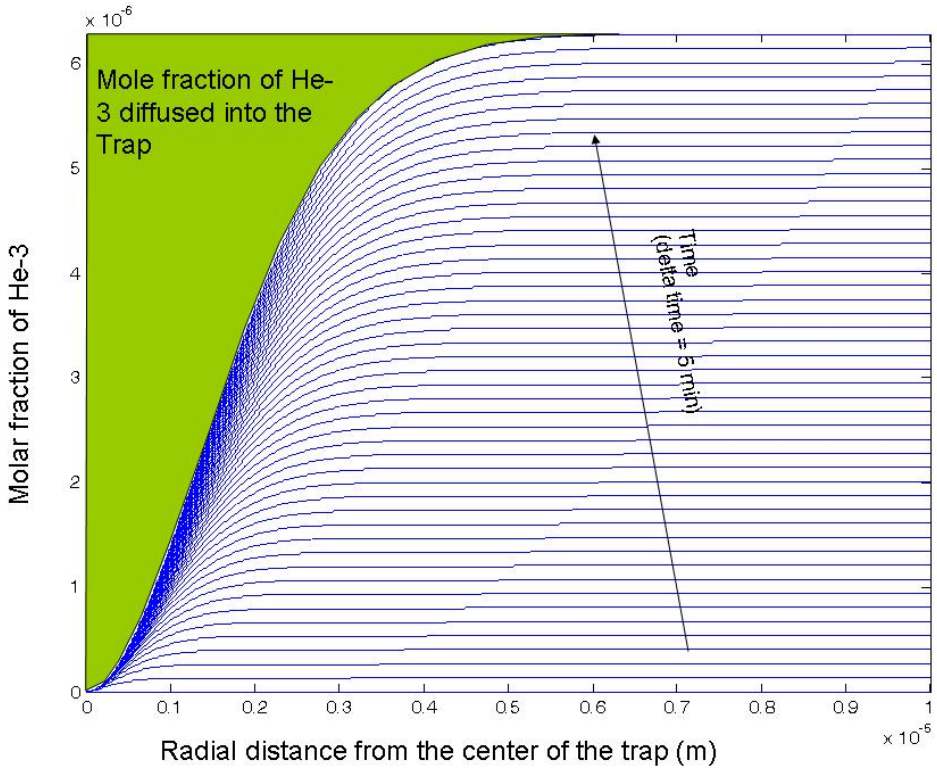


Figure 3.2: The ^{3}He-concentration as a function of the radial distance from the trap is plotted for 300 s time steps. The radius of influence for this plot is 10 microns. The green shaded area indicates the amount of ^{3}He that diffused into the trap in 4 hours.

Next, the effects of the radius of influence are analyzed. By using different sizes of radii of influence (or domain sizes), and plotting the respective final profiles after 4 hours, figure 3.3 was created. We are interested in estimating, how many ^{3}He molecules accumulated in the trap in four hours.

The first observation is that for radii of influence larger than 6 or 7 microns, the profile remains unaffected over the time period considered (4 hours); at a distance larger than 7 microns from the center of the trap, the model predicts a constant concentration of ^{3}He according to the tritium decay.

For smaller radii of influence the profiles change. Physically, imposing a small radius of influence means that two sinks are close enough for them to affect each other. The respective profiles can be extended by a mirror image to describe the adjacent trap.

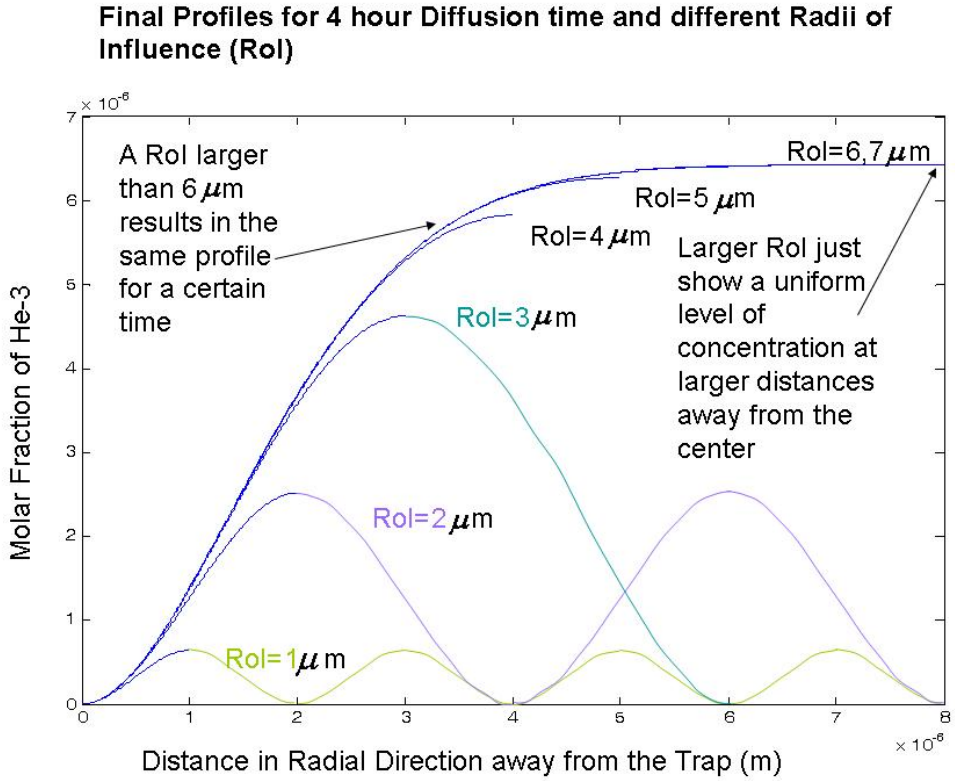


Figure 3.3: The profiles computed by the 1-D diffusion code are drawn here for different radii of influence. If the traps lie closely together, the overall ^{3}He concentration in the domain will remain low, since the ^{3}He gets trapped in the voids quickly after it was produced. If the traps are far apart from each other, they don't influence each other and the amount of ^{3}He trapped will approach a constant value. The minimum distance between two adjacent traps for them not to affect each other is related to the characteristic diffusion length.

Table 3.2 lists the values of ^{3}He atoms trapped in a void using different radii of influence and various diffusion times. Larger values of the radius of influence result in

an increasing number of ^{3}He atoms in the traps. For radii of influence larger than a certain value, the number of ^{3}He atoms in the trap approaches a constant value (for each diffusion time). The value after which the number of ^{3}He atoms in the trap remains constant is related to the diffusion coefficient and the characteristic diffusion length. Figure 3.4 shows graphically the results from table 3.2; the number of ^{3}He atoms per trap is plotted as a function of the radius of influence for different diffusion times. Notice that for longer diffusion times, larger radii of influence are required to reach a constant value of ^{3}He atoms in the trap.

Table 3.2: Numerical values for different diffusion times and radii of influence.

Radius (microns)	1 hour	Radius (microns)	2 hours	4 hours	8 hours	18 hours
1.00E-06	1.41E+05	1.00E-06	3.42E+05	7.50E+05	1.57E+06	3.60E+06
1.30E-06	2.34E+05	2.00E-06	1.49E+06	4.48E+06	1.10E+07	2.73E+07
1.70E-06	3.39E+05	3.00E-06	2.19E+06	9.04E+06	2.86E+07	8.32E+07
2.00E-06	3.91E+05	4.00E-06	2.38E+06	1.17E+07	4.71E+07	1.69E+08
3.00E-06	4.51E+05	5.00E-06	2.40E+06	1.28E+07	6.03E+07	2.66E+08
4.00E-06	4.57E+05	6.00E-06	2.41E+06	1.30E+07	6.77E+07	3.57E+08
5.00E-06	4.57E+05	7.00E-06	2.41E+06	1.31E+07	7.10E+07	4.28E+08
6.00E-06	4.57E+05	8.00E-06	2.40E+06	1.31E+07	7.22E+07	4.78E+08
7.00E-06	4.57E+05	9.00E-06	2.40E+06	1.31E+07	7.26E+07	5.10E+08
8.00E-06	4.57E+05	1.00E-05	2.40E+06	1.31E+07	7.27E+07	5.29E+08
9.00E-06	4.56E+05	1.50E-05	2.40E+06	1.31E+07	7.27E+07	5.46E+08
1.00E-05	4.56E+05	2.00E-05	2.40E+06	1.31E+07	7.27E+07	5.46E+08
1.50E-05	4.55E+05	2.50E-05	2.39E+06	1.31E+07	7.26E+07	5.46E+08

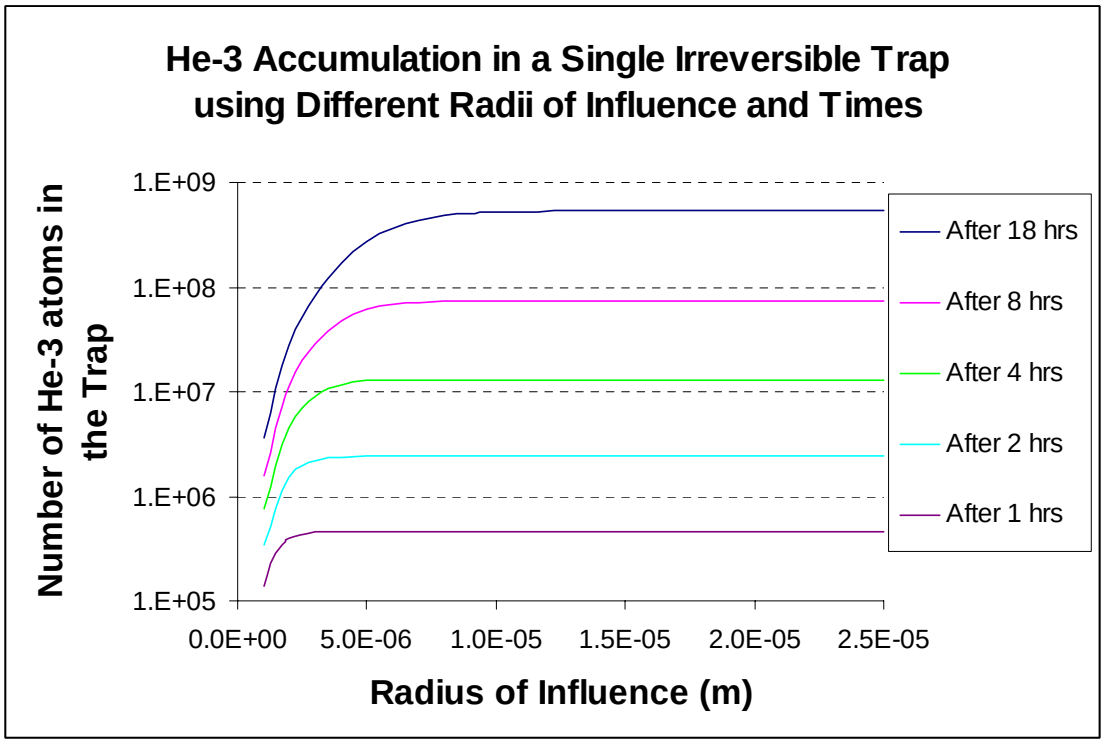


Figure 3.4: The ^{3}He accumulation in a single irreversible trap using different radii of influence and diffusion times are plotted here. The longer the diffusion time, the further the distance between two traps has to be for the two not to affect each other.

Based on these results, a value for the radius of a ^{3}He nucleus can be assigned to the number of ^{3}He atoms in a trap after the phase change from DT solid to liquid has occurred by applying the ideal gas law. These results can be seen in figure 3.5. Assigning a value in the solid state is difficult, because the pressure in the trap is unknown, while the pressure in the liquid phase is easy to calculate.

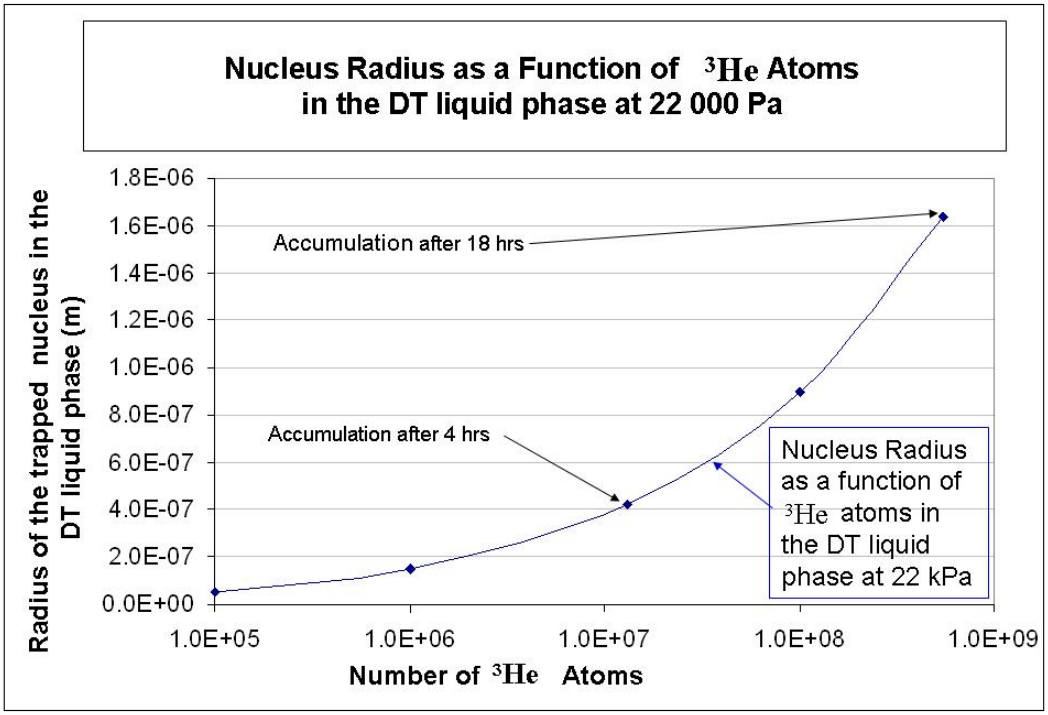


Figure 3.5: This figure relates the number of ^{3}He atoms present in a trap to a ^{3}He gas nucleus radius in the DT liquid phase at 22 kPa that this trap would involve into. The figure also indicates what size nucleus to expect after 4 and 18 hours.

3.6 Relevance to the Bubble Nucleation Model

When relating the ^{3}He gas accumulation in the solid DT lattice to a nucleus size in the liquid phase, we do that in anticipation of modeling a liquid to vapor phase change. This phase change is expected to happen according to classic nucleation theory. In this theory, a nucleus of a certain size is surrounded by a liquid of a certain pressure and temperature. If the temperature of the liquid is increased above a certain value, a thermodynamically unstable equilibrium is created, resulting in a bubble to grow out

of a pre-existing nucleus. If the pre-existing nucleus is created by non-condensable gas bubbles held in suspension in the liquid, as they do in our case, the nucleation mode is called heterogeneous nucleation [17], [18]. Collier states that the presence of dissolved gas reduces the superheat required to grow a bubble out of a nucleus of a certain size. Equation (3.3) relates the pressure inside the bubble to the pressure in the liquid. We observe an increase in pressure in the bubble as compared to the liquid pressure due to surface tension.

$$p_g = p_f + \frac{2\sigma}{r_b} \qquad (3.3)$$

The superheat required to produce an unstable equilibrium of a nucleus with radius r_b in a liquid of uniform temperature is described (in a simplified manner) by equation (3.4):

$$\Delta T_{SAT} = \frac{R \cdot T_{SAT}^{\ 2}}{J \cdot i_{fg} \cdot M} \cdot \frac{2\sigma}{p_f \cdot r_b} \qquad (3.4)$$

(Parameters represented by symbols are listed in the nomenclature section)

Clearly, if we want to avoid bubbles to nucleate in a system, smaller nuclei are desired, since the maximum allowable superheat is increased. In the following parts of this thesis, the relation between the onset of bubble nucleation in heated systems and the presence of nuclei of a certain size in a superheated liquid will be investigated. The results from this chapter will help us estimating what size of initial bubble nucleus to assume.

We can conclude from this chapter, that the decay of tritium into ^{3}He affects the bubble nucleation in heating experiments and will also affect target survival if phase change is allowed.

4. The Model

4.1 Overview

The need for an accurate model to describe the mechanical and thermal response of an IFE target has been previously discussed at length [1], [2]. Research and modeling work so far focused on the consequences of phase changes within the target assuming a one-dimensional geometry. This approach leads to modeling a liquid and vapor layer at the outer edge around the target. As far as the solid liquid phase change is concerned this is a valid simplification, since we do expect a melt layer to form symmetrically around the target, but modeling the liquid to vapor phase change in one dimension only, disagrees with the bubble nucleation theory and experimental heating observations. The Los Alamos National Laboratory (LANL) conducted a series of heating experiments past the triple point on frozen DT targets in 2004 and presented the results at the HAPL meeting in June 2004 at UCLA [6]. The present work will first simulate these experiments and predict a similar behavior by numerical modeling in the cylindrical geometry used in the laboratory and then translate these results into spherical coordinates to establish an accurate design margin for the DT targets.

4.2 The Setup and Results from LANL Experiments

The series of experiments relevant to this study were referred to as “direct-heating of solid DT layers” by the authors of the June 2004 HAPL presentation [6]. In that series of tests, a cylindrical target of 4mm diameter and 0.4mm height was exposed to direct heating from the outside using an electrical coil. The incoming heat flux was intended to be 1 W/cm^2, but was estimated to have a 10% error margin due to heat loss through the outside interface of the coils (Fig. 4.1). Several sets of experiments were conducted using different equilibrium temperatures (16K, 17K, 18K, and 19K) and equilibrium times (4hrs, 18 hrs). Pictures taken during the 100ms heat pulse clearly show bubble nucleation starting at different onset times and with a different bubble density on the outside layer of the target. Observations from that study indicate that there are relations between the equilibrium time and the bubble density as well as between the equilibrium time and the onset of bubble growth. We suspect that the foundation for these relations lies in the time dependant buildup of the ^3He concentration. These conjectures will be analyzed by a 2-D numerical heat transfer model and the results from the 1-D diffusion model from the previous chapter.

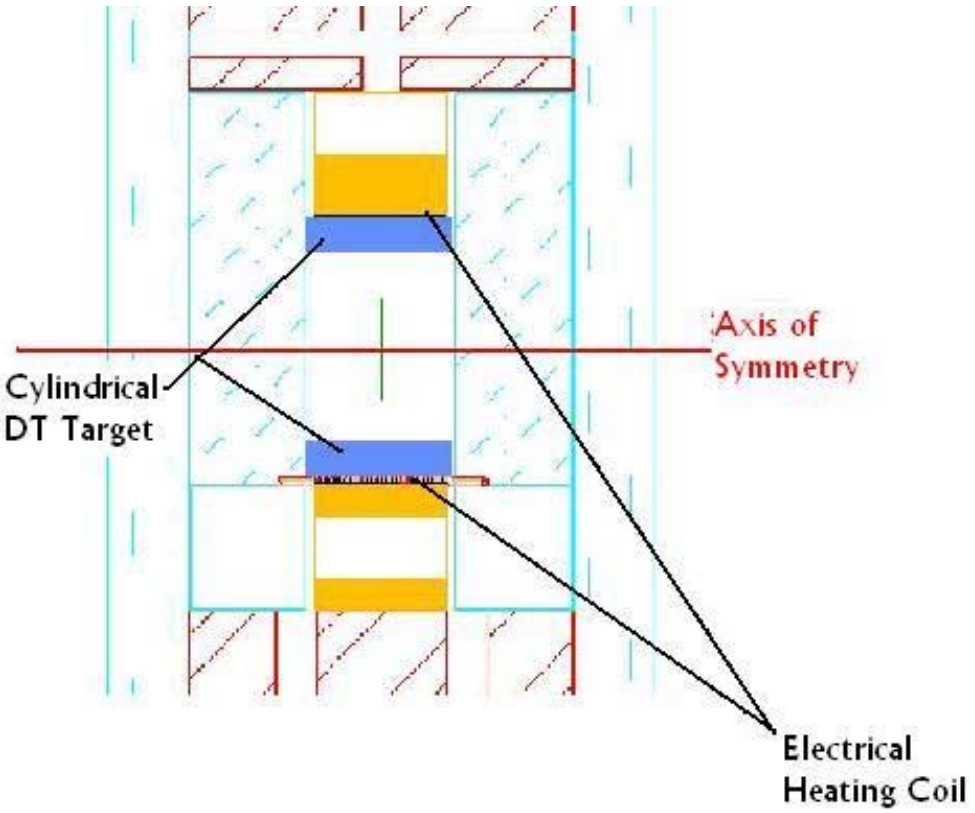


Figure 4.1: The experimental setup used at the LANL experiments is show here. The dark blue rectangles represent the cross section of the DT cylinder. Clearly the heating coils are mounted enclosing the outer surface. The pictures shown in the LANL presentation are taken along the axis of symmetry.

4.3 From 1 -D to 2 –D

Since the LANL experiments impose a constant heat flux around the cylindrical target, a one dimensional model can accurately be used to describe the behavior of the target until bubble nucleation occurs. It has been shown in the previous section 3.6 that bubbles will nucleate, if the liquid is sufficiently superheated and if a nucleus of a certain size is present in the liquid (see also [17], [18]). It has also been shown that these nuclei can occur due to the tritium decay to ^{3}He (section 3.5). If we

assume a nucleus site of a certain size to be present, we can calculate the superheat required for a bubble to nucleate. After using a one dimensional model to simulate the heat transfer until that superheat has been reached at a given distance from the outer edge, the effect of bubble nucleation will be simulated by switching to a 2-D model.

The heat conduction and solid liquid phase change will be modeled in 1-D, since the expansion into the second dimension only increases the computing time required to achieve results. The conduction equation then reads as follows [20]:

$$\frac{\delta T}{\delta t} = \frac{1}{\rho\, c_p(T)}\left[k\,\frac{\delta^2 T}{\delta r^2} + \frac{\delta T}{\delta r}\left(\frac{\delta k}{\delta r} + \frac{k}{r}\right)\right] \qquad (4.1)$$

Note that the only difference with the spherical coordinate system and equation (4.1) is that the last term is multiplied by two in the spherical case.

To account for the rapid change in thermal properties at temperatures in the cryogenic region, and to properly model the phase change, equation (4.1) includes temperature variant coefficients.

Applying uniform spatial and temporal discretization to equation (4.1), it results in the following equation:

:

$$T_i^{n+1} - T_i^n = \left(\frac{\Delta t}{\Delta r^2 \rho\; c_p}\right)\cdot\left[\begin{array}{l}\left(k_i + \dfrac{k_{i+1}-k_{i-1}}{4} + \dfrac{k_i \Delta r}{2\cdot\left(r_0 + \Delta r\cdot(i+1)\right)}\right)\cdot T_{i+1}^{n+1} + \\ \left(-2k_i\right)\cdot T_i^{n+1} + \\ \left(k_i - \dfrac{k_{i+1}-k_{i-1}}{4} - \dfrac{k_i \Delta r}{2\cdot\left(r_0 + \Delta r\cdot(i+1)\right)}\right)\cdot T_{i-1}^{n+1}\end{array}\right] \qquad (4.2)$$

This system of equations can be solved efficiently using the Thomas algorithm to second order accuracy in space and first order accuracy in time with no stability restrictions [21]. The subscripts note the variable's' location in space, while the superscripts note their temporal location.

The boundary conditions imposed in this case are a constant heat flux on the outside and zero gradient at the inner radius (r_0). For the zero-gradient boundary condition at the inner radius of the cylinder, the temperature at the second node is simply copied and set as the value on the first node. The outer edge boundary condition follows equation (4.3).

$$\dot{q}'' = -k \cdot \left(\frac{\delta T}{\delta r} \right) \qquad (4.3\text{ a})$$

$$\dot{q}'' = -k \cdot \left(\frac{T_{i+1}^{n+1} - T_{i-1}^{n+1}}{2 \cdot \Delta r} \right) \qquad (4.3\text{ b})$$

Solving for T_{i+1}^{n+1}:

$$T_{i+1}^{n+1} = -\frac{\dot{q}''}{k} \cdot 2 \cdot \Delta r + T_i^{n+1} \qquad (4.3\text{ c})$$

Equation (4.3 c) will be plugged into equation (4.2) for the outer node [20].

To model the solid to liquid phase change, the apparent c_p method is adapted from the previous model [1]. The idea behind this method lies in the engineering approximation that the phase change and the jump in specific enthalpy affiliated with it happens over a temperature range rather than as a step with an infinite slope.

Once it comes to modeling bubble growth, we need to switch to a two dimensional model. This adds complexity to the numerical simulation, since now the

equations can no longer be represented in a tri-diagonal matrix, but in a penta-diagonal system, which cannot be solved using the Thomas algorithm. Numerous methods are available to solve two and three dimensional systems (ADI, Spectral, …), [Pozrikidis] but the most efficient one in this case is an iterative scheme following the Gauss – Red – Black Algorithm [21]. This scheme converges quickly since the numerical solution doesn't vary largely from one time step to another. The even quicker converging Multigrid method was implemented in the code [22], but fails when the coefficients do not vary smoothly, which is the case with phase changes. It is shown in Appendix C, but remains an academic exercise for this application.

A stretched grid is implemented in the code to circumvent the problem of having a small enough grid spacing on the outer surface of the target (where phase changes occur, and bubble growth and steep temperature gradients are observed) while keeping the number of total grid points within a reasonable computational margin. A hyperbolic tangent function is used to stretch the grid in radial direction. The bubble will be modeled in the center of the pie-shaped domain; a hyperbolic sine function is used to stretch the grid in angular direction. This complicates the set of equations marginally but increases the overall performance and accuracy tremendously. To understand the equation (4.4), it is useful to know that the different grid spacings and locations are saved in a vector at the beginning of the code and just referred to as follows:

r_i Radius at the ith point

Δr_i difference in radius between the i+1th and the ith point

$\Delta \theta_j$ difference in angle between the j+1th and jth point

The set of equations, using a 2-d hollow cylinder with temperature-dependent coefficients and a stretched grid are written in equation (4.4 (a) and (b)) .

