Mass optimisation of cryogenic fluid systems for long-duration space missions

M. Borst

TU Delft

Cover image: modified image of launch of Titan III-Centaur
retrieved from http://mix.msfc.nasa.gov/abstracts.php?p=912

Mass optimisation of cryogenic fluid systems for long-duration space missions

by

M. Borst

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday 20th of April, 2020 at 14:00 AM.

Student number:	4299159	
Project duration:	August 5, 2019 – April 20, 2020	
Thesis committee:	Prof. E. K. A. Gill,	TU Delft SSE, chairman
	Ir. B. Zandbergen,	TU Delft SSE, supervisor
	Ir. M. J. Schuurman,	TU Delft ASM, external committee member

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Abstract

After the space race to the Moon, Mars has been chosen the new destination to be explored by mankind. Cryogenic propellant is necessary to make space beyond the Moon accessible. Boil-off of cryogenic liquids raises the initial mass of spacecraft by increasing the amount of propellant required to carry-on. Therefore, mitigation of boil-off is key in reducing the mass of the propellant tank. The goal of the thesis is to find the mass optimum of a cryogenic propellant tank by varying the different passive and active insulation options. The question what the most mass efficient configuration of propellant and structure of a given cryogenic propellant tank for space applications using active and passive boil-off mitigation options is answered.

Different options to reduce the heat flow to the fluid and vapour are identified. Spray-on foam insulation, multi-layer insulation and coatings are passive methods to prevent propellant from heating. An active component such as a cryocooler can be included in propellant tank design to remove heat from a system. It is also possible to increase the maximum allowable pressure, thereby increasing the saturation temperature of the liquid, or to reduce the initial storage temperature of the propellant to increase the heat capacity of a liquid before the liquid boils.

A tool is developed which analyses different propellant tank design options by calculating the heat flow through the propellant tank insulation and structure to the fluid bulks. The propellant tank structure is divided into separate nodes and the heat flow between different nodes is calculated. The liquid and vapour bulks are also modelled as two nodes of the system. Heat is transferred between these nodes by radiation, conduction or convection. By analysing many options in a Monte Carlo-like system, a design optimised for the total mass can be selected.

The Boil-off Monte Carlo program developed has models for different insulation methods. The conduction equation is used to identify the heat flow through the spray-on foam insulation and the propellant tank shell. For the multi-layer insulation, the empirical modified Lockheed equation is used to determine the heat flow over the layers of insulation. The mass penalty and power required for including a cryocooler to the design are found by using the empirical Ter-Brake and Air-force Laboratory relations for existing cryocoolers. In the mass penalty for the cryocooler, the additional mass for the power system and wiring are included. Different parts of the program have been verified. Due to a lack of experimental and flight data of existing spacecraft, the Boil-off Monte Carlo program is not validated.

Two cases are analysed, a propellant depot orbiting Earth at geostationary orbit and the Centaur upper stage loitering to Mars. For the propellant depot, three mission duration are used to identify when passive-only insulation suffices and from when onward an active component saves mass. The Centaur upper stage mission is used to identify whether the stage can be used for a long space mission by only modifying the insulation.

From the results, it can be concluded that it is most efficient to insulate a propellant tank with passive insulation if the mission duration is under a year. For a mission duration longer than 12 months, active measures reduce the mass of the tank compared to passive-only insulated designs.
The Centaur stage can be used for longer space missions by adding MLI to the existing design. By adding a cryocooler boil-off can be mitigated for any mission. The most mass efficient configuration of the Centaur travelling to Mars is by adding 60 layers of MLI and a cryocooler providing 5 [W] of cooling power to the original design.

Preface

People have been stargazing from the start of our existence. Back then, no one could imagine with it would be like to be up there between the planetary bodies, gas clouds and stars. From the 1950's onward, mankind has taken giant leaps in discovering the universe, both from Earth and by spacecraft. We have set foot on the Moon, but our ambition is endless. The next destination is ought to be Mars. One of the challenges we have to face is increasing the performance of our propulsion systems to be able to reach the "red planet". Liquid hydrogen will play a key role in enabling us to travel further and further. But as for a stove you are careful with the heat to not let the pot boil dry, for liquid hydrogen you should be careful with the heat to not let it simmer at all.

The majority of time in the last 9 months have been about graduating. During this period challenges became obstacles when the enthusiasm was on a low, but eventually obstacles turned into challenges again when a positive attitude towards the process and the topic was reborn. I am grateful to my supervisor Barry Zandbergen for the guidance and discussions we have had. I would like to thank Niels van der Pas for his advice regarding the methodology used, Paul Zaadnoordijk for exchanging thoughts on insulation for cryogenic tanks and Piotr Perczynski for discussing the problem of boil-off.

This master thesis concludes a period of 2.5 years of persuading a Master's degree in Aerospace Engineering from the Delft University of Technology. After you have read this thesis, I hope you have as much faith as I have in mankind reaching further than the Moon.

Mitchell Borst
Delft, April 2020

Nomenclature

Acronyms

ADR	Adiabatic Demagnetisation Refrigerators
AU	Astronomical Unit (149,597,871 [km])
CFD	Computational Fluid Dynamics
COP	Coefficient of Performance
DAM	Double Aluminized Mylar
GEO	Geosynchronous Equatorial Orbit
GFSSP	Generalized Fluid System Simulation Program
GSO	Geostationary Orbit
KSC	Kennedy Space Centre
LBO	Limited Boil-off
LEO	Low Earth Orbit
LH2	Liquid Hydrogen
LOX	Liquid Oxygen
MLI	Multi-Layer Insulation
NIST	National Institute of Science and Technology
SOFI	Spray-on Foam Insulation
VDA	Vacuum Deposited Aluminium
ZBO	Zero Boil-Off

Dimensionless numbers

Nu	Nusselt number
Ra	Rayleigh number

Greek symbols

α	absorptivity
α	thermal diffusivity
ϵ	emittance
η	(Carnot) efficiency
∇	nabla operator
ρ	density
σ_b	Stefan-Boltzmann constant
τ	shear force
ξ	solar zenith angle

Roman symbols

A	surface area
C_1	Placeholder coefficient
C_2	Placeholder coefficient
C_3	Placeholder coefficient
C	specific heat coefficient
dT	temperature change
D	diameter
E	energy in the system
h	coefficient of convection
h	height
k	coefficient of conduction
L	conductive length
L	latent heat of evaporation
M	mass
m	molecular mass
N	number
P	power

Q	heat load
q	heat flux
r	radius
T_1	source temperature
T_2	receiver temperature
T	temperature
t	thickness
t	time
V	volume
ρ_b	bond albedo
$\tilde{N}$	layer density
f	external forces
g_0	standard gravity at ground level on Earth
R_G	ideal gas constant

Subscripts

0	value at t=0 or standard value
alb	planetary albedo
C	cooling
c	characteristic
$Carnot$	Carnot
$cold$	cold side
$conduction$	conduction
$cooler$	cooler
$cryocooler$	cryocooler
$Earth$	Earth
$fuel$	fuel
h	distance h
hot	hot side
i	time instance
in	input
k	node number
max	maximum
$mission$	mission
$mixer$	mixer
MLI	MLI
net	netto
$outer$	outer
p	constant pressure
$parasitic$	parasitic
$penetrations$	penetrations
pla	planetary
$propellant$	propellant
R	distance R
$saturation$	saturation
$shell$	shell
$SOFI$	SOFI
sol	solar
$struts$	support struts
$tank$	tank
$total$	total
vap	vapour
$vapour$	vapour
$convection$	convection
$radiaton$	radiation

List of Figures

List of Tables

Contents

1

Introduction

Mankind has the desire to explore deep space in the coming decades. After the space race to the Moon, a new destination is set. The next planet to be explored by mankind ought to be Mars.[1]
The velocity increment required to go from a Low Earth Orbit (LEO) to any interplanetary orbit is at least $3.61[km/s]$ [1]. From Tsiolkovsky rocket equation[2] comes that a higher specific impulse of a propulsion system reduces the propellant mass required for a space mission. Since every kilogram launched into space results in additional propellant required, a higher specific impulse is favourable over bringing a higher amount of propellant. Conventional storable propellants have a specific impulse at sea level varying from 300 [s] till 350 [s] [2]. Alternatives to conventional storable propellants, are cryogenic liquid propellants.

A major issue with cryogenic liquid propellants is boil-off. Boil-off is the phase transition of a liquid to the vapour state. Due to the increase of vapour mass, the vapour pressure also increases. If the pressure exceeds the maximum allowable pressure, the vapour needs to be released from the system into space to mitigate the risk of the tank bursting. By venting the vapour it is not possible to use this vapour as propellant anymore and to compensate for the lost propellant, additional propellant has to be carried on. This introduction chapter aims to give the problem of losing the propellant by boil-off and why it should be mitigated.

Section 1.1 of this chapter is about the usage of cryogenic liquids as space propellants. The second section, section 1.2 presents the issue of cryogenic propellant boil-off. This results in the research objective and questions, as presented in section 1.3 and 1.4. The last section of this introductory chapter, section 1.5, gives the layout of the rest of this Master thesis report.

1.1. Cryogenic liquids as space propellants

Cryogenic propellant are liquid propellant stored at temperatures below 100 [K] [3]. Different industries use cryogenic liquids for several applications. The automotive industry for example uses cryogenic liquid hydrogen (LH2) as replacement for gasoline [4]. The oil and gas industry liquefies natural gas at cryogenic temperatures to reduce the volume required for shipment. The medical industry uses cryogenic liquids to rapidly cool organs for transport or to store medicines and in the defence industry cryogenic liquid is used in night-vision cameras to cool heat sensitive components. Next to the mentioned industries and applications, also the space industry has many applications of cryogenic liquids. Cryogenic liquids are used to cool space instrumentation, liquid hydrogen and liquid oxygen are used as propellant and liquid oxygen is utilised in life support systems for manned-missions [7]. This research focuses on the use of cryogenic liquid as a propellant. In table 1.1 an overview is given on which launchers use cryogenic propellants, which of the stages, and which cryogenic propellant combination the launchers use.

[1] https://www.theguardian.com/science/2020/jan/05/space-race-moon-mars-asteroids-commercial-launches
[2] https://www.grc.nasa.gov/WWW/K-12/rocket/rktpow.html

Launcher	Stage(s)	Propellant (O&F)
Space Shuttle [5]	orbiter	LOX&LH2
Ariane 5[3]	main stage	LOX&LH2
Ariane 5 ECA	main stage & second stage	LOX&LH2
New Shepard	rocket stage	LOX&LH2
Delta IV [5]	first stage & second stage	LOX&LH2
Ares I	upper stage	LOX&LH2
Saturn V [5]	second stage & third stage	LOX&LH2
Centaur [6]	rocket stage	LOX&LH2
Starship (Space X)[4]	second stage	LOX&CH4

Table 1.1: Existing cryogenic propellant stages used for space missions.

The cryogenic fuel hydrogen will make future deep space travel possible. Liquid hydrogen in combination with liquid oxygen deliver a specific impulse up to 465.5 [s] in a vacuum (Pratt & Whitney Rocketdyne RL10B-2 [6]). Nuclear engines which use liquid hydrogen can produce a specific impulse of up to 800 [s] (level not given by source [8]). Due to the significant higher specific impulse obtainable with hydrogen fuel, total spacecraft mass will be less compared to spacecraft using conventional propellant combinations which do not need to be cooled to cryogenic temperatures. This statement is supported by figure 1.1 from [9], in which the stage mass plus the propellant mass of the mission is plotted for two types of propellant and LH2/LOX as a function of delta-V required. Boil-off is not taken into account.

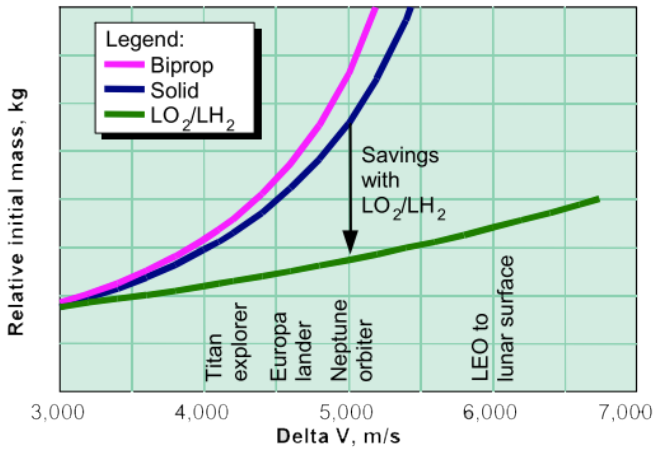

Figure 1.1: Relative initial total mass for two types of propellant compared with LH2/LOX[9].

The mass saving observed in figure 1.1 is explained by the higher specific impulse given by the propellant combination LOX/LH2. The curves are exponential due to that heavier spacecraft need more propellant to have the required delta-V and more propellant results in a bigger and heavier spacecraft. Due to the heavier spacecraft, more propellant is necessary to reach the delta-V target.

The combination of hydrogen and oxygen can be used for many long-duration space missions, especially when nuclear energy is not an option such as with manned missions. Stored as a liquid, the density of the propellant is smaller than the gaseous form. This reduces the volume required for carrying the propellant. The downside of storing the propellant in liquid form is the low temperature required. In figure 1.2 the key concepts which need to be used for cryogenic fluid management are illustrated. Further work in this thesis will focus on two of these concepts: passive storage and active storage.

[3] https://www.arianespace.com/vehicle/ariane-5/
[4] https://www.spacex.com/starship

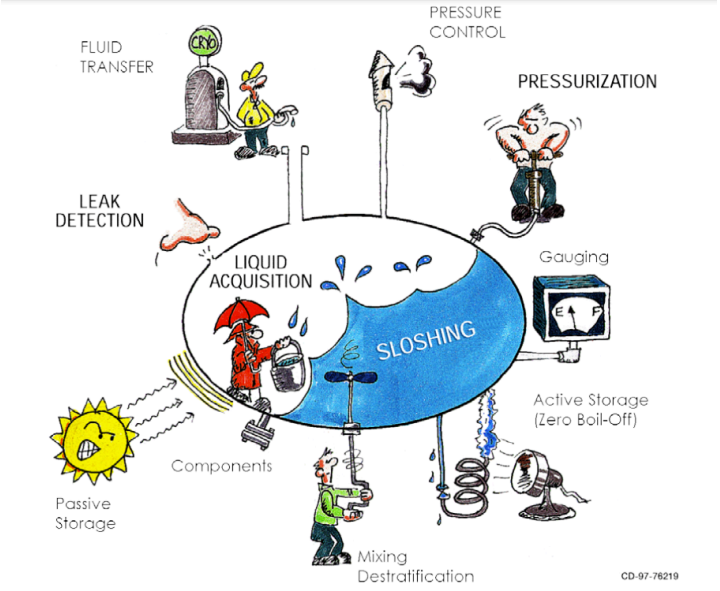

Figure 1.2: Cryogenic fluid management with icons of key concepts. [10]

1.2. Boil-off of cryogenic liquids

Cryogenic liquids are stored at temperature close to the saturation temperature (the temperature from which the liquid boils) for most applications. Cooling the liquid under the saturation temperature comes at a high cost of energy, as can be calculated by multiplying the liquid mass by the heat capacity. Due to heat absorption the liquid temperature can exceed the saturation temperature and this results in the liquid vaporising. In figure 1.3[5] the escape of hydrogen vapour from a feed-line at NASA's Kennedy Space Centre is observed. The gas from the feed-line is called boil-off gas. The rate at which liquid vaporises is the boil-off rate. The boil-off rate depends on the heat flow to and temperature of the liquid, the gas pressure of the vapour, and the latent heat of vaporisation of the liquid. This is apparent from the relations related to phase transition.

Cryogenic storage system development has priority to enable manned-missions to Mars [11]. The Saturn V launcher utilised for the Apollo missions used liquid hydrogen and liquid oxygen (LOX) as fuel and oxidiser. The liquid hydrogen had a boil-off of around 5% of the propellant mass during three hours in LEO [12]. The Atlas-Centaur rocket also used LH2 and LOX. In [9] is has been identified that a maximum of 0.02 % boil-off per day is allowable for a 500-day mission to limit the additional launch mass within 10 % of the original launch mass $(1 - (1 - 0.0002)^{500} = 0.095)$.

[5]retrieved from: https://www.nasaspaceflight.com/2015/03/ksc-shopping-lh2-ahead-sls-launch/

Figure 1.3: Boil-off trail during the ground storage of liquid hydrogen.

1.3. Research objective

This thesis will focus on the boil-off of liquid hydrogen in spacecraft propellant tanks. The ground storage and boil-off during the ground stage is not part of this thesis. For spacecraft, propellant mass fractions (mass of the propellant over the total mass of the stage) of up to 90% are normal for cryogenic rocket stages [13]. By limiting the additional fuel and oxidiser required to compensate for boil-off, mass can be saved.

Development of Zero Boil-Off (ZBO) propellant tanks has increased over the years. However, to meet the requirements set by the long loiter periods for (human) spaceflight beyond the Moon, more effort is required [14]. The need of mitigating boil-off in cryogenic storage tanks is high since deep space travel requires achieving low or ZBO rates, as identified by NASA [15] and illustrated in figure 1.4. By reducing the boil-off, the cost of the rocket and the mass will decrease. Nowadays, excess propellant has to be carried by spacecraft to compensate for boil-off.

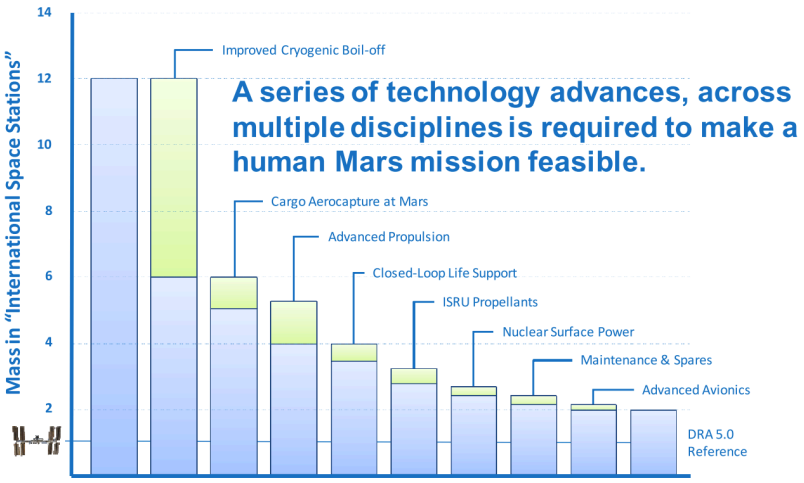

Figure 1.4: NASA has identified that improving cryogenic propellant storage is a major source for mass savings [15].

The need to limit the launch mass by reducing the propellant required by mitigating boil-off results in the research opportunity for this thesis. The research objective of this thesis is to find insulation configurations for spacecraft (propellant) storage tanks for long-duration space missions which gives the least total mass. The aim is to achieve this objective by only doing minor adjustments as adding layer of insulation to existing propellant tank designs.

Table 1.2: Overview of authors and their work with respect to ZBO or LBO and Passive only or including Active components.

Paper	ZBO	LBO	Passive only	Active	Citation
Smolak, Knoll et Wallner (1962)		X	X		[16]
MacNeil, England et Knoll (1988)		X	X		[17]
Hastings, Plachta, Salerno et Kittel (2001)	X			X	[18]
Hedayat, Hastings, Sims et Plachta (2001)	X			X	[19]
Plachta et Kittel (2002)	X		X	X	[20]
Panzarella et Kassemi (2003)	X			X	[21]
Plachta (2004)	X			X	[22]
Guernsey et al. (2005)	X		X	X	[23]
Plachta et al. (2006)	X		X		[24]
Kruif et al. (2007)	X		X		[25]
Plachta et al. (2008)	X		X	X	[26]
Ho et Rahman (2008)	X			X	[27]
McLean et al. (2008)	X			X	[28]
Perczynski et Zandbergen (2009)	X	X	X		[29]
Panzarella et Kassemi (2009)	X			X	[30]
Nast, Frank et Burns (2011)		X		X	[31]
Fesmire, Coffman, Meneghelli et Heckle (2012)		X	X		[32]
Chai et Wilhite (2014)	X	X	X	X	[33]
Sun, Guo et Huang (2015)	X		X		[34]
Kumar (2015)		X	X		[35]
Taylor et al. (2015)	X			X	[36]
Liu et al. (2016)	X			X	[37]
Notardonato et al. (2017)	X			X	[38]
Plachta et al. (2018)	X			X	[14]

In table 1.2 an overview is given of authors who performed research to boil-off in cryogenic propellant tanks. From table 1.2 the research opportunity arises. A research gap exists on researching the mitigation of boil-off while looking at ZBO and Limited Boil-off (LBO) and passive insulated propellant tanks and propellant tanks including active mitigation measures. In chapter 2 the options to mitigate boil-off are given and explained.

1.4. Research questions

To achieve the mitigation of the boil-off phenomenon in the most mass efficient manner, it is important to identify the effect of certain parameters on the physics of the propellant tank. The effect of the key parameters on boil-off will be identified by answering the sub-questions formulated. Knowing what the effect is of the different design variables, an answer can be given to the main research question.

1.4.1. Main research question

The main research question originates from the need to make a spacecraft propellant as mass efficient as possible. Whereas most literature assumes that this involves ZBO storage, it is not identified yet whether this is indeed most mass efficient. The main research question is:

'What is the most mass efficient configuration of propellant and structure of a given cryogenic propellant tank for space applications using active and passive boil-off mitigation options?'

Formulating the main question this way, there is space for boil-off as long as the boil-off mass is less than the additional insulation mass required to mitigate this boil-off. The main research question will be answered by determining the mass saving of finding a new configuration for a conventional propellant tank.

The cost of the spacecraft and reliability are two other important design factors. However, the cost is hard to relate just to design parameters of a spacecraft itself and it is not the scope of this research to identify the cost of cryogenic propellant tank design. A reliable system is achievable by increasing redundancy. This comes with a mass penalty if additional systems are added.

1.4.2. Sub questions

To answer the main research question, additional sub-questions are formulated and accompanying hypotheses.

- How much passive insulation is required to obtain ZBO in a propellant tank? From the work done by Sun [34] it follows that the time which a propellant tank can last without active thermal control devices is proportional to the initial temperature compared to the saturation temperature multiplied with the heat capacity and the propellant mass over the heat flow.
- What passive insulation option is most effective in mitigating boil-off and reducing propellant tank mass?
- What is the effect of increasing or decreasing the liquid fill ratio of the propellant tank on the boil-off rate? Larger propellant tanks with a lower fill level have a larger surface area to receive and emit heat, while smaller propellant tanks have a smaller surface area to receive heat.
- What is the effect on the boil-off of pressurising a propellant tank to the maximum allowable pressure? At higher pressure, the saturation temperature of the liquid is higher and therefore the liquid can absorb more heat from the same reference temperature before it starts to boil. Disadvantage of pressurising the propellant tank is that the propellant tank shell could burst.
- From which mission duration onward is it more mass efficient to add an active cooler to a system compared to passive systems only? In [20] it is stated that for LH2 propellant tanks, it is lighter to have a cryocooler installed on the propellant tank than mitigating boil-off by passive only insulation.

The sub-questions help in gaining an understanding of the boil-off problem. Also, the key parameters can be identified from answering the questions. These parameters are key in finding a lighter configuration if such a configuration is present.

1.5. Report layout

To be able to answer the research questions and meet the research objective of this Master thesis, the following report structure is used. In chapter 2 the information necessary to develop a methodology and tool to predict boil-off for different propellant tank configurations is given. Also, the work done by others will be explained. Chapter 3 gives the methodology used to find the boil-off rate of different propellant tanks. Before the results are produced, in chapter 4 the developed tool from the methodology is verified and a sensitivity analysis is performed. Also, a validation strategy is given. In chapter 5 and 6 two cases are described, results are generated for these cases and the results are discussed. The answer on the main question and sub-questions is given in the conclusion chapter 7. This last chapter will also provide recommendations for future work.

2

Background information

To find the best methodology to achieve the research objective, background information and knowledge about the topic of boil-off of cryogenic propellant is required. Section 2.1 is about the research conducted by other scientist and the methodologies used to predict boil-off in cryogenic propellant tanks. Section 2.2 of this chapter is about how heat propagates in solids and liquids. After that, section 2.3 is about the thermal environment around cryogenic propellant tanks. In section 2.4 the design options on how to mitigate boil-off are studied.

2.1. Research performed to boil-off

This section presents the work done on boil-off by various researchers. From this work, an understanding of which methods are available and what results are achievable is gained.

In [18], it was identified what the NASA initiatives were on mitigation of boil-off and development of zero boil-off concepts. In figure 2.1 it can be seen what components a typical ZBO propellant tank contains according to NASA. NASA has identified that the development of cryocoolers is key to the success of ZBO concepts, in which passive systems should limit the heat leak to the system to the cooling capacity of the active systems.

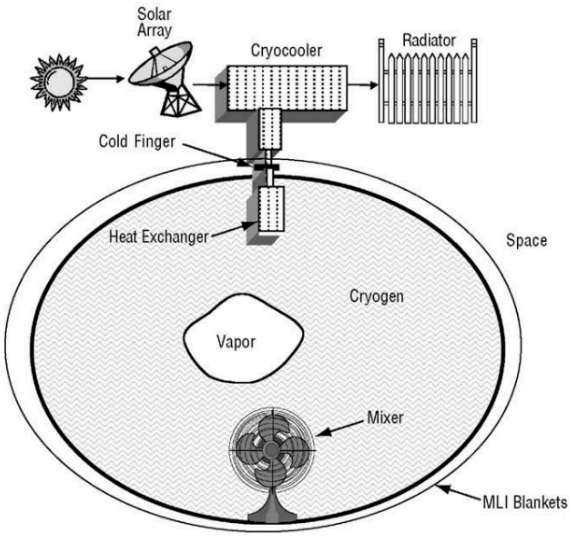

Figure 2.1: Schematic overview on components of a zero boil-off tank according to NASA[18].

In [39] it is presented that the state of the art cryogenic propellant storage performance is the Titan-Centaur 5 launcher, with a 9-hours mission duration. The boil-off performance is estimated on a boil-off

rate of 4.8 % per day of the total tank volume [10]. In the Atlas-Centaur user guide [40] it is stated that the high boil-off rates are acceptable because the mission duration the rocket is designed for is short.

There are a few papers which aim to obtain a propellant tank with reduced boil-off or zero boil-off. The results of these studies differ due to the different tank sizes simulated and assumptions used. From the papers analysed, it is noted that only one ([34]) aims to prove a zero boil-off concept by only using passive insulation. This approach is supported by [23], in which it is stated that having zero boil-off with only passive insulation should be possible if the spacecraft field of view to any planet is limited. The configurations and results of the papers with limited boil-off are presented in table 2.1, only the papers in which the results are clearly stated have been included. The list of three papers is small compared to the list of papers presented in table 1.2 from chapter 1. This could be an indication that the preferable method to store cryogenic propellant for space missions is with zero boil-off.

Table 2.1: Summary of authors and papers written with achieved boil-off rate, insulation type, tank volume and heat flux to the propellant tank.

Paper	Passive only	Active	Achieved boil-off rate	Tank Volume	Heat flux	Citation
Perczynski (2009)	X		0.2-0.006 [%/month]	2,485 [kg]	1290 [W]	[29]
Chai et al. (2014)	X	X	2.5 [%/month]	34,600 [kg]	267 [W] / 262 [K]	[33]
Kruif et Kutter (2007)	X		2.5 [%/day]	48.22 [m^3]/3401 [kg]	146.5 [W]	[25]

The boil-off rates of propellant tanks differ during different stages of a space mission. Typically, the boil-off rates during the ground stage are highest due to the conductive and convective heat transfer taking place between the atmosphere and the spacecraft. Also, during the ground stage of the space mission, it is not only necessary to reduce the heat flow to the propellant tank, but it is also key to reduce the 'cold flow' from the propellant tank to the surroundings. The cold around a cryogenic propellant tank can cause water-ice forming on the outside of the propellant tank and spacecraft. Although this has an advantage that an additional layer of insulation is created, the mass penalty is uncontrollable and the performance of the SOFI can degrade under the influence of water [32].