Cylindrical Coordinates (4.4a):

$$T_{i,j}^{n}\frac{\rho(T)\cdot c_p(T)}{\Delta t}=$$

$$\left\{\begin{array}{l}
\left[\begin{array}{l}\dfrac{k(T_{i,j}^{n+1})}{0.5\cdot(r_{i+1}-r_{i-1})}\cdot\left(\dfrac{1}{\Delta r_{i-1}}+\dfrac{1}{\Delta r_i}\right)+\\ \dfrac{k(T_{i,j}^{n+1})}{\left(0.5\cdot(\theta_{j+1}-\theta_{j-1})\cdot r_i\right)^2}\cdot\left(\dfrac{1}{\Delta\theta_{j-1}}+\dfrac{1}{\Delta\theta_{j+1}}\right)+\dfrac{\rho(T)\cdot c_p(T)}{\Delta t}\end{array}\right]\bullet T_{i,j}^{n+1}\\
\left[\dfrac{-k(T_{i,j}^{n+1})}{0.5\cdot\Delta r_{i-1}(r_{i+1}-r_{i-1})}+\dfrac{1}{\Delta r_{i+1}+\Delta r_i}\left(\dfrac{k(T_{i+1,j}^{n+1})-k(T_{i-1,j}^{n+1})}{\Delta r_{i+1}+\Delta r_{i-1}}+\dfrac{k(T_{i,j}^{n+1})}{r_i}\right)\right]\bullet T_{i-1,j}^{n+1}\\
\left[\dfrac{-k(T_{i,j}^{n+1})}{0.5\cdot\Delta r_i(r_{i+1}-r_{i-1})}-\dfrac{1}{\Delta r_{i+1}+\Delta r_i}\left(\dfrac{k(T_{i+1,j}^{n+1})-k(T_{i-1,j}^{n+1})}{\Delta r_{i+1}+\Delta r_{i-1}}+\dfrac{k(T_{i,j}^{n+1})}{r_i}\right)\right]\bullet T_{i+1,j}^{n+1}\\
\left[\dfrac{1}{r_i^{\,2}}\cdot\left(\dfrac{-k(T_{i,j}^{n+1})}{0.5\cdot\Delta\theta_{j-1}(\theta_{j+1}-\theta_{j-1})}+\dfrac{\left(k(T_{i,j+1}^{n+1})-k(T_{i,j-1}^{n+1})\right)}{(\theta_{j+1}-\theta_{j+1})^2}\right)\right]\bullet T_{i,j-1}^{n+1}\\
\left[\dfrac{1}{r_i^{\,2}}\cdot\left(\dfrac{-k(T_{i,j}^{n+1})}{0.5\cdot\Delta\theta_j(\theta_{j+1}-\theta_{j-1})}-\dfrac{\left(k(T_{i,j+1}^{n+1})-k(T_{i,j-1}^{n+1})\right)}{(\theta_{j+1}-\theta_{j+1})^2}\right)\right]\bullet T_{i,j+1}^{n+1}
\end{array}\right\}$$

Spherical Coordinates (4.4b):

$$T_{i,j}^{n}\cdot\frac{\rho(T)\cdot c_p(T)}{\Delta t}=$$

$$\left\{\begin{array}{l}
\left[\begin{array}{l}\dfrac{k(T_{i,j}^{n+1})}{0.5\cdot(r_{i+1}-r_{i-1})}\cdot\left(\dfrac{1}{\Delta r_{i-1}}+\dfrac{1}{\Delta r_i}\right)+\\ \dfrac{k(T_{i,j}^{n+1})}{\left(0.5\cdot(\theta_{j+1}-\theta_{j-1})\cdot r_i\right)^2}\cdot\left(\dfrac{1}{\Delta\theta_{j-1}}+\dfrac{1}{\Delta\theta_{j+1}}\right)+\dfrac{\rho(T)\cdot c_p(T)}{\Delta t}\end{array}\right]\bullet T_{i,j}^{n+1}\\
\left[\dfrac{-k(T_{i,j}^{n+1})}{0.5\cdot\Delta r_{i-1}(r_{i+1}-r_{i-1})}+\dfrac{1}{\Delta r_{i+1}+\Delta r_i}\left(\dfrac{k(T_{i+1,j}^{n+1})-k(T_{i-1,j}^{n+1})}{\Delta r_{i+1}+\Delta r_{i-1}}+\dfrac{2\cdot k(T_{i,j}^{n+1})}{r_i}\right)\right]\bullet T_{i-1,j}^{n+1}\\
\left[\dfrac{-k(T_{i,j}^{n+1})}{0.5\cdot\Delta r_i(r_{i+1}-r_{i-1})}-\dfrac{1}{\Delta r_{i+1}+\Delta r_i}\left(\dfrac{k(T_{i+1,j}^{n+1})-k(T_{i-1,j}^{n+1})}{\Delta r_{i+1}+\Delta r_{i-1}}+\dfrac{2\cdot k(T_{i,j}^{n+1})}{r_i}\right)\right]\bullet T_{i+1,j}^{n+1}\\
\left[\dfrac{1}{r_i^2}\cdot\left(\dfrac{-k(T_{i,j}^{n+1})}{0.5\cdot\Delta\theta_{j-1}(\theta_{j+1}-\theta_{j-1})}+\cdot\dfrac{\left(k(T_{i,j+1}^{n+1})-k(T_{i,j-1}^{n+1})\right)}{(\theta_{j+1}-\theta_{j+1})^2}+\dfrac{k(T_{i,j}^{n+1})}{\tan(\theta_j)(\theta_{j+1}-\theta_{j+1})}\right)\right]\bullet T_{i,j-1}^{n+1}\\
\left[\dfrac{1}{r_i^2}\cdot\left(\dfrac{-k(T_{i,j}^{n+1})}{0.5\cdot\Delta\theta_j(\theta_{j+1}-\theta_{j-1})}-\dfrac{\left(k(T_{i,j+1}^{n+1})-k(T_{i,j-1}^{n+1})\right)}{(\theta_{j+1}-\theta_{j+1})^2}-\dfrac{k(T_{i,j}^{n+1})}{\tan(\theta_j)(\theta_{j+1}-\theta_{j+1})}\right)\right]\bullet T_{i,j+1}^{n+1}
\end{array}\right\}$$

The imposed boundary conditions remain the same in the radial direction. In the axial direction periodic boundary conditions are used, so technically we could use a spectral method in the angular direction, but this has not been implemented in the present work.

Now we have to face the challenging task of modeling a 3-dimensional spherical bubble in a 2- dimensional cylindrical domain.

4.4 Modeling a Bubble

The superheat required for a bubble of a certain size to grow can be calculated by using equation (4.5) [17]:

$$T_{vapor} - T_{SAT} = \frac{R}{M} \cdot \frac{T_{sat} \cdot T_{vapor}}{i_{fg}} \cdot \frac{2\sigma}{p_f r^*} \qquad (4.5)$$

If we assume a nucleus of a certain size to be present in the melt layer (due to ^{3}He decay), the required superheat can be determined by the model as it is discussed so far. However, modeling the bubble nucleation and growth is a challenging task, since the bubble is spherical in geometry, and it is off center of our overall cylindrical domain. It has been shown though [18], that bubble growth in the time- and size-frame relevant to this study is heat flux restricted, which is convenient for our heat diffusion model. According to S. Van Stralen [18], the bubble grows exactly as fast as the heat can be delivered to the liquid – vapor interface, where it is used to account for the latent heat required by the mass flux of vapor into the bubble.

Assuming that the vapor inside the bubble is at thermodynamic equilibrium, we can predict the temperature inside the bubble to be the saturation temperature at the pressure inside the bubble. We can relate the pressure in the bubble with the pressure of the surrounding liquid by equation (4.6).

$$p_b = p_f + \frac{2\sigma}{r_b} \qquad (4.6)$$

Clearly, the temperature in the bubble and the superheat required to grow the bubble are closely related (see eqs (4.5) and (4.6)).

This suggests that the bubble can be represented by a heat sink imposed into the domain. The temperature in the bubble is lower than the temperature in the surrounding liquid, drawing thermal energy to the surface. Numerically, we can add a heat sink term by assigning a certain temperature to an arbitrary point in the domain, which is lower than the temperature of the neighboring points. Physically, by choosing the location of such a point in the grid, the location of the nucleus and as such the point from which the bubble will start growing is determined. Once the necessary superheat to grow a bubble out of a nucleus of a certain size is reached, the temperature at that point gets dropped, starting the heat sink. This is justifiable since we assume a nucleus filled with ^{3}He to be present in the superheated liquid. Once the temperature in the surrounding liquid is higher than the saturation temperature at the pressure in the bubble, DT starts evaporating into the bubble, making it grow. Once the bubble grows the pressure in the bubble drops even further (see eqn 4.6) lowering the temperature as well. The energy which the heat sink absorbs will be conserved in the system, as energy in form of latent heat is required to evaporate some DT to make the bubble grow. As such, the energy flowing into the heat sink must be tracked and equated to the energy required evaporate enough DT to grow the bubble from one time step to another. This energy can be calculated using the heat flux equation (4.7a) and the area over which this heat flux acts (4.7b). When applying this equation to the grid, we need to realize, that the first term in the brackets of eqn. (4.7b) is the heat transfer in the radial direction, acting over the change in angular direction while the second term represents the heat transferred in angular direction over the change in radius.

$$\dot{q}'' = k \cdot \left(\frac{\delta T}{\delta r} + \frac{1}{r} \cdot \frac{\delta T}{\delta \theta} \right) \tag{4.7 a}$$

$$\frac{Q}{unit\ length} = k \cdot \Delta t \cdot \left(\left(\frac{T_{i+1,j}^{n+1} - T_{i,j}^{n+1}}{\Delta r_i} \right) \cdot \left(\Delta \theta_j R_i \right) + \left(\frac{T_{i,j+1}^{n+1} - T_{i,j}^{n+1}}{R_i \cdot \Delta \theta_j} \right) \cdot \left(\Delta r_i \right) \right) \tag{4.7b}$$

$$\frac{Q}{unit\ length} = \frac{\Delta A}{unit\ length} \rho\, i_{fg} = \frac{\pi \left(r_i^2 - r_{i-1}^{\ 2} \right)}{unit\ length} \frac{p_g}{RT_{bubble}} i_{fg} \tag{4.8}$$

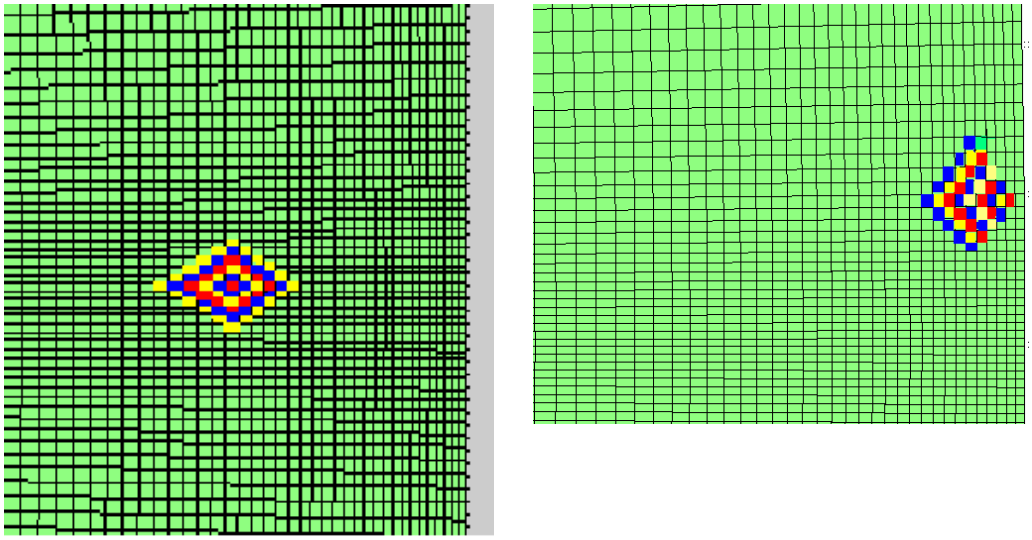


Figure 4.2: Different modes of bubble growth are shown in this figure. On the left, the bubble grows outward radially symmetric from a point in the middle of the domain. On the right, we assume the bubble nucleus to be close to the outer edge of the cylinder, and the bubble growth happening inward.

The obvious physical observation suggests that the energy is used by the system for the liquid to vapor phase change at the interface of the bubble. Because of the discretized nature of our domain, bubble growth can only be modeled step wise. We propose two modes of bubble growth, shown in Figure 4.2. The bubble growth follows the grid either radially symmetric outward (left) or inward from the outer edge

of the domain as shown on the right. More involved modes could be implemented to model a more spherical shape, but the presented results used these two modes.

We are now able to calculate the area that each ‘bubble’ covers on the grid at each step. Noting that the area of the ‘bubble’ on the gird is related to the volume by unit height of the cylinder, we can calculate the amount of heat necessary (per unit height) to grow the bubble from one step to the next. Knowing the heat required to grow the bubble on the one hand and the heat flux into the bubble from the heat sink model on the other (see Figure 4.3), we are now able to model bubble growth by matching the two. The only free parameter left for adjustment is the time step.

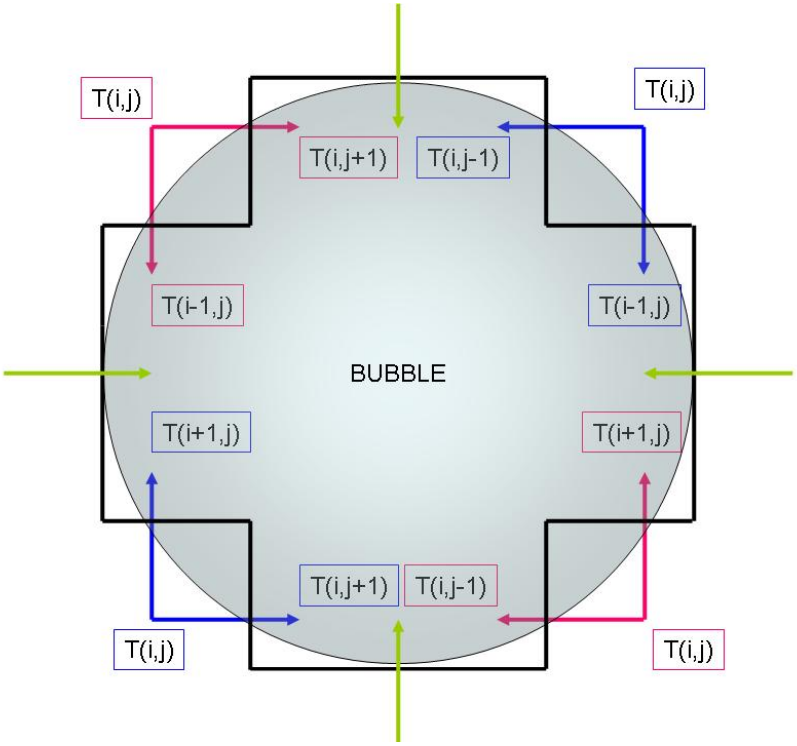


Figure 4.3: this figure shows the schematic approximation of how a bubble is represented in the overall domain. Notice that the heat transfer into the bubble comes from the temperature difference in radial and angular direction.

The numerical simulation consists of the following steps:

First, the grid is created. Since we have a fixed mesh, the bubble sizes that will be represented by the model are determined by that.

Second, the model runs in 1-D for uniform heat fluxes until the superheat at the point specified for bubble nucleation is reached.

Then, we switch to a 2-D model. This model first calculates the heat required for bubble growth at that time step. After that an iterative scheme is used to find the right size of the time step to conserve the energy of the system. Once the time step is found, the bubble grows from one step to the next; the temperature of the points of the new bubble is adjusted to the saturation temperature of the pressure in the bubble, creating a larger heat sink.

Several points must be considered using this approach to accurately compare to the ‘real physics’ of bubble nucleation.

4.5 Approximations to the Bubble nucleation model

The first problem we encounter derives from the discretized (step wise) growth of the bubble we model as compared to the smooth, gradual growth in real physics. In the model, a heat sink is applied on the domain. The profile relaxes over the time step according to the heat diffusion equation. The gradient flattens and subsequently the heat transfer into the bubble slows down as the profile relaxes. The problem we are now facing is which profile to use for the heat flux into the bubble, the one immediately after the heat sink is defined at the beginning of the time step, or the

flatter one at the end of the time step. Both profiles are artificial creations originating from the step wise growth of the bubble. In this model we propose to use half the heat flown into the bubble according to the initial profile, and half of the heat flown in according to the final profile.

The second problem to be handled also results from the step wise growth and concerns the conservation of mass in the system. The density of the vapor in the bubble is about 1000 times lower than the density of the liquid. By creating the bubble, the volume of the whole cylindrical domain increases. For small bubbles in a large domain, this increase in volume is negligible, but this imposes some limitations on the model for larger bubble sizes. One aspect impacting the bubble growth is the way the temperature field around the bubble changes as the bubble grows. Bubble growth pushes the liquid mass surrounding the bubble out. The mass in then distributed then over a larger radius. This means that a point where the temperature has been computed at the previous time step moves to a different location as the bubble grows. We have to calculate the new temperature of the point represented in the grid (Figure 4.4). To get the exact place $r^{**}_{1,2,3,4,5}$ at which to compute the temperature $T\left(r^{n+1}_{1,2,3,4,5}\right)$, we need to account for this radial effect and for the conservation of volume (Figure 4.5).

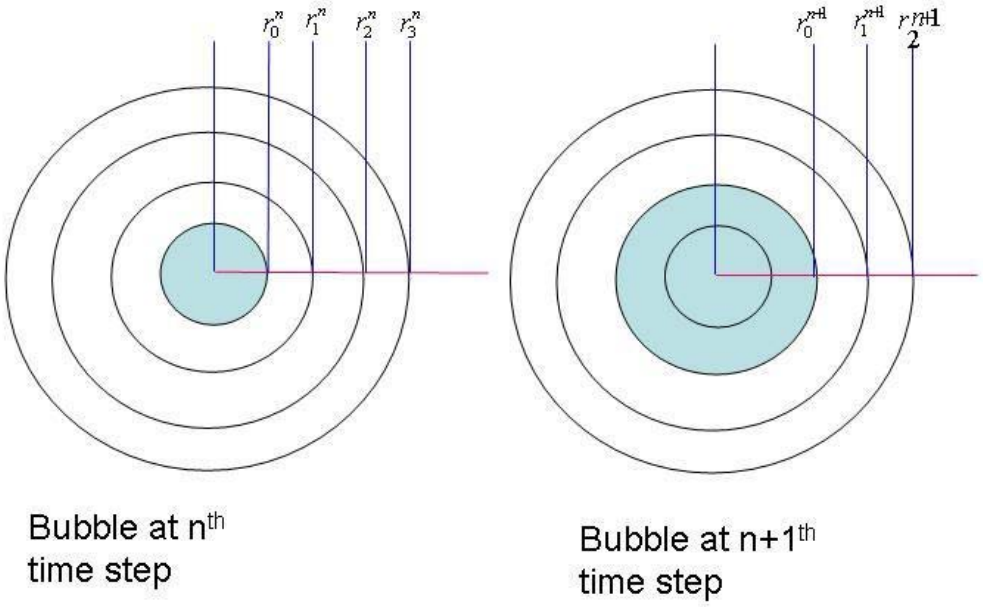


Figure 4.4: The simplified bubble growth modeled in the code makes the bubble size jump from one grid size to another. In doing so, we need to move the profile radially outward. The volume surrounding the bubble on the left between the grid spacing is different than the volume between the grid spacing on the right due to radial effects.

Volume between r_0^n *and* $r_1^n \neq$ *Volume between* r_0^{n+1} *and* r_1^{n+1}

observe:

$$r_0^{n+1} = r_1^n$$

$$r_1^{n+1} = r_2^n$$

We then use linear interpolation between the respective points to get the temperatures.

$$r_{1,2,3,4,5}^{**} = \sqrt{r_{2,3,4,5,6}^{n}{}^{2} - r_1^n + r_0^n} \qquad (4.9)$$

$$\frac{T\left(r_{1,2,3,4,5}^{n+1}\right) - T\left(r_{1,2,3,4,5}^{n}\right)}{T\left(r_{2,3,4,5,6}^{n}\right) - T\left(r_{1,2,3,4,5}^{n}\right)} = \frac{r_{1,2,3,4,5}^{**} - r_{1,2,3,4,5}}{r_{2,3,4,5,6} - r_{1,2,3,4,5}} \qquad (4.10)$$

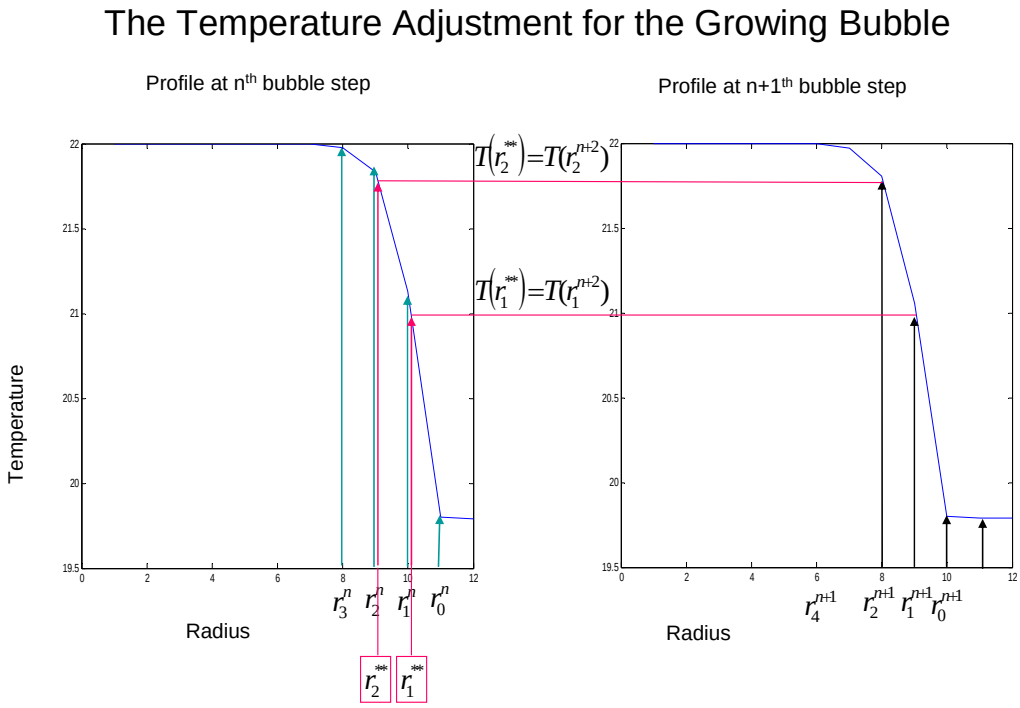


Figure 4.5: As the bubble grows, the temperature field needs adjustment. The temperature at a specific radius away from the bubble is computed by interpolation between the neighboring points at the previous time step.

When modeling the bubble nucleation by just assigning a lower temperature value at the grid point determined to represent the bubble, the model will be unable to resolve any heat transfer through the bubble. Neither will the model be able to express a temperature gradient through the bubble. By looking at the physics, it can be reasoned, that the thermal resistance through the bubble is much higher than through the liquid surrounding it, suggesting, that the heat flowing through the bubble will have no effect compared to the temperature distribution around the bubble. Looking at the mean free path of the gas molecules inside the bubble, we can justify the constant temperature assumption within the bubble.

4.6 The Spherical Bubble Model

When applying the above model to spherical coordinates, a 3-D bubble can be modeled with a 2-D code. Symmetry around one axis makes this possible as will be shown below:

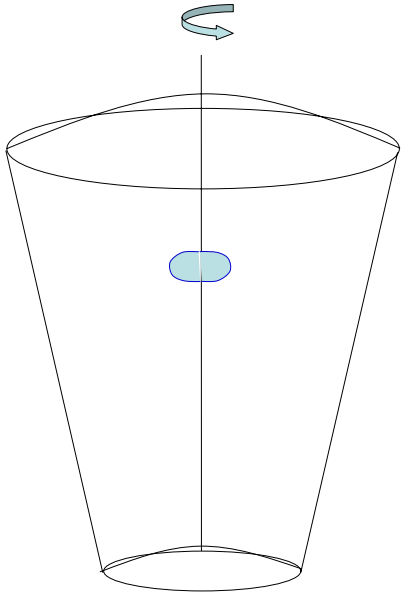


Figure 4.6: When modeling a bubble in 2-D in spherical coordinates, physics allows us to model a 3-D shape, by rotating the (blue) shape around on axis of symmetry. The boundary conditions at the rotating axis must be symmetrical, while we impose (zero-gradient) Newman boundary conditions at the outer edge of the rotating angle.

The model follows the same scheme described in the preceding section, the only difference being the following different formulas for heat diffusion and heat flux:

$$\dot{q}'' = k \cdot \left(\frac{\delta T}{\delta r} + \frac{1}{r} \cdot \frac{\delta T}{\delta \theta} \right) \tag{4.11a}$$

$$Q = k \cdot \Delta t \cdot \left(\begin{array}{l} \left(\frac{T_{i+1,j}^{n+1} - T_{i,j}^{n+1}}{\Delta r_i} \right) \cdot \left(\pi \cdot \left(\left(\theta_{j+1}{}^2 - \theta_j{}^2 \right) \cdot R_i \right)^2 \right) + \\ \left(\frac{T_{i,j+1}^{n+1} - T_{i,j}^{n+1}}{R_i \cdot \Delta \theta_j} \right) \cdot \left(2 \cdot \pi \cdot R_i \cdot \theta_j \cdot \Delta r_i \right) \end{array} \right) \tag{4.11b}$$

Note that in equation (4.10) we integrate the heat flux over a complete area of the bubble as opposed to integrating over a length and leaving the other dimension to be unit length.

In doing so, we have to compute the heat required to grow the bubble also as the total heat instead of heat per unit length.

$$Q = \Delta V \cdot \rho \cdot i_{fg} = \frac{4\pi}{3} \left(r_i^3 - r_{i-1}{}^3 \right) \cdot \frac{p_{bubble}}{R T_{bubble}} \tag{4.12}$$

5. Testing the Model against Analytical Solutions

Before we try to analyze the LANL experimental results or predict the behavior in the IFE target case, the model needs to be checked against analytical solutions for well-defined cases.

5.1 Liquid to Solid Phase Change – Melting and Solidification

In order to establish accuracy of the solid – liquid phase change as it is modeled in the code, an analytical solution had to be found, and the numerical and analytical results needed to be compared. As has been previously proposed, the apparent c_p method is used to model the phase change. This idea has been implemented and tested in the spherical code [1], but remained untested in the cylindrical case.

Ozisik [23] derived an analytical solution for the case of a heat sink in the center of a cylindrical domain. As the heat sink -surrounded by initially superheated liquid- is turned on, a cylinder solidifies growing symmetrically outward in radial direction.

This solution is given by the following set of equations:

Find the coefficient λ using equation (5.1)

$$\frac{Q}{4\pi}\exp\left(-\lambda^2\right)+\frac{k_l\left(T_{init}-T_m\right)}{Ei\left(-\frac{\lambda^2\alpha_s}{\alpha_l}\right)}\exp\left(-\frac{\lambda^2\alpha_s}{\alpha_l}\right)=\lambda^2\alpha_s\rho_s L \qquad (5.1)$$

Use λ to compute the thickness of the solid layer and the temperature profiles in the two phases using equations (5.2) – (5.4):

$$\lambda = \frac{s(t)}{2(\alpha_s t)^{1/2}} \tag{5.2}$$

$$T_s(r,t) = T_m + \frac{Q}{4\pi k_s}\left[Ei\left(-\frac{r^2}{4\alpha_s t}\right) - Ei(-\lambda)\right] \tag{5.3}$$

$$T_l(r,t) = T_{init} - \frac{T_{init} - T_m}{Ei\left(-\frac{\lambda^2 \alpha_s}{\alpha_l}\right)} Ei\left(-\frac{r^2}{4\alpha_l t}\right) \tag{5.4}$$

Ei is the exponential integral and exp the exponential function

The following figures (5.1 and 5.2) show a plot of selected profiles following equations (5.3) and (5.4) for selected times at a certain heat strength and compare those with the results of the code applying the boundary conditions corresponding to the input parameters of the analytical solution.

Numerical problems evolve on both ends of the domain, since we need to impose a close to infinite slope at the center and an unknown slope on the outer edge. The r=0 boundary condition is resolved by computing the heat flux between two points at a certain distance from each other (according to the grid) analytically for a certain sink strength, assuming that it stays constant over time, and imposing this slope between the first two points in the numerical grid. The outer boundary condition is set to be zero heat flux, which is acceptable for small times. To be exact on that side we would need to compute the heat flux as a function of time at the outer edge, and impose that heat flux in the model between the two outermost points.

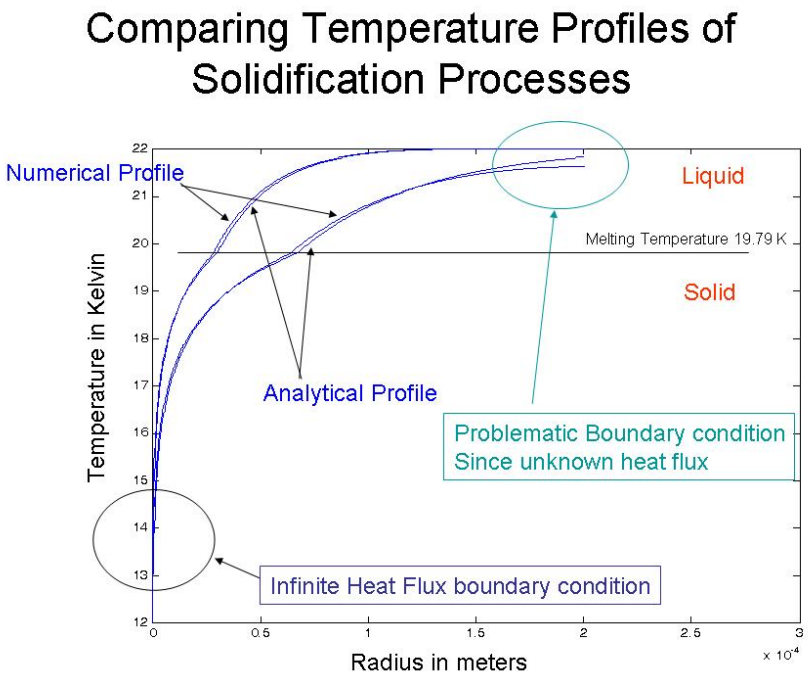


Figure 5.1: This figure shows the profiles for two selected times in the solidification process. Note the discrepancy due to the boundary conditions which could not be imposed in a straight forward manner.

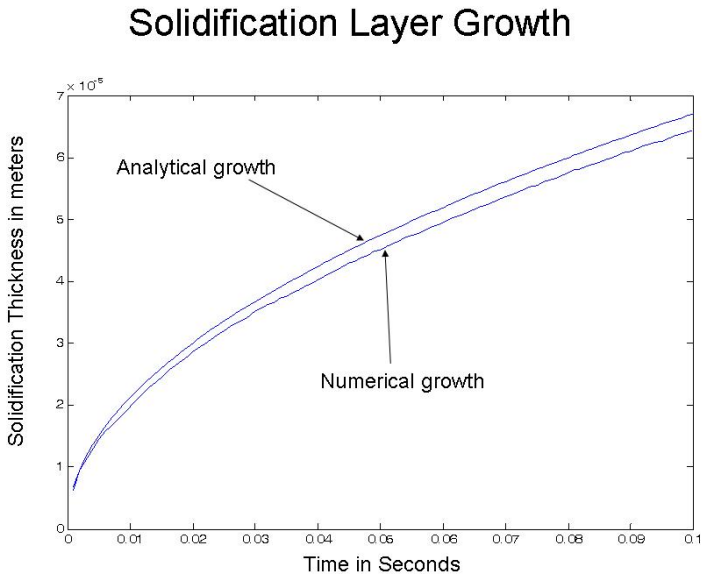


Figure 5.2: This figure shows the time dependent thickness of the solid layer in the solidification process. Both solutions lie within a 5% error margin of each other, which is acceptable for our application.

5.2 Liquid to Vapor Phase Change – Bubble Nucleation and Growth

According to Van Stralen [18], bubble growth can be divided into three different stages. In the first stage (initial mode $t \to 0$), the bubble growth is restricted by hydrodynamic inertia effects. In this stage the bubble growth is linear with time, according to the Rayleigh solution. As the bubble grows, it has to accelerate the surrounding liquid as it pushes it radially outward.

In the second stage (asymptotic mode $t \to \infty$), the description of the bubble growth can be simplified to a heat diffusion model. The bubble growth is proportional to the square root of time. The bubble grows as fast as the heat required to grow the bubble can be delivered to the interface, where it is used to evaporate the liquid.

The third stage is a transient stage, which lies in between the previously mentioned modes. Both of these restricting factors can impact bubble growth.

For the time scale and bubble sizes we are interested in, we can safely assume that the bubble growth is heat flux restricted. This means that the bubble will grow as fast as the heat necessary to grow the bubble can diffuse to the interface. As discussed above, we can easily model this scenario, since we are already simulating heat diffusion in our model; as a result, we can use the information from the code and apply it to bubble growth.

5.3 Equations

Bubble nucleation and growth occur whenever a sufficiently large liquid superheat and a sufficiently large nucleus in the liquid phase [17], [18] are present. Let us start by examining the behavior of vapor bubbles during nucleate boiling: the growth of a free, spherical vapor bubble in an initially in a uniformly superheated liquid of homogeneous composition.

Following the derivation from the 'Bosnjakovic Theory for Isobaric Heat Diffusion Controlled Growth' [18], the bubble behavior can be described by equation (5.5) and (5.6).

$$r_b(t) \approx 2 \cdot Ja \left(\frac{\alpha_l t}{\pi} \right)^{0.5} \qquad (5.5)$$

$$Ja = \frac{\rho_{liquid}\ c_{p,liquid}}{\rho_{vapor}\ i_{fg}} \Delta T_{SAT} \qquad (5.6)$$

Using the definition for the thermal diffusivity, we expand the equation to (5.7):

$$\left. \alpha)_{liquid} = \frac{k}{\rho\, c_p} \right)_{liquid}$$
$$r_b(t) \approx \frac{k_{liquid} \cdot \Delta T_{SAT}}{\rho_{gas} \cdot i_{fg} \sqrt{\alpha_{liquid} \pi}} \cdot 2 \cdot (t)^{0.5} \qquad (5.7)$$

The values used for this calculation are extracted from Souers [16]. They read:

$$\alpha)_{liquid} = 7.8\text{E-}8 \frac{m^2}{s} \qquad \Delta T_{SAT} = 2.00\ \text{K} \qquad \mathrm{k}_{liquid} = 0.1 \frac{W}{m\,K}$$

$$i_{fg} = 1360 \frac{J}{mol} \qquad \rho_{gas} = 130\ \frac{mol}{m^3}$$

The following simplifications have to be implemented in the code:

- The liquid temperature in the whole domain is initially 22 Kelvin (ΔT_{SAT} = 2.00K).
- The temperature in the bubble is 20.00 K at all times. In applying this boundary condition, we neglect the increase in temperature due to the increase in pressure by surface tension effects (eqn. (4.6)). The effect of the pressure and temperature increase can be implemented in the numerical code for experimental simulations, but will be omitted here to model the analytical solution. Also, the real value of the gas temperature at the surrounding pressure (triple point temperature 19.79 K) cannot be used, since, for modeling reason we assume that the solid – liquid phase change occurs over a discrete temperature range between 19.79 and 19.99 K [1]. In order to always account for the latent heat required for the solid to liquid phase change, we have to use the lowest possible value of the liquid temperature, which is 19.99K.
- The density of the gas is computed by using the ideal gas law and is approximated to 133 mol/m^3. In the actual case, as the pressure and the temperature inside the bubble change, the gas density will follow these changes according to the ideal gas law.

5.4 Results for Cylindrical Model (per unit height)

Under the previously stated assumptions, the growth rate is calculated and plotted in Fig 5.3. The figure 5.3 also shows the growth predicted by the numerical model, using the same assumptions. Considering the approximations in the bubble model and in the heat flux causing bubble growth, the modeling results compare reasonably well to the analytical results. The model can then be applied to simulate the LANL experimental results with some degree of confidence.

The model shows good congruency of the plots when different grid stretching coefficients are used. By changing the grid stretching coefficients, the bubble sizes are changed. The more the grid is stretched, the smaller is the difference in bubble size from one size to another.

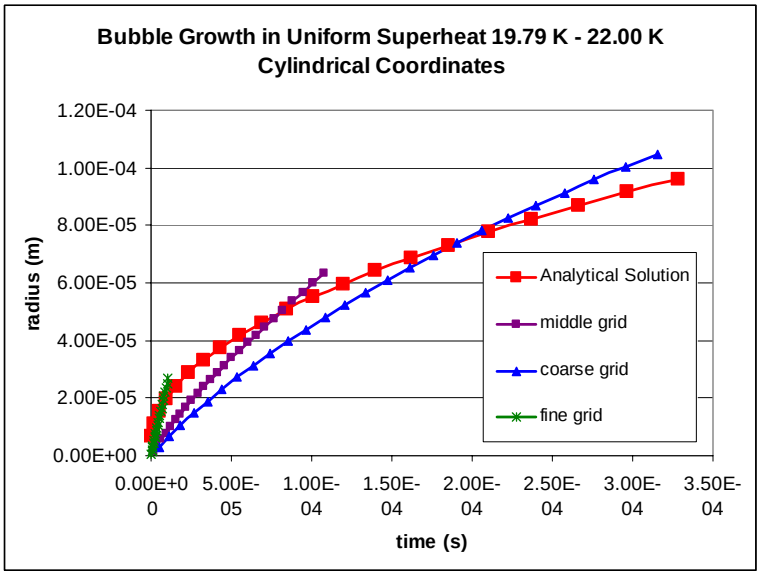


Figure 5.3: The growth of the bubble is plotted in this figure both using the numerical approximation and an analytical result using the previously described boundary conditions. Clearly, the two plots are similar in shape and value, giving us confidence in modeling assumptions.

Similar accordance could be reached using different superheat values in the domain. The “radius of influence” or the temperature boundary layer is found to be sufficiently simulated by the model if the closest 10 or 20 points are updated after each iteration.

The “slide show” in figure 5.4 shows the development of the bubble over time. Notice that the gradient around the bubble remains very steep indicating a rapid heat transfer to the bubble surface.

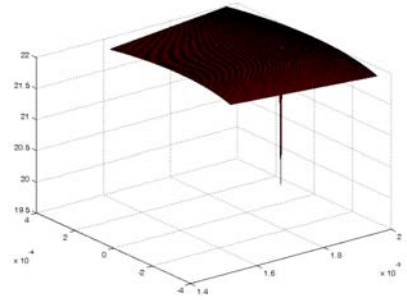
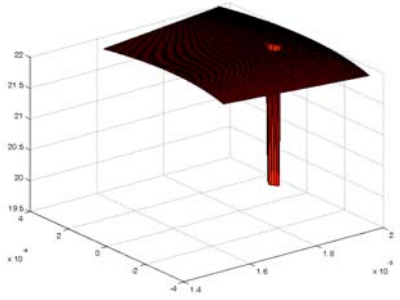
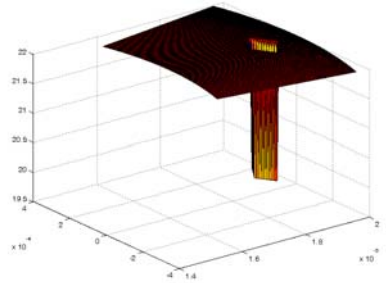
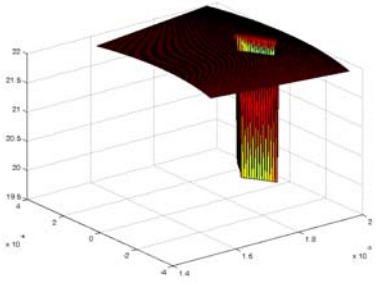
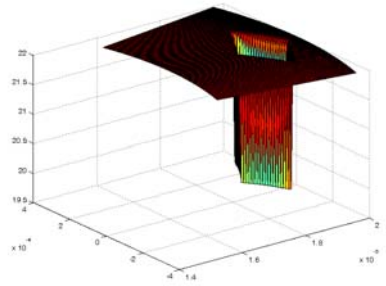
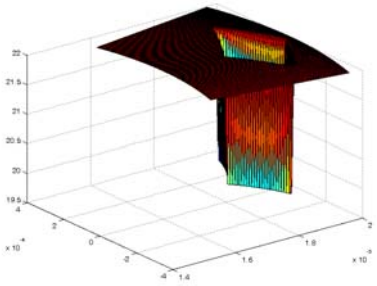
Time: 4.86 E-6s Radius: 2.94 E-6m 	Time: 3.52 E-5s Radius:1.88 E-6m 
Time: 8.53 E-5s Radius:3.97 E-5m 	Time: 1.47E-4s Radius:5.66 E-5m 
Time: 2.23E-4s Radius:8.69 E-5m 	Time: 3.15E-4s Radius:10.5 E-5m 

Figure 5.4: this slide show presents the evolution of the temperature field around the bubble in a pie shaped cylindrical domain. The uniform superheat surrounding the bubble was initially 22 K. We see that the thermal boundary layer around the bubble remains small leading to steep gradients and a high heat flux into the bubble.

Input values used in the code (HEADER) (for future references):

```
Cs= 1.0 (coarse) 2.0 (middle) 3.5 (fine)! stretching factor in radial direction
Cst= 1.3 (coarse) 2.8 (middle) 4.0 (fine)! stretching factor in theta direction
g%nr= 100            ! Grid calculated to have (close to ) quadratic
g%nt= 75             ! patches where we want to model the bubble
n1= 100;             ! # of rb iterations at various points
Rad= 0.002;          ! outer radius of the DT shell
Rad_inner= 0.0016;
pi= 3.1415;
total_angle= pi/8

bubb_loc= 35         ! # of nudes the bubble nucleates inside the
                     ! outer rim of the cylinder
bubble_steps= 25     ! number of bubble steps
surf_tens= 3.9E-3    ! surface tension of liquid DT (temp. dep)
press_liquid= 22000  ! pressure in liquid (will change later, but
                     ! good for now)
lat_heat= 1350       ! latent heat of vaporization (temp. dep)
gas_const= 8.314     ! gas constant
```

5.5 Results for Spherical Model

As discussed in the previous section, the spherical model can simulate a 3-d bubble using the symmetry around one axis. As a result, the only approximation we have in our code is the non-spherical shape of the bubble and the simplifying assumption of utilizing an effective heat transfer into the bubble based on a combination of the initial and final temperature profiles at each time step. The non-spherical shape should not have a great importance as the error induced by it is expected to be small.

The spherical model also shows a good simulation of the analytical bubble growth, even if different grid sizes are chosen. Like in every numerical simulation, the finer the chosen grid is, the closer the solution follows the analytical result.

Figure 5.5 shows the bubble growth predicted by our model for the same case as figure 5.4. The 22 K uniform superheat and the same geometric properties are used; we have simply changed the heat diffusion equation and the heat flux equation from cylindrical to spherical coordinates.

As a grid is chosen which models very small bubbles, the numerical solution follows the analytical plot closely (green curve). But if we are interested in bigger bubbles and longer times, a coarser grid can be chosen to speed up the code. In that case, a large error is observed initially, but for bigger times, the bubble radius predicted by the model agrees with the bubble sizes an analytical calculation would give.

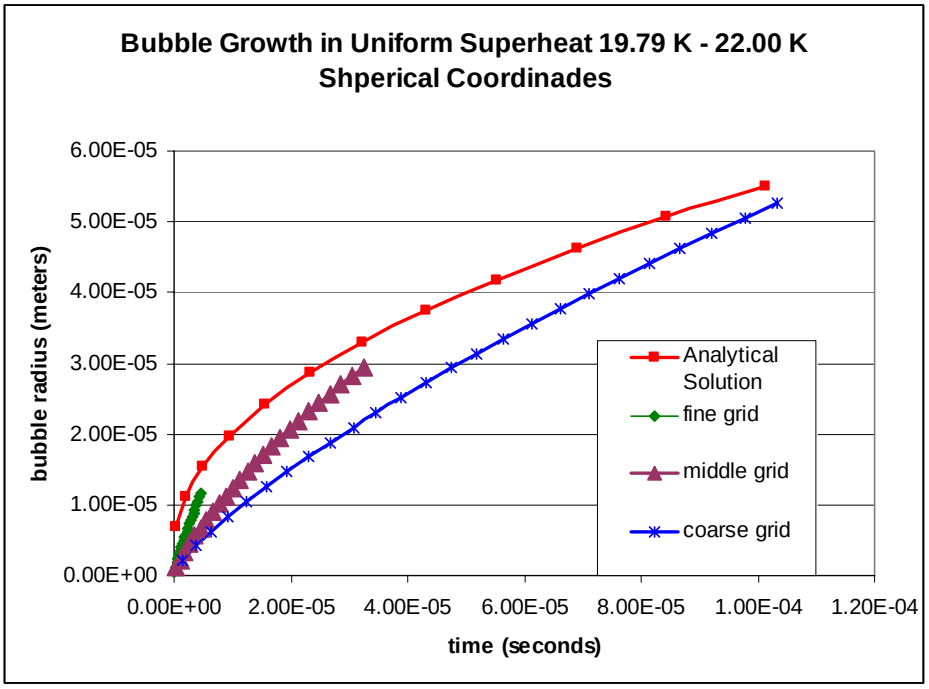


Figure 5.5: Good accordance between the analytical uniform superheat equation and the bubble growth predicted by the model is shown here. Where exactly the divergence comes from and how it can be erased could be further analyzed, but for our desired accuracy this model seems reasonable.

6. Comparing the Results from the Computer Model to the Experimental Results from LANL

After having successfully tested the numerical simulation against analytical solutions for cases with simplified boundary conditions, we used the model to simulate the behavior of the LANL cylindrical targets during heat flux experiments. This modeling will help to explain the physics behind the LANL experimental observation, and to gain insight on the thermo-mechanical behavior of the spherical IFE targets.

6.1 The Solid – Liquid Phase Change

Please see Section 4.2 for the schematics and description of the LANL experimental setup. The experimental values were given to us directly from LANL [24].

As a first step we focused on the solid- liquid phase change. The measured thickness of the melt layer in the experiment and the corresponding results from the code are plotted in figure 6.1. The experimental results are based on post-test examination of pictures of the DT whose range of uncertainty tends to be higher for thinner melt layers.

Clearly there is a large discrepancy between the calculated (numerical) and the observed (experimental results).

The applied boundary conditions are:

Initial temperature - 18K

Heat flux ($\dot{q}''$) - 1W/cm^2

Outer Radius - 2mm

Inner Radius - 1.533 mm

From figure 6.1, it can be seen, that the modeling results show a slope similar to that of the experimental results, but shifted down by about 50 μm on the local axis.

Appendix H shows a computation of the heat required to grow a melt layer according to the LANL observations and compares the result to the total heat induced into the target by the heat flux. From this simple energy calculations based on the latent heat of DT, it seems clear that the melt layer cannot reach the reported thickness within the reported time when a 1 W/ cm^2 heat flux is imposed.

In addition, the model indicates that a time of about 13 ms is required for the DT to reach its triple point (this is a pretty reliable result from the previously verified thermal conduction part of the code). This time, experimental results indicate a jump in the melt layer thickness to about 50 μm within an additional 11 ms, which is difficult to believe.

In search for possible explanations on the apparent discrepancy in the experimental results, the following possible factors resulted from direct discussion with LANL [24]:

- The melt depth measurements carry an uncertainty due to difficulties in determining exactly where the melt layer interface is actually located. The convex DT ice surface, through which the pictures leading to these results were taken distorts the picture and could lead to a misconception about the exact location of the solid to liquid interface.

- Due to the complexity of the heating apparatus, the exact heat flux imposed on the target during the experiment could not be exactly determined. While the electrical power to the resistors can be determined quite accurately, heat losses at the back and ends of the experimental set-up and the possibility of additional heating due to light sources can result in significant uncertainties. These could cause an underestimate of the heat flux by up to a factor of 2 (as an upper limit but probably closer to a factor in the range of 1 – 1.5).
- Non- uniformity of the melt layer could also be a possible reason affecting the experimental results, but this was thought to be rather unlikely based on LANL observations.

Next, we tried to find a heat flux that would give a similar melt layer thickness as the LANL experiments showed. The results are plotted in figure 6.2. The results indicate that the heat flux should be increased by a factor of about 3 in order for the modeling results to reproduce the experimental observations; this is well over the maximum uncertainty factor of 2 on the heat flux and indicates that another factor has to be in play, which is likely to be the uncertainty in the melt layer measurements.

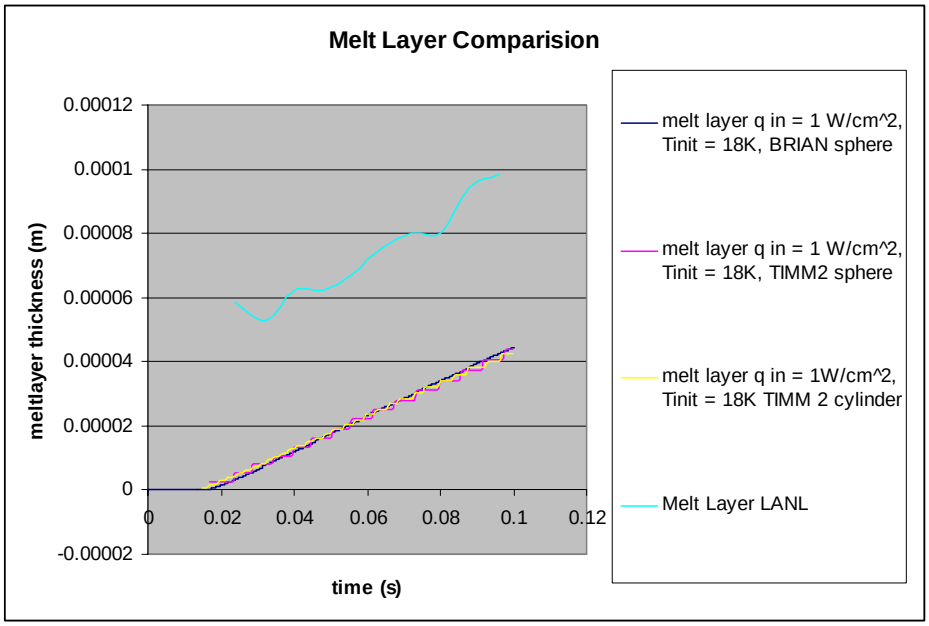


Figure 6.1: the melt layer thickness calculated by different models available (previous and new spherical model (1-d and 2-d) as well as cylindrical case) are plotted here. Notice that there is only a small difference between the cylindrical and the spherical coordinate geometry. There is a clear difference between the numerical simulations and the LANL observations.

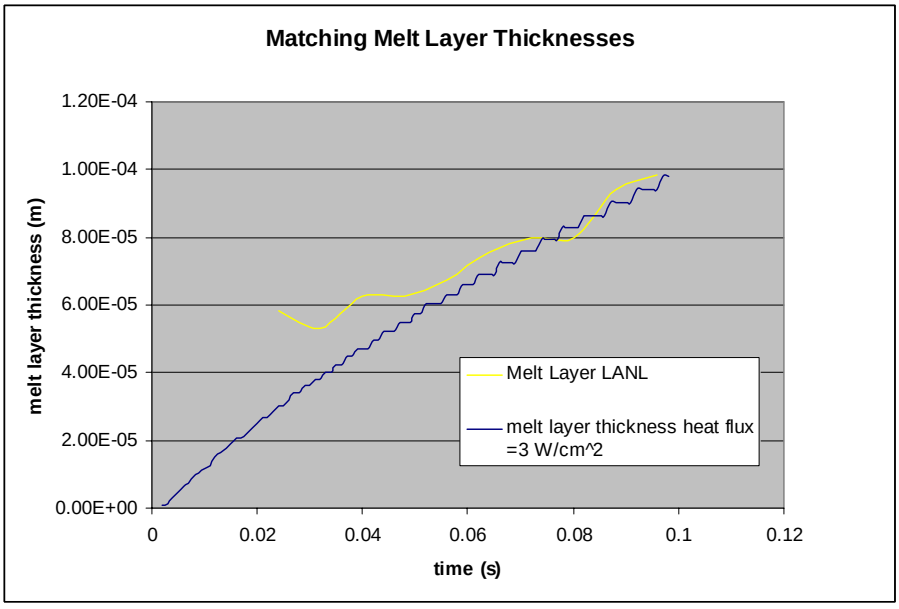


Figure 6.2: The heat flux is increased to get closer to the experimental results. If we apply a heat flux three times as big as previously assumed, the results become close. If that heat flux was imposed though, we should observe a melt layer over times shorter than 20 ms which have not been reported so far.

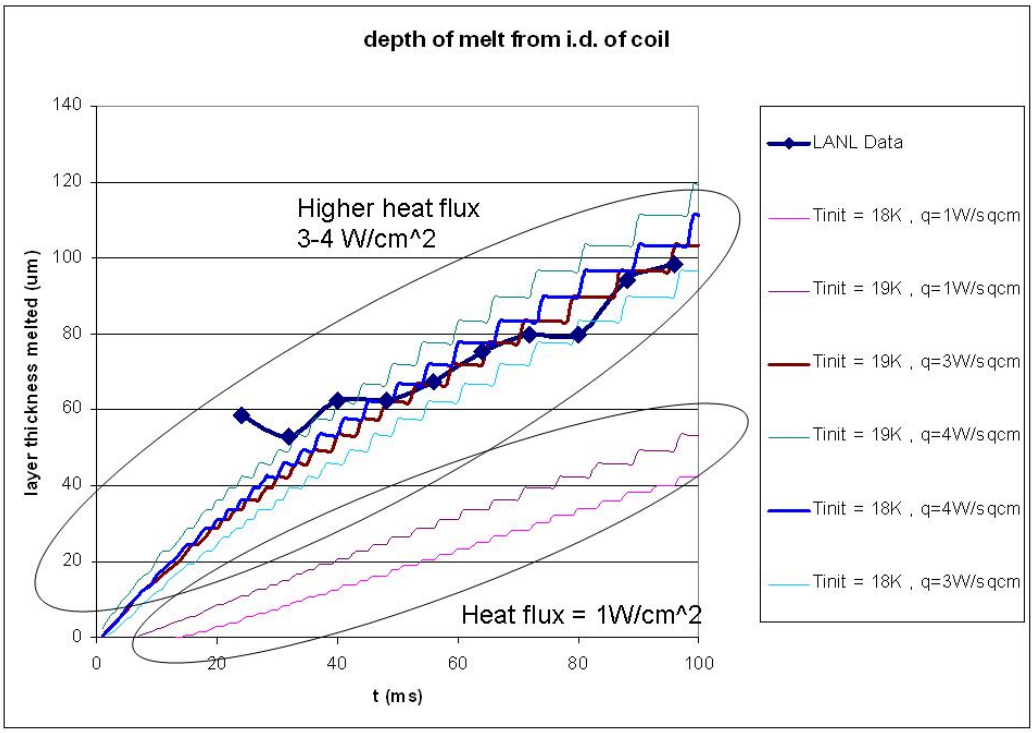


Figure 6.3 superimposes the different scenarios. Different initial temperatures and heat fluxes have been applied. The solution look the closest for 18 K and 4 W/ cm^2 or 19 K and 3 W/ cm^2 for initial temperature and heat flux respectively.

Further comparison between different sets of experimental data would be required to better understand where exactly the discrepancies come from.

6.2 The Bubble Nucleation Simulation

For the initial bubble nucleation analysis, we assumed a heat flux of 1.00 W/ cm^2, consistent with the value of 0.92 W/ cm^2 reported by LANL (i.e. with no correction for possible uncertainties in the heat flux).

The main observation to be recorded when comparing the bubble growth from the LANL heating experiments to the uniform superheated solution is that the bubbles grow much slower in the experimental setup. As we show in this chapter we observe two different stages of bubble growth during the heating experiment. In the first stage explosion-like bubble growth is observed. The temperature in the bubble is much lower than the liquid temperature all around it providing a lot of heat for vaporization and bubble growth. During this stage the bubble growth rate is close to the uniform superheated case.

In the second stage, the temperature in the liquid on the side of the bubble opposing the wall is lower than the temperature in the bubble. In this stage some of the heat flows “through” the bubble into the liquid behind it leaving less energy to grow the bubble.

Figure 6.4 illustrates the temperature profile around the bubble at the two different stages.

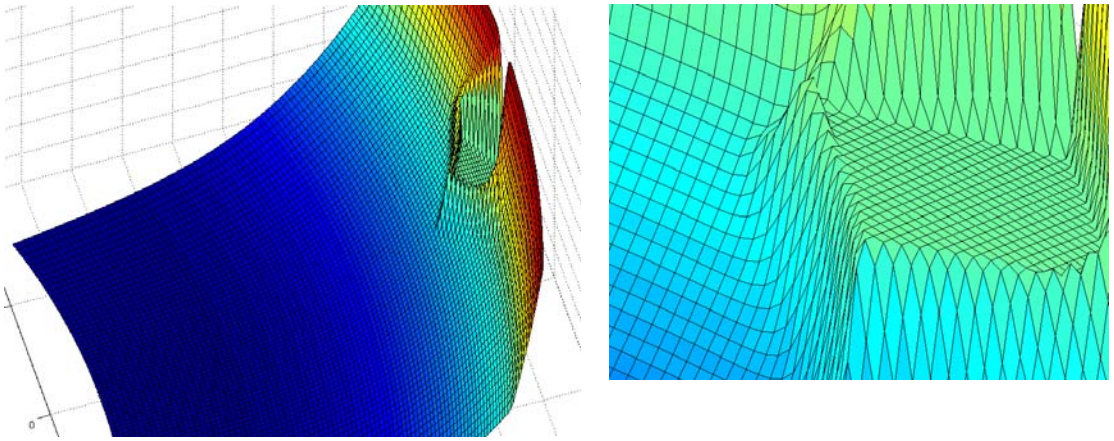


Figure 6.4: On the left side the bubble is shown in the fast stage of bubble growth. The liquid all around the bubble is at a higher temperature than the bubble, causing it to be a strong heat sink. On the right side, the picture zoomed closer to the bubble, showing that the heat flux into the bubble (the back -right side of this picture) can be differentiated from the heat flux out of the bubble (front-left of the picture), as the gradient is reversed for the two cases.

When simulating the bubble growth using the model as described and tested in the uniform superheated case and comparing that data with the observations from the LANL experiments, certain assumptions about the conditions have to be made. These include the pressure in the liquid during the heating experiment, the nucleus size present in the liquid before the heat pulse is started, strength of the heat pulse and the initial temperature. While the heat flux and the initial temperature can be taken from the LANL report, educated guesses have to be made about the nucleus size and the liquid pressure.

Based on the results from the simple ^{3}He diffusion model (chapter 3), two different nucleus sizes have been tested: 1.6 um and 0.4 um for the 18 hours and 4 hours layering time respectively. The liquid pressure was assumed to be 22 000 Pa, slightly higher than the saturation pressure at triple point.

The LANL measurements give two sets of bubble diameters: one is the bubble size measured by looking through the liquid DT, and the other one measured by looking through the solid layer. At this point it is important to know that these measurements are quite rough and can only be used to give us an approximate idea of the bubble sizes. Figure 6.5 shows one of the images used to measure the bubble diameter, from which the difficulty of making an exact measurement can be appreciated [24].

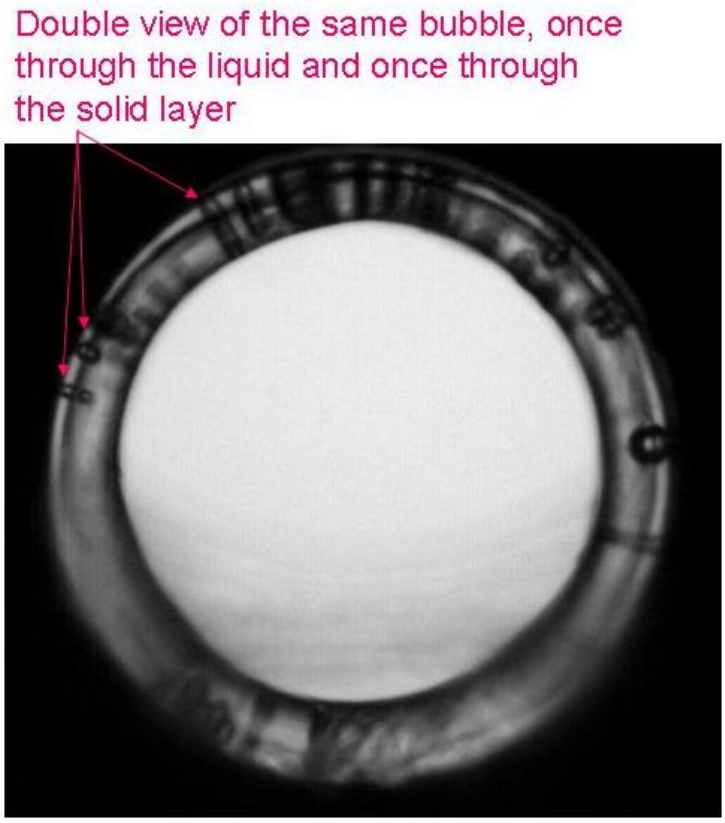


Figure 6.5: This picture was taken during the LANL heating experiment. Frames like this are used to evaluate the bubble diameter at each time step. It can be seen that there are different size bubbles throughout the domain, one of which being selected and measured as the bubble grows. Also very remarkable is the double view of the bubbles, once through the solid and once through the liquid (as indicated).