Next to ground test data, also actual flight mission data is presented in figure 2.2 [41]. The boil-off rates found for the Centaur upper stage vary a lot: for ground hold the boil-off rate is more than 100% per day, while in orbit the boil-off rate is in the order of 1% per day.

The remainder of this section will focus on several papers and their content. There will be a focus on the methods presented to simulate the performance of the configurations. The papers handled are Panzarella and Kassemi [21], Majumdar [42], [43], Liu [44], Perczynski [29], Chai and Wilhite [33], and Sun [34].

Panzarella and Kassemi [21] investigated the effect of pressurising a cryogenic propellant tank. The methodology used involved calculating the mass transfer from the liquid phase propellant to ullage gas. The ullage was considered an ideal gas, from which the pressure could be determined with the equation of state. Next to simulations with only passive cooling modelled, a liquid jet was modelled within the tank to cool the propellant (this is considered an active cooler). The results demonstrated that a liquid jet in a propellant tank could control the boil-off rate and pressurisation by cooling the liquid.
The methodology used several assumptions. The liquid propellant was assumed to be incompressible ($\rho_{liquid} = \text{constant}$) and the Boussinesq approximation was used. The Boussinesq approximation is valid if density differences within the liquid are small and the effect of this approximation is that the density is taken constant in the involved equations, except for when a term involves buoyancy. By using these two assumptions, a set of Navier-Stokes equations can be developed which can be solved numerically. The Navier-Stokes equations and several simplified versions are presented in appendix A.
The boundary conditions set stated that the temperature change within the liquid would be uniform, giving a homogeneous liquid bulk. The liquid jet introduced was sized by solving the Navier-Stokes equations and assuming that the liquid jet should remove as much heat as the liquid absorbs from the propellant tank wall and the vapour. By sizing the active cooler to the heat absorbed by the liquid, the goal of zero boil-off is reached. Ho and Rahman [27] use the same approach as explained in the past

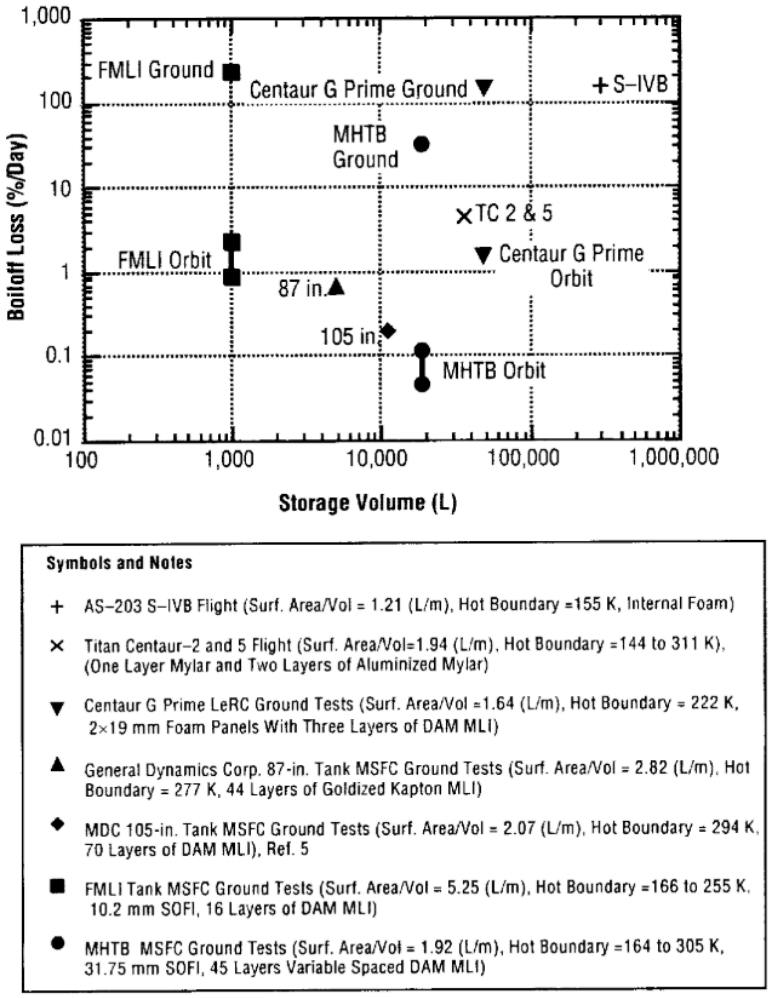

Figure 2.2: Boil-off rate over the volume for several ground tests and space missions [41].

paragraph.

In Majumdar [42] a numerical model to predict the boil-off of a large cryogenic storage tank at the Kennedy Space Centre (KSC) was created in the Generalized Fluid System Simulation Program (GFSSP) developed by NASA. It is only given about the program that the Navier-Stokes equations are solved to determine fluid interaction.

Majumdar states that a simple one-node thermodynamic model which only determines conduction from the shell to the liquid would not be accurate since vapour-liquid interactions would be ignored. The numerical model focused on heat transfer between different parts of the propellant tank, the cryogenic liquid with a homogeneous temperature and density, and the ullage volume, also considered homogeneous. The tank shell is split into 16 parts, each with different boundary conditions depending on the location. Interaction between sections is modelled for every section and its adjacent sections. This method appears valid for tanks with a small size, but for bigger tanks, the assumption to consider the ullage and liquid cryogenic as homogeneous is not valid, if boil-off occurs. The validation is done by comparing simulation data with experimental data from two cryogenic hydrogen tanks present at KSC. For smaller tanks, convection is quick enough through the tank to consider the ullage as one. For bigger tanks, however, the time scale in which the heat is distributed over the tank is too long to consider the ullage as one homogeneous gas.

The previous model is improved to a model of a self-pressure-regulating cryogenic propellant tank [43].

The model is expanded to 37 nodes, which gives higher accuracy. Also, the liquid-gas interface is modelled as an additional node, to increase the accuracy of the boil-off calculations. A self-pressurisation model is included, to determine how much work a sprayer should perform to maintain constant pressure inside of the tank. The thermodynamic module from the previous model functions separately from the self-pressurisation module developed in this work. The developed module simulates that if the pressure is too high, the vapour is vented from the system and the liquid in the propellant tank is cooled by a cryocooler.

A thesis on cryogenic storage for the Earth Departure Stage has been written by Perczynski [29]. The methodology used separates the propellant tank structure from the liquid in two modules. The structure module determines the heat flow through the structure by using a thermal nodes network. In a thermal nodes network, layers of insulation are represented by a node. The node has the properties mass, temperature, thermal resistance and specific heat capacity. With this information, combined with the heat transfer correlations given further in this chapter, the heat flow from node to node is calculated. The result of this module is a heat flow to the liquid.
The second module is the liquid module. This liquid module assumes the liquid bulk to be radially symmetric and vertically homogeneous. The radial component is split in nodes, and by using convection and conduction equations and the specific heat capacity of the fluid, the temperature rise of the fluid is determined. The boil-off is determined by using the heat flow to the fluid to vaporise fluid once the saturation temperature of the fluid is reached.

In Plachta [26] an approach is given to design a ZBO propellant tank with a cryocooler. First, the heat flow through the liquid is determined by calculating the heat flow over the MLI, the heat leak, the heat from a device mixing the fluid and the heat-conducting through the propellant tank support struts. More information on these relations is given further in this chapter. This heat has to be compensated by the cooling power of the cryocooler installed. If the cooler compensates for the heat flow to the liquid, there is no heat to raise the temperature and there is no boil-off. A similar approach is used by Haberbusch [45]. In this paper, however, there is more focus on the total system mass. It is analysed whether a higher cooling power, which allows the heat leak to be bigger, can be lighter than a lower cooling power with a lower heat leak. Although it remains unclear whether the additional mass due to the power demand growth is taken into account, it is concluded that adding a cooler actively cooling the wall reduces the overall mass for a ZBO system.
A more extensive analysis using the method introduced by Plachta is performed by Chai and Wilhite [33]. In this analysis, a propellant depot in LEO is simulated. The relations for the heat leaks are taken from [24] and are scaled by using mass ratios of the propellant. The boil-off is determined by analysing the heat leak and with the latent heat of evaporation, it is determined how much propellant boils-off.

Perczynski aims to design the Earth Departure Stage with only passive insulation. The boil-off rate achieved is not zero. Sun [34] aims to design a ZBO propellant tank by only using passive insulation. The method used is to calculate the heat capacity of the fluid by multiplying the fluid mass with the specific heat capacity and the temperature difference between the fluid and the saturation temperature. This heat capacity divided over the intended mission duration gives the allowable heat leak to the fluid. By adding insulation to the system until the heat leak is lower than this threshold value, a ZBO system is designed.
The passive insulation used to consist of a sun shield and spray-on foam insulation. The propellant tank is aimed with the smallest side to the Sun, reducing the area catching radiation. It is concluded in the paper, that ZBO can be attained with passive insulation only. However, from the method used it can be seen that it is important to know the mission duration. When the cryogenic liquid heats, a stratified layer forms on top of the liquid. In this stratified region, the temperature can be considered uniform and equal to the saturation temperature. The stratified region grows over time if nothing is done. The growth of this region was numerically investigated by Liu [44]. With the Rayleigh number and the characteristic length, it could be investigated what the coefficient of convection would be. This coefficient was used to determine the heat flow to the liquid.
The mass leaving the boundary layer was assumed to enter the stratified layer. This mass minus the boil-off mass is the growth of the stratified layer. It was identified that the higher the fill height of the propellant tank, the faster the growth and the thicker the stratified layer became. It appears that the ullage

temperature is of little influence on the growth of the stratified layer. When the heat flux increases, also the Rayleigh number increases and the mass flow rate from the boundary layer as well. This results in a more rapid increase in the stratified layer thickness. The liquid properties used in this work are taken from the NIST database and the liquid properties are continuously updated.

One of the most important parameters influencing boil-off is the heat flux. For every model it is important to know what the heat flux modelled is, because this is of major influence on the result. Also, with the heat flux known, it is easier to trade-off options on what is effective to mitigate boil-off and what is not.
Simulating in gravity conditions has as an effect that natural convection due to buoyancy forces is not negligible. In zero gravity or micro-gravity, natural convection is weak and thus for a proper mixture of the cryogenic fluid, forced convection has to take place [46].
The saturation temperature is depending on the saturation pressure. With a higher pressure, the saturation temperature rises and thus reduces the boil-off rate. However, this overpressure also needs to be vented from the tank by a valve, to prevent the shell to rupture. By venting the boil-off gas and thus reducing the pressure, the saturation temperature lowers again and the boil-off rate increases. For most models, however, it is assumed that saturation temperature is constant and pressure is only important to determine what needs to be vented.

From the papers analysed, it can be concluded that there are different methods to simulate propellant tank boil-off. However, all methods have in common that to reduce the boil-off, the heat flow to the propellant tank is reduced. The biggest difference between the papers is how liquid hydrogen behaviour is simulated. Little information is given on how the different papers simulate the heat propagation through the insulation material and the propellant tank wall. Most of the papers analyse indicate that the thermodynamic relations described in section 2.2 are used to calculate the heat flow for the propellant tank structure.

2.2. Forms of heat propagation

To understand boil-off and how to mitigate it, it is key to understand how heat propagates into a structure and specifically into a spacecraft propellant tank. In this section, the three forms of heat propagation will be elaborated upon. First, thermal conduction will be explained, then radiation and at last, convection. The basic heat transfer theory from this chapter is taken from [47] and the course AE4S01 Thermal Rocket Propulsion[1].

2.2.1. Conduction

Conduction is heat transfer by diffusion and collision of molecules in gasses and liquid and heat transfer by free electrons displacing energy in solids[2]. Equation 2.1 gives the conductive heat flux $\mathrm{q_{conduction}}$, which multiplied with the surface area of the material A gives the heat flow. In equation 2.1, T_1 and T_2 are the temperature of the source and the receiver respectively. If the temperature of the source is higher, it will lose heat to the receiver and therefore the conductive heat transfer will be negative. L is the distance over which conductance takes place. k is the coefficient of conduction, which is a measure on how well a material propagates heat.

$$\mathrm{q_{conduction}} = -\mathrm{k}\frac{(\mathrm{T_1} - \mathrm{T_2})}{\mathrm{L}} \tag{2.1}$$

From the conduction equation it becomes apparent that the reduce conduction, the thermal conductivity of the material selected should be small and/or the length of the conductive path should be longer. Insulation materials typically have a low value for their thermal conductivity. Increasing the conductive path length comes with a mass penalty, but can be preferable over using material with a lower thermal conductivity since increasing the thickness of insulation does not increase the complexity of the structure while using different materials can be complex or more expensive.

[1] https://studiegids.tudelft.nl/a101_displayCourse.do?course_id=49304
[2] https://www.sciencedirect.com/topics/engineering/conduction-heat-transfer

Conduction is the dominant heat transfer method while the spacecraft is still on Earth [32]. Heat propagates through the atmosphere by conduction and from the atmosphere to the structure heat is passed on by conduction.

2.2.2. Radiation

Radiation heat transfer is the dominant heat transfer mechanism in space[3]. For conduction and convection, a medium in which the heat can propagate is necessary. Radiation will transfer heat without a medium, thus in the vacuum of space, this mechanism is dominant.
Equation 2.2 gives the radiative heat flux from one object to another. $\mathrm{q}_{\mathrm{radiation}}$ is the radiative heat flux, ϵ is the emissivity of the source, σ_b is the Stefan-Boltzmann constant, A is the surface area, T_1 is the source temperature and T_2 is the receiver temperature. The radiation emitted is in the infrared spectrum.

$$\mathrm{q}_{\mathrm{radiation}} = -\epsilon\sigma_b \left(\mathrm{T}_1{}^4 - \mathrm{T}_2{}^4\right) \tag{2.2}$$

From the radiation equation is can be found that the emissivity of the surface is key in increasing the outward radiation of the spacecraft, while a lower emittance of the internal area reduces the heat flux into the spacecraft. The emittance of a structure can be easily manipulated by using coatings. More information on coatings will follow section 2.4.1.

Next to the radiation equation, which can be used to find the radiative flux from one source to an object, there is another major source of radiation: the Sun. Radiation from the Sun is the major source of heat in interplanetary space missions. The radiation flux from the Sun received by the spacecraft can be scaled over the distance squared. The magnitude of the flux is given by equation 2.14 found in section 2.3.2.

Solar radiation also contains alpha (α^{2+}), beta (β^{-}), and gamma (γ) particles, which cause particle heating. Alpha radiation contains particles with two protons and two neutrons. These particles are quite heavy and therefore, their impact ionises molecules at which heat is released. Beta particles consist of one electron. The mass of a beta particle is two thousand times smaller than an alpha particle and therefore has a lower ionising effect on molecules. However, since the beta particle is smaller and faster, it penetrates deeper in most materials. This property makes it harder to protect anything from beta particles, thus also the cryogenic propellant stored in a spacecraft. Nevertheless, shielding material should be able to block all of these particles. Gamma particles have the highest penetrating capability. However, the mass of a gamma particle is negligible. If a gamma particle collides with a molecule the ionising effect is very low. In figure 2.3 an overview is given of the different particles, their penetrating capability and the effect of collision with a molecule [48]. Since either the effect of collision is low or the penetrating power is low, for the remainder of this research ionising particles are not taken into account. This is supported by the papers analysed, since these don't mention electromagnetic radiation as a source for boil-off.

Particle	*Symbol*	*Mass*	*Penetrating Power*	*Ionizing Power*	*Shielding*
Alpha	α	4 amu	Very Low	Very High	Paper Skin
Beta	β	1/2000 amu	Intermediate	Intermediate	Aluminum
Gamma	γ	0 (energy only)	Very High	Very Low	2 inches lead

Figure 2.3: Different particles emitted by the Sun, penetration characteristics and ionising effect. [48]

Aueron and Thomas [49] did research to the effect of beta and gamma radiation to a cryogenic propellant tank next to a thermal nuclear propulsion system. From this research, it is seen that in the worst case, 30 [W] of radiative power is received by the cryogenic hydrogen. This term could be added to the equations, of a thermal nuclear propulsion system is present.

[3] https://science.nasa.gov/science-news/science-at-nasa/2001/ast21mar_1

2.2.3. Convection

The heat within the liquid and gas will propagate due to convection and conduction. The equation for conduction is already given by equation 2.1. The heat flow due to convection is a mixture of heat transfer between molecules by conduction and heat movement due to the movement of particles. Convection can only take place in fluid or gas and not in solids since molecules in a solid are locked. Convection can be natural or forced convection, the first is caused by buoyancy forces (forces due to density differences between two or more media or within one medium), while the latter is caused by some external mechanism which generates the flow movement, such as a pump. Convection is given by equation 2.3, in which $\mathrm{q_{convection}}$ is the convective heat flux, h is the coefficient of convection, A is the characteristic surface area, and T_1 and T_2 are respectively the source and receiver temperature.

$$\mathrm{q_{convection}} = -\mathrm{h}\left(\mathrm{T}_1 - \mathrm{T}_2\right) \tag{2.3}$$

The coefficient of convection is hard to determine since it depends on many factors of a fluid or gas. To find the coefficient of convection of a medium, the Nusselt number (Nu) can be used. The definition of the Nusselt number is the ratio of convective heat transfer over the conductive heat transfer, as given by equation 2.4.

$$\mathrm{Nu} = \frac{\text{convective heat transfer}}{\text{conductive heat transfer}} = \frac{\mathrm{hL_c}}{\mathrm{k}} \tag{2.4}$$

The Nusselt number is the ratio between convection and conduction, but more is necessary to obtain the coefficient of convection. The Rayleigh number (Ra), equation 2.5, is the ratio of heat-diffusivity over heat-convection within the fluid. Diffusive heat transfer is comparable with conductive heat transfer, the diffusivity of a medium is the conductivity over the density times the specific heat capacity at a constant pressure of the medium, see equation 2.6 [50]. When the Nusselt number is lower than one, diffusive heat transfer dominates convective heat transfer, and when the number is bigger than one convective heat transfer dominates diffusive heat transfer. In the latter case, the fluid velocities will be high due to the heat transfer being dominated by moving fluid.

The Nusselt number is related to the Rayleigh number, which is obtained by using the characteristic length (L_c), the temperature difference between the wall and the liquid/vapour (ΔT), the gravity (g) and the thermal expansion coefficient (α), the thermal diffusivity (β) and the kinematic viscosity (η/ρ) of the medium. The latter three parameters are properties of the medium and therefore, can be taken from a database such as the chemistry web book of NASA [51].

$$\mathrm{Ra} = \frac{\text{convective flux}}{\text{diffusive flux}} = \frac{\rho\beta\Delta T l^3 g}{\eta\alpha} \tag{2.5}$$

$$\alpha = \frac{\mathrm{k}}{\rho \mathrm{C_p}} \tag{2.6}$$

From [52] a relationship, equation 2.7, between the Nusselt number and the Rayleigh number is found for a wide range of Rayleigh numbers. With a Rayleigh number below 10^7, the convection is dominated by diffusion. A Rayleigh number between 10^7 and 10^{10} indicates that the internal flow in the tank is laminar., while a Rayleigh number exceeding 10^{10} indicates that the flow in the propellant tank is turbulent. With equations 2.4 and 2.7, the coefficient of convection is found for a liquid, as a function of the Rayleigh number, the coefficient of conduction of the fluid and the characteristic length of the liquid. This function is equation 2.8 in which the Nusselt number found with 2.7 has to be plugged in. In both the equation for the Rayleigh number and the equation for the Nusselt number, a characteristic length scale is necessary. Normally, the characteristic length is defined as the volume over the surface area of an object. However, for equation 2.7 to be valid, the length scale used is defined as the height of the liquid in the propellant tank from the propellant tank bottom [52]. This characteristic length definition is also used by [44].

$$\begin{aligned} \mathrm{Nu_L} &= 0.642\,(\mathrm{Ra_L})^{1/6} && \mathrm{Ra_L} \leq 10^7 \\ \mathrm{Nu_L} &= 0.167\,(\mathrm{Ra_L})^{1/4} && 10^7 < \mathrm{Ra_L} \leq 10^{10} \\ \mathrm{Nu_L} &= 0.00053\,(\mathrm{Ra_L})^{1/2} && 10^{10} < \mathrm{Ra_L} \leq 5\mathrm{x}10^{13} \end{aligned} \tag{2.7}$$

$$\mathrm{h} = \frac{\mathrm{k}}{\mathrm{L_c}}\mathrm{Nu_L} \tag{2.8}$$

Equation 2.7 is developed for liquids. In [52] another relationship is found between the Rayleigh number and the Nusselt number for a gas/vapour. In equation 2.9 the Rayleigh number of the vapour determines the Nusselt number. This Nusselt number is then put into equation 2.8 to find the coefficient of convection of the vapour.

$$\begin{aligned} \mathrm{Nu_L} &= 4.5 && \mathrm{Ra_L} \leq 10^7 \\ \mathrm{Nu_L} &= 0.08\,(\mathrm{Ra_L})^{1/4} && 10^7 < \mathrm{Ra_L} < 10^{12} \end{aligned} \tag{2.9}$$

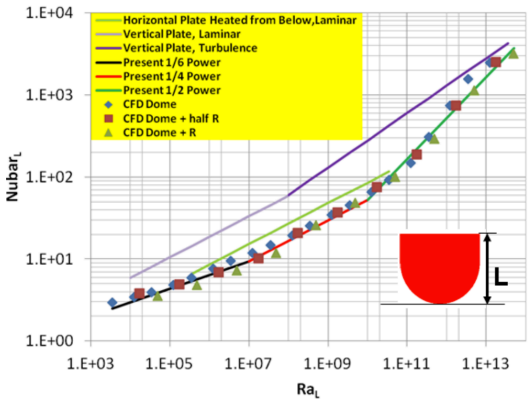

Figure 2.4: Plot of the Nusselt number as a function of the Rayleigh number for the propellant tank liquid (equation 2.7) and accompanying test data [52].

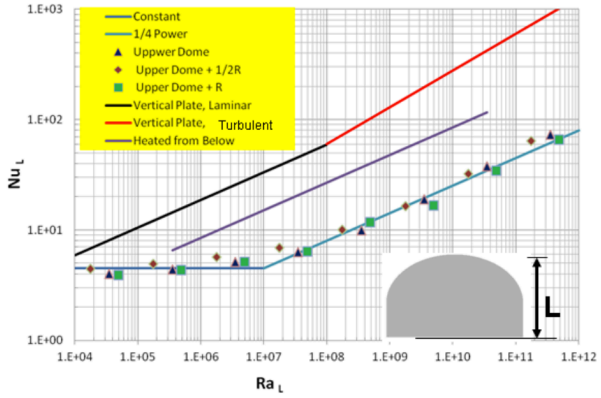

Figure 2.5: Plot of the Nusselt number as a function of the Rayleigh number for the propellant tank vapour (equation 2.9) and accompanying test data [52].

Equation 2.7 is plotted into a graph with test data in figure 2.4 taken from [52]. In this figure, it is seen that the difference between the Computational Fluid Dynamics (CFD[4]) data and the equation formulated has a maximum of about 7.5% for a Rayleigh number close to order of magnitude 10^{12}. The Nusselt number of the vapour as a function of the Rayleigh number is plotted in figure 2.5. It is seen that the maximum deviation of equation 2.9 is 30% for a Rayleigh number of $2 * 10^7$. For higher and lower numbers the deviation of the test data compared to the fitted line is less. Both equations estimate the Nusselt number better than the flat plate relations also plotted in the figures, thus equations 2.7 and 2.9 are considered the best options to find the coefficient of convection for the liquid and the vapour in the propellant tank.

Equations 2.7 and 2.9 do not include the Prandtl number. The Prandtl number of a fluid is defined by the thermal diffusivity over the kinematic viscosity. In the relations of the Nusselt number as a function of the Rayleigh number provided by [53] and [54], the Prandtl number is part of the equation. In the NIST Chemistry Webbook [51] it can be seen, that for the temperature range 20 [K] and higher, the Prandtl number is close to unity and therefore does not change the coefficient of convection over the temperature range given.

The three modes of heat transfer have been treated in this section. Conduction, radiation and convection will together determine how much heat flows to the liquid in the propellant tank. In the next section, the spacecraft thermal environment will be set-out.

[4] https://www.sciencedirect.com/topics/materials-science/computational-fluid-dynamics

2.3. Spacecraft thermal environment

In all methods of heat propagation, the temperature of one object compared to another is important. This section will present the details of the thermal environment of a spacecraft. The internal and external environment will be dealt with separately since both have a very different impact on the spacecraft. The internal environment is defined as everything within the spacecraft structure, the external environment is everything excluding the structure.

2.3.1. Internal environment

The internal environment of a spacecraft can be challenging in mitigating boil-off. Different components of the spacecraft need to be stored at different temperatures since they have a different operating temperature. In table 2.2 the required operating temperatures of different spacecraft components is given.

Table 2.2: Typical operating temperatures of spacecraft subsystems. [55]

system	operating temperature range [K]
digital electronics	273-323
analog electronics	273-313
batteries	283-293
infrared detectors	4-100
solid-state particle detectors	238-273
solar panels	173-398

For launcher systems, an additional challenge comes with managing the propulsion system temperatures. Whereas the adiabatic flame temperature of the hydrogen-oxygen combination is hot, from around 3000 [K] and warmer[5], the storage temperature for the propellant can be very low, e.g. around 20 [K] for LH2. The internal configuration of systems is very important in spacecraft thermal design. In the thermal design, heat sink paths should be taken into account. Nevertheless, finding the optimal internal configuration of systems is not the scope of this thesis.

The internal environment transfers heat for the majority via the insulation of the spacecraft propellant tank. Next to the heat transfer through the insulation material, there are also secondary heat leaks into the propellant tank. To model these heat leaks, a detailed design is necessary to find the magnitudes of these heat leaks. During the preliminary design phase, too little is known to find the exact heat leaks. Therefore, in [24] equations are set to size these heat leaks thereby taking into account the size of the propellant tank and the desired temperatures. The total heat load to the propellant ($\mathrm{Q_{net}}$) is given by adding all the different components, as given by equation 2.10.

The net heat load on the fluid is necessary to determine the temperature rise and boil-off. From equation 2.10, the heat loads through the insulation ($\mathrm{Q_{MLI}}$) and the heat loads of the cryocooler ($\mathrm{Q_{cryocooler}}$ and $\mathrm{Q_{parasitic}}$) will be elaborated upon in a next section. The other heat loads will be explained here. The heat leak through support struts holding the tank ($\mathrm{Q_{struts}}$) is a major heat source. To reduce the impact of the struts, special struts are necessary. These struts are called PODS (Passive Orbital Disconnect Struts). In figure 2.6 it is illustrated what the components are of a PODS. The heat leak of the PODS is found by using equation 2.11, which is derived in [23] and presented in [24]. This equation is only valid for liquid hydrogen. The factor C_1 is a design margin factor.

$$\begin{aligned}\mathrm{Q_{net}} =& \mathrm{Q_{MLI}} + \mathrm{Q_{struts}} + \mathrm{Q_{penetrations}} + \mathrm{Q_{mixer}} \\ &- \mathrm{Q_{cryocooler}} + \mathrm{Q_{parasitic}}\end{aligned} \tag{2.10}$$

$$\mathrm{Q_{struts}} = C_1 * 0.021 * \left(\frac{\mathrm{M_{tank}} + \mathrm{M_{propellant}}}{560}\right) * \left(\frac{\mathrm{T_{hot}} - \mathrm{T_{cold}}}{68.6 - 23.6}\right) \tag{2.11}$$

[5] http://www.braeunig.us/space/comb-OH.htm

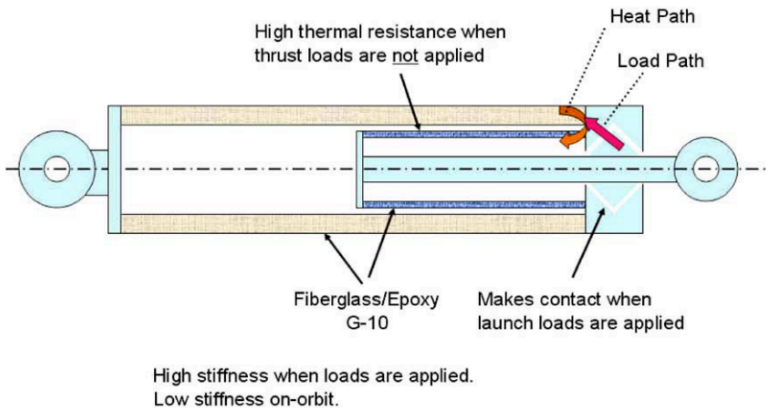

Figure 2.6: Illustration of the components of a Passive Orbital Disconnect Strut. [24]

Although the propellant tank is isolated as proper as possible, there is still a term which estimates the heat leak through insulation penetrations ($\mathrm{Q_{penetrations}}$). This term can be estimated by using equation 2.12, in which C_1 is again a design margin factor.

$$\mathrm{Q_{penetrations}} = C_1 * \left(0.0025 * (\mathrm{T_{hot}} - \mathrm{T_{cold}}) * \sqrt{\mathrm{V_{tank}}}\right) \tag{2.12}$$

Equation 2.12 is derived in [24] by analysing the heat leak data of a 3,500 [kg] LH2 propellant tank. By using the expected hot ($\mathrm{T_{hot}}$) and cold side temperature ($\mathrm{T_{cold}}$) and propellant tank volume, the heat leak found in [18] is scaled to the specific propellant tank.

$$\mathrm{Q_{mixer}} = C_1 * \left(\frac{\mathrm{V_{tank}}}{1.4}\right) * 3.5 * C_2 * \frac{1}{C_3} \tag{2.13}$$

The heat-dissipating from a device mixing the propellant to keep the propellant homogeneous and unstratified ($\mathrm{Q_{mixer}}$) is estimated by using equation 2.13. The equation is derived in [22] by using test data. A 3.5 [W] mixer was used to mix a propellant tank with a 1.4 [m^3] volume, a duty cycle of 90 minutes and a mixture time of 30 seconds. The placeholder coefficient C_2 is set for the duty cycle, while C_1 is used to state the mixture time. Coefficient C_3 is the buoyancy factor, which has a value of 42 for LH2 [24].