Figure 6.6 shows a plot of the bubble diameter as a function of time for the two different nucleus sizes, and Figure 6.7 shows the temperature distribution around the bubble at different stages (as indicated in Figure 6.6).

The results shown in Figure 6.6 suggest that the smaller nucleus size requires a higher superheat temperature to grow into a bubble, causing it to start growing at a later point. Meanwhile, since it is surrounded by liquid at a higher temperature, it grows very fast until the conditions for slower growth are established.

The results from the model for the assumed case seem to simulate reasonable well the experimental results. However, it should be noted that there are uncertainties in the bubble measurements (perhaps of the same order as for the melt layer measurements), which can shift the results. Moreover, even in this case, the slope of the modeling results for bubble growth is comparable to that of the experimental results.

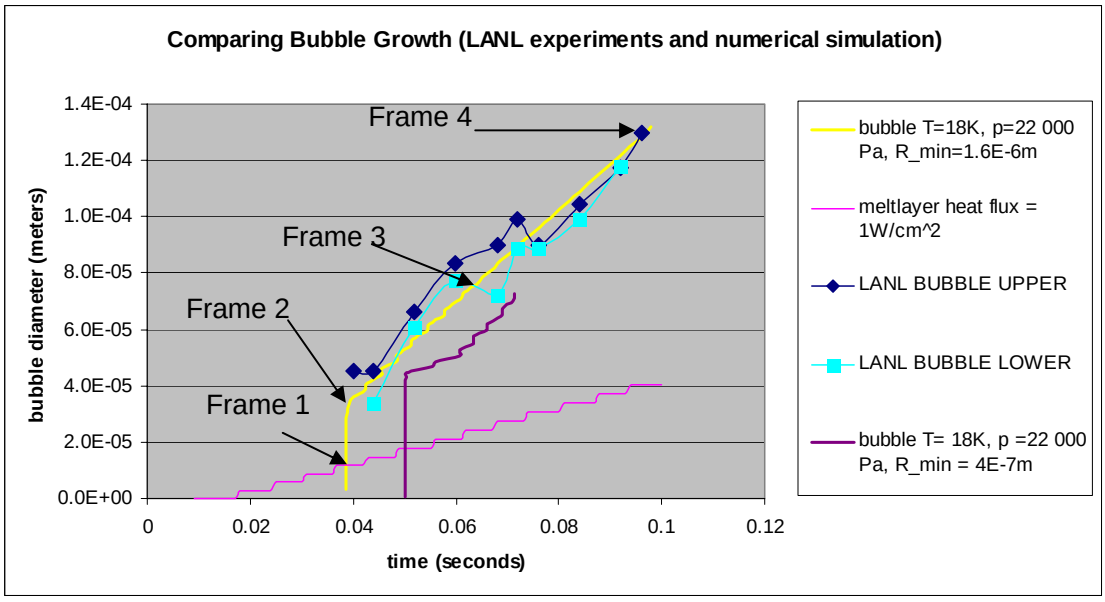


Figure 6.6: The two different stages of bubble growth can be seen: very fast initial bubble growth (the lines appear to be vertical), and slower growth as the temperature of the bubble opposing the wall is higher than the temperature of the liquid that is faces.

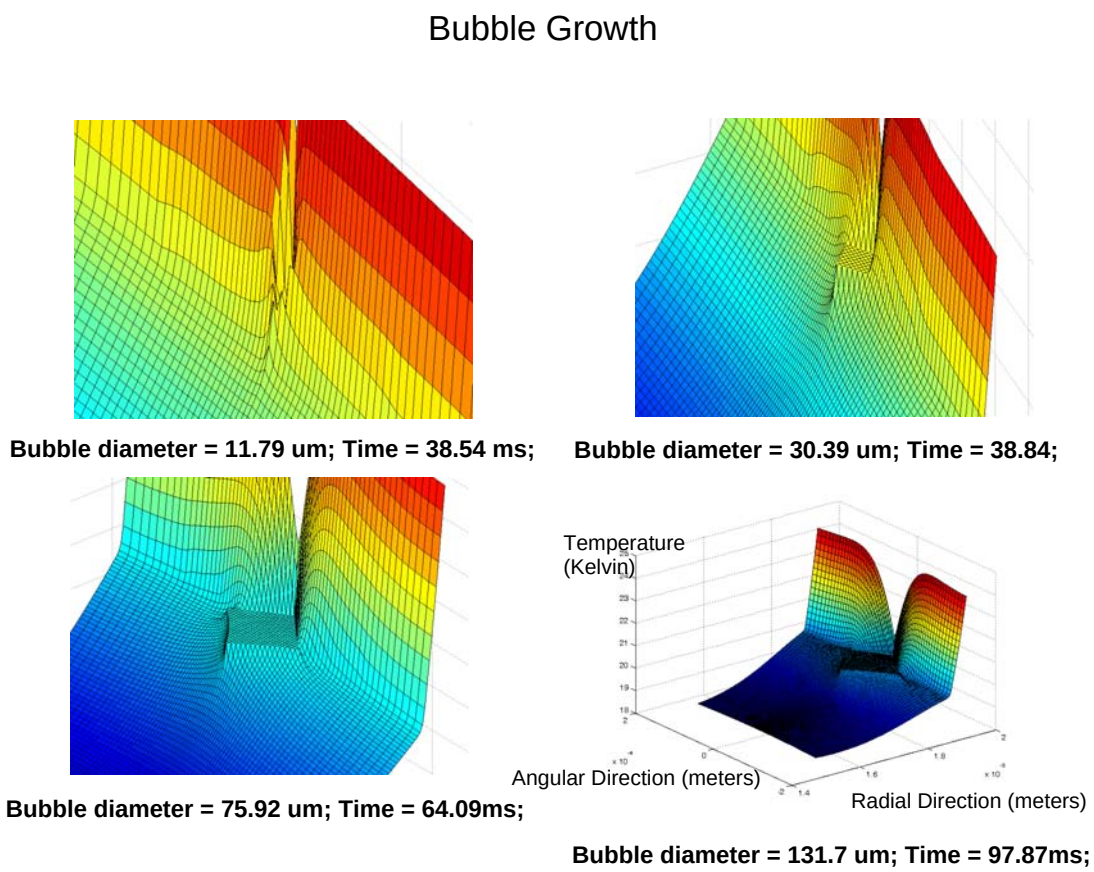


Figure 6.7: the temperature profiles during bubble growth are shown at the indicated times and respective bubble diameters. Notice the correlation between the temperature field in the first two pictures and the corresponding fast growth rates as opposed to the temperature fields in the last two slides matching the conditions for slow growth.

These trends indicate that the assumptions used in the model are reasonable. Notice that, according to the code, the bubble grows deeply into the solid layer. Analyses of solid DT around the TP indicate its low strength, which could infer that the bubbles could grow by pushing the liquid in the melt layer as well as the soft solid (a trend also indicated by LANL's initial observations but requiring more accurate experimental confirmation).

In order to better understand the influence of the nucleus radius on the onset of bubble growth, we plotted the superheat required for bubble growth as a function of the nucleus radius. Figure 6.8 shows the results.

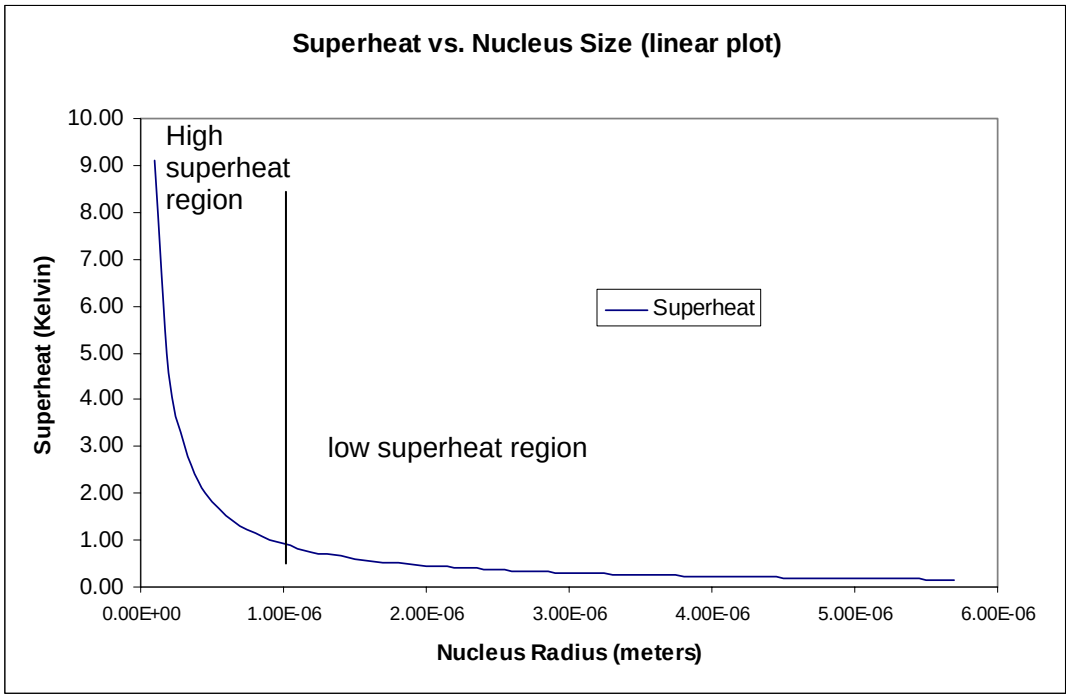


Figure 6.8: the superheat required for different size nuclei to grow into bubbles is plotted. Clearly, once the nucleus is greater than 1.5 microns, the superheat required is less than one degree, with only slight changes for larger nuclei.

From this figure we can see, that a radius bigger than 1.6 micron is only going to have little effect on the superheat temperature (and consequently the onset time of bubble growth assuming a certain heat flux). If the bubbles were smaller than 0.4 microns, very high superheat temperatures will be required leading to even later onset times for bubble growth. We expect the bubble growth in such cases to be extremely fast until bubble sizes similar to the sizes from earlier onsets will be reached.

6.3 Varying different Input Parameters

An interesting study would be to investigate the influence of the surrounding liquid pressure on bubble growth. An increase of the saturated vapor temperature in the bubble would follow an increase of the liquid pressure, causing the bubbles to nucleate later in time. Since at pressures higher than triple point pressures it is possible to have a liquid cooler than the bubble temperature on the side opposing the incoming heat flux, we expect the bubble to remain smaller. Figure 6.9 shows the influence of increased pressure on bubble nucleation and growth.

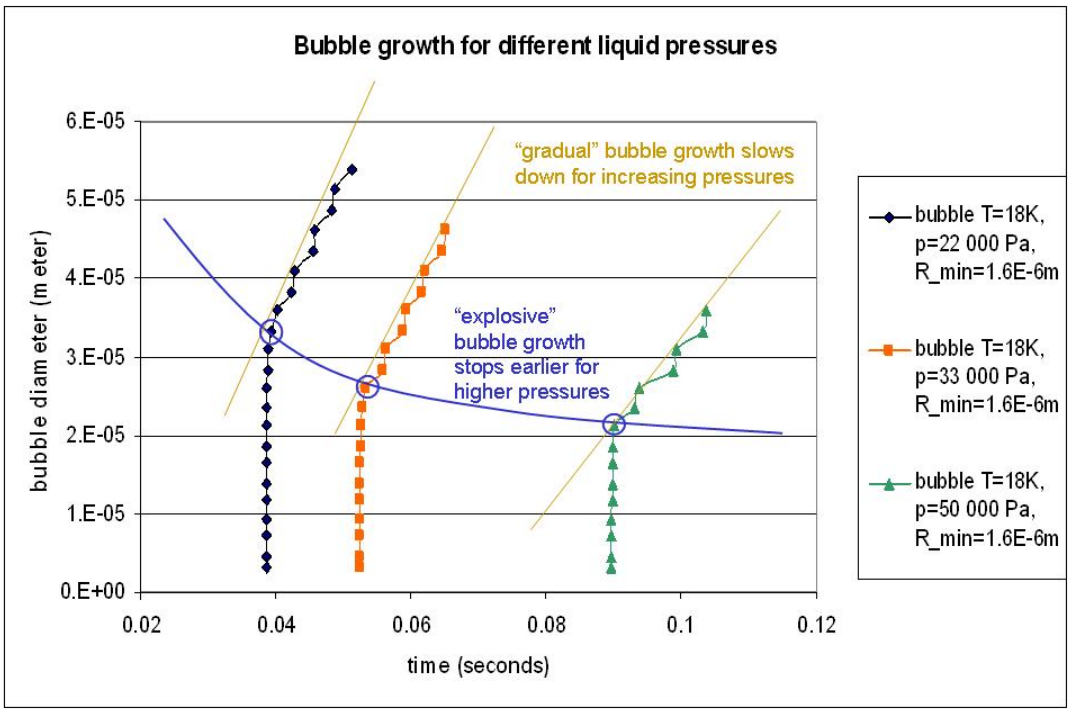


Figure 6.9: Higher liquid pressures influence both the onset of bubble growth as well as the size at which the bubble growth slows down and the speed at which the gradual bubble growth happens.

Because of the importance in the target case, the temperature fields for different pressures are shown in Figure 6.10. Notice the higher temperature in the bubble allowing for a heat flux leaving the bubble without the need of penetrating the solid layer.

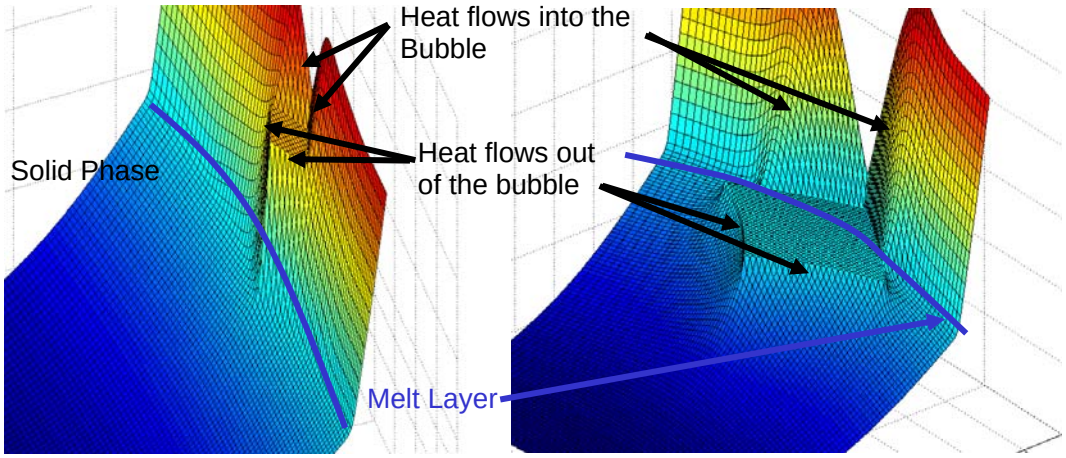


Figure 6.10: The temperature fields of the three phases at different pressure scenarios are shown here. The bubble diameter at which bubble growth stagnates is smaller than the melt layer in the left case (high pressure), and larger than the melt layer on the right side (low pressure).

As a final step in the interpretation of the data sets resulting from the LANL experiments, we need to analyze the relation between the initial temperature and the onset of bubble growth. Figure 6.11 shows the bubble growth for the 1.6 um bubbles applying various initial temperatures.

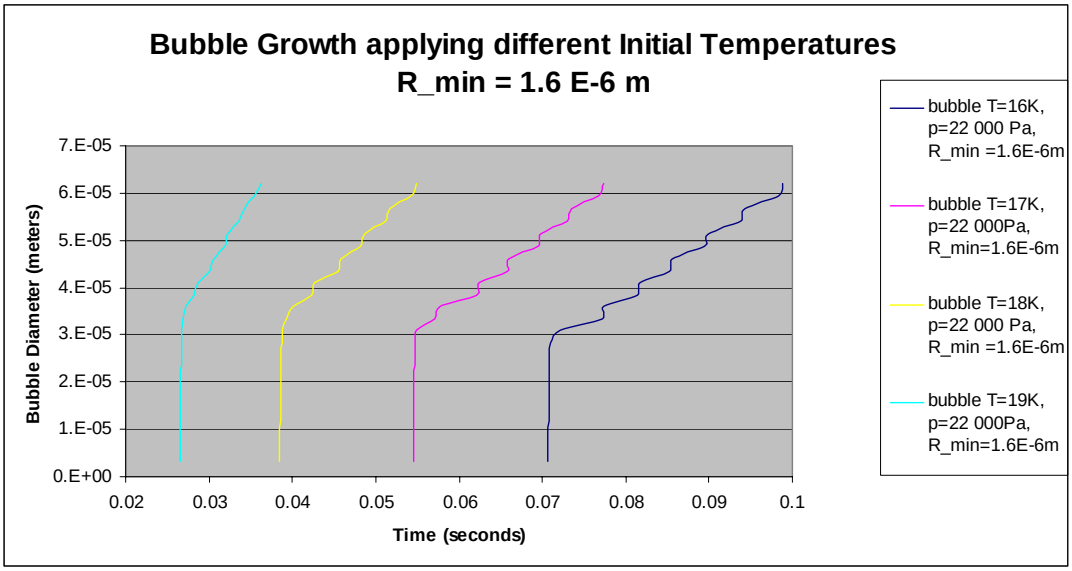


Figure 6.11: As the initial temperature is lowered, the onset of bubble growth is delayed. The speed of gradual bubble growth is also found to decrease with decreasing temperatures (from the modeling results).

6.4 Summary

The following observations can be made from the analysis of the LANL experiments.

As described above (section 6.1) there is a large discrepancy between the results of the model and the experimental results for the melt layer thickness. The main reasons behind these discrepancies are believed to origin from difficulties in measuring and/or determining the exact melt layer location. The drastic increase in heat flux required to match the reported melt layer numerically is believed to be unlikely.

For the bubble growth, the model delivers good explanations for some of the experimental observations. The series of pictures shot during the heating experiments show sudden occurrence of bubble of a significant size at a certain time after the start

of the heat pulse. The temporal resolution of the picture series is too low (4ms between two shots) to resolve the behavior of the bubble before it reaches rather large sizes (40 microns). From the model we know, that the bubble nucleation and initial growth happens very fast (within a few microseconds). Because the bubbles nucleate in an environment characterized by a steep temperature gradient across the domain, we know that stagnation in bubble growth occurs when the temperature gradient on the side opposing the incoming heat flux is negative, causing most of the incoming heat to flow through the bubble rather than being entirely used to grow the bubble.

As a result, two different modes of bubble growth can be distinguished, fast initial nucleation and slower gradual growth. The LANL data can only show the gradual growth, but the size of the suddenly occurring bubbles suggests fast initial nucleation and growth.

At pressures slightly higher than triple point pressure, the model predicts that the bubble grows deeply into the solid layer. Whether this is physically possible or not, needs to be determined by further experiments. Comparing the melt layer thickness data from LANL with the bubble sizes and the fact that solid DT around triple point temperature is reported to have a very low strength, bubble growth into the solid layer can be explained.

If the heat flux could be determined with certainty, the model could be used to estimate the size of the nucleus. Knowing the onset time of bubble growth, the superheat of the liquid can be quantified; from the superheat, conclusion can be drawn for the minimum nucleus radius present in the liquid. The case studied for this thesis,

suggests a nucleus radius of around 1.6μm (assuming the heat flux to be close to 1 W/ cm^2). This result is in good agreement with the estimates from chapter 3.

Additional parametric studies result in the following relations between input parameters and bubble growth:

A larger nucleus radius results in an earlier onset of bubble growth; this statement can be explained by the higher pressure and saturation temperature in a bubble of a smaller radius due to the surface tension (equation 4.6: p ~ p(liquid)+1/r).

The higher the liquid pressure, the more delayed the bubble nucleation. This is to be expected, since a higher liquid pressure will lead to a higher saturation temperature of the vapor in the bubble. In order for the bubble to grow, a higher liquid temperature will be required. Also, as a higher liquid pressure is applied, resulting in a higher temperature in the bubble, the bubble will not grow as deeply into the solid layer.

The influence of the lower initial temperatures is less surprising as has been discussed previously [1].

The next step following the outline of this work is an attempt to predict the thermal behavior within the spherical target as it is injected into the chamber. From the LANL studies and the conclusions drawn after numerically modeling the experiments, we know that the pressure in the liquid phase will have a significant influence in the bubble growth indicating that we need to focus not only on the thermal, but also on the mechanical behavior of the target.

Table 6.1: Numerical Input parameters for the different cases:

Nr = 100	Nt = 75	R inner = 1.533mm	R outer = 2mm	Theta = pi/16
Stretching factors	Radial = 0.9 Radial = 2.0	Angular = 2.0 Angular = 4.1	Large bubble Small bubble	1.6E-6 0.4E-6

7. Application of the Bubble Growth Simulation to the Spherical Target Geometry

Following model validation by comparison with analytical results for controlled example cases and the interesting results from simulation of LANL experimental results, the model was applied to simulate the spherical target behavior in an IFE chamber during injection. The previous model [1] has taken the thermal influence of the plastic shell as well as different properties for the DT – foam region into account. Here, for simplicity, these corrections are neglected, since their influence is small when simulating phase change and bubble growth. We will assume direct heating of the DT due to radiation and convective heat transfer as discussed in chapter 2, and will consider the confining and compressing effect of the plastic shell on the expanding DT melt layer.

7.1 Increase in Pressure due to Melt Layer Growth

When projecting the main conclusions from chapter 6 onto the target case, it becomes apparent that the influence of volume expansion during the solid to liquid phase change inside the plastic shell on the pressure buildup inside the target needs to be studied. As the volume of the DT expands during such a phase change, it is confined by the outer plastic shell and the inner DT solid sphere, leading to an increase in pressure inside the target. This increase needs to be quantified and its effects included in the numerical simulation. In order to do so, an additional loop needs to be included in the code to compute the pressure buildup due to the growth of

the melt layer. The increasing pressure in the liquid phase of the DT results in an increase of the saturation temperature inside a nucleus (which we assume to be present due to the tritium decay as discussed in chapter 3). We can plot the melt layer thickness against the pressure in the target neglecting the small volume changes due to changes in the bubble size. This approximation is reasonable considering how small the change in bubble volume of one bubble is as compared to the volume of the overall domain and the volume change due to the melt layer growth. Of course, as more bubbles grow simultaneously, this effect needs to be considered. However, since the goal of the simulation is to find the parameter space that would avoid bubble growth, this event can be considered outside the bound of this analysis.

In order to compute the pressure buildup, the following reasoning, equations, and material properties are used (explanation of the symbols can be found in the nomenclature section) following [25]:

$$\begin{aligned} V_m &= \frac{4\pi}{3}\left(r_{out}{}^3 - \left(r_{out} - d_{melt}\right)^3\right) \\ moles_{melt} &= v_{DT\,liquid} \cdot V_m \\ \Delta V &= \Delta V_{molar,\,phase\,change} \cdot moles_{melt} \end{aligned} \tag{7.1}$$

Pressure buildup follows equation (7.2):

$$p_f = p_{no\,melt} + \frac{\Delta V}{V} \cdot \frac{4tE}{3d\left(1 - \frac{1}{m}\right)} \tag{7.2}$$

In our case, as the pressure buildup deflects both the DT shell and the plastic shell, we need to apply equation (7.2) simultaneously for the DT and the plastic shell. Note

that the pressure in the liquid due to the two deflections have to be equal, while the volume changes have to be added.

$$\Delta V_{total} = \Delta V_{DT} + \Delta V_{plastic}$$

$$\frac{\Delta V_{plastic}}{\Delta V_{DT}} = \frac{\Delta V_{plastic}}{\Delta V_{total} - \Delta V_{plastic}} = \frac{t_{DT} E_{DT}}{t_{plastic} E_{plastic}} \cdot \frac{d_{plastic}}{d_{DT}} \cdot \frac{\left(1 - \frac{1}{m_{plastic}}\right)}{\left(1 - \frac{1}{m_{DT}}\right)} \qquad (7.3)$$

The right hand side of equation (7.3) is a constant, depending only on the material properties of DT and plastic, which can be found in Souers [16] for DT and material properties handbook for the plastic [26]. In the following equations, the right hand side of equation (7.3) is referred to as κ.

$$\frac{\Delta V_{plastic}}{\Delta V_{total}} = \frac{\kappa}{1+\kappa} \qquad (7.4)$$

Using equation (7.2) again, this leads to a pressure buildup following equation (7.5):

$$p_f = p_{no\ melt} + \frac{\kappa}{1+\kappa} \cdot \frac{4 t_{plastic} E_{palstic}}{3 d_{plastic} \left(1 - \frac{1}{m_{plastic}}\right)} \qquad (7.5)$$

Two different values for the Young's modulus of solid DT were found in the literature [16], [1], differing by an order of magnitude. It was decided to use them in computing upper and lower values for the κ constant. Applying the numbers, we get $\frac{\kappa}{\kappa+1}$ = 0.683998 or 0.95584, using the high and low values of the DT Young's modulus, respectively.

κ is only a ratio of various geometric and material properties of the DT and the plastic shell. Its value would also be affected by parameters such as the thickness of the plastic shell and the choice of materials for the shell. The geometric parameters used in the target in this analysis are based on those used by Christiansen [1] (see table 7.1).

Table 7.1: the geometric parameters and material properties used in the pressure buildup computations are listed here.

	Inner Radius	**Outer Radius**	**Thickness of the shell**	**Young's Modulus**	**Poisson ratio**
DT shell	1.600 mm	2.000 mm	0.400 mm	40 MPa 400 MPa	0.325
Plastic shell	2.000mm	2.002 mm	0.002 mm	3.3 GPa	0.300

The analysis will assume that the pressure in the DT at the moment of injection (no melt layer present) is 22 000 Pa [16]. Following the thermodynamics of the layering process (solidification of the DT inside the plastic shell) [5], [24] a pressure close to the triple point pressure needs to be present in the target. Once the target is layered, there is no mass flux through the plastic shell and out of the target, demanding that the pressure remains at 22 kPa. The growth in melt layer will add to the 22 kPa according to equation (7.5).

Figure 7.1 shows the pressure increase with increasing melt layer thickness for different κ values.

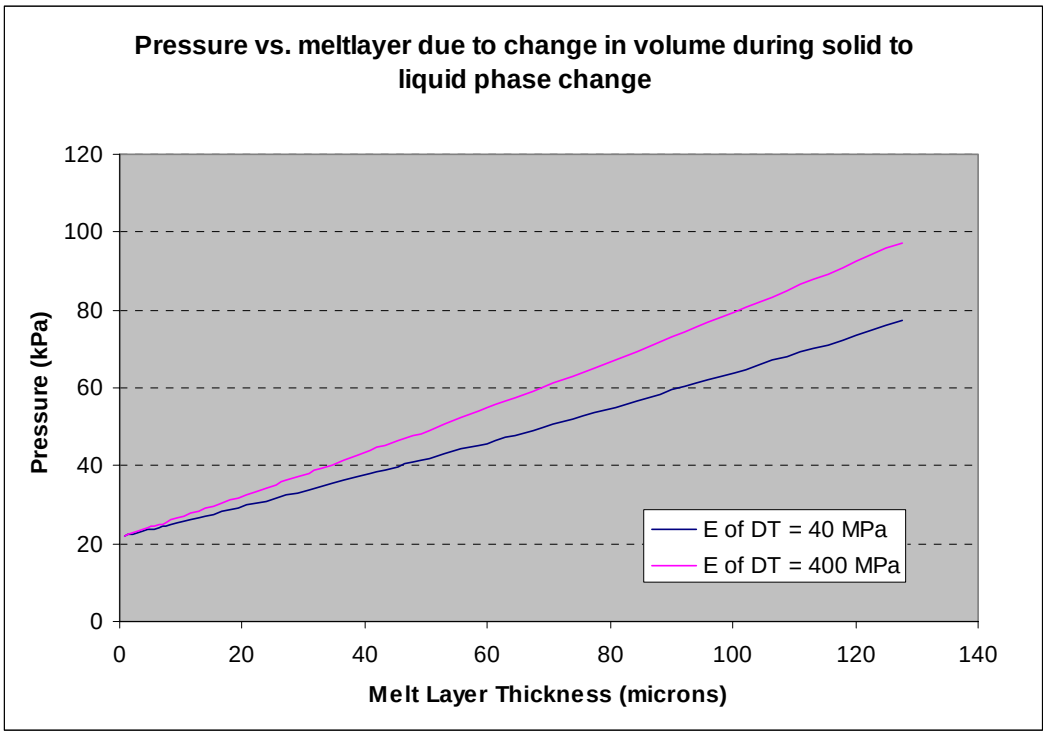


Figure 7.1: Depending on the stiffness (Young's modulus) of the DT shell, the pressure in the target increases according to either of these two lines. Since the properties of the DT are highly temperature-dependent, and a large gradient characterizes the temperature distribution through out the DT shell, a more complex way of computing the real integrated value of the Young's modulus could be applied.

7.2 Results from the Bubble Growth Model

As shown in the cylindrical case, increasing the liquid pressure has a large impact on the bubble growth in a superheated liquid close to the triple point pressure and temperature.

The increase in pressure significantly delays the onset of bubble growth especially for heat fluxes that lead to a thick melt layer. The steady increase in liquid pressure due to the melt layer growth even after the onset of bubble growth also results in an increase of the temperature in the bubble. This can lead to a stagnation of bubble

growth or even collapse of the bubble if the saturation temperature corresponding to the pressure inside the bubble gets higher than the temperature of the surrounding liquid. The present code has been developed to simulate bubble growth and is not capable in its present form to also simulate stagnating or collapsing bubbles. This is because the code can only simulate bubble sizes of certain diameters (based on the mesh size);- it would be very challenging to keep track of the heat flux in and out of the bubble over several time steps in which the bubble doesn't grow from one grid size to another in the code but yet physically changes size between the two values. The possibility of finding time steps long enough for the bubble size to equal the next grid-assigned value can lead to time steps so long that the accuracy of the code becomes doubtful.

It is possible and accurate though, to model the thermal and mechanical behavior inside the target up to the point where bubble nucleation occurs.

Figure 7.2 shows the temperature profile for the outer three points in the domain along with the saturation temperature required to grow a 1.6 μm bubble.

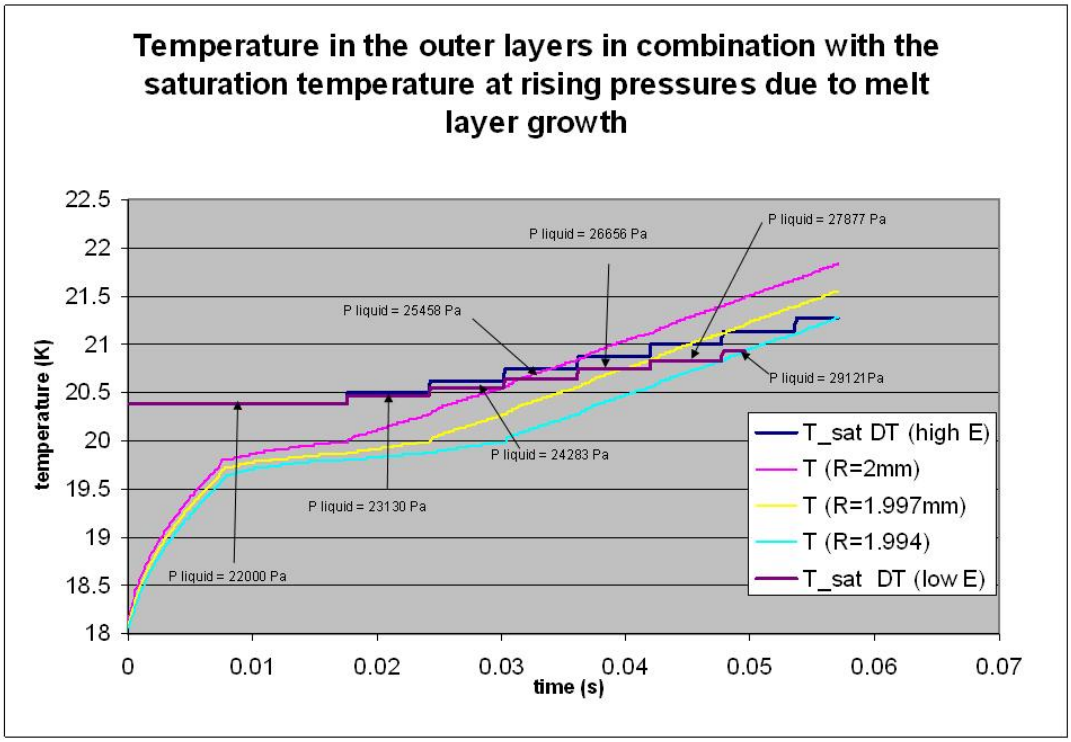


Figure 7.2: In this figure, the temperature histories for the three outer most points of the spherical domain in the radial direction are plotted. The step wise increase of the melt layer (due to the discretized nature of numerical solutions) leads to a step wise increase in pressure in the liquid, and accordingly a step wise increase of superheat temperature required to grow the bubble. As we set the bubble growth criterion to when the second point radially inward from the outer radius (R=1.994mm) reaches a temperature higher than saturation temperature, the bubbles would start growing at the intersection of the light blue line with the step wise growing saturation temperature lines. Bubbles would grow in the plotted case at about 50 ms assuming a lower value for the DT Young's modulus, and at 58 ms assuming DT to have a high E-value.

As we let the code run to simulate bubble nucleation and growth it returns a very fast growth of bubbles (similar to the isobaric cases of the previous chapter) until about eight microns in diameter. After that the heat flowing into the bubble on one side is slightly smaller than the heat flowing out on the other, leading to a stagnation of bubble growth. Meanwhile, as the heating continues, and the melt layer continues growing, the temperature in the bubble also increases further. At some point, the

temperature in the bubble gets higher than the temperature of the surrounding liquid. At that time, physics considerations dictate a flow of heat from the bubble to the liquid, condensing the DT again, leading to a collapse of the bubble. The code cannot simulate this result as it is increasing the time step looking for an equilibrium between the heat flowing in and the heat required to grow the bubble. Instead of equilibrating the heat fluxes for a bubble collapse, the code returns the maximum allowed time step without convergence as figure 7.3 indicates. Thus, this could be considered as the threshold for bubble collapse.

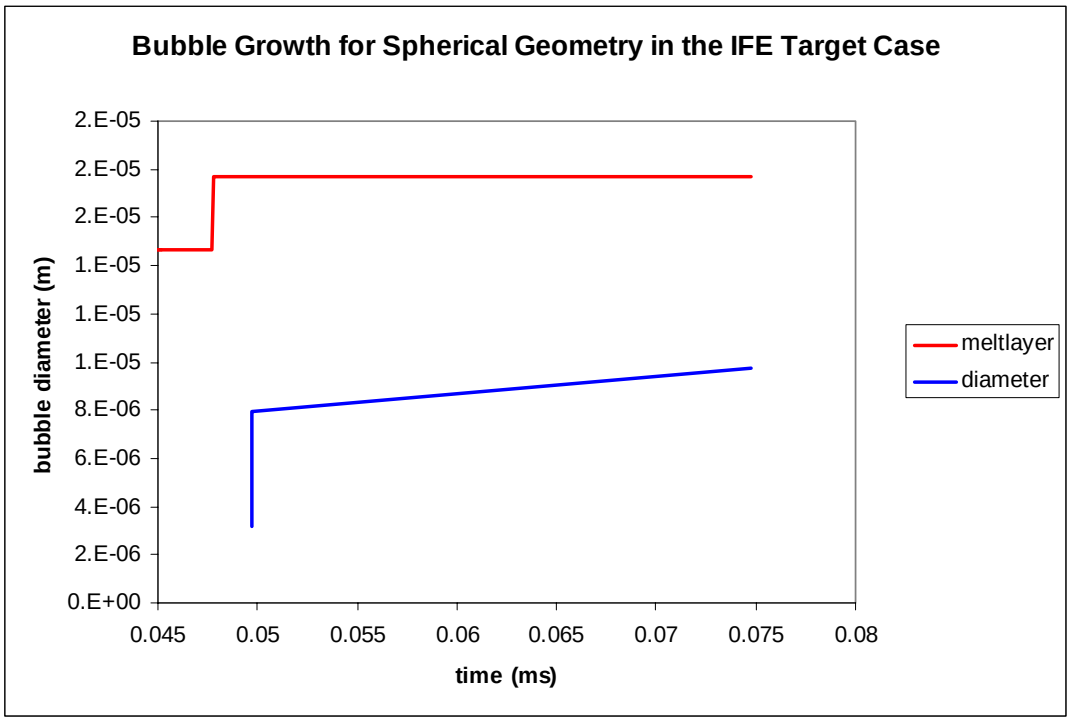


Figure 7.3: Once the superheat temperature in the liquid matches or exceeds the temperature in the bubble, the bubble starts growing. After a short time, in which the bubble grows to a diameter of about 8 μm, the fast growth comes to a stop. While in the real case, the bubble is expected to collapse, the code returns a large value for the time step, which we know to be unreasonable. If the collapse of the bubble is of further interest, the code needs to be modified for this scenario.

In order to fulfill the symmetry and smoothness requirements of a direct drive target, it seems reasonable to assume that bubble nucleation should be avoided.

7.3 Conclusions

It might not be a fully satisfying result that we cannot model the exact behavior of the bubble in the target case since the code cannot model a collapsing bubble. But we can focus on the valuable results from this study: according to the reasoning and computations presented in this work, we are able to lift the conservative restriction of triple point temperature as the maximum allowable temperature. Assuming that a liquid layer around the target does not violate the target physics symmetry requirements we showed that it is possible to temporarily separate the onset of a melt layer growth from the onset of bubble growth. Figure 7.4 shows the impact of this conclusion on the target and chamber design work.

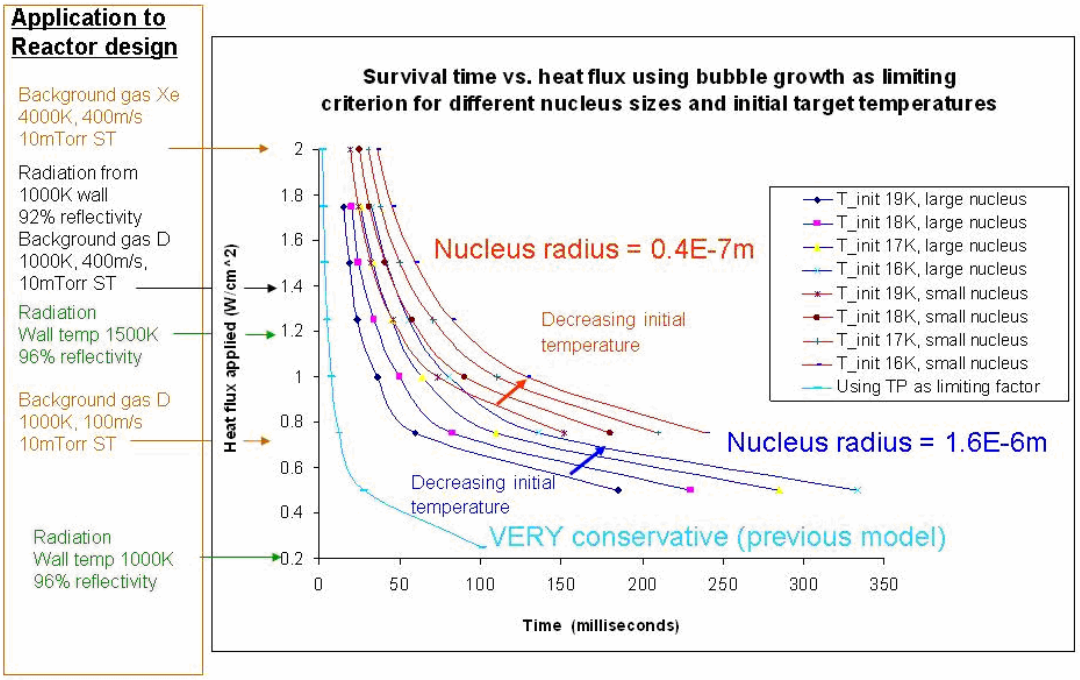


Figure 7.4: Allowing for a solid to liquid phase change, but not allowing for bubble growth (assuming target physics requirements preclude bubble formation) provides a larger design margin, which can be used to either allow a higher heat flux onto the target, or longer target survival time.

We expect the nucleus size in the actual target case to be bound between the two values plotted (0.4 and 1.6 μm), but future work should focus on a better estimate that the one elaborated in chapter 3.

Obviously, by shifting the determining factor for target survival from a maximum temperature limited to the triple point temperature to phase change (melting) but without bubble growth for the estimated ^{3}He nucleus sizes, the maximum allowable heat flux for a fixed time is raised by a factor of around three. Likewise, the survival time for a certain heat flux can be raised by a factor of five. Furthermore, extrapolating from the analysis result, it can be speculated, that if we allow for bubbles to grow and collapse, a self healing effect could turn into action which guarantees target survival as

long as the DT shell can handle the liquid pressure. However, the code cannot model bubble collapse in its current form and this would need to be verified by further modeling and, if so confirmed, ultimately by experiments.

8. Conclusion

Previous research on the thermo-mechanical response of direct drive inertial fusion energy targets was limited because of its one dimensionality. In order to relax the conservative restriction that the target's maximum outer temperature, while exposed to heat flux during injection, had to be below TP temperature, further analysis was required to model all three different phases of the DT fuel which are expected to be present as the triple point gets passed.

First, the heat flux for different chamber designs needed to be quantified. These chamber configurations included the absence of any protective gas, such that the only convective heat flux on the target would be due to helium, deuterium and tritium residuals at very low pressures. Depending on the gas species, the convective heat transfer ranged between 0.5 - 0.8 $\frac{W}{cm^2}$ and 4.5 - 6.5 $\frac{W}{cm^2}$ for 1000 K and 4000 K gas temperature, respectively. Adding the radiative heat flux of estimated 0.2 -1.2 $\frac{W}{cm^2}$ depending on the wall temperature, bounds the expected heat flux on a IFE target between 0.7 and 7.7 $\frac{W}{cm^2}$.

Since the liquid to vapor phase change is characterized by bubble nucleation and growth, we focused our attention to heterogeneous bubble nucleation. Assuming that a nucleus of a certain radius has to be present for a bubble to nucleate within a superheated liquid, we demonstrated the possibility of such nuclei to be present in the DT liquid phase due to tritium decay into ^{3}He and its diffusion into clusters and nuclei.

Assuming a layering time of a few hours, these nuclei could be big enough to serve as nucleation sites for bubble growth in superheated liquid. Depending on the number density of these clusters and the time between layering and heating, we estimated the ^{3}He nuclei to reach a radius of 0.4 microns and 1.6 microns for the 4 hrs and 18 hrs waiting time cases.

Following these considerations a bubble nucleation model was created for both hollow cylindrical and hollow spherical geometry. After testing the models on simple problems applying boundary conditions for which analytical solutions are available, the scenarios of heating experiments at LANL (cylinder) as well as the IFE target case (sphere) were simulated.

While DT - heating experiments at LANL showed bubble nucleation and growth, the exact circumstances under which phase changes occur could not completely be determined. This work identifies liquid pressure, nucleus size and initial temperature as relevant parameters in describing the phase changes and the times at which these phase changes occur in a heating experiment. The observations from LANL could be reproduced numerically with satisfying accuracy. Main results included the forecast of a deep penetration of the bubble into the solid layer as the bubble grows faster than the melt layer. Meanwhile, a significant amount of superheat (around 1K) is required for bubble nucleation to occur, which clearly separates the onset time of solid to liquid phase change from the onset of bubble nucleation (around 20ms for 18K initial target temperature). Increasing the system pressure and decreasing the nucleus radius separates the onset time of the phase changes even further, as a higher superheat temperature will be required for bubble nucleation.

Because of the volume expansion during solid to liquid phase change and the presence of a plastic shell containing the DT fuel in the target case this work predicts a pressure increase within the target as the melt layer grows. This increase in pressure results in a time - delay of the onset of bubble nucleation. Furthermore, the model predicts a collapse of the bubble soon after its nucleation, but could not simulate this collapse.

The main result of the thermomechanical analysis of the target is that a solid to liquid phase change can clearly by separated from bubble nucleation. Assuming that a melt layer does not violate the smoothness restrictions imposed by implosion physics considerations, the time delay between the melt layer growth and the onset of bubble nucleation lifts the maximum allowable temperature restriction of TP imposed by the previous model. Allowing a higher outside temperature and a liquid layer around the target allows for either a higher heat flux onto the target or provide longer survival times for a given heat flux.

APPENDIX A

The effects of Cryocontaminants on the Target Reflectivity

A major part of the heat flux acting on the target during injection results from radiation from the walls, which are estimated to have an average temperature of 700 to 1000 K. In order to reduce this heat flux, it has been proposed [3] to coat the target with a thin layer of highly reflective metal like gold or palladium. The reflectivity of the coated target has been previously analyzed using a computer model based on the Fresnel approximation written by T.K. Mau and Zoran Dragojlovic at the University of California, San Diego [19]. Previous target survival studies have been based on the results from that numerical simulation, which shows an estimated reflectivity between 94 to 96%.

In this study, the effect of cryocontaminants has been taken into consideration. The target temperature is low enough fro impurity gases (such as H_2O or CO_2) to freeze on the surface prior to injection. Estimates [7] predict a thickness of 0.8 micron. Using the reflectivity data for water and carbon dioxide from [27], [28], the results from the Fresnel code show a huge drop in reflectivity as water freezes on the surface, whereas the reflectivity of the gold layer stays almost unchanged as a CO_2 layer builds up (Figure A.1)

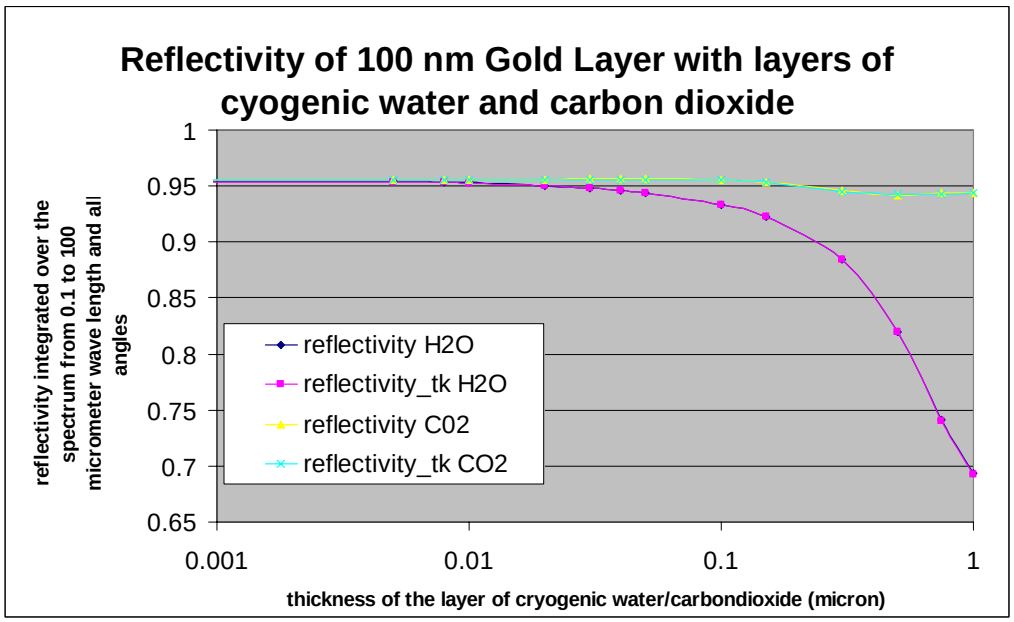


Figure A.1 The reflectivity of a 100 micron Au layer and cryodeposits of water and carbon dioxide is plotted. Clearly, the reflectivity of the target coating drops significantly as the water layer increases. A layer of frozen carbon dioxide hardly affects the overall reflectivity.

APPENDIX B

Minimum Allowable Injection Velocities

The possibility of lowering the injection velocity while accommodating the assumed target survival requirement was assessed. For the two gas temperatures of 1000K and 4000K and chamber pressures of 0.5 to 10 mTorr the DS2V software [4] was used to find the convective heat load values presented in table B.1.

Table B.1: The heat load onto the target imposed by a Deuterium background gas is determined using the DS2V software for different pressures and temperatures.

Heat Load due to Deuterium Protective Gas in the Chamber ($\frac{W}{cm^2}$)				
PRESSURE At ST=300K (mTorr)	4000K		1000K	
	400m/s	100m/s	400m/s	100m/s
0.5	0.34	0.31	0.048	0.038
1.0	0.625	0.600	0.095	0.075
5	3.53	3.20	0.5	0.33
10	6.5	6.0	1.0	0.75

Assuming a chamber gas pressure of 1.0 mTorr at ST (green shaded values) and a wall radiation heat flux of 0.2 $\frac{W}{cm^2}$ (based on 96% reflectivity and 1000K wall temperature), and using the one dimensional thermo-mechanical model [1] in an iterative scheme, parametric analyses were done, whose results are summarized in table B.2 and figure B.1 based on DT reaching the triple point as a survival requirement. The flight times are computed based on a chamber radius of 6.5 m.

Table B.2: Assuming a chamber pressure of 1 mTorr at ST and considering radiative and convective heat fluxes, the following initial temperatures are determined by numerical simulation. As a limiting criterion for target survival a maximum DT-temperature of 19.79 K is used.

1000 K Deuterium Background Gas	Temperature (K)	Injection Velocity m/s	heat flux W/cm^2	Flight time s
	16	25	0.275	0.26
	18.2	100	0.28	0.065
	18.7	200	0.285	0.0325
	19.03	400	0.29	0.01625
4000 K Deuterium Background Gas	Temperature (K)	Injection Velocity m/s	heat flux W/cm^2	Flight time s
	12	54	0.8	0.26
	14.7	100	0.805	0.065
	16.6	200	0.812	0.0325
	17.5	400	0.82	0.01625

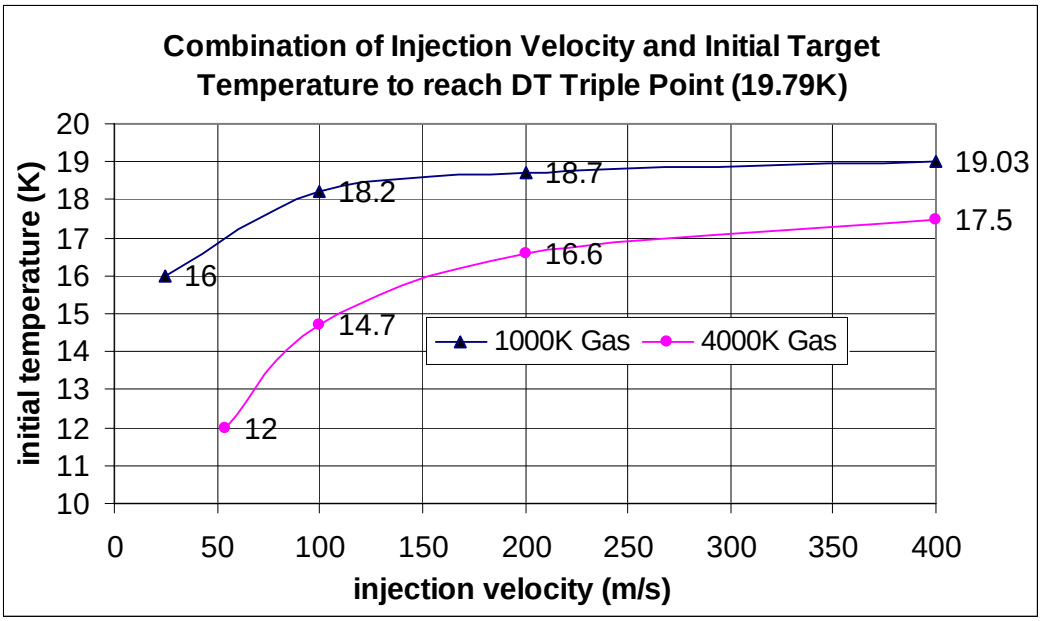
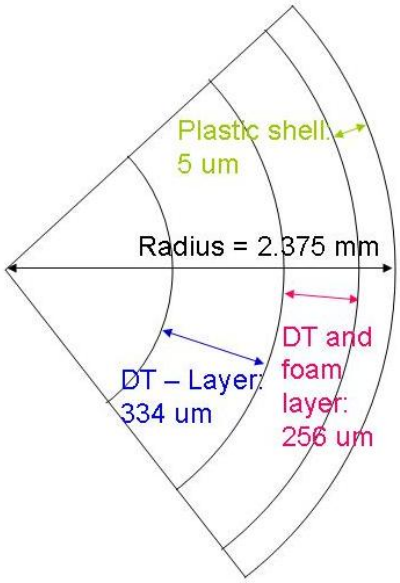


Figure B.1: The initial injection temperature is plotted as a function of the injection velocity for DT to reach its triple point (19.79K) in a chamber with D (1 mTorr at ST) at 1000K and 4000K, respectively and a wall temperature of 1000K. Clearly, the injection temperature must be lower for lower injection velocities.

APPENDIX C

Parametric Study on Larger Target

The exact designs of the different components of the IFE power plant are not determined yet. We are able to predict the target thermo-mechanical behavior for different target geometries, different chamber sizes, protective gas pressures, chamber wall temperatures etc. It is obvious, that all the uncertainties presented about bubble nucleation and growth, liquid pressure buildup and temperature distribution around the bubble become irrelevant, if the target reaches the center of the chamber with a maximum temperature below the triple point. For a different target geometry than the one assumed so far (figure C.1), the relations between heat flux, initial temperature and final maximum temperature have been analyzed using the previous model [1].



Injection Velocity	100m/s
Chamber Radius	10.5 m
Wall Temperature	1000K
Target Reflectivity	96%
Heat Flux	0.5 W/cm^2

Figure C.1: a larger radius and thicker layers have been proposed as a new target design to be analyzed for different heat fluxes and initial target temperatures.

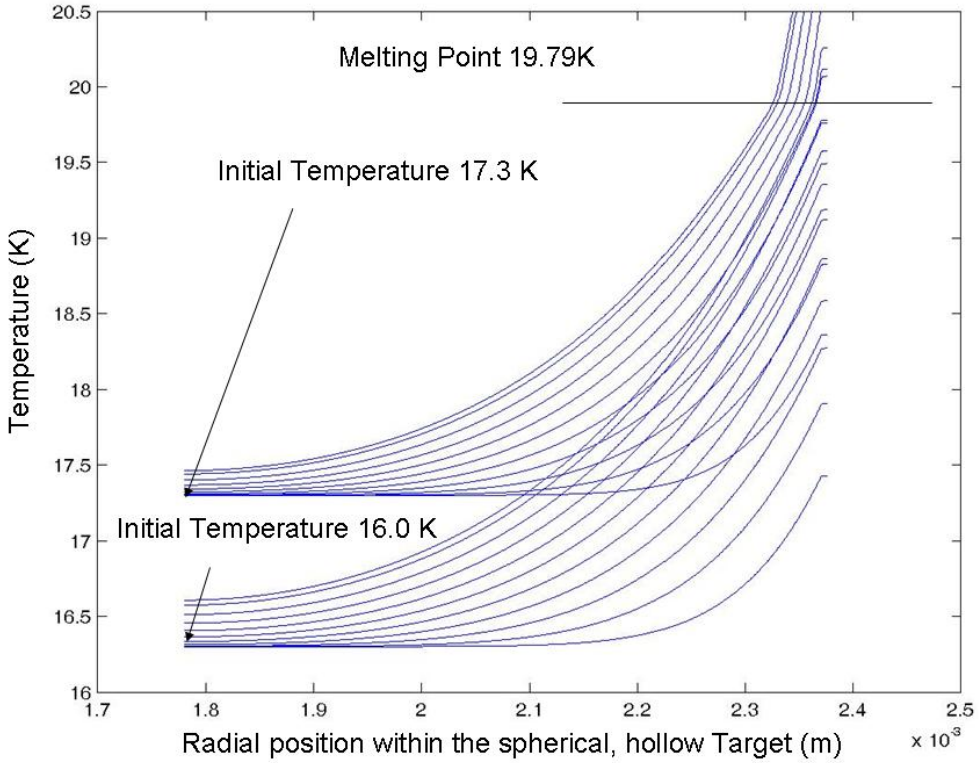


Figure C-2: The temperature profiles for the target with changed geometry have been plotted every ten milliseconds. The incoming heat flux was 0.5 W/cm^2. For an initial temperature of 17.3 K, we observe a temperature of 21.5 K at the outer surface, which corresponds to a superheat of 1.7 K.