2.3.2. External environment

For solar radiation, it is important whether the spacecraft is in eclipse or not and how far the spacecraft is from the Sun. Radiation power at one astronomical unit is $(\dot{q}_{sol})_{R=1} = 1414\,W/m^2$. This scales over the distance squared and results in equation 2.14 for the solar radiation flux, as given by [56].

$$(q_{sol})_R = \frac{(q_{sol})_{R=1}}{R^2} \tag{2.14}$$

Next to the direct heat from the Sun, the heat from the Sun also reflects on third bodies to the spacecraft. Warming up of the satellite due to this phenomenon is called albedo heating. The intensity of albedo heating is not uniform and is depending on the location of the spacecraft to the third body and the Sun. The location of the spacecraft to the Sun and a planet is given by the solar zenith angle ξ and the distance of the spacecraft to the planet. The solar zenith angle is the angle between the vector opposite of the gravity vector and the vector pointing to the Sun. The distance of the spacecraft to the planet is used to determine the form factor defined by $\left(\frac{r_{pla}}{r_{pla}+h}\right)$ where r_{pla} is the radius of the planet and h is the altitude of the satellite above the surface of the planet. It is possible to calculate the simplified albedo flux by assuming a constant for the amount of solar radiation reflected by a varying atmosphere of a planet. This constant ρ is one minus the absorptivity α of the respective planet ($\rho_b = 1 - \alpha$) and

is called the planetary bond albedo. The albedo heating is quantified by equation 2.15 [56]. The bond albedo for planets in Earth's solar system can be found in [57].

$$(q_{alb})_h = (q_{sol})_{R_{planet}} \rho_b \left(\frac{r_{pla}}{r_{pla} + h} \right)^2 \cos \xi \tag{2.15}$$

A planet emits not only the albedo radiation but also infrared radiation originating from the planet itself. The infrared radiation emitted by a planet should be equal to the heat received by the planet, otherwise, the planet is not in thermal equilibrium and will heat up or cool down. Given this fact, the relation is given by equation 2.16 is derived in [56].

$$q_{pla} = \frac{q_{sol}(1 - \rho_b)}{4} \tag{2.16}$$

For a spacecraft at distance h, the received heat flux due to infrared radiation of a planet and the form factor is given by Equation 2.17.

$$(q_{pla})_h = \frac{q_{sol}(1 - \rho)}{4} \left(\frac{r_{pla}}{r_{pla} + h} \right)^2 \tag{2.17}$$

In equation 2.17 the form factor can be omitted if the average infrared radiation from a planet to the spacecraft is to be calculated.Due to Kirchhoff's law [58], which states that the total emissivity of an object becomes the absorptivity of that object times the emissivity of a perfect black body at a given temperature, and the fact that heat from the planet is radiated in the IR spectrum, the heat flux absorbed by the spacecraft is calculated with the emissivity instead of the absorptivity [55]. For the majority of long-duration space missions, the planetary albedo and infrared flux will play a minor role. In both equation 2.17 and 2.15 the form factor $\left(\frac{r_{pla}}{r_{pla}+h} \right)^2$ is present, with h being the altitude of the spacecraft above the planetary body. Already if a space mission crosses a Geosynchronous Equatorial Orbit (GEO)(also called Geostationary Orbit (GSO), which is $h \approx 36,000[km]$, the value of the form factor is 0.022. This results in only 2.2 % of the radiation from the planetary body reaching the spacecraft.

2.4. Methods to reduce heat flow

Propellant tank designs which limit the heat to the propellant are key to access space beyond the Moon. These propellant tank designs can contain different measures to limit the heat flow. The measures are divided in active and passive measures. Active cooling devices use energy to remove heat from the propellant tank system. Passive measures do not require any energy to function.

Insulation limited to passive insulation can be most mass-efficient for short-duration missions, but boil-off can make these designs heavier than ZBO propellant tank for long-duration missions. This trade is illustrated in figure 2.7 [18].

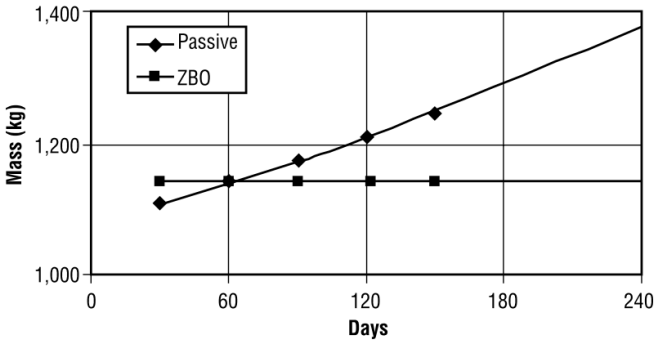

Figure 2.7: Trend-line on using only passive insulation or active insulation over mission duration. [18]

2.4.1. Passive methods

Varies passive insulation methods exist, such as Multi-Layer-Insulation (MLI), Spray-on Foam Insulation (SOFI) and radiation shields. Passive systems can be subdivided into two categories. The first are the measures which limit the heat absorption by and increase the emissivity of the propellant tank. This category functions by reducing the net radiation heat flow. The second category is focused on limiting heat propagation by conduction.

Radiation heat flow reduction

An effective way to reduce radiation heat flow to the propellant tank is by insulating the propellant tank with MLI. MLI consists of an arbitrary amount of layers of thin foil, which are separated from each other by spacers. Since the biggest part of the foil does not touch another layer, the heat transfer within the MLI should be dominated by radiation. However, from [59] it is found that thermal conductance through MLI via the spacers is dominant. The emissivity of the foil is of secondary importance compared to the conductivity of the spacers. The foil has a very low emissivity value, thus per layer of foil, most of the heat flux does not transfer to the next layer by radiation. Since the radiation from one layer to another is low, the conductance due to fabrication imperfections becomes dominant in most thermal environments.

$$\dot{q}_{\mathrm{MLI}} = \dot{q}_{\text{solid conduction}} + \dot{q}_{\text{gas conduction}} + \dot{q}_{\text{radiation}}$$

From [60] comes the modified Lockheed equation, a revision on the equation developed by Lockheed Martin to estimate the performance of MLI. The equation does not only take the radiation between the different blankets into account but does also accommodate a term for conductance through the spacers and conductance through the residual gas between the blankets. Because these elements are included in the equation, the equation provides a realistic estimate on the heat flow over the MLI. The Lockheed equation is given by equation 2.18 in its general form. The values for A, n, B, C and m were obtained from [60] and result in equation 2.19, which is called the modified Lockheed equation. In section 4.1 the verification is done on the relationship given.

$$\begin{aligned} q_{\mathrm{total}} =& A T_{\mathrm{m}} (N*)^{n} \left(T_{\mathrm{H}} - T_{\mathrm{C}}\right) / N_{\mathrm{s}} \\ &+ B \epsilon \sigma_{B} \left({T_{\mathrm{H}}}^{4.67} - {T_{\mathrm{C}}}^{4.67}\right) / N_{\mathrm{s}} + C P \left({T_{\mathrm{H}}}^{m+1} - {T_{\mathrm{C}}}^{m+1}\right) / N_{\mathrm{s}} \end{aligned} \tag{2.18}$$

$$\begin{aligned} q_{\mathrm{total}} =& 2.40 \times 10^{-4} [0.017 + 7 \times 10^{-6} (800 - T) + 0.0228 \ln(T)] (N*)^{2.63} \left(T_{\mathrm{H}} - T_{\mathrm{C}}\right) / N_{\mathrm{s}} \\ &+ 4.944 \times 10^{-10} \epsilon \sigma_{B} \left({T_{\mathrm{H}}}^{4.67} - {T_{\mathrm{C}}}^{4.67}\right) / N_{\mathrm{s}} + 1.46 \times 10^{4} P \left({T_{\mathrm{H}}}^{0.52} - {T_{\mathrm{C}}}^{0.52}\right) / N_{\mathrm{s}} \end{aligned} \tag{2.19}$$

The (modified) Lockheed equation is not the only method to find the heat flow through MLI. By McIntosh [61] another approach is used. This approach calculates the heat flow from layer to layer, which is different than the Lockheed equation which gives the heat flow over the complete set of MLI.

$$q = \sigma \left(T_{\mathrm{h}}^{4} - T_{\varepsilon}^{4}\right) / \left[\left(1/\varepsilon_{\mathrm{h}} + 1/\varepsilon_{\mathrm{c}} - 1\right)\right] + C G P \alpha \left(T_{\mathrm{h}} - T_{\mathrm{c}}\right) + C_{\mathrm{s}} f_{\mathrm{k}} \tag{2.20}$$

Hastings [18] investigated the applicability of both equations on MLI test specimens. From the analysis performed, it is concluded that both models give similar results to the heat flow from the first to the last layer of MLI.

In [62] a relationship has been derived from the modified Lockheed equation to find the optimum layer density as a function of hot- and cold-side temperatures. From figure 2.8, the optimum layer density can be found as a function of the hot- and cold-side temperatures for Perforated Double Aluminized Mylar and Dacron Net MLI. This value can be plugged in into equation 2.19 and by plotting q_{total} as a function of the number of layers MLI N, the optimum amount of layers can be found as well.

The emissivity of the layers of MLI is found in [59] as a function of the temperature, which is $\epsilon = 6.8 \times 10^{-4} T^{0.67}$. This relation is already taken into account for in the (modified) Lockheed equation and the relation is visible in the radiative term as a power 4.67 instead of power to the fourth as presented in the normal radiation equation 2.2.

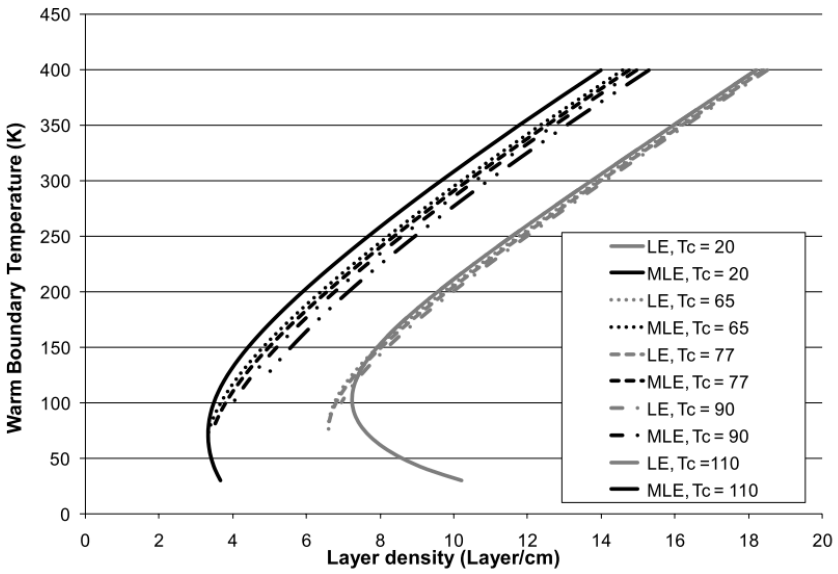

Figure 2.8: Optimum layer density as a function of hot- and cold-side temperatures of MLI. [62]

Next to reducing radiative heat flow by using MLI, a heat shield can also be used to reduce the heat flux to the propellant tank. A heat shield is a layer of material which is not physically connected to the propellant tank. This layer of material absorbs the incoming heat flux and emits the heat further.

By using material or coating with a maximum emissivity over absorptivity ratio, the heat flux which reaches the propellant tank is reduced. From the spacecraft coatings overview found in [63], it is found that the optimum coating is depending on the surface material required for the spacecraft. In figure 2.9 an overview is given with coating categories and their typical emissivity and absorptivity range. A more detailed list of coatings and surface finishes is given in [64].

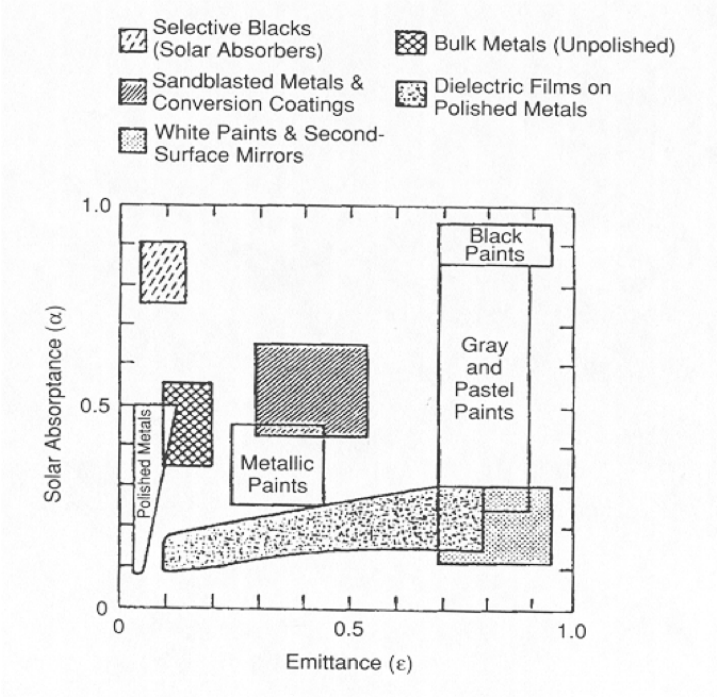

Figure 2.9: Overview of coatings with their emissivity and absorptivity. Adapted from [63]

Conduction heat flow reduction
Reducing the conductive heat flow is a traditional way of reducing the heat flux to a propellant tank. The Centaur upper stage uses SOFI on the propellant tanks to reduce the heat flux to the tank [32]. By analysing equation 2.1 it can be found that to decrease the conduction, either the coefficient of conduction k should decrease, or the conductive pathway L should increase. Both suggested methods are applied in isolating propellant tanks.

Spray-on Foam Insulation is one of the most common methods to isolate a propellant tank passively. During the Apollo program, SOFI was already used, after which the performance has been analysed and improved for the Centaur-upper stage and successive launch vehicles. SOFI is a foam which contains holes within the structure. These holes contain either gas or are (close-to) vacuum. Due to the presence of these holes, the conductive area within the foam is reduced. This results in a low thermal conductivity of these foams. The low conductivity is depending on several factors. Degassing of the foam is important. From [65] it appears that the performance of the SOFI increases in a vacuum, while the conductivity increases by more than 300% in a helium-rich environment due to the helium filling the otherwise empty holes within the SOFI.

Analysis done by [65] reveals that moisture in the SOFI is beneficial for the thermal conductivity of the foam. However, the mass also increases due to the absorption of moisture by the foam. It is expected that unprotected SOFI on a launch vehicle or spacecraft will gain a significant amount of mass by absorption of moisture. A protective layer is therefore necessary.

In [32] is stated that for long-duration space missions, Multi-Layer Insulation will be required to mitigate the heat flow into the propellant tank. However, SOFI will be required to reduce the heat flow to the propellant tank during the early stages of the space mission, such as the ground stage. During the ground stage, SOFI reduces the mass of the spacecraft by mitigating icing phenomena on the outer shell of the spacecraft [66].

2.4.2. Active methods - cryocoolers

From the overview given in chapter 1 it can be seen that most research is done to ZBO. From the papers analysed in section 2.1 it comes that except for [34], all ZBO designs incorporate an active component to mitigate boil-off. Active systems used in space to cool liquids are cryocoolers, actively cooled sun shields or adiabatic-demagnetisation refrigerators (ADRs). The latter is used for instruments which are cooled to near zero (in the order of 10^{-3}) kelvin, but the cooling capacity order of microwatts [67] is not sufficient for propellant cooling. Therefore, ADR's are not considered for this thesis. The first, cryocoolers, are named "necessary evil" in [68], due to the added complexity they give to the spacecraft architecture. In ESA efforts to make long term cryogenic storage possible, the focus is on achieving ZBO. The goal is to find a combination of active and passive components to mitigate the liquid and vapour losses due to boil-off of cryogenic liquid.[6]

Cryocoolers are active cooling systems which remove heat from a system. For long-duration space missions, cryocoolers are the only option because cryocoolers do not consume but reuse liquid. Cryocoolers are available in different thermodynamic cycles. From figure 2.10 [7] it is observed for the 20 [K] regime (the saturation temperature of LH2) that the Joule-Thomson cycle and the Stirling cycle coolers are usable.

The Stirling cycle is a regenerative cycle using isochoric and isothermal processes. The fluid flow in the heat exchanger changes in direction during the different stages of the regenerative cycle. Regenerative cryocoolers are compact because only one channel for the cool and hot flow is used. Since there is only one channel, the cycle can not cool continuously. The Stirling cycle has relatively low complexity and therefore a high reliability and life span expectation, exceeding 10 years .
Joule-Thomson cycle uses isothermal compression, isenthalpic cooldown, adiabatic expansion, and isenthalpic heat up. The cycle is recuperative, which means there are different flow channels for the stream up and down and heat is exchanged between the two channels. A Joule-Thomson cycle cools

[6] https://www.esa.int/Enabling_Support/Space_Engineering_Technology/Shaping_the_Future/Zero-Boil_Off_Propulsion_System_Feasibility_Demonstration

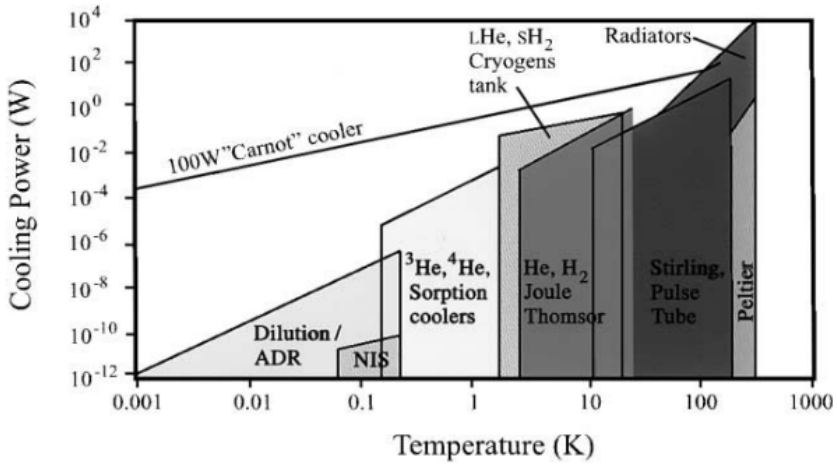

Figure 2.10: Cooling ranges of space coolers as function of temperature [7].

continuously. Heat is absorbed by the evaporation of a cooling agent and heat is rejected by expansion of the same agent [69] [70]. In figure 2.11 the two thermodynamic cycles of the Stirling and Joule-Thomson coolers are graphically presented.

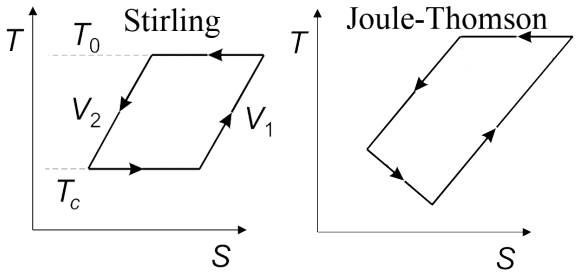

Figure 2.11: Representation of the thermodynamic cycles of the Stirling cycle and the Joule-Thomson cycle.

For the remainder of this research, the type of cryocooler is not considered since the implementation differences and added complexity which using cryocoolers give, are not considered within the scope of this research.

In [71] an extensive survey is conducted to find a relationship between cryocooler power and the mass. The power required can be calculated by using equations 2.21, 2.22 and 2.23, which together give Equation 2.24. The $\mathrm{COP}_{\mathrm{Cooler}}$ is the coefficient of performance of the cooler, a coefficient which gives how much of the power given as input (P_{in}) is converted into cooling power (Q_C). The Carnot COP ($\mathrm{COP}_{\mathrm{Carnot}}$) is the theoretical maximum performance of any cooler and is therefore used to give the efficiency η of the cooler since a cooler functioning at the Carnot COP is functioning at 100% efficiency.

$$\mathrm{COP}_{\mathrm{Cooler}} = \frac{Q_C}{P_{in}} \tag{2.21}$$

$$\mathrm{COP}_{\mathrm{Carnot}} = \frac{T_{cold}}{T_{hot} - T_{cold}} \tag{2.22}$$

$$\eta = \frac{\mathrm{COP}_{\mathrm{Cooler}}}{\mathrm{COP}_{\mathrm{Carnot}}} \tag{2.23}$$

$$P_{in} = \frac{Q_C}{\eta} \frac{T_{hot} - T_{cold}}{T_{cold}} \tag{2.24}$$

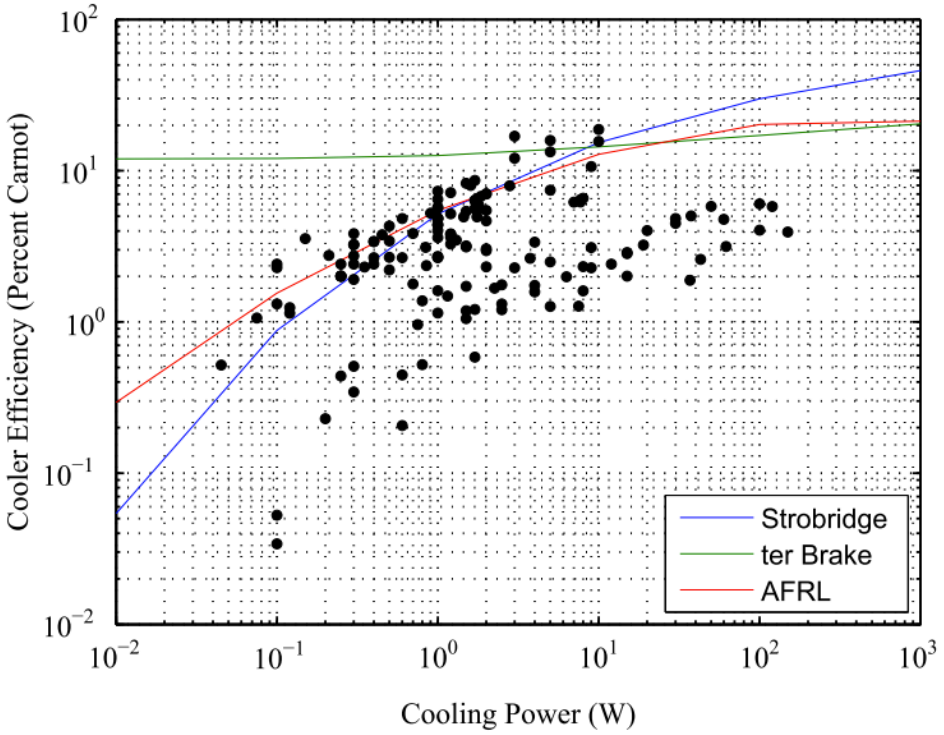

Figure 2.12: Efficiency of a 20K cryocooler as percentage of Carnot efficiency. [33]

Figure 2.12 gives efficiency as a percentage of the Carnot efficiency as a function of the cooling power. In the figure, existing cryocooler data (black dots) and three efficiency relations found by different researchers are plotted. The three relations are the Ter Brake's Limit [72], the AF correlation [71] and the Strobridge's Correlation [73]. These three relations are found from extensive cryocooler surveys performed by the researchers. The efficiency compared to the Carnot efficiency by Ter Brake is based on a survey of 80 [K] cryocoolers, but since the Carnot efficiency (see equation 2.22) will compensate for the temperature difference, it is still taken as a valid relation.
The efficiency of a cryocooler given the cooling power can be found from equation 2.25 given in [33]. This relation gives the minimum of the Ter Brake's Limit and the AF correlation. The Strobridge's Correlation is not used in this equation, because the survey is outdated (1974).

$$\log_{10}(\eta) = \min \begin{cases} -1.26281 + 0.45936\left(\log_{10} Q_C\right) - 0.08743\left(\log_{10} Q_C\right)^2 \\ -0.92237 + 0.07763\left(\log_{10}\left(1 + Q_C\right)\right) \end{cases} \tag{2.25}$$

The cryocooler subtracts heat from the system, but when the cryocooler is disabled, it leaks heat to the system. This heat is called parasitic heat. From [24] equation 2.26 is found. For this thesis a continuous active cryocooler is assumed, thus the parasitic heat leak due to a turned-off cryocooler is not taken into account. A placeholder for future research is however used in the tool to be developed.

$$Q_{\text{parasitic}} = 0.75 * C_1 * C_2 * \left(\left(T_{\text{hot}}^2 - T_{\text{cold}}^2\right) P_{\text{input}}^{\frac{1}{3}} * \left(\frac{1}{5.5 * 10^5}\right)\right) \tag{2.26}$$

Using cryocoolers has a purpose to reduce the boil-off rate if not eliminate it. This does not only cost input power, as is seen from the related equations. A cryocooler also adds mass to the system. By doing an extensive survey on existing cryocoolers in [71] a relationship is found between the cryocooler input power and the mass of the cryocooler. Equation 2.27 is valid for a 20 [K] cryocooler.

$$\mathrm{M}_{\text{cooler}} = 0.1422\mathrm{P}_{\text{in}}{}^{0.905} \tag{2.27}$$

The power demand of the cryocooler and infrastructure required to use a cryocooler also increase the mass of the propellant tank. The additional power results in an increase in the solar array area, which results in a mass increase. In [28] table 2.3 is presented, in which the additional mass as a function

of cooling power is given. The data presented originates from a survey conducted by [74]. The summation of the different penalties and adding those to equation 2.27 gives equation 2.28 to estimate the total mass of a cryocooler as function of the input power.

Vehicle Subsystem	System Penalty (kg/W)
Structure & Heat Transport	0.097
Radiator	0.071
Cold Plumbing & Insulation	0.025
Cables & Misc	0.032
Local Subtotal	*0.225*
Power System	0.100
Total	0.325

Table 2.3: Additional required components and the mass per watt of cooling power for a cryocooler. [28]

$$\mathrm{M_{cooler\text{-}total}} = 0.1422\mathrm{P_{in}}^{0.905} + 0.325\mathrm{P_{in}} \tag{2.28}$$

In equation 2.28 the $\mathrm{M_{cooler-total}}$ is the total cooler mass in [kg] and $\mathrm{P_{in}}$ is the input power in [W] of the cryocooler (thus not the cooling power $\mathrm{Q_{cryocooler}}$). The constant 0.1422 has the unit $[kg/W^{0.095}]$, while the 0.325 has the unit $[kg/W]$.