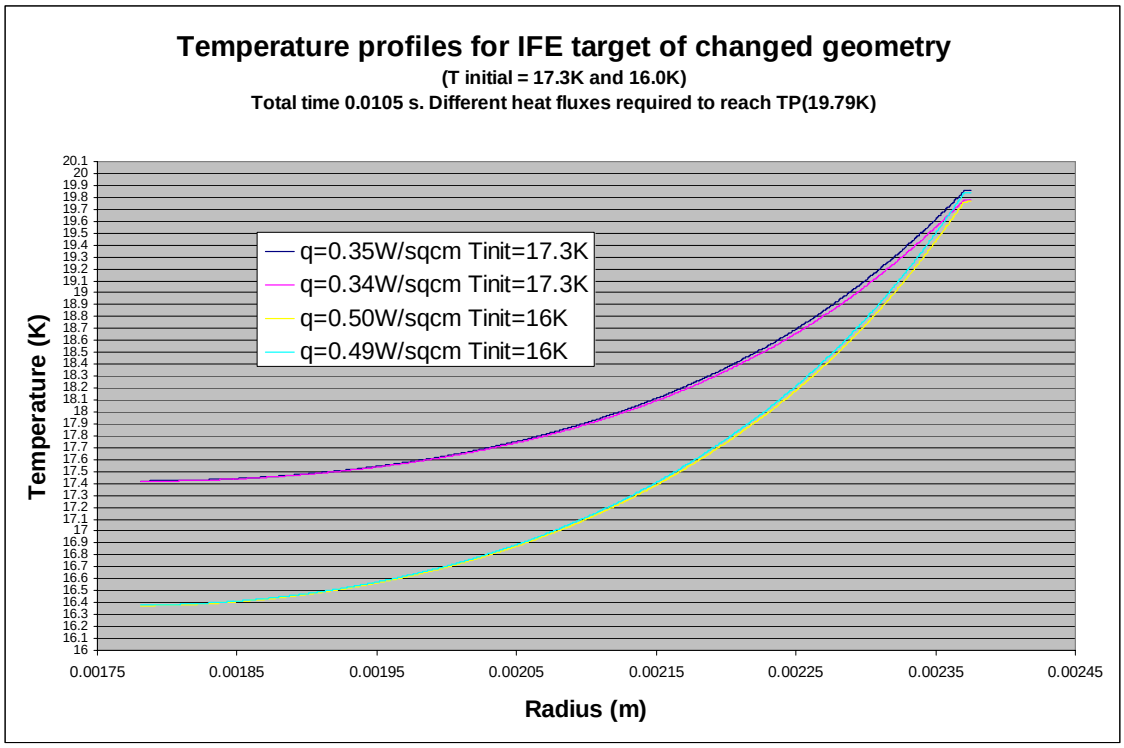


Figure C.3: We computed the maximum allowable heat flux for the two cases with initial target temperatures of 17.3 and 16 K, respectively, based on DT reaching its triple point. The resulting heat fluxes are low, allowing only for a low density of chamber constituents during injection.

APPENDIX D

3-D Model Setup

Since the goal of this study was to model a 3-d spherical bubble off centered in a spherical domain, the possibility of a 3 dimensional code was analyzed. Similar to the presented 2-D case, the heat diffusion equation has been discretized as a first step in the setup of the code. The 3- dimensional spherical heat diffusion equation with constant coefficients reads:

$$\frac{1}{\alpha}\cdot\frac{\delta T}{\delta t}=\frac{1}{r^2}\cdot\frac{\delta}{\delta r}\left(r^2\cdot\frac{\delta T}{\delta r}\right)+\frac{1}{r^2\sin\theta}\cdot\frac{\delta}{\delta\theta}\left(\sin\theta\cdot\frac{\delta T}{\delta\theta}\right)+\frac{1}{(r\sin\theta)^2}\cdot\frac{\delta^2 T}{\delta\phi^2} \quad \text{(D.1)}$$

In discretized form applied on a uniform spaced grid, equation D-1 becomes:

$$\begin{aligned}
T_{i,j,k}^{n} = &\left[\frac{\alpha\Delta t}{\Delta r^2}\cdot\left(2+\frac{2}{i^2\Delta\theta^2}+\frac{2}{i^2\sin^2\theta\cdot\Delta\varphi^2}\right)+1\right]\cdot T_{i,j,k}^{n+1}+\\
&\left[\frac{\alpha\Delta t}{\Delta r^2}\cdot\left(-\frac{1}{i}-1\right)\right]\cdot T_{i+1,j,k}^{n+1}+\\
&\left[\frac{\alpha\Delta t}{\Delta r^2}\cdot\left(\frac{1}{i}-1\right)\right]\cdot T_{i-1,j,k}^{n+1}+\\
&\left[\frac{\alpha\Delta t}{\Delta r^2}\cdot\left(-\frac{1}{i^2}\cdot\frac{\cos\left(j\cdot(\Delta\theta)\right)}{\sin\left(j(\Delta\theta)\right)}\cdot\frac{1}{2\Delta\theta}-\frac{1}{i^2\left(\Delta\theta^2\right)}\right)\right]\cdot T_{i,j+1,k}^{n+1}+\\
&\left[\frac{\alpha\Delta t}{\Delta r^2}\cdot\left(\frac{1}{i^2}\cdot\frac{\cos\left(j\cdot(\Delta\theta)\right)}{\sin\left(j(\Delta\theta)\right)}\cdot\frac{1}{2\Delta\theta}-\frac{1}{i^2\left(\Delta\theta^2\right)}\right)\right]\cdot T_{i,j-1,k}^{n+1}+\\
&\left[\frac{\alpha\Delta t}{\Delta r^2}\cdot\left(-\frac{1}{i\sin^2\left(j\Delta\theta\right)\Delta\varphi^2}\right)\right]\cdot T_{i,j,k+1}^{n+1}+\\
&\left[\frac{\alpha\Delta t}{\Delta r^2}\cdot\left(-\frac{1}{i\sin^2\left(j\Delta\theta\right)\Delta\varphi^2}\right)\right]\cdot T_{i,j,k-1}^{n+1}
\end{aligned} \quad \text{(D.2)}$$

The resulting matrix has a checkerboard structure similar to the 2-D case, with the only difference of being heptadiagonal (seven entries on each line, rather than five). The proposed Gauss Red Black algorithm as well as the multi grid algorithm discussed in Appendix E can be applied to this problem, solving it quickly and efficiently. The computational expense to solve a problem of this size seemed too large as to implement this system in code. If several adjacent bubble are to be analyzed, a 3-d code would become necessary, while a single bubble can be exactly modeled using the symmetry in the φ- direction.

APPENDIX E

The Multigrid Algorithm

The general methodology behind solving the penta-diagonal system at hand (as any two dimensional system will result in), is chosen to be the Gauss Red Black algorithm. This special case of a splitting method uses an iterative scheme that converges into the exact solution at each time step up to machine precision when enough iterations are performed. Gauss Red Black converges faster than other splitting methods leveraging the checkerboard structure of the problem. It lies in the nature of the algorithm that errors oscillating largely over the domain will only converge very slowly, while fast oscillating errors get resolved quickly [21].

In order to shorten the computational time, the multigrid algorithm has been considered to solve this problem. Press et alt. have shown a faster convergence, if the grid spacing of the problem is altered in the search for a converging solution. Hereby the linear elliptic operator describing the system is applied on the grid to smoothen the quickly oscillating errors in the domain. After that, the operator is applied on every second grid point only, and smoothened, now only using half of the grid points. This scheme is used until the operator is only applied to a four by four grid describing the domain. Once this system is solved, we use a step wise linear interpolation scheme to assign values to the whole domain again.

The details of how the mechanism of the multi grid algorithm can be researched from reference [22].

This faster converging algorithm has been implemented in the problem at hand, the code can be found in the addenda.

A measure to compare the Gauss Seidel and the multi-grid method is how much the maximum error changes with each iteration. While Gauss Seidel improves by a factor of 1.1 – 1.3, the multi-grid algorithm gives an improvement factor of 10-30 per iteration.

Although the multi-grid algorithm proves to be more powerful than the Gauss Seidel method used in the code, it has not fully been implemented in the code for the following reasons:

- The multi-grid algorithm fails if the coefficients don't vary smoothly over the domain. Both when we model the solid – liquid phase change as well as bubble nucleation, we don't have smoothly varying parameters any more.
- By using the 2- D code iteratively guessing different time steps, an exact solution is not required to get to a better guess for the next iteration. Even a few (100 to 150) Gauss Seidel Iterations give us a good idea whether the guess for the time step was good or not. As the guess for the time step gets better and better, eventually Gauss Seidel delivers a converged solution before getting to the next time step.
- Nevertheless the algorithm was implemented for a uniform spaced grid and constant coefficients, but it remains an academic exercise.

```
program test_poisson_mg
! A 2D Poisson solver on uniform mesh using Multigrid and
Red/Black Gauss-Seidel
! By Kurt Boehm after an idea from Thomas Bewley.

integer    :: ii, jj, smooth, main, time
real*8, dimension(:,:,:), allocatable :: T_save
real*8, dimension(:,:), allocatable :: T,Tp
real*8     :: e, o

   include 'header'

! Define "offset" and "max" grid variables
   g%ro=2; g%rm=g%nr+1;
   g%to=2; g%tm=g%nt+2;

   allocate(T(g%rm,g%tm) , Tp(g%rm,g%tm)); T=0.0; Tp=0.0
   allocate(T_save(timesteps,g%rm,g%tm)) ; T_save=0.0

   DO jj=1,g%tm                ! set the initial conditions
   DO ii=1,g%rm
     Tp(ii,jj)=Tinit;
     T(ii,jj)=Tinit;
   END DO
   END DO

   DO time=1,timesteps

   CALL max_error(T,Tp,g,g%rm,g%tm,e)

   DO smooth=1,n1

     CALL poisson_rb(T,Tp,g,g%rm,g%tm)

   END DO

   DO  main=1,150
     o=e;
     CALL poisson_mg(T,Tp,g,g%rm,g%tm,n2,n3);
     CALL max_error(T,Tp,g,g%rm,g%tm,e);
     print(*,*),'Iteration =',main, 'error =',e, 'factor =',o/e
     if (o/e==1 .or. e.lt.1E-12) then
     print(*,*),'Converged', e, time
     GOTO 10
   end if

   END DO

10   Tp=T;
   DO ii=1,g%rm
     DO jj=1,g%tm
        T_save(time,ii,jj)=T(ii,jj);
     END DO
   END DO

   open(7,FILE='output.txt')
   write(7,*) T_save(time,:,:)

   END DO

   close(7)

   print(*,*), 'last profile = ', T(:,3)

   deallocate(T,Tp,T_save)

end program test_poisson_mg

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!1
subroutine enforce_bcs_fine(T,Tp,g,grm,gtm)

real*8   :: temp3
```

```
real*8, dimension(grm,gtm) :: T, Tp
integer  :: jj,ii

include 'header'

! Enforce the Neumann and periodic boundary conditions

!_____________Neuman BCS in the inner cylinder__________

  DO  jj=1,g%tm
    T(1,jj)=T(2,jj);
  END DO

!___________constant heat flux at the boundary_______________

  delta_r=(Rad-Rad_inner)/(g%nr-1);
  temp3=Rad_inner/delta_r+g%rm;

   DO jj=1,g%tm
    T(grm,jj)=q_in*delta_r*(2.0*temp3-1)/(2.0*0.3*temp3)+ &
        (temp3-1)*T(grm-1,jj)/(temp3);
   END DO


!_______ periodic boundary conditions in theta______________

   DO ii=2,g%rm-1

   T(ii,1)=T(ii,g%tm-1);
   T(ii,g%tm)=T(ii,2);

   END DO

 end subroutine enforce_bcs_fine
!%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
subroutine enforce_bcs_coarse(T,Tp,g,grm,gtm)

integer  :: grm, gtm
real*8, dimension(grm,gtm) :: T, Tp
integer  :: jj,ii

! Enforce the Neumann and periodic boundary conditions
 !________Neuman BCS in the inner and outer cylinder____________

  DO  jj=1,gtm
    T(1,jj)   = T(2,jj);
    T(grm,jj) = T(grm-1,jj);
  END DO

!_____________ periodic boundary conditions in
theta________________

   DO ii=2,grm-1

   T(ii,1)=T(ii,gtm-1);
   T(ii,gtm)=T(ii,2);

   END DO

 end subroutine enforce_bcs_coarse
!%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

subroutine poisson_rb(T,Tp,g,grm,gtm)

!% Apply Red/Black Gauss-Seidel smoothing, with L derived from the
Poisson equation

integer  :: jj, ii, rb, m
real*8   :: temp3, norm, temp4, temp5, temp6
real*8, dimension(grm,gtm) :: T, Tp
```

```
include 'header'

delta_r=(Rad-Rad_inner)/(grm-2);
delta_t=pi/(16.0*(gtm-3));

temp1=alpha*delta_time*((delta_r)**(-2));

DO rb=0,1
  DO ii=2,g%rm-1       ! % update "red" points first, then "black"
points.

    m=2+mod(ii+rb+g%ro+g%to,2);

    temp3 = Rad_inner/delta_r+ii-1;
    norm=1.0/(2.0+2.0/((temp3*delta_t)**2)+1.0/temp1);

    DO jj=m,(g%tm-1),2

    T(ii,jj)=norm*((T(ii,jj+1)+T(ii,jj-1))*((temp3*delta_t)**(-2))+&
       T(ii+1,jj)*(1.0+1.0/(2.0*temp3))+T(ii-1,jj)*(1.0-
1.0/(2.0*temp3))+&
       (Tp(ii,jj)/temp1));

   END DO

   IF (grm == fine_grid) THEN
     CALL enforce_bcs_fine(T,Tp,g,g%rm,g%tm);
   ELSE
     CALL enforce_bcs_coarse(T,Tp,g,g%rm,g%tm);
   END IF
  END DO
END DO
end subroutine poisson_rb
!%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

subroutine max_error(T,Tp,g,grm,gtm,e)

real*8   :: temp3, e
real*8, dimension(grm,gtm) :: T, Tp
integer  :: jj, ii

include 'header'

delta_r=(Rad-Rad_inner)/(g%nr-1);
delta_t=pi/(16.0*(g%nt-1));
temp1=(alpha*delta_time)/(delta_r**2);

e=0;

  DO ii=2,g%nr
    DO jj=2,g%nt

      temp3 = Rad_inner/delta_r+ii-1;

     e=max(e,abs(Tp(ii,jj)-T(ii,jj)-temp1*(&
       T(ii,jj)*(2.0+2.0/((temp3*delta_t)**2))+&
       T(ii+1,jj)*(-1.0-(1.0/(2.0*temp3)))+&
       T(ii-1,jj)*(-1.0+(1.0/(2.0*temp3)))-&
       (T(ii,jj+1)+T(ii,jj-1))*((temp3*delta_t)**(-2)))));

    END DO
  END DO

END SUBROUTINE max_error
!%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

recursive subroutine poisson_mg(Tf,Tpf,gf,gfrm,gftm,n2,n3)

integer  :: jj, ii, gfrm, gftm, smooth, ic, jc

real*8, dimension(gfrm,gftm)            :: Tf, Tpf
real*8, dimension(:,:),allocatable     :: Tc, Tpc
```

```
include 'header'

!% Apply Multigrid with Red/Black Gauss-Seidel smoothing to the
Poisson eqn
gc%nr=gf%nr*0.5;
gc%nt=gf%nt*0.5;

! % Define coarse grid
gc%ro=2; gc%rm=gc%nr+1;
gc%to=2; gc%tm=gc%nt+2;

DO smooth=1,n2
  CALL poisson_rb(Tf,Tpf,gf,gf%rm,gf%tm)
END DO       ! % n2 iterations of rb smoothing

allocate(Tc(gc%rm,gc%tm),Tpc(gc%rm,gc%tm)); Tc=0.0; Tpc=0.0

DO ic=2,gc%rm-1          ! % Calculate residual and perform the
restriction
  DO jc=2,gc%tm-1        ! % (using direct injection) in a single step.

    ii=2.0*(ic-gc%ro)+gf%ro; jj=2.0*(jc-gc%to)+gf%to;

    delta_r=(Rad-Rad_inner)/(gf%nr-1);
    delta_t=pi/(16.0*(gf%nt-1));

    temp1=(alpha*delta_time)/(delta_r**2);
    temp3 = Rad_inner/delta_r+ii-1;

    Tpc(ic,jc)=Tpf(ii,jj)-Tf(ii,jj)-temp1*(&
      Tf(ii,jj)*(2.0+2.0/((temp3*delta_t)**2))+&
      Tf(ii+1,jj)*(-1.0-(1.0/(2.0*temp3)))+&
      Tf(ii-1,jj)*(-1.0+(1.0/(2.0*temp3)))-&
      (Tf(ii,jj+1)+Tf(ii,jj-1))*((temp3*delta_t)**(-2)))
      Tc(ic,jc)=Tpc(ic,jc);
  END DO
END DO

CALL enforce_bcs_coarse(Tc,Tpc,gc,gc%rm,gc%tm);
IF (gc%nr > 3 .and. gc%nt > 3 .and. mod(gf%nr,4)==0 .and.&
 mod(gf%nt,4)==0) THEN
   CALL poisson_mg(Tc,Tpc,gc,gc%rm,gc%tm,n2,n3);
ELSE
   DO smooth=1,20; CALL poisson_rb(Tc,Tpc,gc,gc%rm,gc%tm);
   END DO  !% Solve small system (almost exactly)
END IF
DO ic=2,gc%rm ! % Prolongation (bilinear interpolation) on the black int.
pts
  DO jc=2,gc%tm !% the next call to poisson_rb will take care of the red
pts
    ii=2.0*(ic-gc%ro)+gf%ro; jj=2.0*(jc-gc%to)+gf%to;
    if (jj<=gf%tm) then
      Tf(ii-1,jj)=Tf(ii-1,jj)+(Tc(ic-1,jc)+Tc(ic,jc))*0.5; end if
    if (ii<=gf%rm) then
      Tf(ii,jj-1)=Tf(ii,jj-1)+(Tc(ic,jc-1)+Tc(ic,jc))*0.5; end if
  END DO
END DO
IF (gf%rm == fine_grid) THEN
   CALL enforce_bcs_fine(Tf,Tpf,gf,gf%rm,gf%tm);
ELSE
   CALL enforce_bcs_coarse(Tf,Tpf,gf,gf%rm,gf%tm);
END IF
DO smooth=1,n3           ! % n3 iterations of smoothing
   CALL poisson_rb(Tf,Tpf,gf,gf%rm,gf%tm);
END DO

deallocate (Tc,Tpc)
END SUBROUTINE poisson_mg
```

APPENDIX F

The Heat Conduction Code including Solid- Liquid Phase Change and Bubble Nucleation

The following is a listing of the program and subroutines used in the model that was written in FORTRAN. The subroutines that returned the material properties are not shown due to their length and simplicity.

Common variables are defined at the beginning of each program and subroutine, the user input is listed in the header for convenience, which is included into any subroutine as well.

program test_poisson_mg

- Define the grid
- Calculate the bubble sizes that will be modeled using a certain grid
- Define the factors that will be used to move the profile
- Initialize the profile
- Calculate the required superheat temperature
- Call the 1-D heat diffusion code: CALL SUBROUTINE temp_profile
  - return the 1-D profile after one time step has passed
  - get the necessary material properties using SUBROUTINE (dtdensity, heatdt, conddt)
  - solve the tridiagonal system using SUBROUTINE thomas
- Check whether superheat is reached at the outer edge of the domain
- Move forward in time using the 1-D model, until superheat is reached
- Read in the profile for the 2-D domain (still symmetrical)
- ONSET OF THE BUBBLE GROWTH
- Calculate the pressure in the bubble
- Determine the temperature in the bubble
- Determine the amount of heat required to grow a bubble of that size
- Set the temperature in the bubble according to saturation temperature at the pressure
- Guess a time step
- CALL SUBROUTINE total_heat_calc:
  - Uses SUBROUTINE poisson_rb to perform a certain number of Gauss Red Black iterations.
  - After each iteration SUBROUTINE max_error is used to check how big the error is, between the exact solution and the solution at hand.
  - If that error is close to machine precision:
  - We use this temperature field to compute the heat flown into the target by:

  - Computing the heat flux at all the points around the bubble using the profile after it converged and summing them up
  - Computing the heat flux into the bubble using the profile before it converged at each point around the bubble and summing them up
  - As the actual heat flux into the bubble we apply the initial and the final profile over half a time step respectively
  - Returns the heat that would have flown into the bubble using this time step
- Get a better guess for the time step using bisection method.
- Output on the screen:
  - Time step
  - Heat flown in
  - Heat necessary to grow the bubble
- Adjust the profile to prepare for the next time step
- Start the next time step