The state of the art cryocooler is the 20K-20W cryocooler developed by Creare [14]. Cryocoolers producing more than 100 [W] of cooling power at 20 [K] are not developed yet.

2.4.3. Pressure and temperature in the propellant tank

The initial pressure in the propellant tank is important since the initial pressure determines the initial saturation temperature of the liquid in the propellant tank, as can be seen from equation 2.29.

$$\mathrm{T_{saturation}} = \left(\frac{1}{\mathrm{T_0}} - \frac{\mathrm{R_G} \ln \frac{\mathrm{p}}{\mathrm{p_0}}}{\mathrm{L_{vap}}} \right)^{-1} \tag{2.29}$$

The vapour in the propellant tank also absorbs heat. This heat absorption raises the temperature of the vapour, which results in an expansion of the vapour. However, since the volume of the propellant tank is constrained, the vapour cannot expand and the pressure grows. The pressure growth due to heating of the vapour is given by equation 2.30. This equation originates from the ideal gas law and uses the assumption that the temperature increases isochorically.

$$\left(\frac{\mathrm{dp_{vapour}}}{\mathrm{dt}} \right)_{\mathrm{expansion}} = \frac{\mathrm{m_{vapour}} \mathrm{R_G} \mathrm{dT}}{m \mathrm{V_{vapour}}} \tag{2.30}$$

If the liquid propellant in the tank boils, this increases the vapour mass. This also results in the pressure raising. Equation 2.31 is derived from the ideal gas theory and gives the pressure raise due to boil-off. For a more in depth derivation, see [75].

$$\frac{dp_v}{dt} = \left(\frac{L}{V c_v T_s} + \frac{R_G T_s (\rho_l - \rho_s)}{L m \rho_l V} - \frac{R_G T_s (\rho_l - \rho_s) \rho_s}{m \rho_l p_v V} \right) Q + \left(\frac{R_G T_s (\rho_l - \rho_s) \rho_l}{m \rho_l p_v V} - \frac{\rho_l - \rho_s}{\rho_l V} + \frac{(\rho_l - \rho_s) \rho_s L}{\rho_l p_v V} - \frac{(\rho_l - \rho_s) \rho_l L}{\rho_l p_v V} \right) Q \tag{2.31}$$

For hydrogen and helium $\frac{\rho_v}{\rho_l} << 1$, thus the terms $-\frac{R_G T_s \rho_s}{L m \rho_l V}$, $-\frac{R_G T_s (\rho_l - \rho_s) \rho_s}{m \rho_l p_v V}$, $-\frac{\rho_s}{\rho_l V}$, and $\frac{(\rho_l - \rho_s) \rho_s L}{\rho_l p_v V}$ can be neglected. Also, the specific heat and pressure work terms are less significant, since it is assumed all

heat is used for phase transition, thus keeping the temperature and volume constant. This omits the terms $\frac{LQ}{Vc_vT_s}$, $\frac{R_GT_s(\rho_l-\rho_s)\rho_l}{m\rho_lp_vV}$ and $-\frac{(\rho_l-\rho_s)\rho_lL}{\rho_lp_vV}$. The relationship becomes equation 2.32.

$$\left(\frac{\mathrm{dp_v}}{\mathrm{dt}}\right)_{\mathrm{boil-off}} = \frac{\mathrm{R_G T_{saturation}}}{\mathrm{L_{vap}} m \mathrm{V}} \mathrm{Q} \tag{2.32}$$

The raise in pressure has to be prevented at a maximum pressure. This is done with a pressure valve, which releases vapour when the maximum pressure is reached. In [28] it is already suggested that releasing vapour from the propellant tank can increase the cryogenic insulation performance. By venting (releasing) propellant, energy is released from the system by work. This results in a reduction of the temperature. The reduction of the temperature is an adiabatic process. In an adiabatic process no heat is lost to the surroundings, energy change is only due to work[7]. In [9] is stated that venting LH2 is very efficient due to the relatively high change in enthalpy achieved by expanding LH2. A high enthalpy change results in a high change in temperature since enthalpy and temperature for an adiabatic process is given by equation 2.33. The relation of the pressure change and enthalpy by work is given by equation 2.34.

$$dH = \mathrm{mC_p dT} \tag{2.33}$$

$$dH = \mathrm{Vdp} \tag{2.34}$$

Combining the two equations given for enthalpy results in equation 2.35, which gives the change in temperature of the remaining vapour by releasing gas. In equation 2.35 the specific heat coefficient under constant pressure of the vapour mixture ($\mathrm{C_{P_{mix}}}$) can be calculated by using the ratio of gas masses contained by the vapour mixture[8]. The vapour mixture contains the initial pressurant gas and the boil-off gas, which can be used in equation 2.36 to calculate the coefficient if the masses are known.

$$\mathrm{dT_{venting}} = \frac{V_{vapour}\mathrm{dp_{vapour}}}{\mathrm{m_{vapour}C_{P_{mix}}}} \tag{2.35}$$

$$\mathrm{C_{P_{mix}}} = \frac{\mathrm{m_{gas\ 1}} * \mathrm{C_{P_{gas\ 1}}} + \mathrm{m_{gas\ 2}} * \mathrm{C_{P_{gas\ 2}}}}{\mathrm{m_{mix}}} \tag{2.36}$$

Next to the cooling of the vapour in the propellant tank by releasing vapour from the propellant tank, regenerative cooling utilizing the hydrogen boil-off vapour is possible. By Isselhorst in [76], it is demonstrated that the vapour which needs to be vented can be expanded to reduce the temperature. In the mentioned paper, the cooled hydrogen is used to cool the engine, but in theory, also the propellant tank itself could be cooled by the expanded hydrogen. Due to time constraints, this option is not further investigated.

2.5. Heat flow reduction design option tree

The design options discussed in section 2.4.1 and section 2.4.2 and the pressure and thermal conditions discussed in section 2.4.3 can be summarised in a design option tree. In the design options tree, the options not considered for this research are excluded. The design options are given in figure 2.13. The options define the search space to find a lighter configuration for a given propellant tank.

The options given in figure 2.13 are also used in literature. [33] and [34] focus on using MLI for mitigating the heat flow in the propellant tank. [24] focuses on mitigating boil-off by adding SOFI. By Fesmire [32] it is concluded that SOFI is required to mitigate ice forming on the spacecraft during the ground phase of a space mission. MLI is effective when heat transfer occurs via radiation [77], but is ineffective against conduction heat transfer. SOFI functions effectively when conduction is dominant, such as during the ground stage [32], but is less effective against radiation. Therefore, since different environments with different dominant heat transfer mechanisms are encountered during the lifespan of a spacecraft, combining these passive insulation methods is recommended.

[7] https://www.nuclear-power.net/nuclear-engineering/thermodynamics/thermodynamic-processes/adiabatic-process/adiabatic-expansion-adiabatic-compression/

[8] https://thermtest.com/thermal-resources/rule-of-mixtures

Two options given in the design options tree are reducing the storage temperature of the liquid and/or increasing the maximum allowable pressure. The effect of taking these measures is analysed in section 4.2.
The design options not considered include a double shell insulated propellant tank. This option is not used in any of the literature reviewed, because a second shell will increase the mass significantly, while the shell itself does not isolate properly. A second shell could be simulated though, by using an MLI blanket of one layer.

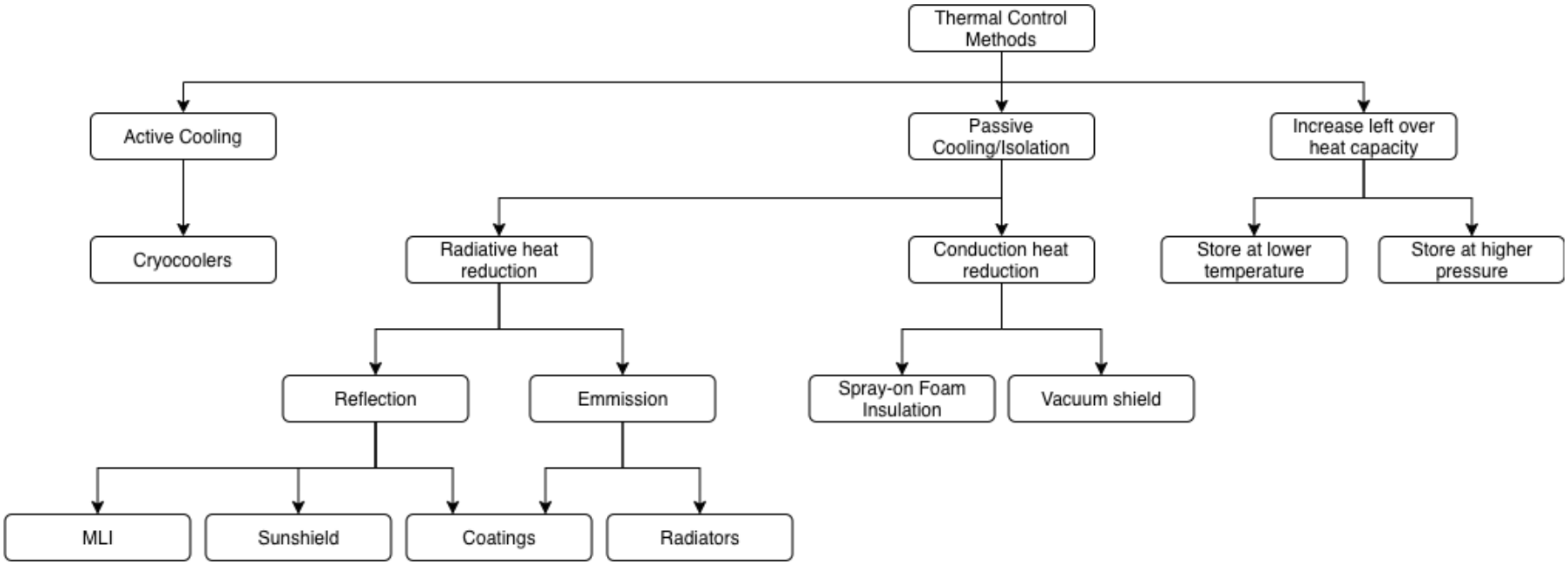

Figure 2.13: Design option tree of mitigation measures for boil-off by isolating a propellant tank.

3

Methodology

To answer the research questions and to identify what the optimal configuration concerning the mass for a spacecraft propellant tank is, a propellant tank design tool is developed. In this chapter, the theory and background information of chapter 2 is translated into a methodology for a preliminary design tool.

The input of the propellant tank design tool consists of design parameters and mission conditions. The design parameters are the properties of the SOFI, properties of MLI, properties of the cryocooler, tank wall thickness, propellant mass and fill ratio and the maximum propellant tank radius. With the mission conditions, the mission time and environment in which the spacecraft will operate are given.

From the literature review in section 2.1 two general methods can be distinguished for simulating the liquid and one method is mentioned for the propellant tank structure. The structure of the propellant tank is split up in nodes and the heat flows from one node to another is determined. This methodology will be used for this research as well.
For the liquid there are two different options: solving the Navier-Stokes equations numerically for the liquid or taking the liquid as one bulk and only determine the temperature gradient. Solving the Navier-Stokes equations is complex and a small time-step is necessary to accurately calculate the fluid movement and temperature gradient of the liquid. It is expected from the literature review and background information, that having ZBO or a small boil-off rate (< 1% per month) will give the most optimal design with respect to the mass. If there is (almost) no boil-off, it is less important what happens to the liquid. Therefore, a one bulk method is used to calculate the boil-off of the liquid.

The preliminary design tool has to consist of several subroutines. The three modules are the propellant tank external environment, the propellant tank structure including insulation materials, and the liquid and vapour bulk including the optional cryocooler. These modules will be explained after section 3.1, in which numerical integration methods are explained and one is chosen for the program.

3.1. Time integration methods

A numerical model needs to be accurate while the computation time of the model is to be limited to a minimum. For this, different time integration methods exist, which determine how an iteration node is calculated from other nodes and what the steps between these nodes are. In this subsection, different integration methods will be discussed.

For the discretisation of equations governed with the model, explicit and implicit methods are distinguished. The explicit method is only depending on previous steps, while for the implicit method also current step is taken into account, e.g. a derivative of a previous node. The advantage of the explicit method is that it is quick since only old values are used to calculate a new value. However, this method is only stable for small time steps, since otherwise, the error becomes too large. Since small time steps are required to obtain a stable solution, many time steps are necessary to analyse a system. The advantage of the implicit method is that is stable for relatively large time steps. The disadvantage is that

the calculations for every time step are more complex since the partial solutions of the time step have to be used as well.

Next to a chosen fixed time step, also algorithms exist which use a variable or adaptive time step. The time step is chosen by monitoring the error induced in a step, estimated from a certain relation. If the time step is too big and the error is above a threshold, the time step is redone with a smaller step. This way, it is assured that the total error stays within a limit while speeding up the program by using the biggest time step possible. In table 3.1 different integration schemes are presented.

Table 3.1: Numerical integration schemes [78].

Integration Method [source]	*explicit or implicit*	*Formulation (with $O(h^p)$ as truncation error)*	*Advantage*	*Disadvantage*
Euler(-Cauchy)	explicit or implicit	For explicit: $y_{n+1} = y_n + \Delta t \frac{dy_n}{dt}_{t_n} + O(h^2)$ For implicit: $y_{n+1} = y_n + \Delta t \frac{dy_{n+1}}{dt}_{t_{n+1}} + O(h^2)$	+ Algebraically seen easy. + Quick due to easiness of the solution.	- Error propagates fast due to low order of convergent $O(h^2)$. - Diverges for big time steps, due to low order of convergence.
Heun ()Trapezoidal)	implicit	$y_{n+1} = y_n + \Delta t \frac{1}{2}\left(\frac{dy_{n+1}}{dt}_{t_{n+1}} + \frac{dy_n}{dt}_{t_n}\right) + O(h^3)$	+ Slope at start and end taken into account	- Not functional close to limit
Runge-Kutta	implicit	Most often used is the RK4 version. The general formulation can be seen in equation 3.1. In the equation, $h = \Delta t$.	+ More accurate since the slope is defined at 4 points. + Relatively simple since derivatives are used from known points.	- Not suitable for non-linear differential equations with big fluctuations in slope. - Error estimation is hard to be done.
Backward Difference Formula 2	implicit	$\alpha_0 y_{n+1} = \alpha_1 y_n + \alpha_2 y_{n-1} + \Delta t \frac{dy}{dt}_{t_{n+1}} + O(h^3)$	+ Method is simpler to solve than RK.	- Not accurate for highly dynamic functions

$$y_{n+1} = y_n + h\frac{1}{6}\left(\frac{dy_n}{dt}_{t_n} + 2\frac{dy_{n+k_1 h/2}}{dt}_{t_{n+h/2}} + 2\frac{dy_{n+k_2 h/2}}{dt}_{t_{n+h/2}} + \frac{dy_{k_3 h}}{dt}_{t_{n+h}}\right) \tag{3.1}$$

For the implicit methods, the forward slope has to be known. This forward slope is hard to predict. For this program, the explicit Euler method is chosen. This method is most simple to apply and due to the expectation that the propellant tank will be in steady state, it is not deemed necessary to invest time in an implicit method.

3.2. Propellant tank external environment modelling

In section 2.3 the thermal environment in space was analysed. The identified sources of heat can be merged into one heat flow to the most outer layer of the propellant tank. On which side of the propellant tank the heat flow is directed is depending on the angle of incidence. Every surface defined by the different nodes has an angle of orientation, which determines together with the angle of incidence what the incoming heat flow is to that surface.

For simplicity a constant heat flux to the propellant tank is assumed. The heat flux used, depends

on the target mission. Since the dominant heat flux source in the space environment is the solar flux, the heat flux is assumed to consist only of the solar flux. Equation 2.14 is therefore used to obtain the heat flux. For the biggest part of interplanetary travel, the assumption that only solar radiation is relevant holds. Nevertheless, some orbital manoeuvres could make the spacecraft go by planets relatively close. This will result in that the albedo radiation and planetary infrared radiation also become a major source of heat to take into account. This is supported by [23], which states that for storage of cryogenic propellant, it should be avoided to loiter around planetary bodies.

Depending on location in space, the environment temperature can be higher or colder than the propellant tank structure. Therefore, the environment temperature can function as a heat source ($T_{environment} > T_{structure}$) or a as a heat sink ($T_{environment} < T_{structure}$). In the first case, the propellant tank emits energy to its surroundings through infrared radiation. The heat flux from the propellant tank to its surroundings is calculated by using equation 2.2, with ϵ_{IR} having the value of the infrared emissivity of the propellant tank coating. The reference temperature is the temperature of the propellant tank, while the receiving temperature is the temperature of outer space, which is 3 [K][1]. To obtain the total heat flow out of the propellant tank to the surroundings, the heat flux is multiplied by the surface area of the tank.

In chapter 5 and chapter 6 an overview will be given of the values used for the mentioned parameters. The cryogenic propellant tank boil-off tool will allow for a file to be uploaded which contains the solar radiation, the planet albedo, the surrounding temperature and other heat sources for every 10 minutes of simulation time. More information on the environment of the propellant tank is not deemed necessary for the scope of this thesis.

3.3. Modelling of the propellant tank insulation and structure

For the thermal analysis of the structure, a thermal nodes network is created [63]. A thermal nodes network is a method to reduce the complexity of a thermal analysis by simplifying the structure to a set of nodes with properties. This method is explicitly used by Perczynski [29], but from the analysis of papers in chapter 2 it is found that most papers use a similar approach.

The structure module uses the output and fixed parameters of the external environment module to determine the heat flow propagating through the thermal insulation and structure. The propagation will be via conduction and radiation. The output of this module will be the temperature of the cold side of the propellant tank structure.

To model the propellant tank structure, the shape of the propellant tank is split up in twelve sections. In figure 3.1 it is graphically presented on the right-hand side how the propellant tank is divided into different sections. It is assumed that the spacecraft propellant tank is a cylindrical propellant tank with spherical endcaps, making it possible to split the structure up into a cylindrical part and two half-sphere parts. From the work done by Perczynski [29] it was identified that using several smaller nodes to represent one structure is recommended. Therefore, it is decided to use eight parts for the cylinder and four parts for the spherical endcaps. The twelve segments involved are separated in a node for every layer of insulation, e.g. a node for the SOFI and a node for the shell.

To calculate the heat transfer past each node, a thermal network is created with these nodes. The properties of each node and the heat transfer method determine the heat transfer from one to another. The method of heat transfer between the structural nodes is by conduction or by radiation, e.g. in case of using MLI. In the case of using MLI, the modified Lockheed equation as presented in section 2.4.1 is used to determine the heat transfer. The interface heat transfer between the MLI and SOFI is determined by using Kirchhoff's Law [2][58] or by conduction, depending on how the interface is defined. For the cases defined in this work, it is assumed that the coldside of the MLI is attached to the SOFI and the SOFI outer temperature has the same temperature as the MLI coldside. This is a worst case scenario, since in reality the connection of the MLI and the SOFI will not be a full contact between the

[1] https://sciencing.com/temperatures-outer-space-around-earth-20254.html

[2] the total emissivity of an object becomes the absorptivity of that object times the emissivity of a perfect black body at a given temperature

two surfaces. The connection would be via spacers, eliminating the biggest part of conductive surface, but adding a radiative term to the balance.

Every section is sub-divided into nodes, which represent a layer of insulation or structure. The heat flow to and from a node is calculated and by subtracting the outgoing flow from the incoming flow, the net heat load at any time increment for every node is known. The net heat load is divided by the heat capacity of the node to get the temperature raise or drop of every node. For the next time increment, this temperature increase or decrease is added or subtracted from the last temperature value. How a segment is split into nodes is graphically presented in figure 3.1. The integration scheme described is found in section 3.6.

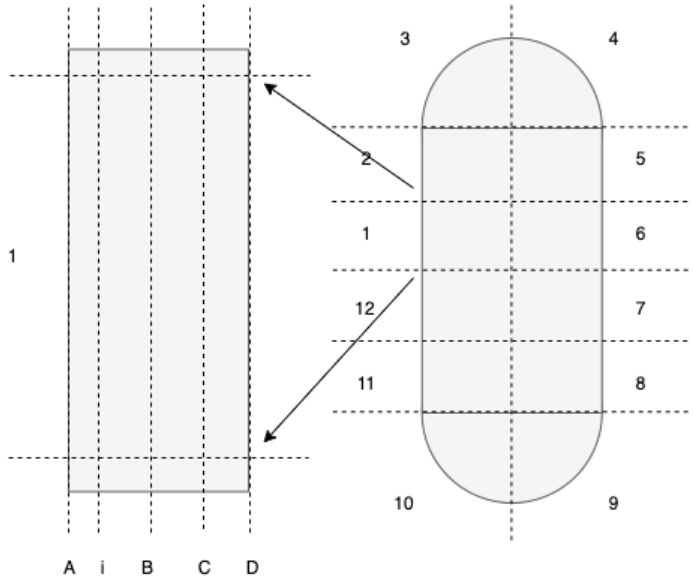

Figure 3.1: Division of the sections in sub-nodes. Left: segment split in nodes. Right: Tank split in segments.

In figure 3.2 a graphical representation is given of how the different nodes relate to each other in the thermal network. The colour of the arrows used in the figure give the method of heat transfer.

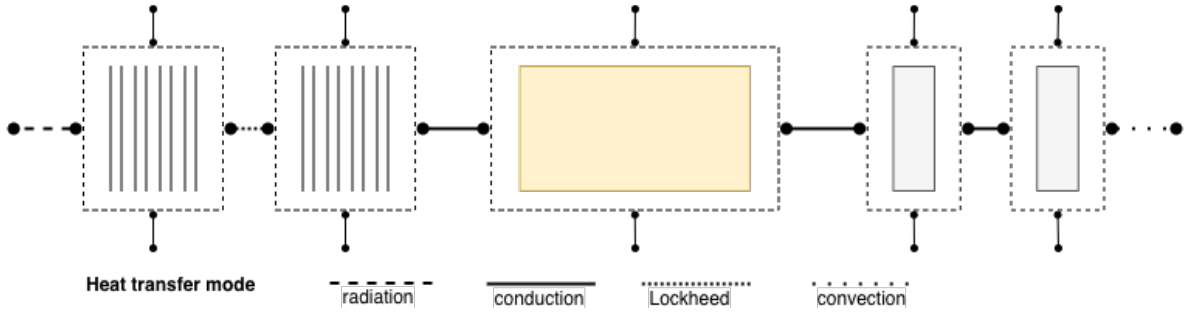

Figure 3.2: Heat flow between the sub-nodes of the thermal network.

To calculate the heat flow from one node to another, of every node the temperature, heat capacity and surface area need to be known. Next to these parameters, for the nodes which receive radiation, the absorptivity has to be known as well and from the nodes transmitting radiation, the emissivity is necessary. In figure 3.3 the information hold per node is presented graphically.

Next to the structural nodes, the liquid and vapour also need to be represented by nodes. In the following section, it is elaborated upon how these nodes are defined

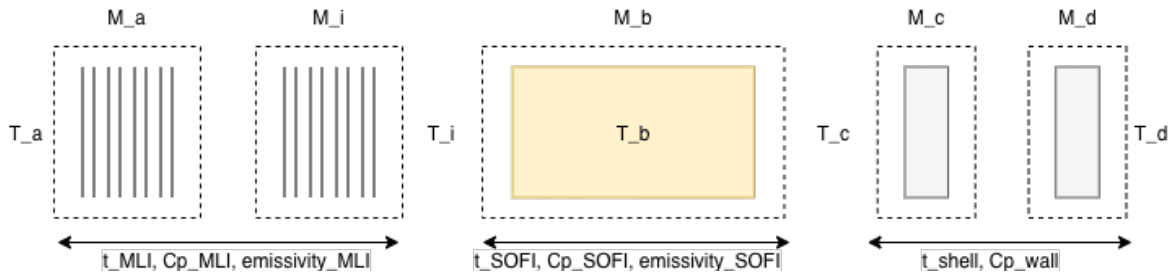

Figure 3.3: Temperature definition and mass volume of the sub-nodes of the thermal network.

3.4. Thermal modelling of the liquid, vapour and boil-off

The cold side of the propellant tank structure represents the hot side of the liquid and vapour bulk module. In this module, the mass transfer (boil-off) between the liquid and vapour will be calculated as the amount of vapour which needs to be vented due to the high pressure in the propellant tank.

The heat flow from the inner structural nodes propagates to the fluid in the propellant tank. The liquid and vapour are heated or cooled by the heat flow, depending on the temperature differences. In chapter 2 it was explained how the Rayleigh number and the characteristic length of the liquid determine the coefficient of convection for liquid in a propellant tank and for vapour in a propellant tank. This coefficient of convection is used to determine the heat flow from the solid nodes to the liquid node. The heat flow from the solid nodes to the vapour is transferred by convection, as is the heat flow from the vapour bulk to the liquid bulk.

3.4.1. Liquid node

The propellant fill level is determined by dividing the liquid volume at the time instance over the total volume of the propellant tank. Depending on the fill level, the heat flow originating from a node is either allocated to the liquid bulk or the vapour bulk.

Heat is transferred from the solid structure to the fluid by convection. To determine the heat flow, the Rayleigh number and characteristic length, which is assumed to be the fill height[52], are calculated. These parameters are used to calculate the coefficient of convection of the liquid by equation 2.8 from chapter 2. By using this coefficient of convection and by assuming the liquid is homogeneous due to an installed fluid mixing device as done by [26], [33], it is not necessary to determine the thermal boundary layer for this heat transfer problem. It is assumed that there is no propellant stratification because of the installed mix device, which gives a heat penalty.

The heat flow to the liquid bulk is summed and the result is used together with the specific heat capacity under constant pressure (since the pressure in the propellant tank is maintained) and the mass of the liquid to determine the homogeneous temperature rise of the liquid. Before this is done, it is checked whether the liquid is at a saturation temperature. If the liquid temperature is equal to the saturation temperature, all the heat is used for a phase transition. For this to be valid, it is assumed that phase transition is isentropic and thus reversible. The mass transfer rate from liquid to vapour is calculated by dividing the heat flow over the latent heat of vaporisation of the LH2. This mass flux is subtracted from the liquid bulk mass and added to the vapour bulk mass.

Last part of modelling the fluid bulk is adding a cryocooler. The cryocooler cooling capacity will result in a term in the simulation which subtracts heat from the system. This heat is radiated into space via a radiator. Using a cryocooler also comes with some penalties. These penalties have been identified in chapter 2. The result for the methodology is that a parasitic heat flow term is added to the heat balance of the liquid. The mass relationship for a cryocooler as presented in chapter 2 is used to determine the mass penalty due to using a cryocooler.

All the fluid properties, thus both liquid and vapour, are taken from the National Institute of Science and Technology (NIST) Chemistry Webbook[51].

3.4.2. Vapour node

Next to the temperature of the liquid bulk, also the temperature and pressure of the vapour bulk are calculated at every time increment. The vapour pressure determines the saturation temperature, which is used to determine the heat used for the phase transition from liquid to vapour and the heat used for a temperature rise of the fluid. The coefficient of convection of the vapour bulk is determined by using equation 2.9 from chapter 2. This coefficient determines the heat flow, which is used to calculate the temperature rise of the vapour. For this temperature change, the heat capacity coefficient of the vapour mixture is used. In section 2.4.3 it is given how the coefficient of the gas mixture is found.

The pressure rise of the vapour is determined by two factors: the thermal expansion of the already present vapour and the additional vapour due to boil-off. The expansion of the present vapour results in a pressure increase given by equation 3.2. This equation is from the ideal gas theory, given that the process is isochoric (thus the volume in which the gas sits is constant).

$$\frac{\mathrm{dp_v}}{\mathrm{dt}} = \frac{\mathrm{R_G} m}{\mathrm{m_{vapour}} * \mathrm{V_{vapour}}} \mathrm{dT} \tag{3.2}$$

The pressure increase due to boil-off is estimated by equation 3.3, as given by [75].

$$\frac{\mathrm{dp_v}}{\mathrm{dt}} \approx \frac{\mathrm{R_G T_{sat}}}{\mathrm{LV_{vapour}} m} \mathrm{dP_{liquid}} \tag{3.3}$$

Both relationships are used to determine the pressure increase of the vapour. If the pressure exceeds a threshold value, vapour needs to be vented from the system to reduce the pressure. This venting is done by a pressure valve, and by venting the vapour, the remaining vapour is cooled. The reduction of the temperature is given by equation 2.35 in section 2.4.3.