Cylindrical Code for Inward bubble Growth

```
!----*|--.---------.---------.---------.---------.---------.---------.-|-------
! This file contains definitions of global parameters and global
variables.
!----*|--.---------.---------.---------.---------.---------.---------.-|-------
      type grid;
           integer  :: nr, nt, ro, to, rm, tm, rmi
           end type grid
           type(grid) :: g
           real*8   :: Rad_inner, Rad, pi, T_super, Temp_in_bub
           real*8   :: q_in, delta_time, delta_time_b, T_init, Cs, Cst
           real*8   :: surf_tens, press_liquid, lat_heat, gas_const
           integer  :: grm, gtm, timesteps, n1, n2, n3, bubble_steps

!%%%%%%%%%%%%%%%% User input%%%%%%%%%%%

           Cs = 0.9              ! stretching factor in radial direction
           Cst = 2.0             ! stretching factor in theta direction
           g%nr=100              ! Grid calculated to have (close to )
quadratic
           g%nt=75               ! patches where we want to model the
bubble

           delta_time_o=1.000E-4;    !  size of time steps
           Tinit=18.0;               ! initial temperature
           n1=100;               ! # of rb iterations at various points
           Rad=0.002;                ! outer radius of the DT shell
           Rad_inner=0.001533;
           q_in=-1.0E4;              ! incoming heat flux
           pi=3.141592653589793
           total_angle=pi/16
           bubble_steps=55           ! number of bubble steps
           surf_tens = 3.9E-3   ! surface tension of liquid DT (temp. dep)
           press_liquid = 22000
           lat_heat = 1350           ! latent heat of vaporization (temp.
dep)
           gas_const = 8.314         ! gas constant

%%%%%%%% END OF USER INPUT %%%%%%%%%

program test_poisson_mg
! A 2D Poisson solver on uniform mesh using Multigrid and Red/Black
Gauss-Seidel
! By Kurt Boehm, Rene Raffray and Thomas Bewley

COMMON/GRID/ R(102), delta_r(102), delta_rsq(102), Theta(80),
delta_t(80), delta_tsq(80), ld1, ld2, ud1, ud2, T_sat

integer    :: ii, jj, smooth, main, time, bubb_step, dummy, steps, bubble
integer    :: diag1, diag2, radii

real*8, dimension(:,:), allocatable :: T,Tp,T_temp
real*8, dimension(:,:), allocatable :: e_rad_star, pro_fac

real*8, dimension(101) :: profile, prev_profile
real*8, dimension(:), allocatable   :: bubb_vol, equiv_rad

real*8     :: e, o, total_heat_u,total_heat_l, Kf
real*8     :: delta_time_upper, delta_time_lower
real*8     :: interval, heat_upper, heat_lower, heat_in
real*8     :: press_bubb, delta_moles, moles, pre_moles
real*8     :: bubb_vol_1, bubb_vol_2, bubb_vol_3
real*8     :: pro_factor1, pro_factor2
real*8     :: e_rad_1, e_rad_2, e_rad_3
real*8     :: e_rad_star1, e_rad_star2, melt_layer_thickness(1002)
```

```
    include 'header'

! user iput done in the header file
! Define "offset" and "max" grid variables
g%ro=2; g%rm=g%nr+1;
g%to=2; g%tm=g%nt+2;

allocate(T(g%rm,g%tm) , Tp(g%rm,g%tm), T_temp(g%rm,g%tm));

!! define the variable for the moving profile
steps = bubble_steps+21
allocate(bubb_vol(steps) , equiv_rad(steps)); bubb_vol=0.0;
equiv_rad=0.0
allocate(e_rad_star(bubble_steps,20),pro_fac(bubble_steps,20));
pro_fac=0.0; e_rad_star=0.0

T=0.0; Tp=0.0; Profile=0.0; T_temp=0.0
bubb_vol=0.0; bubb_vol_2=0.0; heat_in=0.0; equiv_rad=0.0;
press_bubb=0.0; T_sat=0.0;
delta_moles=0.0; moles=0.0; pre_moles=0.0

!_______ _____set up the stretched grid_____________________

DO ii=0,g%nr+1
  R(ii)=Rad_inner+(TANH(Cs*((1.0*ii-
1.0)/g%nr))/TANH(Cs))*(Rad-Rad_inner)
END DO

DO ii=1,g%nr
   delta_r(ii)=R(ii+1)-R(ii)
   delta_rsq(ii)=0.5*(R(ii+1)-R(ii-1))
END DO

DO jj=-1,g%nt+1
  Theta(jj+1)=0.5*total_angle* (SINH(-
Cst+(2.0*Cst*(jj)/(g%nt+1)))/(SINH(Cst)))
END DO

DO jj=0,g%nt
  delta_t(jj+1)=Theta(jj+1)-Theta(jj)
  delta_tsq(jj+1)=0.5*(Theta(jj+1)-Theta(jj-1))
END DO

! This section calculates the equivalent radii and the factors by which we
! need to move the profile

diag1 =  (g%nt+3.0)/2.0+g%nr
diag2 = -(g%nt+3.0)/2.0+g%nr

DO bubble = 1,steps
  !! calculate all the bubble volumes
  DO ii=1,g%nr
    DO jj=1,g%nt
      if(((ii+jj).ge.(diag1+1-bubble)).and.((ii+jj).le.(diag1)).and.&
          ((ii-jj).ge.(diag2+1-bubble)).and.((ii-jj).le.(diag2)))then

        bubb_vol(bubble)=bubb_vol(bubble)+&
            0.5*(R(ii)+R(ii+1))*delta_r(ii)*delta_t(jj)
      end if
    END DO
  END DO
  equiv_rad(bubble)= (bubb_vol(bubble)/pi)**0.5
END DO

DO bubble = 1,bubble_steps
  DO radii = 1,20
    e_rad_star(bubble,radii)=(equiv_rad(bubble)**2-&
        (equiv_rad(bubble+1))**2+equiv_rad(bubble+1+radii)**2)**(0.5)
  END DO
  DO radii = 1,20
    pro_fac(bubble,radii)=(e_rad_star(bubble,radii)-
equiv_rad(bubble+radii))/&
```

```
        (equiv_rad(bubble+radii+1)-equiv_rad(bubble+radii))
  END DO
END DO

write(*,*),'Maximum Size Bubble:', equiv_rad(bubble_steps)
write(*,*),'Minimum Size Bubble:', equiv_rad(1)
write(*,*),'roundness of bubble = ',delta_r(100), R(100)*delta_t(39)



! Initialize the profile with the initial temperature

DO ii=1,g%rm
  profile(ii)=Tinit
  prev_profile(ii)=Tinit
END DO

!!! calculate pressure in minimum size bubble
     press_bub = press_liquid + 2.0*surf_tens/equiv_rad(1)
!!! calculate saturation temp. at int. pressure (by interpolation)
     T_sat = 19.99+2.0*(press_bub-22000)/(47400-22000) !! 19.79
     write(*,*),'T_sat = ', T_sat

DO time=1,1000
  call temp_profile(prev_profile,profile,g,g%rm)

! Melt Layer Thickness Subroutine
  IF (mod(time,10)==0)then
    DO ii=1,g%rm
      If (profile(ii) .ge. 19.99) then
        melt_layer_thickness(time) = 0.002 - R(ii)
       open(23,FILE='meltlayer.txt')
        write(23,*) melt_layer_thickness(time), time*0.0001
        goto 23
      END IF
    END DO
  END IF
23 continue

  IF (profile(g%nr-2).gt.T_sat) THEN
    GOTO 20
  END IF
END DO

20 pause
close(23)

open(15,FILE='t_r.txt')
write(15,*) time

!__These are the initial conditions for the uniform superheat case_

DO jj=1,g%tm              ! set the initial conditions
  DO ii=1,g%rm
    Tp(ii,jj)= profile(ii)
    T(ii,jj)=  profile(ii)    !  Initial guess of the next time step
  END DO
END DO

bubb_step=0.0
time = 0.0

ud1 = (g%nt+3)/2+g%nr
ud2 =-(g%nt+3)/2+g%nr

DO time=time+1,time+bubble_steps !timesteps

  ! check the heat flux boundary condition
  call conddt(T(g%nr,12),Kf)
  temp1=Kf*(T(g%nr+1,12)-T(g%nr,12))/delta_r(g%nr)
  print(*,*),'heat flux =', temp1
```

```
      ld1= ud1-bubb_step;
      ld2= ud2-bubb_step;

!!! calculate pressure in bubble
      press_bub = press_liquid + 2.0*surf_tens/equiv_rad(time)
!!! calculate saturation temp. at int. pressure (by interpolation)
      T_sat = 19.99+2.0*(press_bub-22000)/(47400-22000) !! 19.79

      write(*,*),'T_sat = ', T_sat

!!! save previous number of moles
      pre_moles = moles
!!! calculate moles
      moles = (press_bub*bubb_vol(time))/(gas_const*T_sat)

      delta_moles = moles - pre_moles
!!! calculate heat in
      heat_in = delta_moles*lat_heat

      Do ii=1,g%nr
        Do jj=1,g%nt
          if(((ii+jj).ge.ld1).and.((ii+jj).le.ud1).and.&
              ((ii-jj).ge.ld2).and.((ii-jj).le.ud2))then
! set the temperature in the bubble to saturation temperature at bubble
pressure
            Tp(ii,jj)= T_sat
          end if
        end do
      end do

!_________ ____initial guess for the time step
delta_time_lower=1E-7

if (time==1) then
delta_time_upper=3E-3
elseif (time>1 .and. time < 15) then
delta_time_upper=1E-2
elseif (time> 14.and. time < 25) then
delta_time_upper=1E-2
elseif (time> 24) then
delta_time_upper=1E-2
end if

interval = delta_time_upper-delta_time_lower

CALL
total_heat_calc(T,Tp,g,g%rm,g%tm,delta_time_upper,total_heat_u,total_
heat_l,bubb_step,equiv_rad)
heat_upper = total_heat_u

CALL
total_heat_calc(T,Tp,g,g%rm,g%tm,delta_time_lower,total_heat_u,total_
heat_l,bubb_step,equiv_rad)
heat_lower = total_heat_u

DO dummy=1,50
  delta_time=(delta_time_upper+delta_time_lower)/2.0

CALL
total_heat_calc(T,Tp,g,g%rm,g%tm,delta_time,total_heat_u,total_heat_l,b
ubb_step,equiv_rad)

IF (heat_in.gt.total_heat_u) THEN
  delta_time_lower = delta_time
  heat_lower = total_heat_u
ELSE
  delta_time_upper = delta_time
  heat_upper = total_heat_u
END IF

interval=interval/2
```

```
IF (time.lt.20 .and.(abs(heat_in-total_heat_u) < 1E-9)) THEN
  write(*,*), 'time_step found>>>>>>>>>>>>>>>>>>>>>>'
  write(*,*), 'heat_in**********', heat_in
  write(*,*), 'total_heat_u*******', total_heat_u
  write(*,*), 'delta_time*********', delta_time
  write(*,*), 'equiv_rad(time)****', equiv_rad(time)
  write(*,*), '>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>'
  GOTO 35

ELSEIF (time.ge.20 .and. (abs(heat_in-total_heat_u) < 1E-7)) THEN
  write(*,*), 'time_step found>>>>>>>>>>>>>>>>>>>>>>'
  write(*,*), 'heat_in**********', heat_in
  write(*,*), 'total_heat_u*******', total_heat_u
  write(*,*), 'delta_time*********', delta_time
  write(*,*), 'equiv_rad(time)****', equiv_rad(time)
  write(*,*), '>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>'
  GOTO 35

END IF
  write(*,*), 'heat_in           ', heat_in
  write(*,*), 'total_heat_u      ', total_heat_u
  write(*,*), 'delta_time        ', delta_time
  write(*,*), 'interval iteration ', dummy

END DO

35 open(7,FILE='output.txt')
  write(7,*) T(:,:)

  open(15,FILE='t_r.txt')
  write(15,*) delta_time, equiv_rad(time)

Tp=T
T_temp=T

!! here is where we have to move the profile out to prepare the next
!! bubble step, we update the profile according to e_rad_star
!! you can pull the corners out and update them out here:

DO radii=1,20

DO ii=1,g%nr  !!(g%nr-bubb_loc-2*bubb_step),g%nr
  DO jj=1,g%nt

    IF ((ii+jj)==(ld1-(1+radii)).and.(ii.lt.g%nr-0.5-0.5*bubb_step).and.&
       ii.ge.(g%nr-bubb_step-(radii)))THEN

Tp(ii,jj)=0.5*(T_temp(ii+1,jj)+T_temp(ii,jj+1))+pro_fac(time,radii)*&
          (T_temp(ii,jj)-0.5*(T_temp(ii,jj+1)+T_temp(ii+1,jj)))

    END IF

    IF ((ii-jj)==(ld2-(1+radii)).AND.(ii.lt.g%nr-0.5-0.5*bubb_step).and.&
       ii.ge.(g%nr-bubb_step-(radii)))then

      Tp(ii,jj)=0.5*(T_temp(ii,jj-
1)+T_temp(ii+1,jj))+pro_fac(time,radii)*&
          (T_temp(ii,jj)-0.5*(T_temp(ii,jj-1)+T_temp(ii+1,jj)))

    END IF
    IF (jj==((g%nt+3.0)/2.0).AND.(ii== g%nr-(bubb_step+1+radii)))
THEN
      Tp(ii,jj)=T_temp(ii+1,jj)+pro_fac(time,radii)*(T_temp(ii,jj)-
T_temp(ii+1,jj))
    END IF
!!--------------differentiate whether odd or even bubb_step
    IF (mod(bubb_step,2)==0)THEN
!-------- even
       IF   (ii == (g%nr-bubb_step*0.5).AND.&
          (jj == ((g%nt+3.0)/2.0+(1+0.5*bubb_step+radii))))THEN
         Tp(ii,jj)=T_temp(ii,jj-1)+pro_fac(time,radii)*&
            (T_temp(ii,jj)-T_temp(ii,jj-1))
```

```
        END IF
        IF   (ii == (g%nr-bubb_step*0.5).AND.&
            (jj == ((g%nt+3.0)/2.0)-(1+0.5*bubb_step+radii)))THEN
          Tp(ii,jj)=T_temp(ii,jj+1)+pro_fac(time,radii)*&
              (T_temp(ii,jj)-T_temp(ii,jj+1))
        END IF
      ELSE
!------- odd
        IF   (ii == (g%nr-bubb_step*0.5+0.5).AND.&
            (jj == ((g%nt+3.0)/2.0)-(0.5+0.5*bubb_step+radii)))THEN
          Tp(ii,jj)=T_temp(ii,jj-1)+pro_fac(time,radii)*(T_temp(ii,jj)-
T_temp(ii,jj-1))
        END IF
        IF   (ii == (g%nr-bubb_step*0.5-0.5).AND.&
            (jj == ((g%nt+3.0)/2.0)-(0.5+0.5*bubb_step+radii)))THEN
          Tp(ii,jj)=T_temp(ii,jj+1)+pro_fac(time,radii)*(T_temp(ii,jj)-
T_temp(ii,jj+1))
        END IF
        IF   (ii == (g%nr-bubb_step*0.5+0.5).AND.&
            (jj == ((g%nt+3.0)/2.0)+(0.5+0.5*bubb_step+radii)))THEN
          Tp(ii,jj)=T_temp(ii,jj-1)+pro_fac(time,radii)*(T_temp(ii,jj)-
T_temp(ii,jj-1))
        END IF
        IF   (ii == (g%nr-bubb_step*0.5-0.5).AND.&
            (jj == ((g%nt+3.0)/2.0)+(0.5+0.5*bubb_step+radii)))THEN
          Tp(ii,jj)=T_temp(ii,jj+1)+pro_fac(time,radii)*(T_temp(ii,jj)-
T_temp(ii,jj+1))
        END IF
      END IF
    END DO
END DO
END DO

bubb_step=bubb_step+1

END DO
```

```
close(15)
close(7)
close(16)
deallocate(T,Tp)

end program test_poisson_mg

subroutine enforce_bcs(T,Tp,g,grm,gtm,delta_time)

  COMMON/GRID/ R(102), delta_r(102), delta_rsq(102), Theta(80),
delta_t(80), delta_tsq(80), ld1, ld2, ud1, ud2, T_sat

  real*8   ::  Kf, kp, ro, Cpdf, temp2, norm
  real*8, dimension(grm,gtm) :: T, Tp
  integer  :: jj,ii

  include 'header'

! Enforce the Neumann and periodic boundary conditions
!________Neuman BCS in the inner cylinder_____________
  DO  jj=1,gtm
    T(1,jj)=T(2,jj);
  END DO
!_____constant heat flux at the outer boundary___________

  DO jj=1,gtm

    call conddt(T(g%rm-1,jj),Kf)
    kp = Kf

    call conddt(T(g%rm,jj),Kf)
    call dtdensity(T(g%rm,jj),ro)
    call heatdt(T(g%rm,jj),Cpdf)

    temp2 = (2.0*delta_time*Kf)/(ro*Cpdf*(delta_r(g%rm-1)**2))
    norm = 1.0/(1.0+temp2)
```

```
    T(g%rm,jj)= norm*(Tp(g%rm,jj)&
        -((2.0*delta_time*q_in)/(ro*Cpdf*delta_r(g%rm-1)*Kf))&
        *(Kf+0.5*(Kf*delta_r(g%rm-1)/R(g%rm)))&
        +T(g%rm-1,jj)*(temp2))

  END DO
!_________ periodic boundary conditions in theta_______

  DO ii=2,grm
    T(ii,1)=T(ii,gtm-1);
    T(ii,gtm)=T(ii,2);
  END DO

end subroutine enforce_bcs

subroutine poisson_rb(T,Tp,g,grm,gtm,delta_time)

  COMMON/GRID/ R(102), delta_r(102), delta_rsq(102), Theta(80),
delta_t(80), delta_tsq(80), ld1, ld2, ud1, ud2, T_sat

!% Apply Red/Black Gauss-Seidel smoothing, with L derived from
the Poisson equation
!% (normalized s.t. the diagonal elements of L are 1) on the grid g, to
Lv=d.

  real*8   :: temp1, norm, Kf, ki, kp, kn, ktp, ktn, ro, cpdf
  real*8, dimension(grm,gtm) :: T, Tp
  integer  :: jj, ii, rb, m

  include 'header'

  DO rb=0,1
    DO ii=2,g%rm-1       ! % update "red" points first, then "black"
points.

m=2+mod(ii+rb+g%ro+g%to,2);

DO jj=m,(g%tm-1),2

   if(((ii+jj).ge.ld1).and.((ii+jj).lt.ud1).and.&
       ((ii-jj).ge.ld2).and.((ii-jj).lt.ud2))then

     T(ii,jj)=T_sat
     else
  call conddt(T(ii,jj),Kf)
  ki  =  Kf
  call conddt(T(ii-1,jj),Kf)
  kp  =  Kf
  call conddt(T(ii+1,jj),Kf)
  kn  =  Kf
  call conddt(T(ii,jj-1),Kf)
  ktp =  Kf
  call conddt(T(ii,jj+1),Kf)
  ktn =  Kf

  call dtdensity(T(ii,jj),ro)
  call heatdt(T(ii,jj),Cpdf)

  temp1 = delta_time/(ro*cpdf);

  norm=1.0/((ki/delta_rsq(ii))*(1.0/delta_r(ii)+1.0/delta_r(ii-1))&
      +(ki/(delta_tsq(jj)*(R(ii)**2)))*(1.0/delta_t(jj-1)+&
      1.0/delta_t(jj))+1.0/temp1);

  T(ii,jj)=norm*( &
      T(ii,jj+1)*((1.0/(R(ii)**2))*(ki/(delta_tsq(jj)*delta_t(jj))+&
      (ktn-ktp)/(4.0*delta_tsq(jj)**2)))+ &
     T(ii,jj-1)*((1.0/(R(ii)**2))*(ki/(delta_tsq(jj)*delta_t(jj-1))-&
      (ktn-ktp)/(4.0*delta_tsq(jj)**2)))+ &
      T(ii+1,jj)*((ki/delta_rsq(ii))*(1.0/delta_r(ii)+0.5/R(ii))+&
      ((kn-kp)/(4.0*delta_rsq(ii)**2)))+ &
```

```
            T(ii-1,jj)*((-ki/delta_rsq(ii))*(-1.0/delta_r(ii-1)+0.5/R(ii))-&
             ((kn-kp)/(4.0*delta_rsq(ii)**2)))+ &
             Tp(ii,jj)/temp1);

          END IF
        END DO
        CALL enforce_bcs(T,Tp,g,g%rm,g%tm,delta_time);
      END DO
    END DO
end subroutine poisson_rb
!%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

subroutine max_error(T,Tp,g,grm,gtm,e,delta_time)

COMMON/GRID/ R(102), delta_r(102), delta_rsq(102), Theta(80),
delta_t(80), delta_tsq(80), ld1, ld2, ud1, ud2, T_sat

  real*8   :: temp1, e, Kf, ki, kp, kn, ktp, ktn, ro, cpdf
  real*8, dimension(grm,gtm) :: T, Tp
  integer  :: jj, ii

  include 'header'

  e=0.0

  DO ii=2,g%rm-2
    DO jj=2,g%tm-2

!!_we have to exclude the boundary of the bubble from the max error
calc
       if((ii+jj).ge.ld1.and.(ii+jj).lt.ud1.and.&
          (ii-jj).ge.ld2.and.(ii-jj).lt.ud2) then
         e=0.0
       else
         call conddt(T(ii,jj),Kf)
         ki  =  Kf
         call conddt(T(ii-1,jj),Kf)
         kp  =  Kf
         call conddt(T(ii+1,jj),Kf)
         kn  =  Kf
         call conddt(T(ii,jj-1),Kf)
         ktp =  Kf
         call conddt(T(ii,jj+1),Kf)
         ktn =  Kf

         call dtdensity(T(ii,jj),ro)
         call heatdt(T(ii,jj),Cpdf)

         temp1 = delta_time/(ro*Cpdf);

         e=max(e,abs(Tp(ii,jj)-T(ii,jj)-temp1*( &
           T(ii,jj)*((ki/delta_rsq(ii))*(1.0/delta_r(ii)+1.0/delta_r(ii-1))+&
             (ki/(delta_tsq(jj)*R(ii)**2))*&
             (1.0/delta_t(jj)+1.0/delta_t(jj-1)))+ &
             T(ii,jj+1)*((1.0/(R(ii)**2))*(-ki/(delta_tsq(jj)*delta_t(jj))-&
             (ktn-ktp)/(4.0*delta_tsq(jj)**2)))+ &
           T(ii,jj-1)*((1.0/(R(ii)**2))*(-ki/(delta_tsq(jj)*delta_t(jj-1))+&
             (ktn-ktp)/(4.0*delta_tsq(jj)**2)))+ &
             T(ii+1,jj)*((-ki/delta_rsq(ii))*(1.0/delta_r(ii)+0.5/R(ii))&
             -(kn-kp)/(4.0*delta_rsq(ii)**2))+ &
             T(ii-1,jj)*((-ki/delta_rsq(ii))*(1.0/delta_r(ii-1)-0.5/R(ii))&
            +(kn-kp)/(4.0*delta_rsq(ii)**2)))));
       end if
    END DO
  END DO
END SUBROUTINE max_error

subroutine temp_profile(prev_profile,profile,g,grm)

! A one D cylindrical model is developed here to model the heat transfer
until
! the superheat temperature required for a bubble to nucleate is reached
```

```
  COMMON/GRID/ R(102), delta_r(102), delta_rsq(102), Theta(76),
delta_t(76), delta_tsq(76), T_sat

  integer  :: ii
  real*8   :: temp1, Kf, ki, kp, kn, ro, cpdf
  real*8, dimension(101) :: profile, prev_profile,a,b,c

  include 'header'

  a=0.0;  b=0.0;  c=0.0

call conddt(prev_profile(grm),Kf)
temp1=Kf*(prev_profile(grm)-prev_profile(grm-1))/delta_r(grm-1)
!print(*,*),'heat flux =', temp1

  DO ii=2,grm-1

    call conddt(profile(ii),Kf)
    ki  =  Kf
    call conddt(profile(ii-1),Kf)
    kp  =  Kf
    call conddt(profile(ii+1),Kf)
    kn  =  Kf

    call dtdensity(profile(ii),ro)
    call heatdt(profile(ii),Cpdf)

    temp1 = delta_time_o/(ro*cpdf);

    a(ii)=temp1*((ki/delta_rsq(ii))*(0.5/R(ii)-1.0/delta_r(ii-1))&
            +(kn-kp)/(4.0*delta_rsq(ii)**2))

    b(ii)=1+temp1*(ki/delta_rsq(ii))*(1.0/delta_r(ii)+1.0/delta_r(ii-1))

    c(ii)=temp1*((-ki/delta_rsq(ii))*(1.0/delta_r(ii)+0.5/R(ii))&
                -(kn-kp)/(4.0*delta_rsq(ii)**2))

  END DO

! Boundary conditions zero gradient in the center
  a(1)=0.0
  b(1)=1.0
  c(1)=-1.0
  prev_profile(1)=0.0

! and const. heat flux at the outer edge
   call conddt(profile(grm),Kf)

    a(grm)=-(2.0*delta_time_o*Kf)/(ro*Cpdf*delta_r(grm-1)**2)
    b(grm)=1+(2.0*delta_time_o*Kf)/(ro*Cpdf*delta_r(grm-1)**2)
    c(grm)=0.0

    prev_profile(grm)= prev_profile(grm)&
        -((2.0*delta_time_o*q_in)/(ro*Cpdf*delta_r(grm-1)*Kf))&
        *(Kf+0.5*(Kf*delta_r(grm-1)/R(grm)))

!_____________________solve with thomas algorithm

  CALL thomas(a,b,c,prev_profile,grm)

  DO ii=1,grm
    profile(ii)=prev_profile(ii)
  END DO

end subroutine temp_profile

subroutine thomas(a,b,c,prev_profile,grm)

  real*8, dimension(101) :: prev_profile,a,b,c
  integer  :: jj,ii, grm
```

```
 DO ii=1,grm-1

   a(ii+1)= -a(ii+1)/b(ii)
   b(ii+1)= b(ii+1) + a(ii+1)*c(ii)
   prev_profile(ii+1)= prev_profile(ii+1) + a(ii+1)*prev_profile(ii)

 END DO

 prev_profile(grm)=prev_profile(grm)/b(grm)

 DO jj=grm-1,1,-1

   prev_profile(jj)=(prev_profile(jj)-c(jj)*prev_profile(jj+1))/b(jj)

 END DO

end subroutine Thomas

subroutine
total_heat_calc(T,Tp,g,grm,gtm,delta_time,total_heat_u,total_heat_l,
bubb_step,equiv_rad)
! this subroutine calcualtes the total heat transferred into the bubble
! assuming a certain size time step (which is the input)

 COMMON/GRID/ R(102), delta_r(102), delta_rsq(102), Theta(80),
delta_t(80), delta_tsq(80), ld1, ld2, ud1, ud2, T_sat

integer    :: ii, jj, smooth, bubb_step

real*8, dimension(grm,gtm) :: T,Tp,heat, heat2
real*8     :: e, o, total_heat_l, total_heat_u, Kf, equiv_rad, check

include 'header'

  DO smooth=1,n1
    o=e;
    CALL poisson_rb(T,Tp,g,g%rm,g%tm,delta_time);
    CALL max_error(T,Tp,g,g%rm,g%tm,e,delta_time)
    if (o/e==1 .or. e.lt.1E-13) then
    write(*,*),'error = ', e, 'inter = ', smooth, 'timestep = ', 'bubble = ',
bubb_step
    write(*,*),'Converged_____________________________________'
     GOTO 10;
    end if
  END DO

10 heat=0.0;heat2=0.0
write(*,*), 'error = ',e

ud1 = (g%nt+3)/2+g%nr
ud2 =-(g%nt+3)/2+g%nr
    ld1= ud1-bubb_step;
    ld2= ud2-bubb_step;

!_calculate the heat transfer into bubble from the slope on the edge__

  DO ii=1,g%nr+1
    DO jj=1,g%nt
!! along lower diagonal one -- cylindrical bubble of unit height
      IF ((ii+jj)==((g%nt+3)/2+g%nr-bubb_step-1).AND.&
         (ii.ge.g%nr-(bubb_step)).AND.(ii.lt.g%nr-0.5-
0.5*bubb_step))then
        call conddt(T(ii,jj),Kf)

        heat(ii,jj)=Kf*delta_time*(&
           ((T(ii,jj)-T(ii,jj+1))*delta_r(ii))&
           /(R(ii)*delta_t(jj)) +&
           ((T(ii,jj)-T(ii+1,jj))*(0.5*(R(ii)+R(ii+1))*delta_t(jj)))&
           /delta_r(ii))
      END IF

!! along upper diagonal one --
```

```
      IF ((ii+jj)==((g%nt+3)/2+g%nr+1).AND.(ii.gt. g%nr+0.5-
0.5*bubb_step)&
          .AND.(ii.le.(g%nr)))then
        call conddt(T(ii,jj),Kf)
        heat(ii,jj)=Kf*delta_time*(&
            ((T(ii,jj)-T(ii,jj-1))*delta_r(ii-1))&
            /(R(ii)*delta_t(jj-1)) +&
            ((T(ii,jj)-T(ii-1,jj))*(0.5*(R(ii)+R(ii-1))*delta_t(jj-1)))&
            /delta_r(ii-1))
      END IF

!! along lower diagonal two --
      IF ((ii-jj)==(-(g%nt+3)/2+g%nr-bubb_step-1).AND.&
          (ii.ge.g%nr-(bubb_step)).AND.(ii.lt.g%nr-0.5-
0.5*bubb_step))then
           call conddt(T(ii,jj),Kf)
        heat(ii,jj)=Kf*delta_time*(&
            ((T(ii,jj)-T(ii,jj-1))*delta_r(ii))&
            /(R(ii)*delta_t(jj-1)) +&
            ((T(ii,jj)-T(ii+1,jj))*(0.5*(R(ii)+R(ii+1))*delta_t(jj-1)))&
            /delta_r(ii))
      END IF

!! along upper diagonal two --
      IF ((ii-jj)==(-(g%nt+3)/2+g%nr+1).AND.&
          (ii.gt.g%nr+0.5-0.5*bubb_step).AND.(ii.le.g%nr))then
        call conddt(T(ii,jj),Kf)
        heat(ii,jj)=Kf*delta_time*(&
            ((T(ii,jj)-T(ii,jj+1))*delta_r(ii-1))&
            /(R(ii)*delta_t(jj)) +&
            ((T(ii,jj)-T(ii-1,jj))*(0.5*(R(ii)+R(ii-1))*delta_t(jj)))&
            /delta_r(ii-1))
      END IF

! lower corner --
      IF ((jj)==((g%nt+3.0)/2.0).AND.(ii == g%nr-(bubb_step+1)))THEN
        call conddt(T(ii,jj),Kf)
        heat(ii,jj)=Kf*delta_time*(&
            ((T(ii,jj)-T(ii+1,jj))*(0.5*(R(ii)+R(ii+1))*delta_t(jj)))&
            /delta_r(ii))
      END IF

! upper corner --
      IF (ii==g%nr+1.AND.(jj == (g%nt+3.0)/2.0))THEN
        call conddt(T(ii,jj),Kf)
        heat(ii,jj)=Kf*delta_time*(&
            ((T(ii,jj)-T(ii-1,jj))*(0.5*(R(ii)+R(ii-1))*delta_t(jj)))&
            /delta_r(ii-1))
      END IF
!---------------------------------------------------------------------
      IF (mod(bubb_step,2)==0) THEN
! -- even bubble steps
! -- left corner
      IF (ii ==(g%nr-bubb_step*0.5).AND.&
        (jj ==((g%nt+3.0)/2.0-(1+bubb_step*0.5))))THEN
        call conddt(T(ii,jj),Kf)
        heat(ii,jj)=Kf*delta_time*(&
            ((T(ii,jj)-T(ii,jj+1))*delta_r(ii))&
            /(R(ii)*delta_t(jj)))
      END IF

! right corner --
      IF (ii ==(g%nr-bubb_step*0.5).AND.&
        (jj ==((g%nt+3.0)/2.0+(1+bubb_step*0.5))))THEN
        call conddt(T(ii,jj),Kf)

        heat(ii,jj)=Kf*delta_time*(&
            ((T(ii,jj)-T(ii,jj-1))*delta_r(ii))&
            /(R(ii)*delta_t(jj-1)))
      END IF
    END IF
```

```
!_____________________________________________
      IF (mod(bubb_step,2)==1) THEN
! -- odd bubble steps
! -- left corners
      IF (ii ==(g%nr-bubb_step*0.5+0.5).AND.&
        (jj ==((g%nt+3.0)/2.0-(0.5+bubb_step*0.5))))THEN
        call conddt(T(ii,jj),Kf)
        heat(ii,jj)=Kf*delta_time*(&
            ((T(ii,jj)-T(ii,jj+1))*delta_r(ii))&
            /(R(ii)*delta_t(jj)))
      END IF
      IF (ii ==(g%nr-bubb_step*0.5-0.5).AND.&
        (jj ==((g%nt+3.0)/2.0-(0.5+bubb_step*0.5))))THEN
        call conddt(T(ii,jj),Kf)
        heat(ii,jj)=Kf*delta_time*(&
            ((T(ii,jj)-T(ii,jj+1))*delta_r(ii))&
            /(R(ii)*delta_t(jj)))
      END IF

! right corners --
      IF (ii ==(g%nr-bubb_step*0.5+0.5).AND.&
        (jj ==((g%nt+3.0)/2.0+(0.5+bubb_step*0.5))))THEN
        call conddt(T(ii,jj),Kf)
        heat(ii,jj)=Kf*delta_time*(&
            ((T(ii,jj)-T(ii,jj-1))*delta_r(ii))&
            /(R(ii)*delta_t(jj-1)))
      END IF
      IF (ii ==(g%nr-bubb_step*0.5-0.5).AND.&
        (jj ==((g%nt+3.0)/2.0+(0.5+bubb_step*0.5))))THEN
        call conddt(T(ii,jj),Kf)

        heat(ii,jj)=Kf*delta_time*(&
            ((T(ii,jj)-T(ii,jj-1))*delta_r(ii))&
            /(R(ii)*delta_t(jj-1)))
    END IF
    END IF
  END DO
END DO

!!!Calculate the heat flux assuming the final profile instead of the initial
!_ _calculate the heat transfer into bubble from the slope on the edge

  DO ii=1,g%nr+1
    DO jj=1,g%nt

!! along lower diagonal one -- cylindrical bubble of unit height
      IF ((ii+jj)==((g%nt+3)/2+g%nr-bubb_step-1).AND.&
          (ii.ge.g%nr-(bubb_step)).AND.(ii.lt.g%nr-0.5-
0.5*bubb_step))then
        call conddt(Tp(ii,jj),Kf)
        heat2(ii,jj)=Kf*delta_time*(&
            ((Tp(ii,jj)-Tp(ii,jj+1))*delta_r(ii))&
            /(R(ii)*delta_t(jj)) +&
            ((Tp(ii,jj)-Tp(ii+1,jj))*(0.5*(R(ii)+R(ii+1))*delta_t(jj)))&
            /delta_r(ii))
      END IF

!! along upper diagonal one --
      IF ((ii+jj)==((g%nt+3)/2+g%nr+1).AND.(ii.gt. g%nr+0.5-
0.5*bubb_step)&
          .AND.(ii.le.(g%nr)))then
        call conddt(Tp(ii,jj),Kf)
        heat2(ii,jj)=Kf*delta_time*(&
            ((Tp(ii,jj)-Tp(ii,jj-1))*delta_r(ii-1))&
            /(R(ii)*delta_t(jj-1)) +&
            ((Tp(ii,jj)-Tp(ii-1,jj))*(0.5*(R(ii)+R(ii-1))*delta_t(jj-1)))&
            /delta_r(ii-1))
      END IF

!! along lower diagonal two --
      IF ((ii-jj)==(-(g%nt+3)/2+g%nr-bubb_step-1).AND.&
```

```
          (ii.ge.g%nr-(bubb_step)).AND.(ii.lt.g%nr-0.5-
0.5*bubb_step))then
          call conddt(Tp(ii,jj),Kf)
          heat2(ii,jj)=Kf*delta_time*(&
              ((Tp(ii,jj)-Tp(ii,jj-1))*delta_r(ii))&
              /(R(ii)*delta_t(jj-1)) +&
              ((Tp(ii,jj)-Tp(ii+1,jj))*(0.5*(R(ii)+R(ii+1))*delta_t(jj-
1)))&
              /delta_r(ii))
        END IF

!! along upper diagonal two --
        IF ((ii-jj)==(-(g%nt+3)/2+g%nr+1).AND.&
           (ii.gt.g%nr+0.5-0.5*bubb_step).AND.(ii.le.g%nr))then
          call conddt(Tp(ii,jj),Kf)
          heat2(ii,jj)=Kf*delta_time*(&
              ((Tp(ii,jj)-Tp(ii,jj+1))*delta_r(ii-1))&
              /(R(ii)*delta_t(jj)) +&
              ((Tp(ii,jj)-Tp(ii-1,jj))*(0.5*(R(ii)+R(ii-1))*delta_t(jj)))&
              /delta_r(ii-1))
        END IF

! lower corner --
        IF ((jj)==((g%nt+3.0)/2.0).AND.(ii == g%nr-
(bubb_step+1)))THEN
          call conddt(Tp(ii,jj),Kf)
          heat2(ii,jj)=Kf*delta_time*(&
              ((Tp(ii,jj)-Tp(ii+1,jj))*(0.5*(R(ii)+R(ii+1))*delta_t(jj)))&
              /delta_r(ii))
        END IF

! upper corner --
        IF (ii==g%nr+1.AND.(jj == (g%nt+3.0)/2.0))THEN
          call conddt(Tp(ii,jj),Kf)
          heat2(ii,jj)=Kf*delta_time*(&
              ((Tp(ii,jj)-Tp(ii-1,jj))*(0.5*(R(ii)+R(ii-1))*delta_t(jj)))&
              /delta_r(ii-1))
        END IF
!---------------------------------------------------------------------
        IF (mod(bubb_step,2)==0) THEN
! -- even bubble steps
! -- left corner
        IF (ii ==(g%nr-bubb_step*0.5).AND.&
          (jj ==((g%nt+3.0)/2.0-(1+bubb_step*0.5))))THEN
          call conddt(Tp(ii,jj),Kf)
          heat2(ii,jj)=Kf*delta_time*(&
              ((Tp(ii,jj)-Tp(ii,jj+1))*delta_r(ii))&
              /(R(ii)*delta_t(jj)))
        END IF

! right corner --
        IF (ii ==(g%nr-bubb_step*0.5).AND.&
          (jj ==((g%nt+3.0)/2.0+(1+bubb_step*0.5))))THEN
          call conddt(Tp(ii,jj),Kf)
          heat2(ii,jj)=Kf*delta_time*(&
              ((Tp(ii,jj)-Tp(ii,jj-1))*delta_r(ii))&
              /(R(ii)*delta_t(jj-1)))
        END IF
      END IF
        IF (mod(bubb_step,2)==1) THEN

! -- odd bubble steps
! -- left corners
        IF (ii ==(g%nr-bubb_step*0.5+0.5).AND.&
          (jj ==((g%nt+3.0)/2.0-(0.5+bubb_step*0.5))))THEN
          call conddt(Tp(ii,jj),Kf)
          heat2(ii,jj)=Kf*delta_time*(&
              ((Tp(ii,jj)-Tp(ii,jj+1))*delta_r(ii))&
              /(R(ii)*delta_t(jj)))
        END IF

        IF (ii ==(g%nr-bubb_step*0.5-0.5).AND.&
```

```
          (jj ==((g%nt+3.0)/2.0-(0.5+bubb_step*0.5))))THEN
          call conddt(Tp(ii,jj),Kf)
          heat2(ii,jj)=Kf*delta_time*(&
              ((Tp(ii,jj)-Tp(ii,jj+1))*delta_r(ii))&
              /(R(ii)*delta_t(jj)))
        END IF

! right corners --
        IF (ii ==(g%nr-bubb_step*0.5+0.5).AND.&
          (jj ==((g%nt+3.0)/2.0+(0.5+bubb_step*0.5))))THEN
          call conddt(Tp(ii,jj),Kf)
          heat2(ii,jj)=Kf*delta_time*(&
              ((Tp(ii,jj)-Tp(ii,jj-1))*delta_r(ii))&
              /(R(ii)*delta_t(jj-1)))
        END IF

        IF (ii ==(g%nr-bubb_step*0.5-0.5).AND.&
          (jj ==((g%nt+3.0)/2.0+(0.5+bubb_step*0.5))))THEN
          call conddt(Tp(ii,jj),Kf)
          heat2(ii,jj)=Kf*delta_time*(&
              ((Tp(ii,jj)-Tp(ii,jj-1))*delta_r(ii))&
              /(R(ii)*delta_t(jj-1)))
      END IF
      END IF

    END DO
  END DO

    total_heat_u = sum(heat(:,:))
    total_heat_l = sum(heat2(:,:))

  write(*,*),'total_heat_u',total_heat_u
  write(*,*),'total_heat_l',total_heat_l
    total_heat_u = 0.5*total_heat_u+0.5*total_heat_l

  end subroutine total_heat_calc
```