Altogether, the theory and assumptions used and stated in this section result in the heat and mass flows as presented in figure 3.4. In section 3.5 a list of all the assumptions used is given.

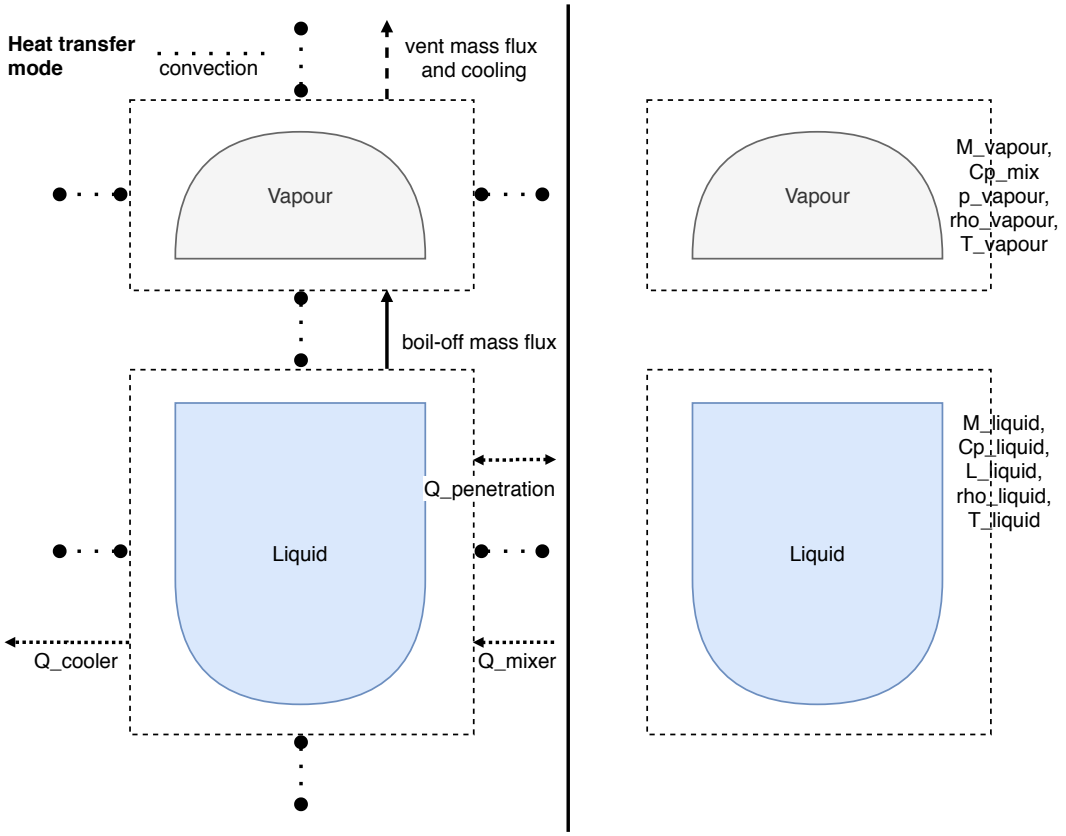

Figure 3.4: Liquid and vapour nodes of the thermal network including the heat and mass flows.

3.5. List inputs and list of assumptions

The following inputs have to be provided by the user of the program:

Table 3.2: List of input parameters which have to be provided by the user of the boil-off program.

- cooler power
- propellant mass
- initial temperature
- initial pressure
- maximum tank radius
- time increment
- mission time
- initial fill level
- SOFI thickness
- SOFI density
- SOFI thermal conductivity
- SOFI heat capacity
- SOFI clean emissivity
- shell heat capacity
- shell thermal conductivity
- shell density
- shell thickness
- MLI layers
- MLI layer density
- MLI density
- MLI thermal conductivity
- MLI heat capacity
- MLI emissivity
- MLI interstitial gas pressure
- external coating emissivity
- external coating absorptivity
- solar incidence angle
- gravity
- outer space temperature
- solar radiation

The methodology for the propellant tank boil-off tool contains several assumptions. In this section,

these assumptions are listed.

- A constant heat flux to the propellant tank is assumed over time.
- The spacecraft propellant tank is a cylindrical propellant tank with spherical endcaps.
- The characteristic length used to determine the coefficient of convection is defined by the fill height of the liquid in the propellant tank.
- The liquid is homogeneous due to an installed fluid mixing device and there is no propellant stratification because of the installed mix device. In the comparisons presented in this thesis, the mixer mass is not taken into account, since all the design options used are assumed to contain the same mixer.
- Phase transition of the liquid to vapour is isentropic.
- It is assumed for this research that the initially tanked hydrogen is all para-hydrogen, thereby eliminating the issue of heat release by the ortho-hydrogen becoming para-hydrogen. This assumption is valid and realistic, it only requires the hydrogen to be stored at 20 [K] for two days ahead of launch.

The output of the program will focus on the mass of the different components, the steady-state temperatures and heat flow, the boil-off rate and the total boil-off mass. In the list below all the parameters stored in output CSV files are given:

- Temperature of all the nodes
- Heat load of all the nodes
- Boil-off mass
- Time at which boil-off starts
- Vapour vent mass
- SOFI mass
- MLI mass
- Cooler mass
- Total mass (SOFI, MLI, cooler and boil-off)

Next to the standard output of the program, the user can modify the program code to output all the parameters involved manually.

3.6. Iteration scheme

Using the methodology described requires a time loop for simulating the thermodynamics of a cryogenic propellant tank. In this time loop, for every time instance (i) the properties of all the nodes ($k = 1, k = 2, k = 3, etc.$) are calculated.
For the net heat flow to node k, the heat flow to all the adjacent nodes is summed:

$$(\mathrm{dP_{net}})_{\mathrm{i}}^{\mathrm{k}} = \mathrm{dP}_{\mathrm{i}}^{\mathrm{k-(k-1)}} + \mathrm{dP}_{\mathrm{i}}^{\mathrm{k-(k+1)}}$$

Depending on the material of the node, $dP_i^{k-(k-1)}$ can be determined by either using the equation for conduction, if the material of the node is SOFI or the shell, by the Modified Lockheed equation, if the material of the node is MLI, or by convection, then the node is a shell node on the interface with the liquid bulk. This results in the following relations:

$$\mathrm{dP}_{\mathrm{i}}^{\mathrm{k-(k-1)}} = -\frac{\mathrm{k}}{\mathrm{L}^{\mathrm{k-1}}}\left(\mathrm{T}_{\mathrm{i}}^{\mathrm{k}} - \mathrm{T}_{\mathrm{i}}^{\mathrm{k-1}}\right)\mathrm{A}^{\mathrm{k}}$$

$$\mathrm{dP}_{\mathrm{i}}^{\mathrm{k-(k-1)}} = -\epsilon\sigma_b\left(\mathrm{T}_{\mathrm{i}}^{\mathrm{k}^4} - \mathrm{T}_{\mathrm{i}}^{(\mathrm{k-1})^4}\right)\mathrm{A}^{\mathrm{k}}$$

$$\mathrm{dP}_{\mathrm{i}}^{\mathrm{k-(k-1)}} = -\mathrm{h}\left(\mathrm{T}_{\mathrm{i}}^{\mathrm{k}} - \mathrm{T}_{\mathrm{i}}^{\mathrm{k-1}}\right)\mathrm{A}^{\mathrm{k}}$$

To determine the temperature change, the total heat flow of node k is divided by the heat capacity of the node. The heat capacity of node k is the specific heat capacity of node k times the mass of node k:

$$dT_i^k = \frac{(dP_{net})_i^k \Delta t}{C_p^k M^k}$$

For the temperature of the instance, an iteration scheme is used for which the temperature of the instance is determined from the temperature of the last time instance $i-1$ plus the determined change of the instance for any k:

$$T_i^k = T_{i-1}^k + dT_i^k$$

Initially, the system is not in steady-state. All the nodes are cooled to 20 [K]. After a settling period, a steady-state is reached. The hot- and cold-side temperatures (T_H and T_C) the MLI, SOFI and the wall will settle at equilibrium temperatures depending on the heat flow and the thickness of the mentioned components.

This iteration scheme is used in a Monte Carlo analysis-like. A Monte Carlo analysis[3] is an analysis in which a program is running a lot of times to see what the general output would be. A Monte Carlo analysis does not cut options, it analyses all the options. Since not all options are randomised and checked a thousand times, the simulation performed here is not an exact Monte Carlo analysis, but Monte Carlo analysis-like. A Monte Carlo analysis is preferred over an optimiser because more than two parameters are used to find a mass optimal solution. If an optimiser is used, the solution is expected to converge towards a local optimum too soon instead of a global optimum.

The first part of the simulation is not part of the Monte Carlo analysis. In figure B.1 in appendix B an extensive flow diagram of the set-up parameters for the simulation is given. An overview of the steps is given in figure 3.5. For most design parameters, fixed values should be given here. If the input parameters are given, the Monte Carlo Analysis is initiated. In figure B.2 in appendix B the flow diagram of the Monte Carlo analysis is presented.

The propellant tank boil-off program will be written in Python. Python is an open-source program language and is taught to the students of the Aerospace Engineering faculty of the Delft University of Technology. Therefore, Python is considered most suitable to create the program in. The exact python script will be presented in appendix C.

[3] https://towardsdatascience.com/an-overview-of-monte-carlo-methods-675384eb1694

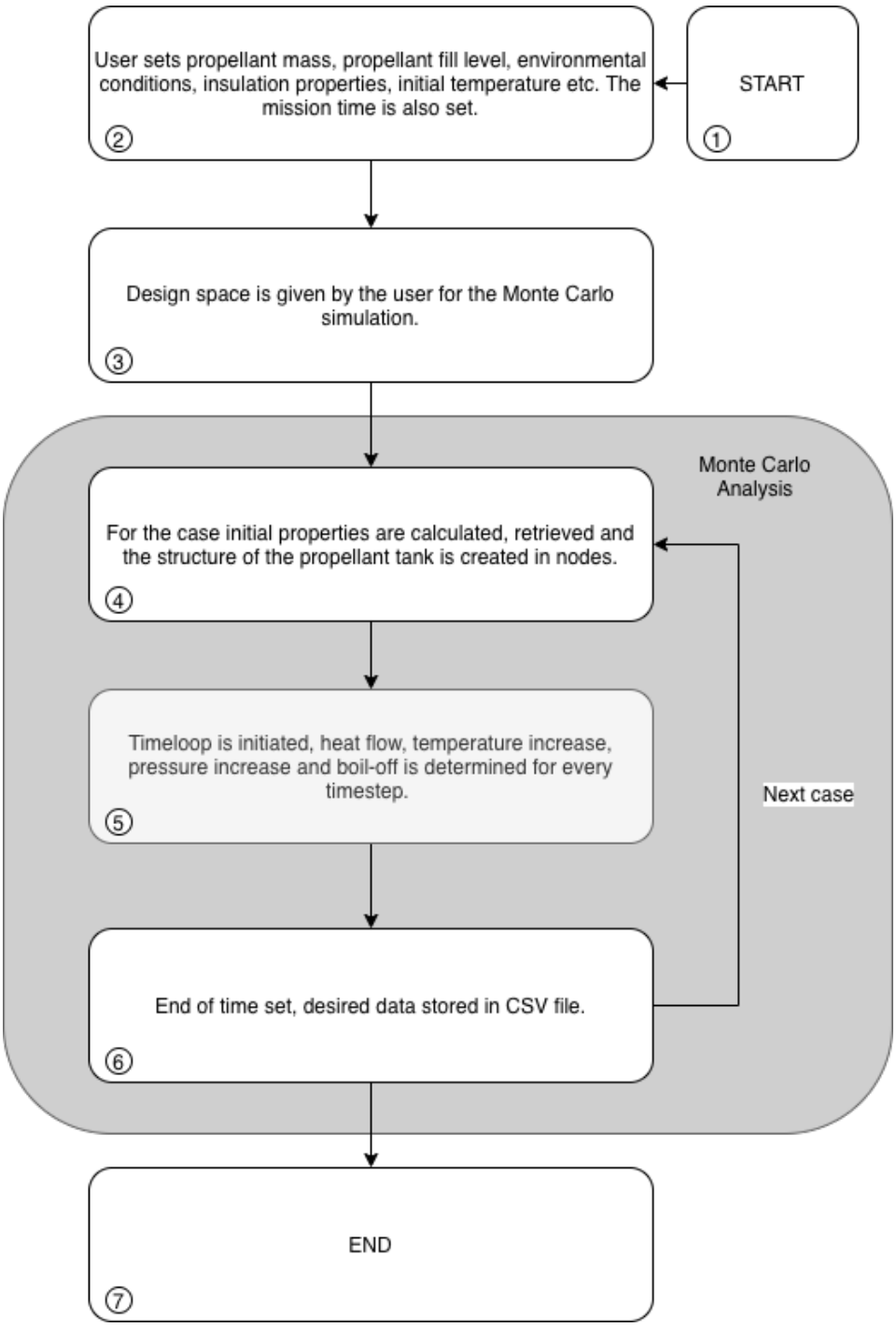

Figure 3.5: Flow diagram of the simulation performed in global steps.

4

Verification, sensitivity analysis and validation strategy

In the previous chapters, relevant background theory was discussed and a methodology was given which is used to develop a tool which predicts the boil-off of different cryogenic propellant tank designs. In this chapter, the focus is on verifying the different components of this tool developed. The verification of the SOFI module, the MLI module, the cryocooler module and the liquid/vapour module is done in section 4.1.
Further in this chapter, also a sensitivity analysis is done. From this sensitivity analysis, the influence of different components on the result is identified. The purpose is twofold: first, it can be used to identify which parameters have to be known accurately, secondly, the results can be used to determine which design parameter has the most effect on the end result. This knowledge can be used to come up with different designs.
The last part of this chapter, section 4.3, will focus on the validation strategy.

To generate results, verify the program and to assure reproduce-ability, two cases are analysed in this thesis. The input parameters used in the two cases are given in sections 5.1 and 6.1. The first case is a propellant tank depot orbiting the Earth, similar to the case handled in [33]. The second case is an analysis of how to improve the cryogenic liquid storage performance of the Centaur upper stage. The verification plus sensitivity analysis are done with the parameters used for the propellant depot. This parameters are given in section 5.1.

4.1. Verification

Verification of the simulation is done by identifying whether the output of the separate equations and modules used is as expected. Verification is checking whether what is build, is built right and according to the requirements. The difference with validation is that validation is checking what is built is the right tool to come to the desired result [79]. The verification done indicates whether the simulation could be valid or not, if the results of the verification are compared with results from other papers and the values are reasonably similar.

Verification can also be done by comparing the results of a program with the results of others. For this to be done, sufficient information has to be available to compare one research with another. In the methodology chapter, chapter 3, the input parameters are given used in the program developed. To be able to compare the results of this program with that of others, those input parameters have to be known for other sources as well.

4.1.1. Verifying Spray-on Foam Insulation simulated performance

In simulating the performance of SOFI, the conduction through the SOFI needs to be calculated and the heating of the SOFI itself. The SOFI determines the thickness and k-value to be used in the conduction equation. It is expected that the performance of the SOFI becomes better, thus reduces the

heat flow, when the SOFI thickness increases or the k-value decreases. The heat flux through the SOFI is determined using the conduction equation as presented in chapter 2. In figure 4.1, the simulated performance of the SOFI is plotted. From the figure, it can be seen that a lower k-value reduces the heat flux and a thicker layer of SOFI also reduces the heat flux. Note that the scales are logarithmic.

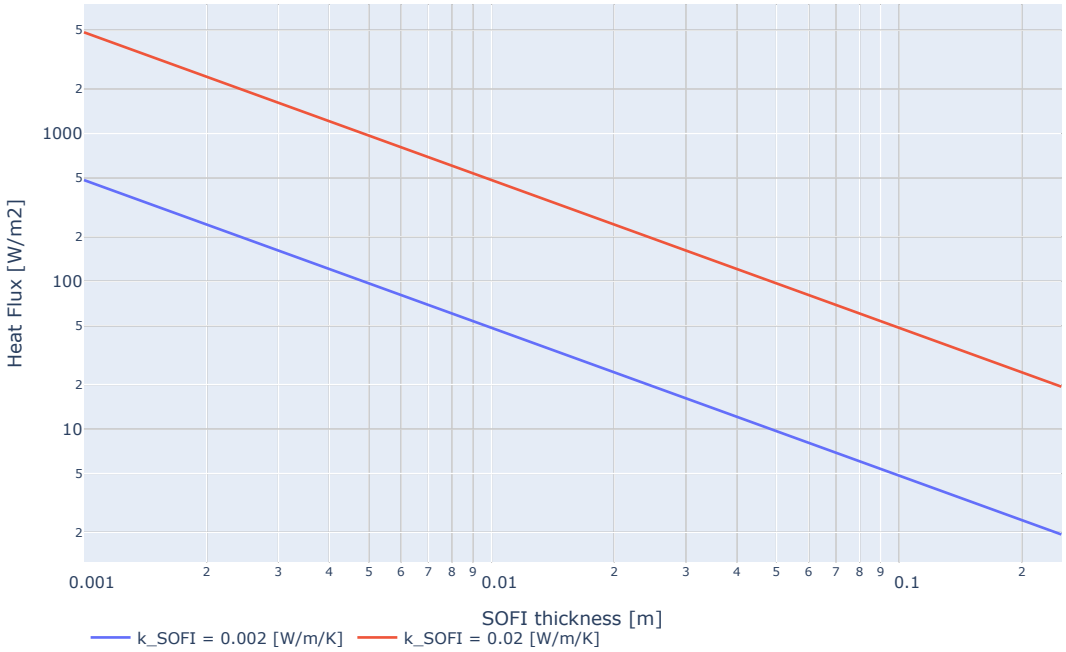

Figure 4.1: Simulation results SOFI with $T_H = 262[K]$,$T_C = 20[K]$ and two values for the coefficient of conduction ($k_{SOFI} = 0.002[W/m/K]$ and $k_{SOFI} = 0.02[W/m/K]$).

From figure 4.1 the effect of changing the k-value and the thickness of SOFI can be seen concerning the heat flux. From this observation, it follows that the boil-off rate should decrease with increasing SOFI thickness. To verify whether this statement is true, a complete simulation of the model is done. In this simulation, a varying thickness of SOFI is applied on top of the constant thickness shell. The result of this simulation is seen in figure 4.2. The figure proves the statement to be verified. It is observed in the figure, that the thicker the SOFI, the less benefit there is from increasing the thickness. The rate with which the boil-off rate is decreasing with increasing SOFI thickness is decreasing. The thickness of the SOFI returns in the conduction equation as $1/L$. The shape of the line is therefore justified since it is similar to a $1/L$ line.

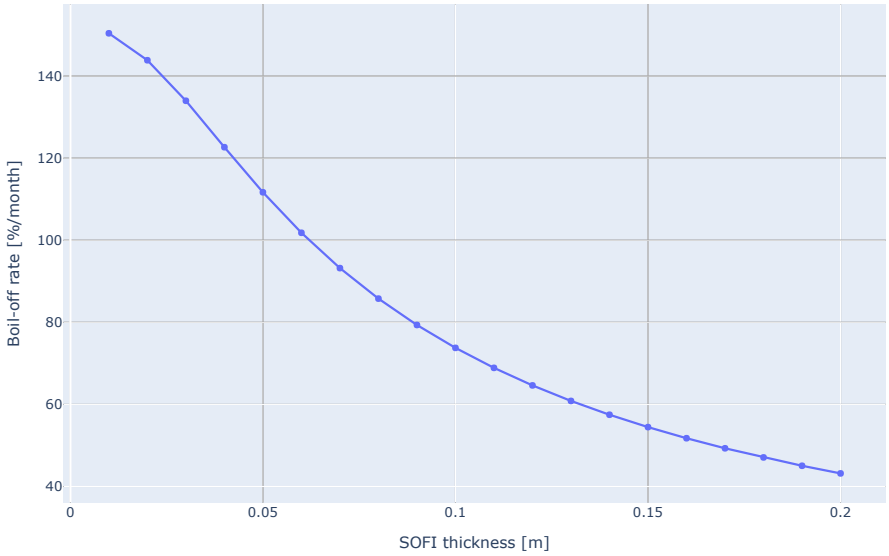

Figure 4.2: Simulation results for the boil-off rate per month over the SOFI thickness. Mission time is 12 months.

4.1.2. Verifying Multi-Layer Insulation simulated performance

It is expected that when the number of layers of MLI increase, the heat flow through the MLI decreases and the boil-off rate of the propellant decreases as well. If the layer density of the MLI increases, it is expected that the heat flow through the MLI increases and the boil-off rate also increases. This latter can be counter-intuitive, but by analysing equation 2.19 from chapter 2 it can be found that an increasing layer density increases the heat flow by solid conduction.

From [33] figure 4.3 is taken. In the figure, the relationship between heat flux through the MLI as a function of the amount of MLI layers is given for an analytical model and the Lockheed equation. Both methods have a comparable output and the output is similar to what is expected from analysing equation 2.19. Figure 4.3 is used to validate the analytical model with the empirical Lockheed model in [33]. In figure 4.4 the results of the tool developed are presented. The two lines presented are the Modified Lockheed equation, as presented in equation 2.19, and the original Lockheed equation given by equation 4.1 [60]. On the horizontal axis, the number of MLI layers is given and on the vertical axis the heat flux through the MLI.

$$\begin{aligned} \mathrm{q_{total}} =& 7.30 \times 10^{-8} * \mathrm{T_m} (\mathrm{N*})^{2.63} \left(\mathrm{T_H} - \mathrm{T_C}\right) / \mathrm{N_s} \\ &+ 7.07 \times 10^{-10} \epsilon \left({\mathrm{T_H}}^{4.67} - {\mathrm{T_C}}^{4.67}\right) / \mathrm{N_s} + 1.46 \times 10^{4} \mathrm{P} \left({\mathrm{T_H}}^{0.52} - {\mathrm{T_C}}^{0.52}\right) / \mathrm{N_s} \end{aligned} \tag{4.1}$$

Comparing the Lockheed equation results from figure 4.4 and the results in figure 4.3 it can be observed that the lines presenting the Lockheed model are similar in both figures with the maximum deviation for only 1 layer of MLI ($\approx 80\%$, but to $\approx 20\%$ at 10 layers and no deviation at 1000 layers).
From figure 4.4 it is observed that the modified Lockheed equation gives a heat flux which is approximately double the value given by the conventional Lockheed equation.

In figure 4.5 the heat flux given in figure 4.4 is split in the conduction terms, both gas conduction and spacer conduction, and the radiation term. From the figure, it is observed that the modified Lockheed equation compared to the Lockheed equation estimates the conduction through the spacers with a higher value. The importance of estimating the heat leak through the MLI by conduction is stressed by [59], in which it was concluded that emittance of the MLI plays a secondary role compared to spacer conductance in the thermal performance of both high- and low-temperature MLI blankets. It should be noted, that in [60] it was concluded that for cold boundary-temperatures ($T_{boundary} < 160[K]$) the actual heat leak is underestimated by 30 % since the thermal conductivity of the spacers is underestimated. At $T = 300[K]$ the deviation between the model and the test is 8%, which indicates that the

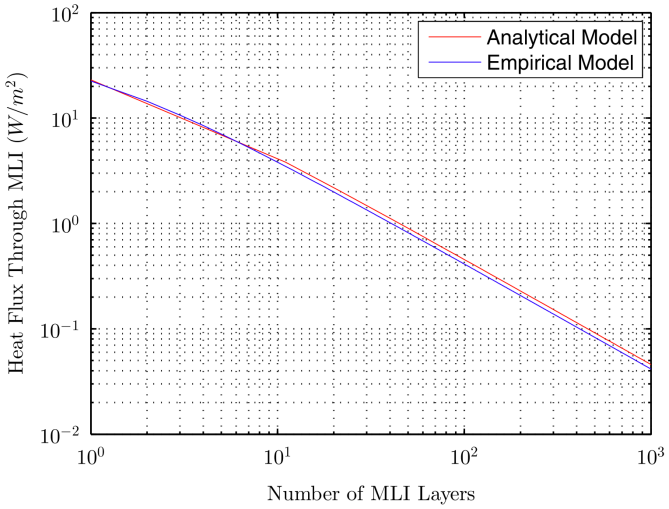

Figure 4.3: Literature results Lockheed equation and Analytical MLI model with $T_H = 262[K]$,$T_C = 20[K]$ and $\rho_{layers} = 40[layers/cm]$ [33].

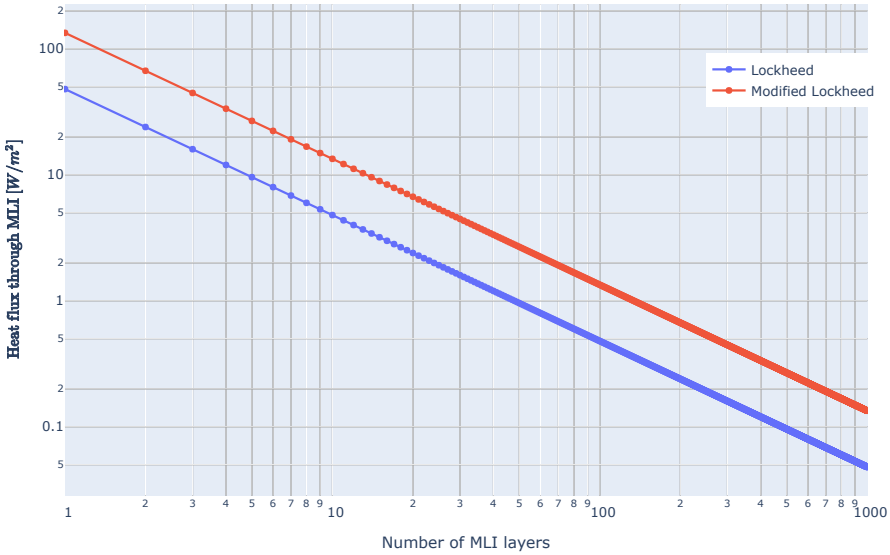

Figure 4.4: Comparison results Lockheed equation and Modified Lockheed equation with $T_H = 262[K]$,$T_C = 20[K]$ and $\rho_{layers} = 40[layers/cm]$.

performance of the equation improves with higher boundary temperatures. These higher boundary temperatures are expected for the simulations and therefore the modified Lockheed equation is used in the program.
Also observed in figure 4.5 is that the heat flux due to radiation is estimated higher in the Lockheed equation compared to the modified Lockheed equation. The difference is 40% over the whole range of MLI layers. Nevertheless, the effect is negligible since the magnitude of the radiative flux is much smaller than the conductive flux.

The expectations that the heat flux through the MLI reduces with an increasing amount of layers and the heat flux increases with a higher layer density are true. Although the results demonstrate the verification of the conventional Lockheed equation, the reasons which are given in [60] and [59] suggest that it is better to use the modified Lockheed equation. From [33] figure 4.6 is taken. This figure gives the relationship between the boil-off rate per month and amount of MLI layers for a 34,600 [kg] LH2 tank. This relationship can be used to analyse the effect of using the modified Lockheed equation instead of

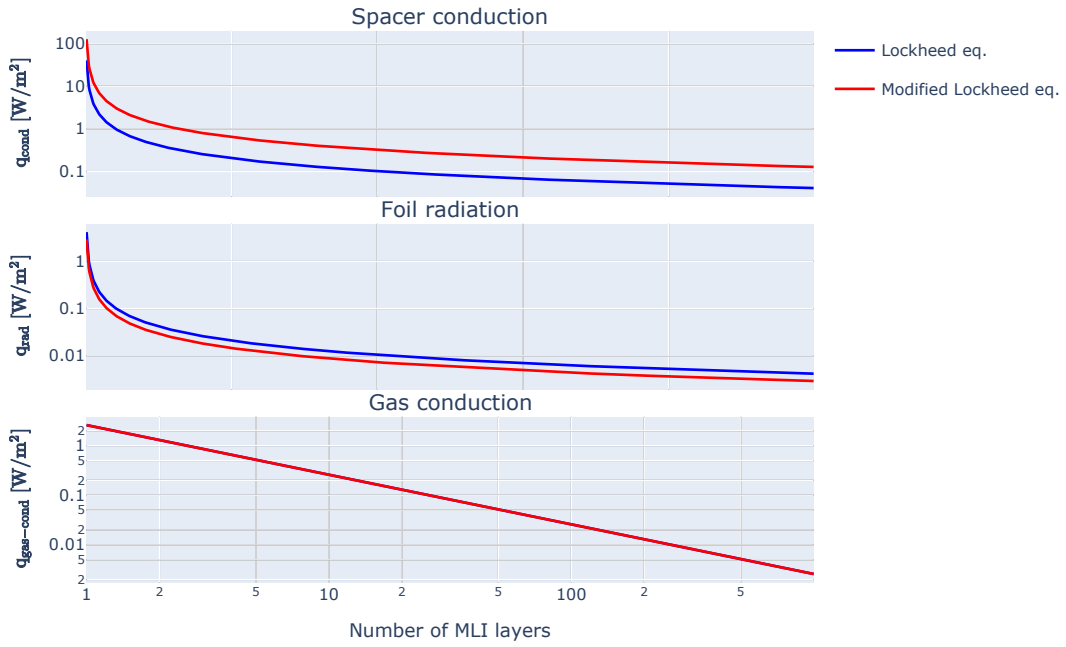

Figure 4.5: Heat flux contribution conduction (gas and spacer) and radiation of the Lockheed equation and modified Lockheed equation as function of the number of layers.

the conventional Lockheed equation. The layer densities used will be used to verify the results, however, from equation 2.19 it is clear that a lower layer density is beneficial for the thermal performance of the propellant tank MLI since the conductive term reduces in magnitude.