Spherical Code for Inward Bubble Growth

(Only significantly different parts)

Adding the pressure increase due to melt layer thickness:

```
(…)
DO time=1,5000
  call temp_profile(prev_profile,profile,g,g%rm)

! Melt Layer Thickness Subroutine

     DO ii=1,g%nr
       If (profile(ii) .ge. 19.99) then

         melt_layer_thickness(time) = 0.002 - R(ii)

         open(23,FILE='meltlayer.txt')
         write(23,*) melt_layer_thickness(time),
press_liq_melt,time*0.0001

         goto 25
       END IF
     END DO

25 continue

  IF (profile(g%nr-2).gt.T_sat) THEN
    GOTO 20
  END IF

  volume_melt  = 4.0*pi*(0.002**3-(0.002-
melt_layer_thickness(time))**3)/3.0
  moles_melt   = volume_melt * 44110 !could be more exact
  delta_volume = 2.72*moles_melt*1E-6
  d_vol_ovr_vol= delta_volume*3.0 / &
      (4.0*pi*(0.002-melt_layer_thickness(time))**3)

print(*,*),'time', time
print(*,*),'melt layer thickness', melt_layer_thickness(time)


!! include pressure buildup due to melt layer growth
  press_liq_melt = press_liquid + 3142867 * d_vol_ovr_vol * 0.95589
  press_bub = press_liq_melt + 2.0*surf_tens/equiv_rad(1)
!!! calculate saturation temp. at int. pressure (by interpolation)
  T_sat = 19.99+2.0*(press_bub-22000)/(47400-22000) !! 19.79

print(*,*),'press_bubb', press_bub, press_liq_melt
print(*,*),'T_sat', T_sat
print(*,*),'T_out', profile(g%nr)

open(6797, FILE='T_sat_out.txt')
write(6797,*), 0.0001*time, T_sat, profile(g%nr-2), profile(g%nr-2),
profile(g%nr)

END DO
(…)
```

Spherical Equations:

```
(…)
!___ _constant heat flux at the outer boundary_____________
  DO jj=1,gtm
```

```
    call conddt(T(g%rm-1,jj),Kf)
    kp = Kf
    call conddt(T(g%rm,jj),Kf)
    call dtdensity(T(g%rm,jj),ro)
    call heatdt(T(g%rm,jj),Cpdf)
    temp2 = (2.0*delta_time*Kf)/(ro*Cpdf*(delta_r(g%rm-1)**2))
    norm = 1.0/(1.0+temp2)
    T(g%rm,jj)=  norm*(Tp(g%rm,jj)&
        -((2.0*delta_time*q_in)/(ro*Cpdf*delta_r(g%rm-1)*Kf))&
        *(Kf+(Kf*delta_r(g%rm-1)/R(g%rm)))&
        +T(g%rm-1,jj)*(temp2))
  END DO
```

(…)

```
subroutine poisson_rb(T,Tp,g,grm,gtm,delta_time)

  COMMON/GRID/ R(102), delta_r(102), delta_rsq(102), Theta(77),
delta_t(76), delta_tsq(76), ld1, ld2, ud1, ud2, T_sat

!% Apply Red/Black Gauss-Seidel smoothing, with L derived from
the Poisson equation
!% (normalized s.t. the diagonal elements of L are 1) on the grid g, to
Lv=d.

  real*8   :: temp1, norm, Kf, ki, kp, kn, ktp, ktn, ro, cpdf
  real*8, dimension(grm,gtm) :: T, Tp
  integer  :: jj, ii, rb, m

  include 'header'

  DO rb=0,1
    DO ii=2,g%rm-1       ! % update "red" points first, then "black"
points.
      m=2+mod(ii+rb+g%ro+g%to,2);
      DO jj=m,(g%tm-1),2
  if(((ii+jj).le.ud1).and.&
     ((ii-jj).ge.ld2))then
    T(ii,jj)=T_sat
  else
    call conddt(T(ii,jj),Kf)
    ki  =  Kf
    call conddt(T(ii-1,jj),Kf)
    kp  =  Kf
    call conddt(T(ii+1,jj),Kf)
    kn  =  Kf
    call conddt(T(ii,jj-1),Kf)
    ktp =  Kf
    call conddt(T(ii,jj+1),Kf)
    ktn =  Kf
    call dtdensity(T(ii,jj),ro)
    call heatdt(T(ii,jj),Cpdf)
    temp1 = delta_time/(ro*cpdf);
  norm=1.0/((ki/delta_rsq(ii))*(1.0/delta_r(ii-1)+1.0/delta_r(ii))&
      +(ki/(delta_tsq(jj)*(R(ii)**2)))*(1.0/delta_t(jj-1)+&
      1.0/delta_t(jj))+1.0/temp1);
  T(ii,jj)=norm*( &
      T(ii,jj+1)*((1.0/(R(ii)**2))*(ki/(delta_tsq(jj)*delta_t(jj))+&
        (ktn-ktp)/(4.0*delta_tsq(jj)**2) &
        +ki/(2.0*tan(Theta(jj))*delta_tsq(jj))))+ &
      T(ii,jj-1)*((1.0/(R(ii)**2))*(ki/(delta_tsq(jj)*delta_t(jj-1))-&
        (ktn-ktp)/(4.0*delta_tsq(jj)**2) &
        -ki/(2.0*tan(Theta(jj))*delta_tsq(jj))))+ &
      T(ii+1,jj)*((ki/delta_rsq(ii))*(1.0/delta_r(ii)+1.0/R(ii))+&
        ((kn-kp)/(4.0*delta_rsq(ii)**2)))+ &
      T(ii-1,jj)*((ki/delta_rsq(ii))*(1.0/delta_r(ii-1)-1.0/R(ii))-&
        ((kn-kp)/(4.0*delta_rsq(ii)**2)))+ &
      Tp(ii,jj)/temp1);

  END IF
END DO
CALL enforce_bcs(T,Tp,g,g%rm,g%tm, delta_time);
```

```
    END DO
  END DO

end subroutine poisson_rb

subroutine max_error(T,Tp,g,grm,gtm,e,delta_time)

COMMON/GRID/ R(102), delta_r(102), delta_rsq(102), Theta(77),
delta_t(76), delta_tsq(76), ld1, ld2, ud1, ud2, T_sat

  real*8   :: temp1, e, Kf, ki, kp, kn, ktp, ktn, ro, cpdf
  real*8, dimension(grm,gtm) :: T, Tp
  integer  :: jj, ii

  include 'header'

    e=0.0

  DO ii=2,g%rm-1
    DO jj=2,g%tm-1

   !!_we have to exclude the boundary of the bubble from the max
error calc
       if(((ii+jj).le.ud1).and.&
           ((ii-jj).ge.ld2))then
         e=0.0
       else
         call conddt(T(ii,jj),Kf)
         ki  =  Kf
         call conddt(T(ii-1,jj),Kf)
         kp  =  Kf
         call conddt(T(ii+1,jj),Kf)
         kn  =  Kf
         call conddt(T(ii,jj-1),Kf)
         ktp =  Kf
         call conddt(T(ii,jj+1),Kf)
         ktn =  Kf
         call dtdensity(T(ii,jj),ro)
         call heatdt(T(ii,jj),Cpdf)
         temp1 = delta_time/(ro*Cpdf);

         e=max(e,abs(Tp(ii,jj)-T(ii,jj)-temp1*( &
             T(ii,jj)*((ki/delta_rsq(ii))*(1.0/delta_r(ii)+1.0/delta_r(ii-1))+&
             (ki/(delta_tsq(jj)*R(ii)**2))*&
             (1.0/delta_t(jj)+1.0/delta_t(jj-1)))+ &
             T(ii,jj+1)*((1.0/(R(ii)**2))*(-ki/(delta_tsq(jj)*delta_t(jj))-&
             (ktn-ktp)/(4.0*delta_tsq(jj)**2)&
             -ki/(2.0*tan(Theta(jj))*delta_tsq(jj))))+ &
             T(ii,jj-1)*((1.0/(R(ii)**2))*(-ki/(delta_tsq(jj)*delta_t(jj-1))+&
             (ktn-ktp)/(4.0*delta_tsq(jj)**2)&
             +ki/(2.0*tan(Theta(jj))*delta_tsq(jj)))) + &
             T(ii+1,jj)*((-ki/delta_rsq(ii))*(1.0/delta_r(ii)+1.0/R(ii))&
             -(kn-kp)/(4.0*delta_rsq(ii)**2)) + &
             T(ii-1,jj)*((-ki/delta_rsq(ii))*(1.0/delta_r(ii-1)-1.0/R(ii))&
             +(kn-kp)/(4.0*delta_rsq(ii)**2))));

      end if
    END DO
  END DO

END SUBROUTINE max_error

subroutine temp_profile(prev_profile,profile,g,grm)

! A one D cylindrical model is developed here to model the heat transfer
until
! the superheat temperature required for a bubble to nucleate is reached

  COMMON/GRID/ R(102), delta_r(102), delta_rsq(102), Theta(76),
delta_t(76), delta_tsq(76), T_sat

  integer  :: ii
```

```
  real*8   :: temp1, Kf, ki, kp, kn, ro, cpdf
  real*8, dimension(101) :: profile, prev_profile,a,b,c

  include 'header'

  a=0.0;  b=0.0;  c=0.0

call conddt(prev_profile(grm),Kf)
temp1=Kf*(prev_profile(grm)-prev_profile(grm-1))/delta_r(grm-1)

  DO ii=2,grm-1

     call conddt(profile(ii),Kf)
     ki  =  Kf
     call conddt(profile(ii-1),Kf)
     kp  =  Kf
     call conddt(profile(ii+1),Kf)
     kn  =  Kf
     call dtdensity(profile(ii),ro)
     call heatdt(profile(ii),Cpdf)
     temp1 = delta_time_o/(ro*cpdf);
     a(ii)=temp1*((ki/delta_rsq(ii))*(1.0/R(ii)-1.0/delta_r(ii-1))&
              +(kn-kp)/(4.0*delta_rsq(ii)**2))
     b(ii)=1+temp1*(ki/delta_rsq(ii))*(1.0/delta_r(ii)+1.0/delta_r(ii-1))
     c(ii)=temp1*((-ki/delta_rsq(ii))*(1.0/delta_r(ii)+1.0/R(ii))&
              -(kn-kp)/(4.0*delta_rsq(ii)**2))
  END DO

! Boundary conditions zero gradient in the center

  a(1)=0.0
  b(1)=1.0
  c(1)=-1.0
  prev_profile(1)=0.0
! and const. heat flux at the outer edge
  call conddt(profile(grm),Kf)
  a(grm)=-(2.0*delta_time_o*Kf)/(ro*Cpdf*delta_r(grm-1)**2)
  b(grm)=1+(2.0*delta_time_o*Kf)/(ro*Cpdf*delta_r(grm-1)**2)
  c(grm)=0.0

  prev_profile(grm)= prev_profile(grm)&
      -((2.0*delta_time_o*q_in)/(ro*Cpdf*delta_r(grm-1)*Kf))&
      *(Kf+(Kf*delta_r(grm-1)/R(grm) ))

!_____________________solve with thomas algorithm

  CALL thomas(a,b,c,prev_profile,grm)
  DO ii=1,grm
    profile(ii)=prev_profile(ii)
  END DO
end subroutine temp_profile
```

Heat flux into the bubble:

```
!___calculate the heat transfer into bubble from the slope on the edge____

   DO ii=1,g%nr+1
     DO jj=1,g%nt
!! along upper diagonal one --
        IF ((ii+jj)==(2+g%nr+1).AND.(ii.gt. g%nr+0.5-
0.5*bubb_step).AND.          (ii.le. (g%nr)))then
 ! Area when calculating  Qr
          area_rad =  pi*((0.5*(R(ii)+R(ii-1)))**2)*&
              ((0.5*(Theta(jj)+Theta(jj+1)))**2-&
              (0.5*(Theta(jj)+Theta(jj-1)))**2)/delta_r(ii-1)
 ! Area when calculating Qt
          area_theta = pi*delta_r(ii)*(Theta(jj)+Theta(jj-1))/delta_t(jj-1)
          call conddt(T(ii,jj),Kf)
          heat(ii,jj)=Kf*delta_time*((T(ii,jj)-T(ii-1,jj))*area_rad &
              +(T(ii,jj)-T(ii,jj-1))*area_theta)
```

```
      END IF

!! along lower diagonal two --
      IF ((ii-jj)==(g%nr-2-1-bubb_step).AND.&
          (ii.ge.g%nr-bubb_step).AND.(ii.lt. g%nr-0.5-
0.5*bubb_step))then
        area_rad =  pi*((0.5*(R(ii)+R(ii+1)))**2)*&
            ((0.5*(Theta(jj)+Theta(jj+1)))**2-&
            (0.5*(Theta(jj)+Theta(jj-1)))**2)/delta_r(ii)
        area_theta = pi*delta_r(ii)*(Theta(jj)+Theta(jj-1))/delta_t(jj-1)
        call conddt(T(ii,jj),Kf)
        heat(ii,jj)=Kf*delta_time*((T(ii,jj)-T(ii+1,jj))*area_rad &
            +(T(ii,jj)-T(ii,jj-1))*area_theta)
      END IF

! lower corner --
      IF (jj==2.AND.(ii == g%nr-(bubb_step+1)))THEN
        area_rad =  pi*((0.5*(R(ii)+R(ii+1)))**2)*&
            ((0.5*(Theta(jj+1)+Theta(jj)))**2)/delta_r(ii)
        call conddt(T(ii,jj),Kf)
        heat(ii,jj)=Kf*delta_time*(T(ii,jj)-T(ii+1,jj))*area_rad
      END IF

! upper corner --
      IF (jj==2.AND.(ii == g%nr+1))THEN
        area_rad =  pi*((0.5*(R(ii)+R(ii-1)))**2)*&
            ((0.5*(Theta(jj+1)+Theta(jj)))**2)/delta_r(ii-1)
        call conddt(T(ii,jj),Kf)
        heat(ii,jj)=Kf*delta_time*(T(ii,jj)-T(ii-1,jj))*area_rad
      END IF
!!-------------------------------------------------------------------
!! we have to distinguish between odd and even bubble steps

      IF (mod(bubb_step,2)==0) THEN
! --- even bubble steps
! right corner --
        IF ((jj==2+(1+bubb_step*0.5)).AND.(ii == g%nr-
bubb_step*0.5))THEN
          area_theta = pi*delta_r(ii)*(Theta(jj)+Theta(jj-1))/delta_t(jj-1)
          call conddt(T(ii,jj),Kf)
          heat(ii,jj)=Kf*delta_time*(T(ii,jj)-T(ii,jj-1))*area_theta
        END IF
      ELSE

! --- odd bubble steps
! right corners ---
        IF (ii ==(g%nr-bubb_step*0.5+0.5).AND.&
            (jj ==(2.0+(0.5+bubb_step*0.5))))THEN
          area_theta = pi*delta_r(ii)*(Theta(jj)+Theta(jj-1))/delta_t(jj-1)

          call conddt(T(ii,jj),Kf)
          heat(ii,jj)=Kf*delta_time*(T(ii,jj)-T(ii,jj-1))*area_theta
END IF
 IF (ii ==(g%nr-bubb_step*0.5-0.5).AND.&
            (jj ==(2.0+(0.5+bubb_step*0.5))))THEN

          area_theta = pi*delta_r(ii)*(Theta(jj)+Theta(jj-1))/delta_t(jj-1)
          call conddt(T(ii,jj),Kf)
          heat(ii,jj)=Kf*delta_time*(T(ii,jj)-T(ii,jj-1))*area_theta
    END IF
  END IF
END DO
END DO
```

APPENDIX G

The Diffusion Model

```
%% This code includes some parametric studies in the
%% diffusion of He3 through solid DT assuming that there is a single
%% void that acts as a irreversible trap for the He3.
%% This code will also account for the He3 buildup due to
%% the Tritium decay. Written by Kurt Boehm

clear all

D=1.155E-16;          %% diffusion coefficient (m^2/s)
R=0.3E-6;             %% different radii  (m)
delta_t=300;          %% time step (s)
grid_pts=5000;        %% number of grid points along the radius
delta_r=R/grid_pts;   %% delta radius (m)
n_time=12;            %% number of time steps
RATE = 4.46363E-10;   %% production rate (mols of He-3/second)

temp=delta_t*D/((delta_r)^2);
%% calculate grid
for i=1:grid_pts+1
grid (i)      = (i-1)*delta_r;
end

tot_vol=0;

for i=1:grid_pts
delta_vol (i) = (grid(i+1)^3 - grid(i)^3)*4*pi/3;
end

%% initialize the points

C(1:grid_pts+1,1)=0;      %% initial concentration
totHe3(1)=0;          %%

for t=1:n_time
   t
   for y=1:grid_pts+1   %% adding He-3 due to decay over the time step
       g(y)=C(y,t)+RATE*delta_t;
    end

   for j=2:grid_pts

   a(j)=(-1/j-1)*temp;
   b(j)=1+2*temp;
   c(j)=(1/j-1)*temp;

   %% set boundary conditions:

   end

   a(grid_pts+1)=-1;
   b(grid_pts+1)=1;
   b(1)=1;
   c(1)=0;

   g(1)=0;
   g(grid_pts+1)= 0;

   g=thomas(a,b,c,g,grid_pts+1);

   C(:,t+1)=g(:);



% this little loop will calculate the molar flux of Helium 3 into bubble
% mol of He-3/mol of DT at the present time step
```

```
   totHe3(t+1)=totHe3(t)+delta_t*RATE ;

   for z=1:grid_pts

      %% calculate the difference in concentration (mols of He-3 per
mols)
      %% of DT
      delta_concentration (z) = totHe3(t+1)-
0.5*(C(z,t+1)+C(z+1,t+1));

      %% particles in the bubble are calculated
      part_in_bub(z,t)= delta_concentration(z)*(6.022E23/19.88E-
6)*delta_vol(z);

   end
     He3_in_bub(t)=sum(part_in_bub(:,t));

end

 figure(1),plot(grid,C(:,t+1))
 hold on

% figure(2),plot(He3_in_bub)

He3_in_bub(t)

%plotting function:

for i = 1:n_time
figure(2),plot(grid,C(:,i))
hold on

axis([0 R 0 abs(max(C(:,n_time)))])

pause(.01)
end
```

APPENDIX H

Analyzing the LANL Melt Layer Thickness

Since the reports from the LANL experiments [24] reported an unexpected large melt layer for very early times after the start of the heat pulse, this calculation has been done to check whether it is physically possible to achieve such high melt layer thicknesses that early in the experiment.

In the present scenario, we assume the temperature of the cylindrical target to be 19.79 K solid DT. All the heat flowing in will be used to melt the DT, the liquid will not be heated up past 19.79 K. First, we figure out, how much heat is required to melt 50 um of DT.

The melted volume per unit height is: $V = \pi \cdot \left({r_{outer}}^2 - {r_{inner}}^2 \right) = 6.2046 \cdot 10^{-7} m^2$

In moles $\frac{V}{v_{liquid}} = \frac{6.2046E-11}{22.6E-6} = 2.74 \cdot 10^{-2} moles$

Energy = number of moles * latent heat of vaporization:

$2.745E-6\, moles \cdot 244 \frac{J}{mole} = 6.69$ Joules per unit height

Then we compute the heat flown into the target

Q= heat flux * time * surface area

$Q = 10000 \frac{J}{s \cdot m^2} \cdot 0.02 s \cdot 2 \cdot \pi \cdot r_{outer} = 2.51$ Joules per unit height

Even if all the incoming heat is used to melt the solid DT, the melt layer cannot have moved inward for more than

$$\sqrt{-\frac{2.51J}{244\frac{J}{mole}} 22.6E-6 \frac{1}{\pi} - {r_{outer}}^2} = 1.9814 mm$$

This corresponds to a melt layer thickness of 18.5 um.

APPENDIX I

Plotting in MATLAB

```
%% CYLINDER CASE
 clear all

Ri=0.001533;
Cs=0.9;
Cst=2.0;
Nr=100;
Nt=75;
Ang=pi/16.0;
          for j=1:Nt+1
   theta(j)=0.5*Ang*(sinh(-Cst+(2*Cst*(j-1)/Nt))/(sinh(Cst)));
end
for j=1:Nt
   dt_inner(j)=(theta(j+1)-theta(j))*0.002;
   dt_outer(j)=(theta(j+1)-theta(j))*Ri;
end
for i=1:Nr+1
   r(i)=Ri+(tanh(Cs*((i-1)/Nr))/tanh(Cs))*0.000467;
end
for i=1:Nr
   dr(i)=r(i+1)-r(i);
end
[th,r]=meshgrid(theta,r);
[X,Y]= pol2cart(th,r);
timesteps=28;
for time=1:timesteps, for i=1:Nr+1, for j=1:Nt+1
   Temp(time,i,j)=output(time,(j-1)*101+i);
end, end, end
for f=1:timesteps
Z(:,:)=Temp(f,:,:);
surf(X,Y,Z)
% if Z > 20.46
%     return
% end
pause(0.5)
end

%% SPHERICAL CASE
clear all
Ri=0.001533;
Ro=0.002;
Cs=0.92;
Cst=3.15;
Nr=100;
Nt=75;
Ang=pi/16;
for j=1:Nt+2
   theta(j)=Ang*(sinh((Cst*(j-1.5)/Nt))/(sinh(Cst)));
end
for i=1:Nr+1
   r(i)=Ri+(tanh(Cs*((i-2)/(Nr-1)))/tanh(Cs))*(Ro-Ri);
end
for i=1:Nr
   dr(i)=r(i+1)-r(i);
end
[th,r]=meshgrid(theta,r);
[X,Y]= pol2cart(th,r);
timesteps=5;
for time=1:timesteps, for i=1:Nr+1, for j=1:Nt+2
   Temp(time,i,j)=output(time,(j-1)*(Nr+1)+i);
end, end, end
for f=1:timesteps
Z(:,:)=Temp(f,:,:);
surf(X,Y,Z)
end
```

References

1. B. R. Christiansen, *Thermal and Mechanical Analysis of IFE Direct-Drive Targets.* M aster of Science Thesis, University of California, San Diego, 2004.

2. B.R. Christiansen, A.R. Raffray and M.S. Tillack, *Modelling DT Vaporization and Melting in a Direct Drive Target.* 2004

3. B.R. Christiansen, A.R. Raffray and M.S. Tillack, *Thermal Loading in Rarefied Gas.* 2004

4. DS2V Version 2.1, GAB Consulting

5. D. Goodin, personal conversation, General Atomics, San Diego. October 2004.

6. J.D. Sheliak, D.A. Geller, J.K. Hoffer, "Update on Solid DT Studies", Presentation given at the June 3-4 HAPL meeting in Los Angeles, CA, http://aries.ucsd.edu/HAPL

7. R. Petzolt, D. Goodin and N. Alexander, *Estimates of the Cryogenic Frozen Gas Buildup on Targets During Handling.* Rev. 2, 2005

8. H. Trinkhaus, B.N. Singh, *Helium accumulation in metals during irradiation – where do we stand?* Journal of Nuclear Materials, 323, 2003 pp. 229-242

9. Paul Shewmon, *Diffusion in Solids,* The Minerals, Metals and Materials Society, Warrendale, PA, 1989

10. Th. Wichert, et alt., *Migration of Helium Atoms in Copper at 25 K.* Physical Review Letters, Vol. 55, 7, 1985, pp 726 – 729

11. M.I. Baskes, et alt., *On the low-temperature Nucleation and Growth of Bubbles by Helium Bombardment of Metals.* Journal of Nuclear Materials, 102 (1981), pp 235 – 245

12. N.M. Ghoniem, et alt., *Theory of helium Transport and Clustering in Materials under Irradiation.* Journal of Nuclear Materials, 117 (1983), pp. 96- 105

13. W.D. Wilson, C.L. Bisson. M.I. Baskes, *Self Trapping of Helium in Metals.* Physical Review B, Vol. 24, 10, 1981, pp. 5616 – 5624

14. N. Kawamura, et alt., *Measurements of He-3 accumulation effect on muon catalyzed fusion in the solid/liquid DT mixtures.* Physical Letters B, 465, 1999, pp. 74-80

15. I.F. Silvera, *Solid Molecular hydrogens in the condensed phase.* Rev. Mod. Phys., Vol. 52, 2, April 1980, pp. 422-423.

16. P.C. Souers, *Hydrogen Properties for Fusion Energy*. University of California Press, Berkeley, 1986.

17. J. G. Collier, *Convective Boiling and Condensation.* McGraw Hill Book Company, New York, 1981.

18. S. Van Stralen, R. Cole, *Boiling Phenomena.* Hemisphere Publishing Corporation, Washington, 1979.

19. Z. Dragojlovic, *Radiation Energy Deposition by Full Wave Integration*, Presentation to General Atomics, 2003.

20. M.N. Ozisik, *Finite Difference Methods in Heat Transfer.* CRC Press, Boca Raton, 1994.

21. T. Bewley, personal communication, Spring-Summer 2005. Course material for MAE 190 A and MAE 223 at UCSD 2004 – 2005.

22. Press, William H. et al., *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, pg 871 - 888 http://www.nr.com, 1988 – 1992

23. M.N. Osizik, *Heat Conduction*, J.Wiley & Sons, 2nd Edition, New York, 1993.

24. D. Geller, personal conversation, LANL, Los Alamos, NM, 2005-2006.

25. S. Timoshenko, *Theory of Plates and Shells*, McGraw Hill, New York, 1987.

26. J.E. Mark, *Polymer Data Handbook*, Oxford University Press, New York, 1999.

27. C.-K. Liu, C.L. Tien, *Cryocontamination of Optical Solar Reflectors and Mirrors*. Advances in Cryogenic Engineering, 19, 1974, pp. 474-481

28. R. Viskanta, *Thermophysics and Thermal Control.* Progress in Astronautics and Aeronautics, Vol 65, New York, 1978.

Bibliography

S. Sharafat and Nasr Ghoniem, *Stibility of Helium – Vacancy Clusters During Irradation,* Journal of Nuclear Materials 122& 123, 1984, pp. 531-536

R. Laesser, *Tritium and He-3 in Metals.* Springer Verlag, Belin, 1989.

E. Saatdjian, *Transport Phenomena*. J.Wiley & Sons, New York, 2000.

R.W. Lewis et alt., *Numerical Methods in HEAT TRANSFER,* John Wiley & Sons, New York, 1981.

C. Pozrikidis, *Numerical Computation in Science and Engineering,* Oxford University Press, New York, 1998.