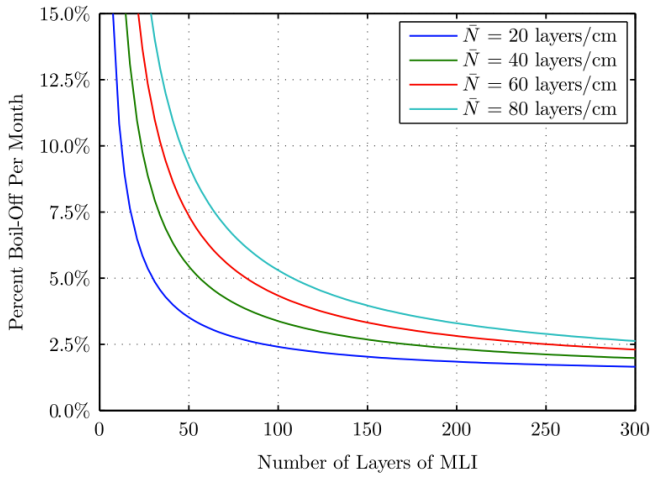

Figure 4.6: Boil-off rate per month over the amount of MLI layers from literature [33].

From figures 4.6, 4.7 and 4.8 it is observed that the boil-off rate varies between the modified Lockheed model, the Lockheed model and the analytical model used by Chai et Wilhite [33]. Chai et Wilhite validated the analytical model used by comparing the results of the analytical model with the Lockheed equation. This is seen in figure 4.3. The analytical equation used, gives the same result as the Lockheed equation for a layer density of $\rho_{layers} = 40[layers/cm]$.

In table 4.1 the results from the figures 4.6, 4.7 and 4.8 are tabulated for the validated layer density of $\rho_{layers} = 40[layers/cm]$. From table 4.1 it is observed that the differences between the modified

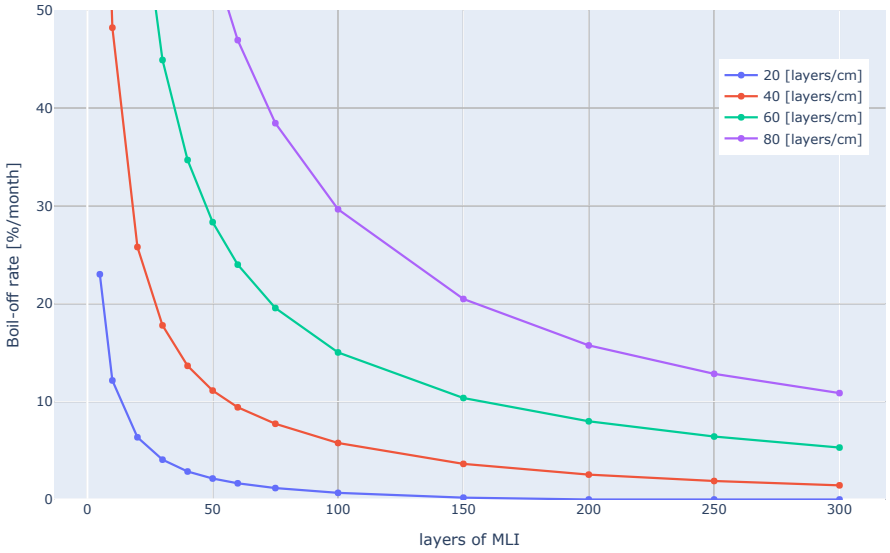

Figure 4.7: Simulation results for the boil-off rate per month over the amount of MLI layers with the Modified Lockheed equation. Total mission time is 12 months.

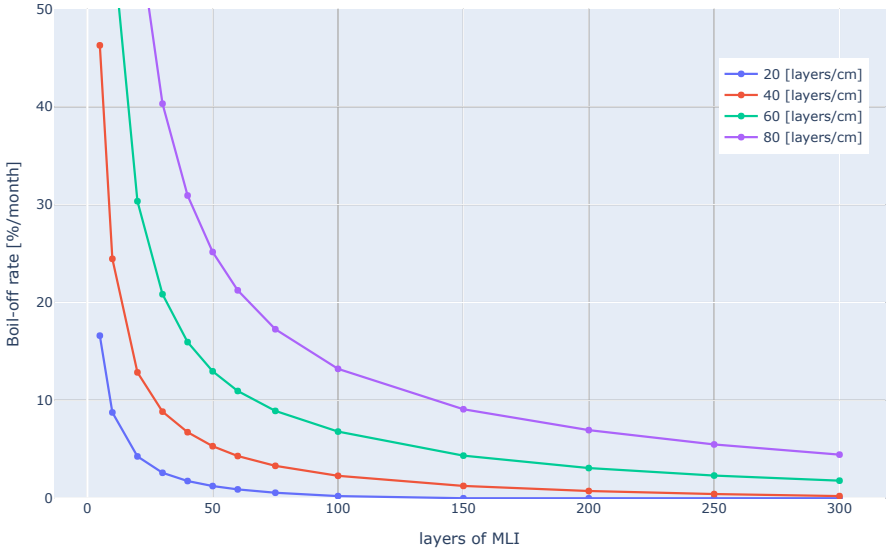

Figure 4.8: Simulation results for the boil-off rate per month over the amount of MLI layers with the Lockheed equation. Total mission time is 12 months.

Table 4.1: Boil-off rate per month for the modified Lockheed model, Lockheed model and Chai et Wilhite [33] for a layer density of $\rho_{layers} = 40[layers/cm]$.

MLI layers [-]	50	100	150	200	250	300
modified Lockheed model	11%	6%	4%	3%	2%	1%
Lockheed model	5%	2%	1%	1%	0%	0%
Chai et Wilhite analytical model [33]	6%	3%	3%	2%	2%	2%

Lockheed model and the Chai et Wilhite model are larger than the differences between the Lockheed model and the Chai et Wilhite model. This is explained by that Chai et Wilhite used the Lockheed model to validate their model. The difference between the Lockheed model and the modified Lockheed model is between the 100 % and 400 %. This is also observed by Fesmire [80], in which it is further concluded that the modified Lockheed model has a better fit to cryogenic propellant tank test data than the Lockheed model.

Next to the data of Chai et Wilhite, no source is found with sufficient information to further verify the functioning of the MLI module. Nevertheless, from the little information presented there is evidence that the MLI module gives sensible results.

4.1.3. Verifying cryocooler mass simulated performance

In the program, the mass of the cryocooler is estimated by using historical data of cryocoolers and data on the mass per power unit. By using the equations provided in chapter 2, the expected mass of the cryocooler can be found. From the relations presented in section 2.4.2, figures 4.9 and 4.10 are made.

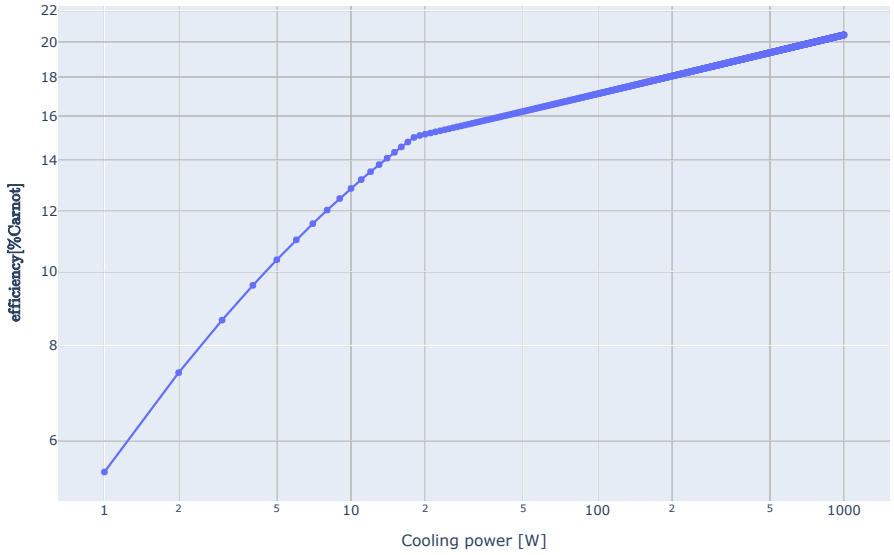

Figure 4.9: The efficiency of Carnot efficiency as a function of the cooling power in [W].

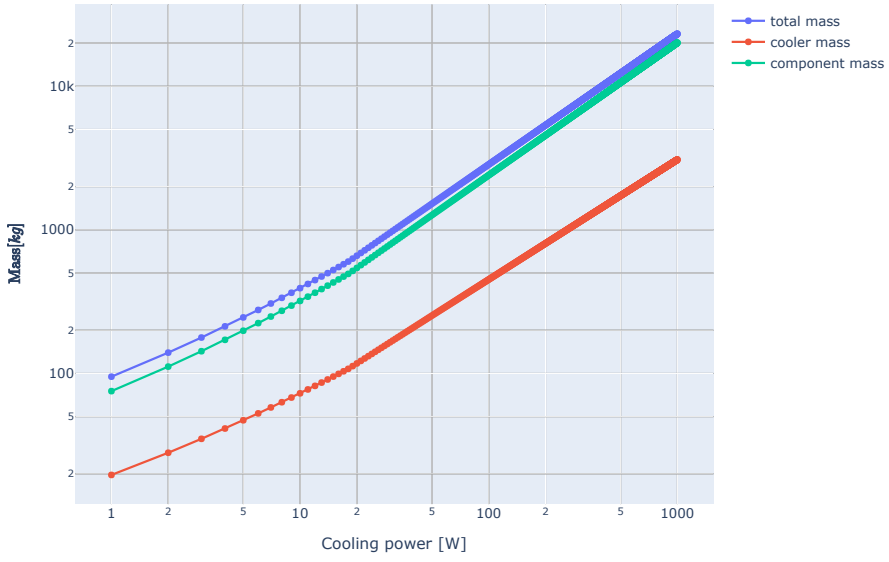

Figure 4.10: Total mass, cooler mass and additional system mass as a function of cooling power in [W].

The efficiency of Carnot as a function of the cooling power of a cryocooler is presented in figure 4.9. At around 18 [W] of cooling power, a change in slope in the graph is seen. This is because the relations give the minimum of the Ter-Brake correlation and the AF correlation and the equation changes between relations at this point. When comparing figure 4.9 with figure 2.12 it is observed that the fraction

of Carnot efficiency presented is similar.

In figure 4.10 the total mass, cryocooler mass and additional system mass, consisting out of the power system, power cables, additional structural components, etc., to accommodate the cryocooler are presented. From the figure, it is seen that the additional system mass accounts for the biggest part of the total cryocooler system mass. This is as expected from equation 2.28 in section 2.4.2.
In [71] it is stated that the performance of the relations presented is not validated due to the lack of commercial cryocoolers cooling at a range of 20 [K]. It should be noted that there is a big uncertainty in the validity of the total mass relationship. The biggest uncertainty is in the assumption that all the power necessary to run the cryocooler, results in additional mass to the system. In the case calculated, that gives an additional 523 [kg] compared to 113 [kg] for the cooler itself. Many systems use a nominal power and have a peak power. In the worst case, all the peaks have to be supplied simultaneously. Most of the time, there is power left of the power possibly supplied by the system. If half of the power could be taken from the left-over power, this results already in a 41 % mass savings.

4.1.4. Verifying coefficient of convection simulated performance

The coefficient of convection determines how much heat is transferred from the wall to the liquid and the vapour. One of the key parameters of the performance of the model is the Nusselt number. In chapter 2 it has been explained that the Nusselt number is necessary to find the coefficient of convection of the vapour and the liquid. Relations from [52] are used to couple the Rayleigh number to the Nusselt number and thereby to the coefficient of convection. It is expected that for a low Nusselt number, the dominant heat transfer method in the fluid is by conduction. Since the Nusselt number and Rayleigh number are coupled, a low Nusselt number also indicates a low Rayleigh number. For the fluid characteristics, this would result in a long time scale for thermal transport via convection, or a short time scale for thermal transport via diffusion. This supports the expectation that for a low Nusselt number, conduction is dominant.

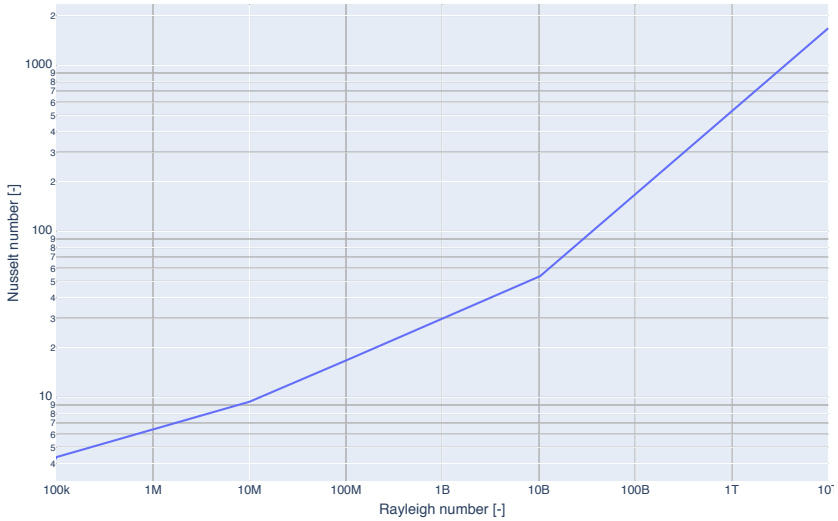

Figure 4.11: Nusselt number as a function of the Rayleigh number for LH2.

In figure 4.11 the calculated Nusselt number is plotted. The Nusselt number is seen as a function of the Rayleigh number. The plot shows the same results as can be observed from figure 2.4 in section 2.2.3. In the figure, two changes of the slope can be observed. The relation between the Rayleigh number and the Nusselt number depends on the value of the Rayleigh number, as can be seen from equation 2.7. The two changes are due to the relations changing.

The coefficient of convection is calculated by the relations developed by Yang [52]. These relations have a validity range to the Rayleigh number. The Rayleigh number of the liquid should be lower than $Ra_{liquid} < 5 * 10^{13}[-]$ and the Rayleigh number of the vapour should not exceed $Ra_{vapour} < 10^{12}$. In

figure 4.12 the Rayleigh number of both the vapour and liquid is plotted over time. From the figure, it is observed that both phases do not exceed the validity ranges given.

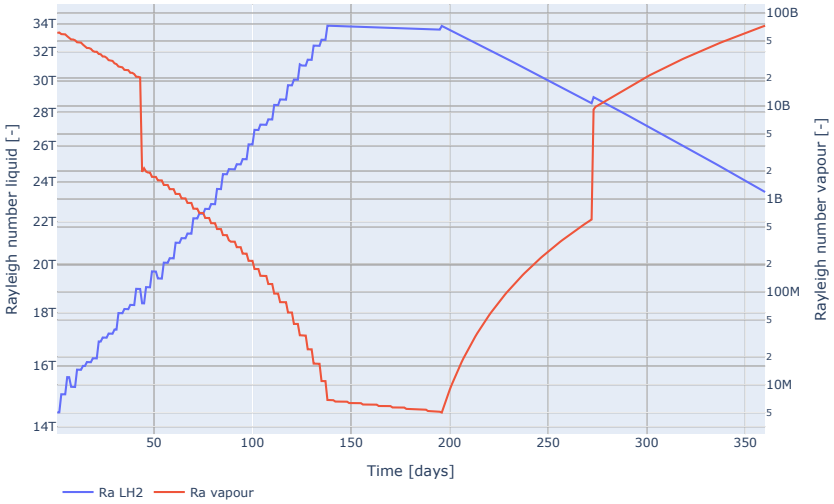

Figure 4.12: Rayleigh number over time for LH2 and the vapour for a propellant tank of 34,600 [kg] LH2, SOFI thickness 0.01 [m], 20 layers of MLI with density 16 [layers/cm].

To verify the functioning of the coefficient of convection modelling, the coefficient is plotted for the period of 12 months, for the condition with an MLI layer density of 16 [layers/cm], 20 layers of MLI, a SOFI thickness of 1.0 [cm] and no cryocooler power. The result of this simulation can be seen in figure 4.13.

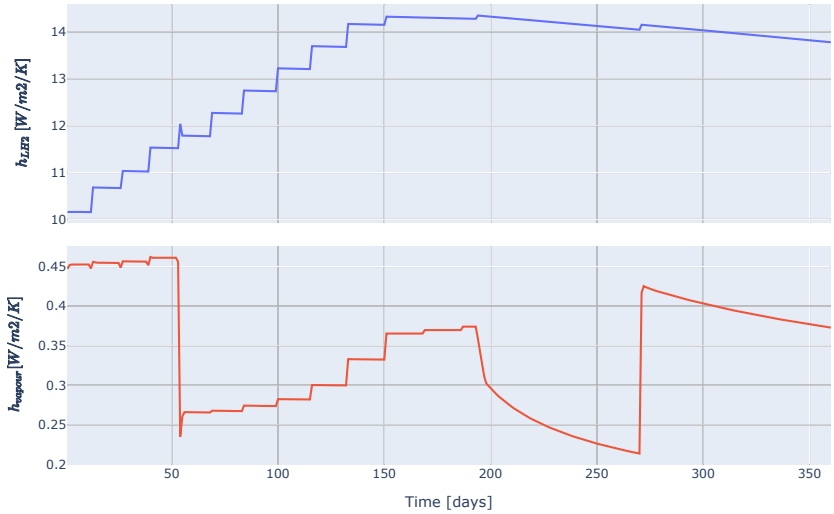

Figure 4.13: Coefficient of convection over time for LH2 and the vapour for a propellant tank of 34,600 [kg] LH2, SOFI thickness 0.01 [m], 20 layers of MLI with density 16 [layers/cm].

From figure 4.13 it is observed that initially, the coefficient of convection of the LH2 starts to increase, the stepwise nature is explained by updating of the properties of the liquid every 0.5 [K]. The increase of the coefficient of convection is due to the warming of the LH2, which results in a higher Rayleigh number and therefore a higher coefficient. The same happens with the coefficient of convection of the vapour. At 54 days, there is a drop in the coefficient of convection of the vapour. This is due to the fill level of the tank growing and the upper node of the propellant tank starts to release heat to the liquid as well as the vapour. The vapour heats from that moment onwards and the line increases again.
At 194 days the liquid starts to boil. A drop in the vapour coefficient of convection is again visible,

this time due to the hydrogen vapour changing the average mass of the vapour. This reduces the coefficient of convection since the Rayleigh number decreases. At around 270 days the liquid fill level decreases drastically, thereby increasing the heat to the vapour and decreasing the heat to the liquid. This explains the rise of the coefficient of convection of the vapour, while the reduction of the fill height explains the reduction of the coefficient of convection of the liquid.

4.1.5. Steady state temperatures and heat balance

A steady state of a system is a state in which the heat flows are (close to) zero. A thermal equilibrium is reached between the different components. In figure 4.14 the temperature of the space-propellant tank interface nodes are presented. It is seen in the figure, that from the start a steady state is reached. The temperature over time of all the nodes, is more or less constant.

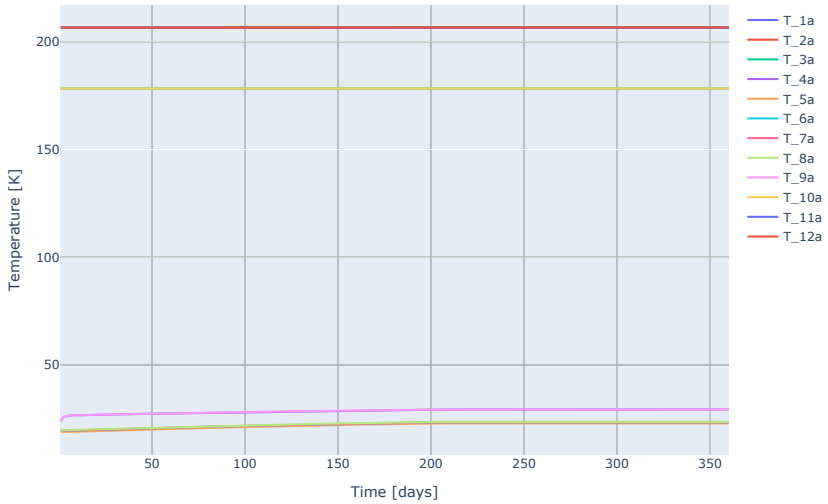

Figure 4.14: Temperatures of the outer nodes of the propellant tank over time.

Next to the steady state of the outer nodes, it is also expected that the other nodes will reach an equilibrium. In figure 4.15 the temperature of the nodes on the liquid-structure interface is presented. Several observations can be done from the figure. The first observation is the interface node 3D has a sudden drop of temperature at day 54. This is explained by the liquid fill level increasing due to expansion of the liquid and thereby the liquid touches the node after 54 days. Since the liquid has another temperature than the vapour present, the steady state temperature changes in this situation. The second observation is the increase of the temperature of node 3D at 190 days. This is when boil-off of the liquid starts and the opposite of what has been described before occurs. The liquid fill level reduces and node 3D is completely exposed to vapour again. This behaviour is not seen for node 4D, which faces the same vapour and liquid. This is explained by the fact that node 4D is at the shadow side of the spacecraft, thus the thermal equilibrium is different.

The temperatures presented in figure 4.14 are constant over time and the temperatures in figure 4.15 are constant from the moment boil-off starts. This indicates that the heat balance for these nodes is zero, since otherwise the temperatures should increase.

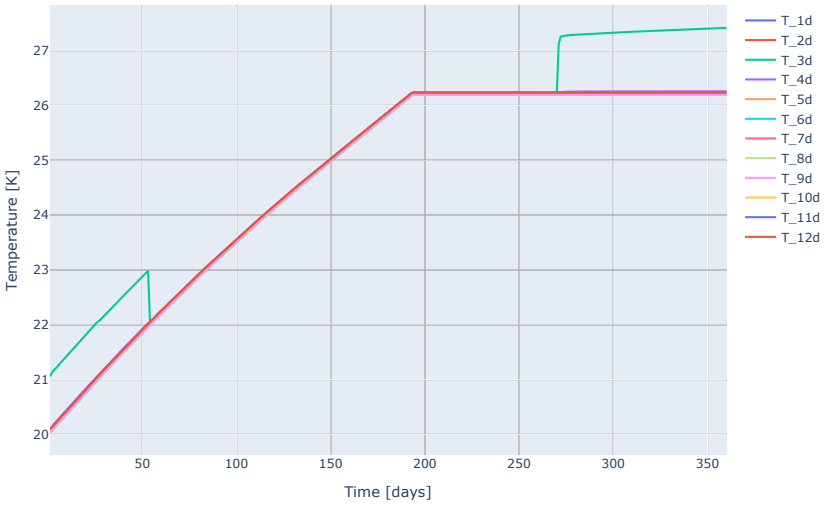

Figure 4.15: Temperatures of the interface nodes between the propellant tank structure and the liquid and vapour over time.

4.2. Sensitivity analysis

The simulations performed, use the assumptions listed in chapter 3. Although the functioning of the different aspects of the tool has been analysed, it is key to understand the effect of estimating a parameter more or less accurate. A sensitivity analysis is done to quantify the effect of changing a parameter on the result. This analysis is performed for three groups of parameters: the material and liquid properties in section 4.2.2, the numerical properties in section 4.2.1 and the design parameters in section 4.2.3.

4.2.1. Numerical properties

The two numerical properties independent in the program are the number of nodes in which the propellant tank structure is split and the time step taken in the simulations. The program has been set to simulate 4 different cases for both analyses. The 4 conditions are combinations of different propellant tank sizes and amount of MLI layers. This is done to see the sensitivity of the parameters not only for one setting, so a generalised conclusion can be drawn. In table 4.2 the four cases are given. All the other properties used are similar to the propellant depot case given in section 5.1.

Table 4.2: Case description for the cases used to see numerical sensitivity of the amount of nodes and time step used.

	$M_{propellant}[kg]$	$layers_{MLI}[-]$
case 1	34600	20
case 2	34600	10
case 3	17300	20
case 4	17300	10

In table 4.3 the boil-off rate is seen for the four different settings. The mission time has been set to 12 months. The other settings can be found in the case definitions given in section 5.1.

Table 4.3: Boil-off in [kg] for 6 cases and different time steps.

-	$dt = 1.$ [s]	$dt = 1.5$ [s]	$dt = 2.$ [s]	$dt = 2.5$ [s]	$dt = 3.$ [s]	$dt = 3.5$ [s]	$dt = 4.$ [s]
Case 1	4648.0	4648.0	4648.0	4648.0	4648.0	4648.0	4648.0
Case 2	12394.7	12394.7	12394.7	12394.7	12394.7	12394.7	12394.7
Case 3	2454.3	2454.3	2454.3	2454.3	2454.3	2454.3	2454.3
Case 4	6440.4	6440.4	6440.4	6440.4	6440.4	6440.4	6440.4

The range of time steps given in the table 4.3 has an upper limit of $dt = 4[s]$. With a larger time step,

it is found that the tool does not give a solution. This is due to a numerical instability which is called overshooting. Overshooting means that in a first time-step, the change between the two nodes is big since the time-step is relatively big. In a second time step, the program should compensate for this, but with overshooting the program overcompensates to the other end, resulting in a dis-convergence which grows over time.
In the table, it is observed that there is no difference in the boil-off between the steps up to $dt = 4[s]$ for any of the cases analysed. It can be concluded that for these cases the solution is independent of the time step chosen and therefore it is expected that the solutions given by the tool are independent of the time step used, as long as the time step is smaller or equal to 4 [s] per step. The following strategy is suggested. First, run the complete set of variables with a time step of 3 or 4 seconds. From the results of the analysis, the options with ZBO and the options with a high boil-off rate are known. For the region in between, the design options should be recalculated with a time step of 1 [s], to check the validity of the solutions for any case. With this information, it is possible to conclude which design option is most mass efficient without it being necessary to simulate all options with a small time-step.

Next to the time step, also the amount of nodes is a numerical property. To observe whether the solution is independent of the number of nodes used, the four cases simulated for the time step are again simulated, but now with a constant time step of $dt = 1.[s]$. In figure 4.16 the heat flow to the liquid bulk, the temperature of the liquid bulk and the boil-off mass of case 1 is given over time.

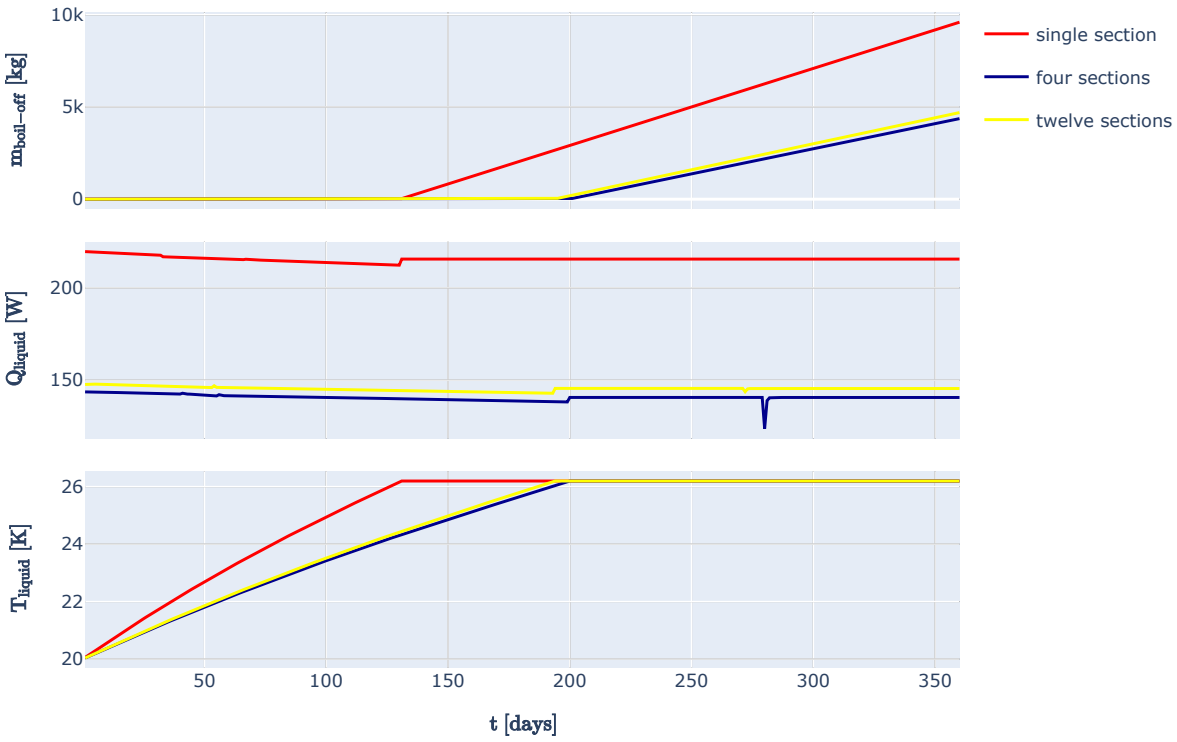

Figure 4.16: Boil-off mass in [kg], temperature of LH2 in [K] and heat flow to the liquid in $[W/m^2]$ for case 1 over time in [days] for 1, 4 or 12 propellant tank sections.

Table 4.4: Outer shell temperatures in [K] for case 1 with the propellant tank split in 1, 4 or 12 tank sections.

sections	**1**	**4**	**12**
$T_{average}$	171.5	112.7	112.6
T_{low}	-	24.3	23.5
T_{high}	-	173.5	206.7

In figure 4.16 it is observed that there is a 100% difference between the boil-off mass at the end between

the single section simulation and the four and twelve section simulation. This is explained by looking at the two other graphs. The heat flow to the liquid over time is 50% higher for the single section simulation compared to the two other simulations. Also, the time at which boil-off starts is around 60 days sooner. This is also explained by the higher heat flow.
To explain the higher heat flow, the average outer shell temperature is analysed. In figure 4.4 the average, hot- and cold side temperatures for case 1 for the three set-ups are given. These temperatures are constant over time and are therefore tabulated rather than plotted. The average temperature is given by the summation of the temperature of the section times the area of the section divided over the total area. This gives the area-averaged temperature:

$$T_{avg} = \frac{\sum T_{section} A_{section}}{A_{total}}$$

From table 4.4 it is observed that the average shell temperature is higher for the single section set-up. For a single section, the temperature of the outer layer is constant around the fluid. The heat rejection from the propellant tank to space is underestimated since heat conduction from section to section is not modelled and therefore all the heat flows from the structure to the liquid. For the propellant tank split into 4 or 12 sections, not all the sections are heated by the Sun. Therefore, these sections function as a heat sink for the sections which are illuminated by the Sun. Although the hot nodes of the 4 and 12 sections simulations have a different temperature, the average temperature is about equal and therefore the boil-off is similar.

In figure 4.17 the results of the 1, 4 and 12 sections simulations of case 2 can be observed. From this figure, similar conclusions can be drawn as for case 1. The two spikes visible in the heat flow to the liquid at 160 and 300 days are numerical errors, but it is seen from the graphs that those are restored right after.

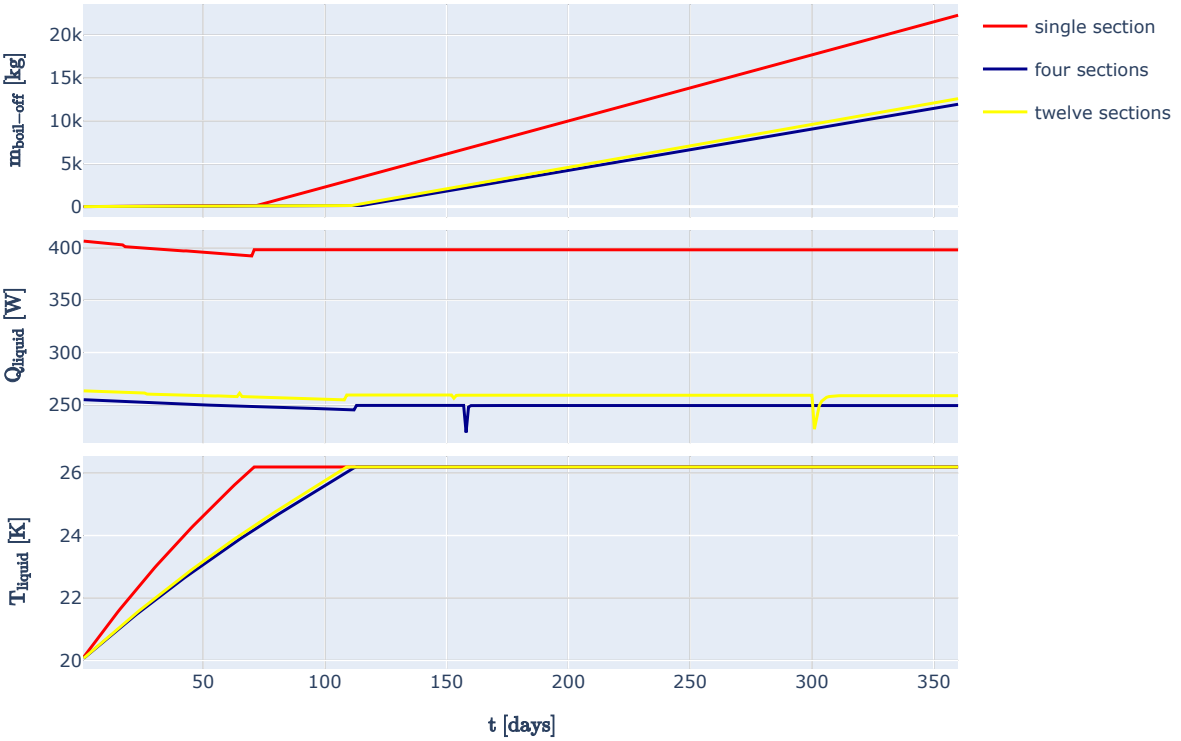

Figure 4.17: Boil-off mass in [kg], temperature of LH2 in [K] and heat flow to the liquid in [W/m^2] for case 2 over time in [days] for 1, 4 or 12 propellant tank sections.

In figure 4.18 the results of the 1, 4 and 12 sections simulations of case 3 are presented. As for case 1 and case 2, the boil-off mass of the single section simulation is highest. For the 4 sections and

12 sections simulations, the boil-off mass differs 21 % over a year. Compared to the previous two cases, the heat flow to the liquid of the 4 and 12 sections simulations are not about equal. Also, if table 4.5 is observed, it is seen that the average shell temperature is different between the simulations. The average shell temperature of the 4 sections simulation is higher. This is explained by the top and bottom node having a higher average temperature together than the top and bottom nodes of the 12 sections, which has half of the nodes in the shade side. This effect is less for case 1 and case 2 since the cylindrical sections are larger and therefore have more influence on the average temperature.

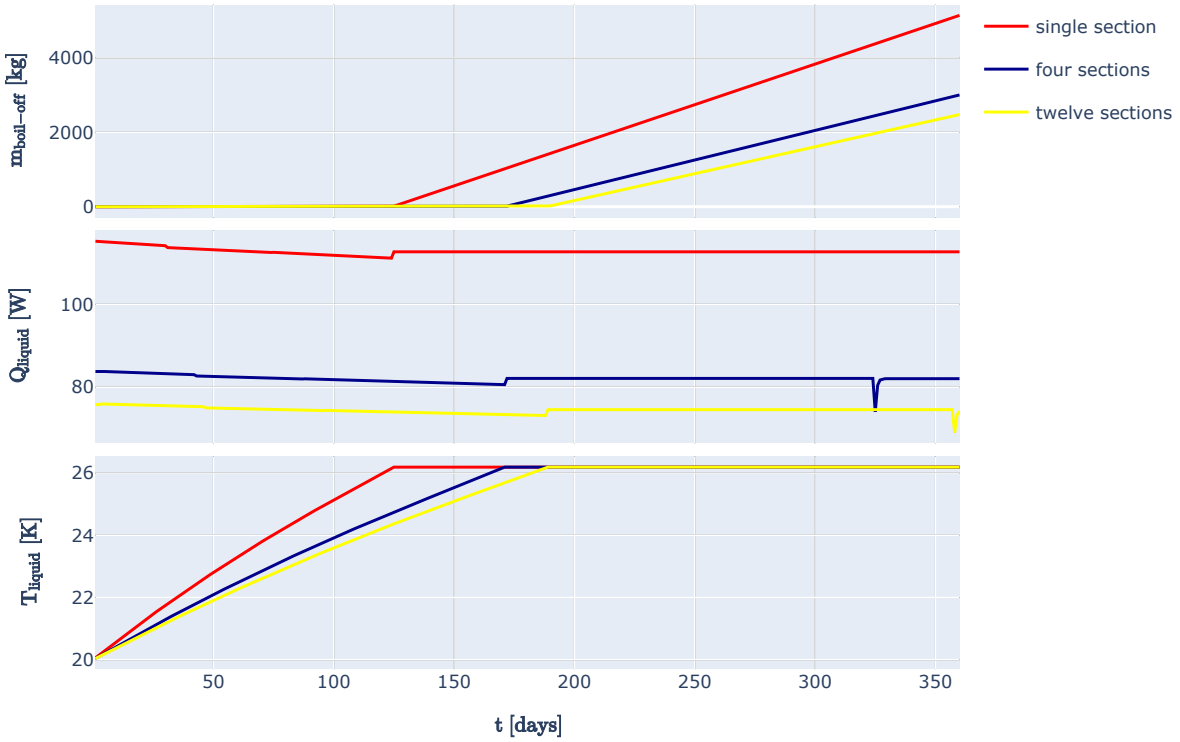

Figure 4.18: Boil-off mass in [kg], temperature of LH2 in [K] and heat flow to the liquid in [W/m^2] for case 3 over time in [days] for 1, 4 or 12 propellant tank sections.

Table 4.5: Outer shell temperatures in [K] for case 3 with the propellant tank split in 1, 4 or 12 tank sections.

sections	**1**	**4**	**12**
$T_{average}$	169.6	125.0	110.5
T_{low}	-	25.7	23.5
T_{high}	-	173.5	206.7

Similar observations which are done for case 3 can be done for case 4 in figure 4.19 and table 4.6. Comparing case 3 and case 4, it can be concluded that the difference does not increase with a higher boil-off rate since the difference in boil-off mass for case 4 between 4 sections and 12 sections is 16 % for a mission of 1 year. Further, it is also seen in table 4.6 that the average temperature of the outer shell is lower for more sections. This explains the differences in heat flow to the liquid, boil-off mass and the difference in slope of the rising liquid temperature.

The results of the sensitivity analysis to the nodes indicate that the solution is not independent of the number of sections used. However, the difference is reducing with more sections used. For a larger propellant tank, the difference between the number of sections is reducing. The first observation indicates that there should be an optimum amount of sections used compared to the complexity of the system. Due to time constraints of this thesis, it is not further analysed what the effect would be of using more sections.

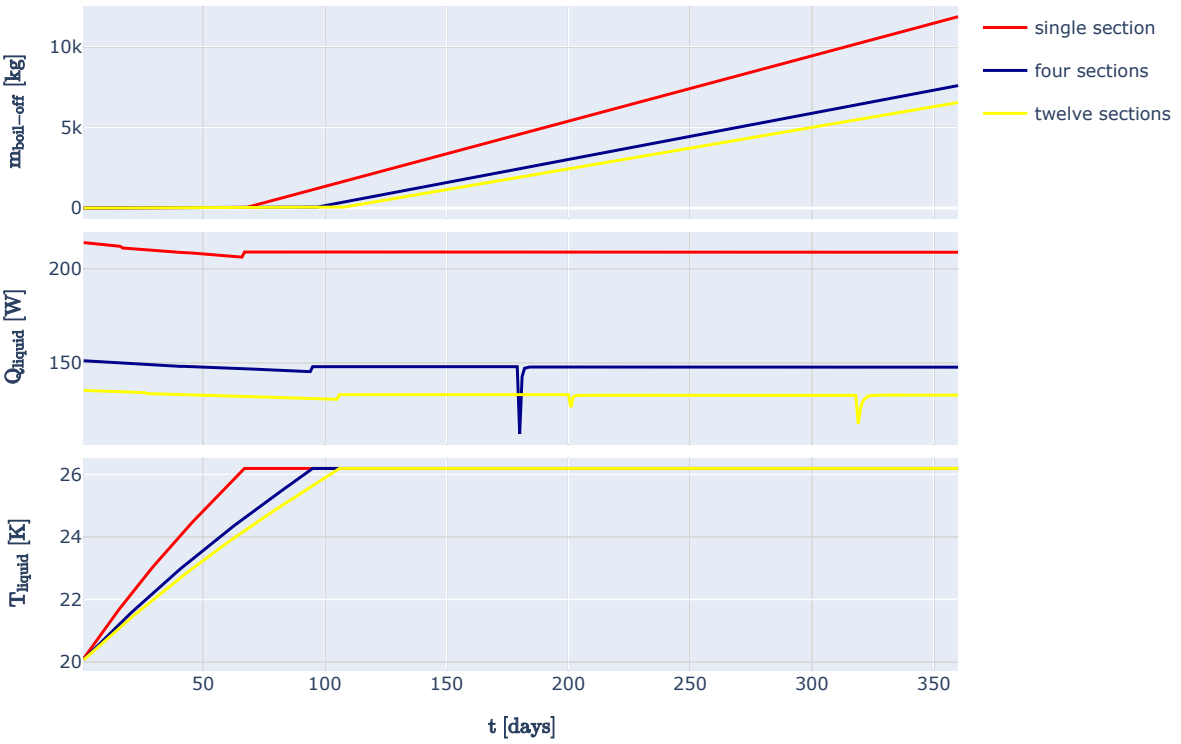

Figure 4.19: Boil-off mass in [kg], temperature of LH2 in [K] and heat flow to the liquid in [W/m^2] for case 4 over time in [days] for 1, 4 or 12 propellant tank sections.

Table 4.6: Outer shell temperatures in [K] for case 4 with the propellant tank split in 1, 4 or 12 tank sections.

sections	**1**	**4**	**12**
$T_{average}$	169.1	124.4	110.0
T_{low}	-	24.9	24.2
T_{high}	-	173.0	206.2

Using one node instead of multiple results in underestimating the performance of propellant tank design. From the cases done, it is seen that the single node always gives the highest boil-off rate and highest average outer shell temperature. Heat rejection by the cold side of a spacecraft cannot be determined by a single node model.
From the information provided by Chai and Wilhite [33] it seems that a single node model is used. The single components from which the work exists are validated by comparing to previous studies and relationships used. Nevertheless, validation as a whole is not done and therefore it cannot be said whether a single node would suffice in simulating a propellant tank. In section 4.3 a validation strategy is provided to eventually find out whether a single node model would be good enough to size a spacecraft.

4.2.2. Material and liquid properties

The material and liquid properties are retrieved from different databases, with the sources given after the values are mentioned. The liquid properties are all retrieved from the NIST database. Most properties vary with temperature. Therefore, for every physical property, a trade-off should be made between keeping the parameter constant during the simulation or retrieving the property for every change in temperature. The latter will be more accurate but comes with a time penalty on the simulation. In this section, the effect of changing these properties is analysed. With the knowledge on the effect of changing a parameter, an educated decision is made on whether to vary a parameter with temperature or whether to keep the property constant during the simulation. The parameters of liquid hydrogen are taken from the NIST database.

The properties for which the sensitivity analysis is performed are the specific heat of the MLI, and the SOFI; the coefficient of conduction of the shell, the MLI and the SOFI; the latent heat of vaporisation of liquid hydrogen, the emissivity of the MLI and the SOFI and the densities of the MLI and the SOFI. The results of the analysis are shown in figure 4.20. On the x-axis, the percentage change of the boil-off rate per month is presented with on the y-axis the different parameters. The left side of the x-axis is the sensitivity with a 10 % decrease of the property compared to the initial value, while the right side of the axis is the sensitivity compared to a 10 % increase of the property. A 10 % change is not always realistic for example in case of the latent heat of vaporisation, which would be known within that margin, but it gives a proper indication whether a parameter has much influence or not. The baseline set up for the simulation is the set up described as the propellant depot case. The initial values of the parameters are listed in table 4.7.

Table 4.7: List of initial values of the material and liquid properties used for sensitivity analysis.

- $\rho_{SOFI} = 38.44[kg/m^3]$
- $\epsilon_{SOFI} = 0.3[-]$
- $k_{MLI} = 0.24[W/m/K]$
- $k_{SOFI} = 0.02[W/m/K]$
- $k_{shell} = 6.7[W/m/K]$
- $C_{MLI} = 1170[J/kg/K]$
- $C_{SOFI} = 1300[J/kg/K]$
- $\rho_{MLI} = 0.047[kg/m^2/layer]$
- $\epsilon_{MLI} = 0.03[-]$

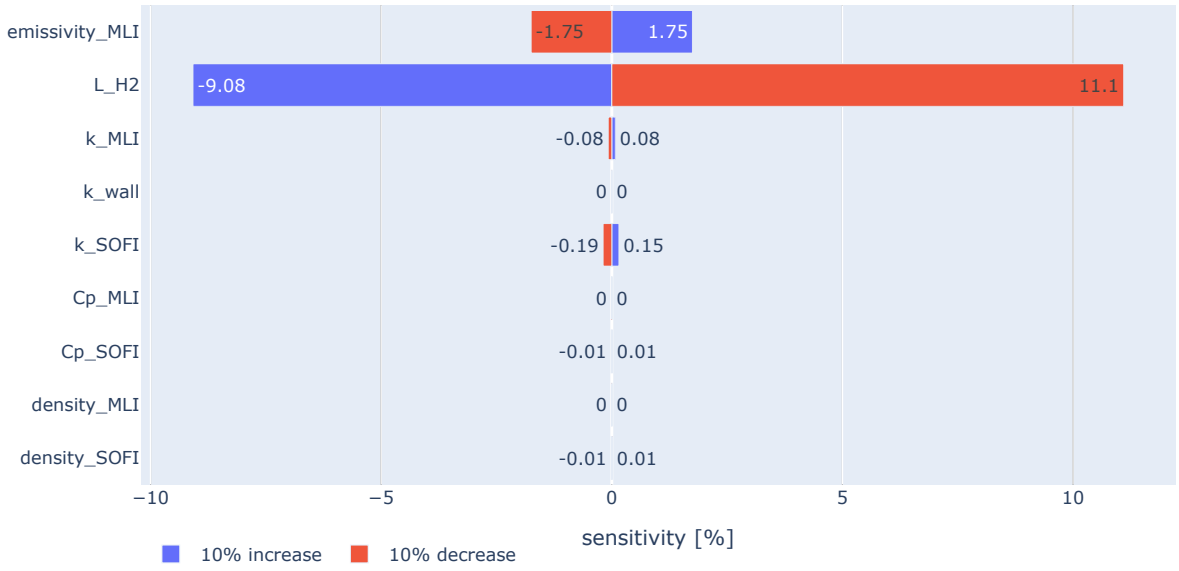

Figure 4.20: Sensitivity analysis on the physical properties with a 10% increase or decrease of the parameter.

In figure 4.20 the sensitivity in percentage is presented on the horizontal axis, while the different parameters are presented on the vertical axis. From figure 4.20 it is seen that most physical properties do not influence the result of a 6-month simulation if changed by 10%. Changing the latent heat of vaporisation of the fluid with 10% results in an 11.1% increase in boil-off if the change decreases the value and a 9.04% decrease in boil-off if the change increases the value. The emissivity of the MLI also influences the performance by 1.75 % for a 10% to any side.

4.2.3. Design parameters

The design parameters are to be set before every simulation. Some design parameters can have a great effect on the boil-off, while others do not. For an equal volume, a spherical propellant tank has an advantage compared to a cylindrical propellant tank, since the volume over area ratio is larger. The

smaller the area is, the less heat flow can enter the propellant tank. In figure 4.21 the sensitivity analysis on the design properties is presented. The design properties considered are the initial temperature, the initial pressure, the maximum radius of the propellant tank, the mission duration, the propellant fill level, the thickness of the shell, the emissivity of the outer coating, the absorptivity of the outer coating, the solar incidence angle, the gravity and the solar flux. All these parameters are again changed by 10% plus and minus to estimate their effect on the boil-off rate. The initial design parameters are listed, the other parameters used are defined in the propellant depot case in section 5.1.

- $T_{initial} = 20.[K]$
- $p_{initial} = 300000.[Pa]$
- $r_{tank} = 2.7[m]$
- $t_{mission} = 6[months]$
- $fill\_level = 0.9[-]$
- $t_{shell} = 0.005[m]$
- $\epsilon_{coating} = 0.66[-]$
- $\alpha_{coating} = 0.08[-]$
- solar incidence angle = 180 deg
- $g_{GSO} = 0.22[m/s^2]$
- $q_{Sun} = 1350[W/m^2]$

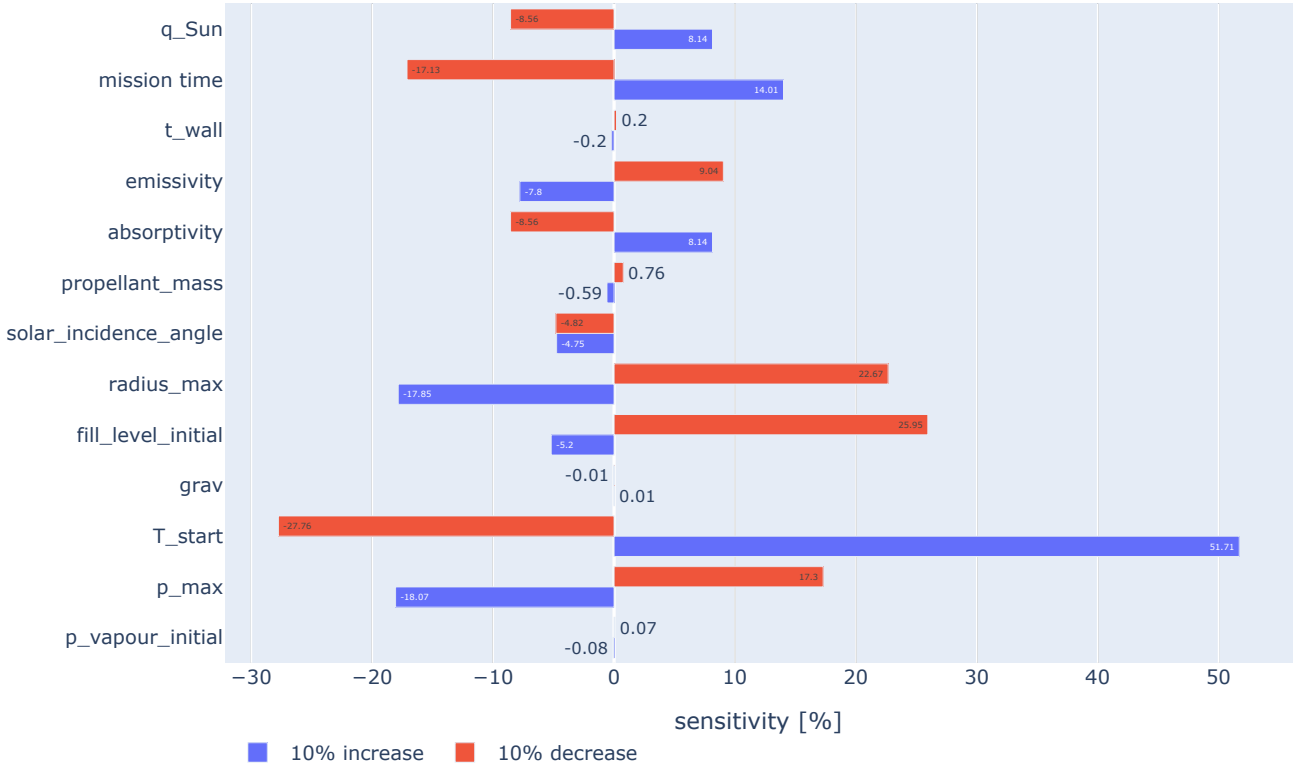

Figure 4.21: Sensitivity analysis on the design properties with a 10% increase or decrease of the parameter.

In figure 4.21 the sensitivity of some design parameters is given on the horizontal axis, with the different design parameters on the vertical axis.
For the time-sensitivity, the result gives that a 10 % decrease in mission duration will reduce the boil-off by 17 %, while a 10 % increase of the mission duration will increase the boil-off by 14 %. This difference is explained by realising that the start time of the boil-off would be the same for any design and is not depending on the mission duration. A longer mission will have more boil-off and the average boil-off will be higher, since the period in which there is no boil-off is as long for any mission duration if the design is equal.
From the sensitivity analysis bar chart it is apparent that some design parameters have a major influence on the performance of the propellant tank. As expected has the maximum radius of the propellant tank significant influence on the result. A larger radius results in a higher volume to area ratio and thereby for smaller heat flow to the propellant tank for an equal volume, since the area becomes smaller

and the area multiplied with the solar flux gives the heat flow to the propellant tank.
The design parameter with the highest sensitivity is the initial temperature. If the initial temperature changes, also the propellant tank size changes due to the density changing. With a lower temperature, there is a higher density and the propellant tank becomes smaller. A smaller propellant tank has a higher area over volume ratio, which has a negative influence on the boil-off rate.

$$V = 4/3\pi R^3 + \pi R h$$

$$h = \frac{V - 4/3\pi R^3}{\pi R^2}$$

$$A = 4\pi R^2 + 2\pi R h = 4\pi R^2 + 2\pi R \left(\frac{V}{\pi R^2} - 4/3\pi R \right)$$

Nevertheless, decreasing the initial temperature of the liquid by 10% reduces the boil-off by 28%. Increasing the temperature would have the opposite effect, although the propellant tank size increases due to the aforementioned analogy, a 10% increase in initial temperature increase the boil-off by 52%. If the propellant tank size is maintained, there is another factor negatively influencing the boil-off rate. Due to the higher density, the liquid volume is less and for a constant tank volume, the fill level decreases. From figure 4.21 it can also be observed that a reduction in the fill level will increase the propellant tank boil-off rate.

The initial vapour pressure has a limited effect on performance. This can be explained by the fact that a lower or higher initial pressure will only hold for a short time, due to the vapour rising in temperature. Any advantage will be lost after a short amount of time compared to the total mission time.
Increasing the maximum allowable pressure does positively influence the performance. Higher allowable pressure results in a higher saturation temperature and therefore in a larger heat capacity before boiling is initiated. The downside is increasing the allowable pressure is the increase in structural mass due to stresses in the shell. In this sensitivity analysis, this is not taken into account.

Gravity does not influence the boil-off rate of the simulation. This will hold as long as stratification is not taken into account by assuming a mixer is present. When stratification occurs, it is expected that gravity influences the boil-off rate.

Both the solar radiation and the orientation of the spacecraft propellant tank influence the boil-off. A 10% increase or decrease in solar radiation power increases or decreases the boil-off of the propellant tank by 8% for a mission duration of 6 months. The orientation of the spacecraft is also important. Changing the solar incidence angle from 180 degrees to 162 or 198 degrees reduces the boil-off by 5%. This is explained by the fact that for 180 degrees, the biggest frontal area is illuminated. This becomes less for any change in angle, except for -180 degrees, which would illuminate the opposite side of the propellant tank.
The absorptivity and the emissivity of the outer coating of the propellant tank can be used to increase or decrease the boil-off. A 10% change in absorptivity will change the boil-off by about 8% to the equal side (increase or decrease). For the emissivity, the opposite is seen. A 10% increase of the emissivity will result in an 8% decrease of boil-off since more infrared radiation is emitted from the propellant tank. A 10% decrease of the emissivity will reduce the infrared radiation emitted and will result in a 9% increase of the boil-off.

In section 6.1 it is mentioned that the real Centaur has a shell thickness of 0.0005 [m], but the boil-off tool does not such a small node size. Therefore a 0.005[m] shell thickness was used. From the sensitivity analysis it appears that as expected, the shell thickness has little influence on the boil-off performance of a design.

4.3. Validation

A verified program delivers an output as desired, but the validity of the output is not checked yet. To ensure that the results delivered are accurate to reality, validation of the program has to be done. The

definition given in the course AE4S12 [79] for validation is: ”Proof by, an examination of objective evidence, that the product accomplishes the intended purpose. Validation is performed to ensure that the product is ready for a particular use, function, or mission.”.
In [81] it was stated that the biggest issue with the numerical research to boil-off done is the lack of validation. Although cryogenic hydrogen is seen as the propellant for the future, there is still no flight data (publicly) available to validate tools as build with. Therefore, in this section, only a strategy will be provided on how to validate this the program once suitable data is available.

Validation of a simulation tool is often done by comparing experimental test data with the output of the simulation. An experiment can be performed to see whether the output of the simulation is representative, or a simulation can be adjusted to an existing experiment to see whether the output matches the results of the experiment. Large scale experiments have been performed at NASA's Kennedy Space Centre and NASA's Marshall Space Flight Centre, there is no test data nor experimental data for cryogenic propellants used during space missions. At the TU Delft, Ir. Hermsen has done experiments with pressure development in a cryogenic nitrogen tank [82]. The data of the experiments performed by NASA is not publicly available and the data provided by Hermsen is not suitable for validation of the Boil-off Monte Carlo program.

To validate the program created, space mission data is necessary. The data which should at least be available contains the propellant mass over time, the heat flow to the liquid over time, the orientation of the Sun to the propellant tank over time, the altitude of the propellant tank over time, the tank pressure over time and the liquid temperature over time. With these parameters known next to the design parameters of a propellant tank, sufficient information should be available to reconstruct the propellant tank in the program and propagate the time with the environmental conditions given. If the temperature, pressure and liquid mass output over time given by the program match the space mission data, the program is validated for the conditions given.

Although it is not possible to validate the results of the simulation, the output of the simulation can still be valuable. The simulation results can be used as a guide to preliminary design. Also, it is possible to verify the results of other researchers. The exact results of other research may not be reproducible utilizing this simulation, but the conclusions can be verified.

One of the cases considered in this thesis is the extendability of the Centaur upper stage for mission time in space. Although a real validation is not possible, the results from the simulation can be used to state what difference there is between the simulation and reality and what can be done on the long term to improve the tool, so it can be validated. In chapter 6 the results will be presented and discussed.

5

Propellant depot case

In this chapter, the results for a propellant depot case will be given and discussed. The case is described in section 5.1. The results generated by using the parameters given in that case description are presented in section5.2. In section 5.3 the results for the propellant depot are discussed.

5.1. Propellant depot case description

The first case study is a propellant tank depot as proposed by Chai and Wilhite [33]. This case study is chosen because the environment of the depot is constant. The propellant tank depot orbits the Earth in GSO. In GSO, the effect of planetary radiation is negligible[1]. Only solar radiation of a magnitude $q_{sun} = 1350[W/m^2]$ is used, constant over the mission time. The temperature around the spacecraft is assumed to be the outer space temperature of $T_{space} = 3[K]$. In GEO, the gravity of Earth is $g = 0.22[m/s^2]$.

The fuel mass carried by the depot is $M_{fuel} = 34,600[kg]$. The shell is made of $t_{shell} = 0.005[m]$ thick Titanium (Ti-6Al-4V). The specific heat coefficient for this Titanium alloy is $C_p = 526.4[J/kg/K]$, the coefficient of conduction is $k_{shell} = 6.7[W/m/K]$. The density of the Titanium alloy is $\rho_{shell} = 4430[kg/m^3]$ [83].

The SOFI used is NCFI 24-124 foam. A constant coefficient of conduction of $k_{SOFI} = 0.02[W/m/K]$ is used [65], this is the conductivity of the foam when it exceeds the 100 [K] temperature. The density of the NCFI 24-124 foam is $\rho_{SOFI} = 38.[kg/m^3]$ [84]. The heat capacity of the foam is not found, but similar polystyrene foams have a heat capacity of $C = 1300 - 1500[J/kg/K]$ [2], the minimum value of $C_{SOFI} = 1300[J/kg/K]$ is used since that would give the worst result. It was identified in [32] that SOFI is effective on Earth, but MLI is more effective in reducing the heat flow in space. Therefore, only a limited number of SOFI thicknesses are simulated. The thicknesses used are 0.01, 0.02, 0.03 and 0.05 [m].

The modified Lockheed equation discussed in section 2.4 gives the heat flow through an N_{MLI} number of MLI layers with a MLI layer density $\tilde{N}_{MLI}$. This is for Double Aluminized Mylar (DAM) with Dacron Net Spacers, a type of MLI produced by Sheldahl[3] and Dunmore[4] [85]. From Johnson [62] the optimum layer density for a cold side temperature of $T_C = 20[K]$ and a $T_H \approx 400[K]$ for DAM MLI is $\tilde{N}_{MLI} = 16[layers/cm]$. The case is initially analysed with 10, 20, 40, 50, 75, 100, 125 and 150 layers of MLI. Later 25, 30, 35, 45, 55 and 60 layers of MLI are also included.
DAM MLI has an emissivity of $\epsilon_{MLI} = 0.03[-]$ [85]. From Johnson [86] it comes that the density of MLI is $\rho_{MLI} = 0.047[kg/m^2/layer]$. The specific heat coefficient of DAM is not found, however, the biggest part of DAM consists out of Mylar. Therefore, the specific heat coefficient of Mylar is used:

[1] $q_{IR} + q_{albedo} = 237 * (6371./(6371. + h_{orbit}))2. + 0.306 * 13502 * f * 0.08 << 1350$
[2] https://www.engineeringtoolbox.com/specific-heat-capacity-d_391.html
[3] https://www.sheldahl.com/technology/thermal-control-materials/multi-layer-insulation
[4] https://www.dunmore.com/industries/aerospace.html

$C_{MLI} = 1170[J/kg/K]$ [87]. The thermal conductivity to the side nodes is also taken into account. The coefficient of conduction of the MLI is $k_{MLI} = 0.24[W/m/K]$.
In [88] it is concluded that the lower the interstitial gas density between layers of MLI, the lower the heat flux. Since degassing will be crucial for most missions which require cryogenic propellants, it is assumed that the interstitial gas pressure is close to vacuum: $10^{-6}[torr]$, which is $1.33 * 10^{-5}[Pa]$.

The propellant depot carries a continuous cooling cryocooler. The cooling power of the cooler was initially 0, 10, 20, 30, 40, 50, 75 or 100 [W]. Later 35 amd 45 [W] were added for some number of MLI layers.
A cooling power of more than 20 [W] at 20 [K] is not feasible yet, but it is expected that cryocooler development will make 100 [W] cooling power at 20 [K] possible [33]. The mass of the cooler is calculated with equation 2.28 and includes the additional mass of the power system as well.

Table 5.1: List of input parameters used for the propellant depot simulation.

- $Q_{cooler} =$ See text
- $M_{fuel} = 34,600[kg]$
- $T_{initial} = 20[K]$
- $p_{initial} = 300000[Pa]$
- $r_{tank} = 2.7[m]$
- $dt = 4[s]$
- $t_{mission} = 60 * 60 * 24 * 30 * 24 = 62208000[s]$
- $fill\_level = 0.9[-]$
- $t_{SOFI} =$ See text
- $\rho_{SOFI} = 38.44[kg/m^3]$
- $k_{SOFI} = 0.02[W/m/K]$
- $C_{SOFI} = 1300[J/kg/K]$
- $\epsilon_{SOFI} = 0.3[-]$
- $C_{shell} = 526.4[J/kg/K]$
- $k_{shell} = 6.7[W/m/K]$
- $\rho_{shell} = 4430[kg/m^3]$
- $t_{shell} = 0.005[m]$
- $N_{MLI} =$ See text
- $\tilde{N}_{MLI} = 16[layers/cm]$
- $\rho_{MLI} = 0.047[kg/m^2/layer]$
- $k_{MLI} = 0.24[W/m/K]$
- $C_{MLI} = 1170[J/kg/K]$
- $\epsilon_{MLI} = 0.03[-]$
- $p_{interstitial} = 1.33 * 10^{-5}[Pa]$
- $\epsilon_{coating} = 0.66[-]$
- $\alpha_{coating} = 0.08[-]$
- solar incidence angle = $180\,\mathrm{deg}$
- $g = 0.22[m/s^2]$
- $T_{space} = 3[K]$
- $q_{Sun} = 1350[W/m^2]$
- $q_{Earth-IR} = 0[W/m^2]$
- $q_{albedo} = 0[W/m^2]$

5.2. Results propellant depot simulation

Using the parameter ranges given in the case description, the program is used to simulate the propellant tank with the given characteristics over time. The simulation is done for a period of 24 months, with the intermediate results stored at 6 and 12 months. The results over time can indicate the behaviour changing with the mission time.

In this section, the total mass is defined by the boil-off mass plus the MLI mass plus the SOFI mass plus the cryocooler mass. These are the only masses changing per design and therefore indicate which design is the lightest concerning the mass. In the graphs presented in this chapter, black dots are the data points from the simulated designs.

5.2.1. 6 month simulation

The first set of results given is for a propellant depot orbiting the Earth in GSO for 6 months without refuelling. Two figures are given in this section, giving the combined MLI, SOFI, cooler and boil-off mass for passive only solutions and systems including a cryocooler. The first figure presented is the combined mass for passive insulation only systems.

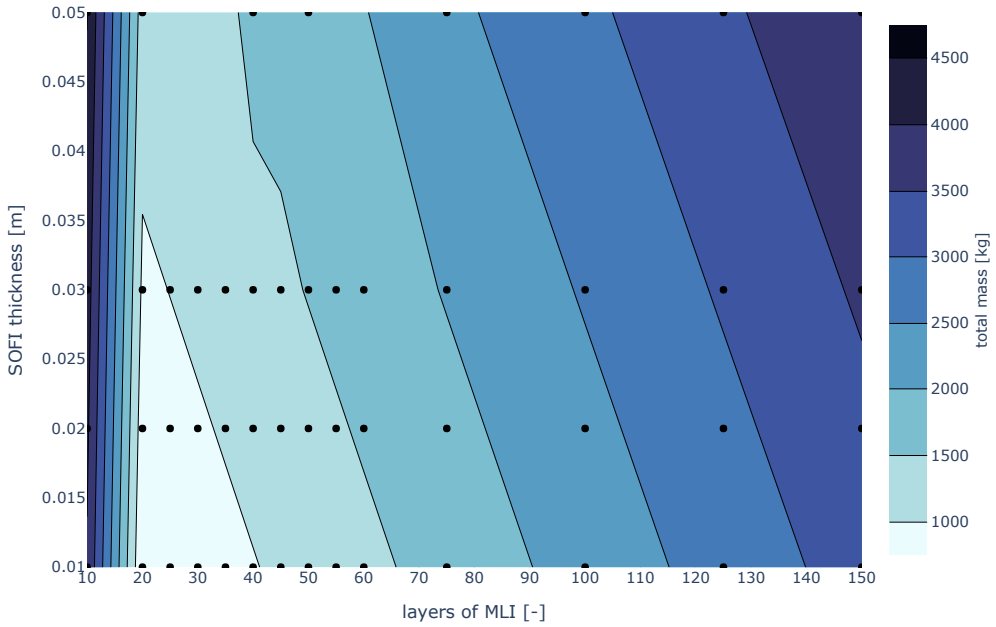

Figure 5.1: Total mass in kilogram for a mission time of 6 months with passive insulation only.

Figure 5.1 presents the number of MLI layers on the horizontal axis and the SOFI thickness in [m] on the vertical axis. The colour bar on the right-hand side indicates the mass in [kg]. From the figure, it is observed that the mass decreasing up to 20 layers of MLI, after which the mass increases again after more than 40 layers of MLI for a SOFI thickness of 0.01 [m] and increases after more than 25 layers for a SOFI thickness of 0.03 [m]. The mass increases with the SOFI thickness. The vertical lines present in the figure are a result of the gradient between the simulations with 10 layers of MLI and the simulations with 20 layers of MLI, as can be seen by the black dots present in the figure. The diagonal lines visible in the figure are a result of interpolation by the plotting program between the different data points. The area of interest is in the range of 20 to 60 MLI layers and a SOFI thickness of 0.01 till 0.03 [m]. Figure 5.2 focuses on this area.

In figure 5.2 a similar pattern is observable compared to the figure 5.1 area of more than 60 layers of MLI. The diagonal pattern is explained by the boil-off rate not reducing by a thicker layer of SOFI, but the mass increases due to the additional insulation material.

Figure 5.3 presents the mass for the simulations including a cryocooler and with a constant SOFI thickness of $t_{SOFI} = 0.01[m]$ for a mission duration of 6 months.

In figure 5.3 the number of MLI layers is given on the horizontal axis and the cooling power in [W] is given on the vertical axis. The colour bar on the right-hand side presents the mass in [kg]. As for the passive insulation only figure, it is observed that the mass decreases from 10 to 20 layers of MLI and after 20 layers of MLI, the mass increases again. The total mass also becomes more with more cooling power.

In table 5.2 the five designs with the least mass are given. From the table, it is observed that all five designs are passive only. Also, none of the designs has boil-off.

In the next section, the results of a mission of 12 months for the propellant depot will be given.

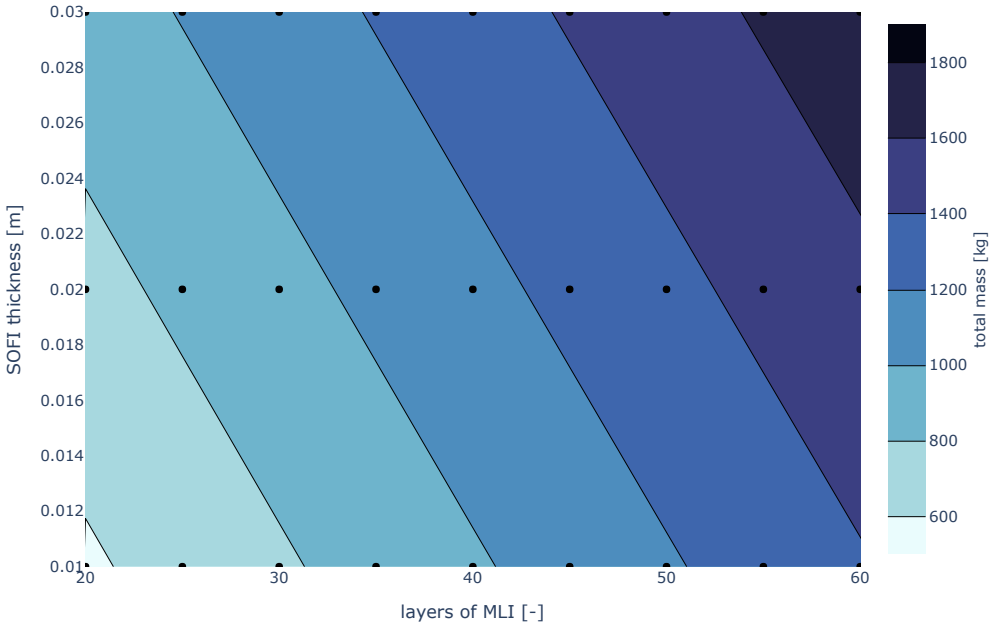

Figure 5.2: Focus area total mass in kilogram for a mission time of 6 months with passive insulation only.

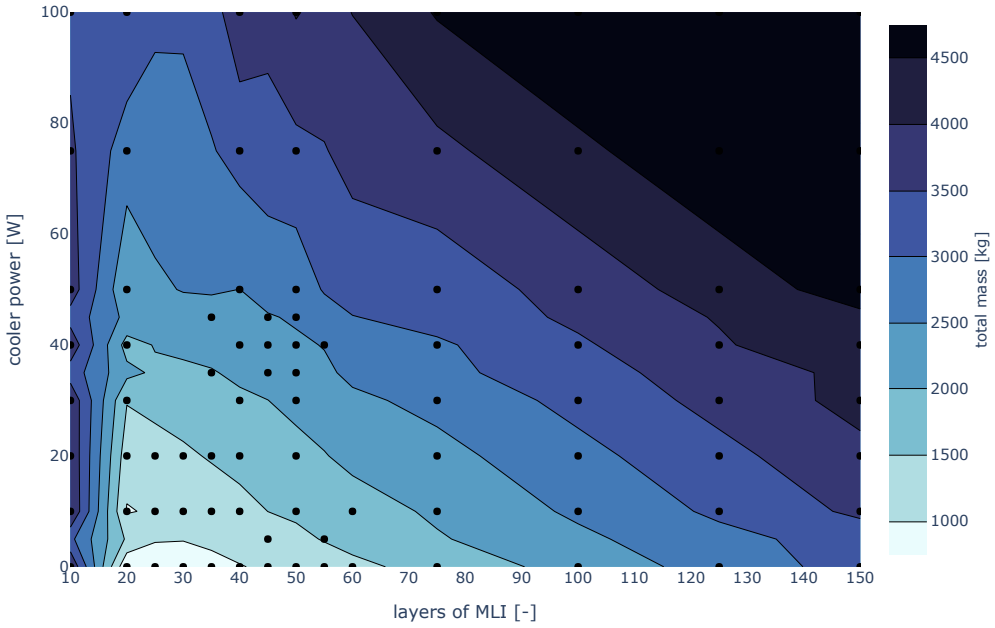

Figure 5.3: Total mass in kilogram for a mission time of 6 months with SOFI thickness of 0.01 [m].

Table 5.2: 5 designs with the least total mass for a propellant depot orbiting Earth in GSO for 6 months.

#	t_{SOFI} [m]	M_{SOFI} [kg]	layers of MLI [-]	M_{MLI} [kg]	Q_{cooler} [W]	M_{cooler} [kg]	Boil-off rate [%/month]	$M_{boil-off}$ [kg]	M_{total} [kg]
1	0.01	165	20	405	0	0	0	0	571
2	0.01	165	25	507	0	0	0	0	672
3	0.02	332	20	407	0	0	0	0	739
4	0.01	165	30	608	0	0	0	0	773
5	0.02	332	25	509	0	0	0	0	841

5.2.2. 12 month simulation

In this section, the results of the simulation representing a 12-month mission for the propellant depot will be given. In figure 5.4 the boil-off rate as a function of the thickness of the SOFI and the number of MLI layers is presented. On the horizontal axis, the number of MLI layers is given for an MLI layer density of 16 [layers/cm]. The vertical axis presents the SOFI thickness in meters. The bar on the right side of the figure gives the colour scale for the boil-off rate in [%/month].

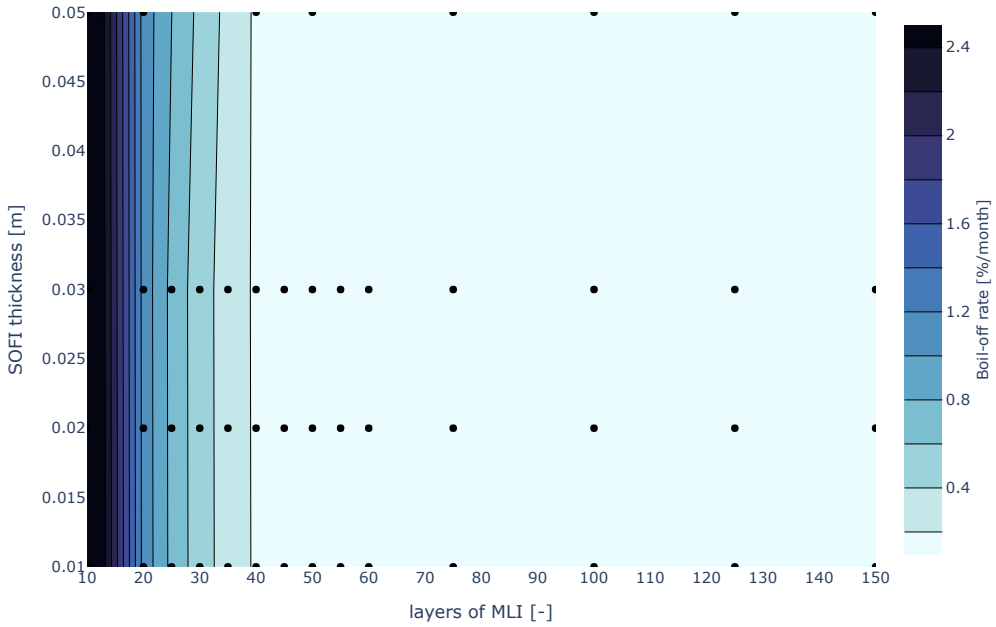

Figure 5.4: Average boil-off rate in percent per month for a mission time of 12 months with passive insulation only.

From the figure it is observed that the SOFI thickness does barely influence the boil-off rate. There is a trend that with more layers of MLI, the boil-off rate decreases up to 40 layers of MLI. With more than 40 layers of MLI, the boil-off rate is zero. The majority of options investigated does not have boil-off for a mission duration of 12 months. The boil-off rate of all the options investigated does not exceed 3.0 [%/month], which is 34.6 [kg/day].

Figure 5.5 presents the boil-off rate in per cent per month of the 12-month mission duration as a function of the number of MLI layers and the cooling power, with a constant thickness of SOFI of 0.01 [m]. The vertical axis presents the cooling power, while on the horizontal axis the number of MLI layers is given. The vertical bar to the right of the figure indicates the boil-off rate per colour in [%/month].
In figure 5.5 it is seen that the boil-off rate is zero for the design options with 0.01 [m] SOFI and more than 40 layers of MLI. It is observed that with increasing cooling power, the boil-off rate decreases. From the pattern visible in the figure an equation can not be provided to predict the boil-off as a function of the layers of MLI and the cooling power. The lines visible are a result of interpolation between the data points given in black.

Figure 5.6 gives the boil-off, SOFI and MLI mass together as a function of the SOFI thickness on the vertical axis and the number of MLI layers on the horizontal axis. The vertical bar on the right of the figure gives which colour is which mass in [kg]. There is no cryocooler used in the designs.
From the figure it is seen that adding SOFI to the design, does not safe mass for any number of MLI layers. By increasing the number of MLI layers the total mass is reduced up to 45 layers of MLI. With more than 65 layers of MLI, the mass increases again. This is a similar pattern as seen in figure 5.1. The centre of the pattern is shifted from 30 layers of MLI in the 6-month mission to 50 layers of MLI for the 12-month mission.

Next to the total mass as a function of SOFI thickness and MLI layers, figure 5.7 gives the total mass in

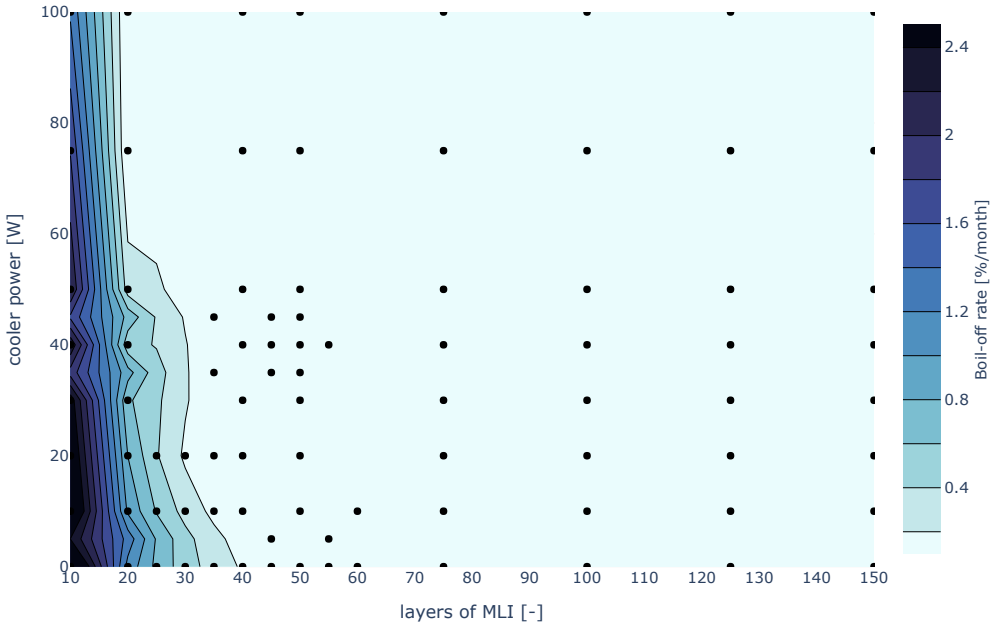

Figure 5.5: Average boil-off rate in percent per month for a mission time of 12 months with SOFI thickness of 0.01 [m].

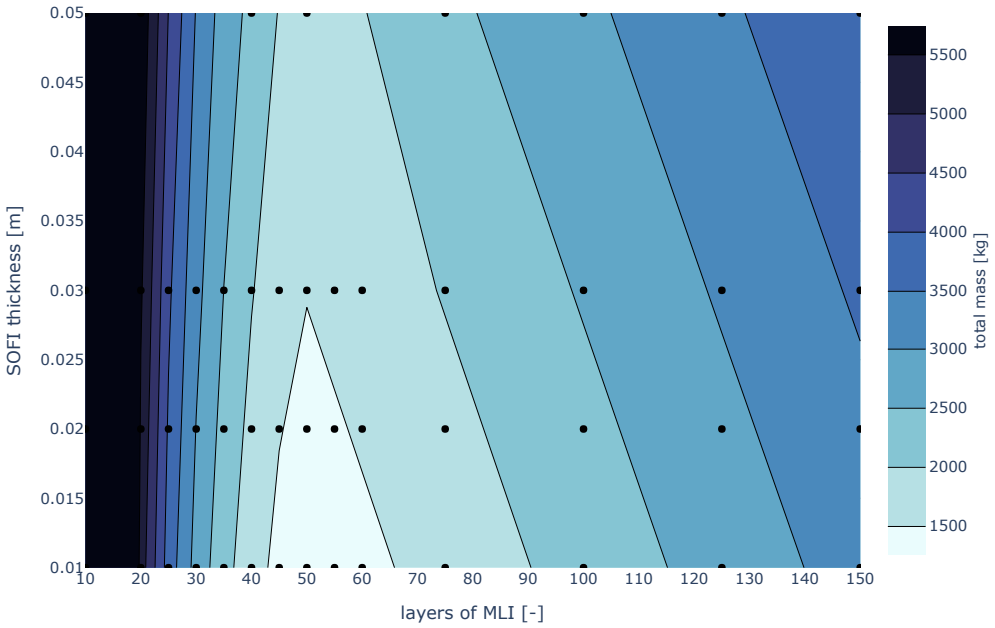

Figure 5.6: Total mass in kilogram for a mission time of 12 months with passive insulation only.

relation with the number of MLI layers and the cooling power of the cryocooler. The vertical bar gives the legend of the contour plot.
As for the previous figure, it is seen that with an increasing number of MLI layers the total mass decreases up to 40 and 50 layers of MLI if the cooling power is low. With 20 [W] cooling power or more, the mass only increases with more layers of MLI.

In table 5.3 the 10 designs which give the smallest total mass are shown. Only 3 out of the 10 best designs uses cooling power. The 3 best results all have a SOFI thickness of 0.01 [m]. All designs in the table have 40 to 60 layers of MLI, with most of the designs having 50 or 55 layers of MLI.

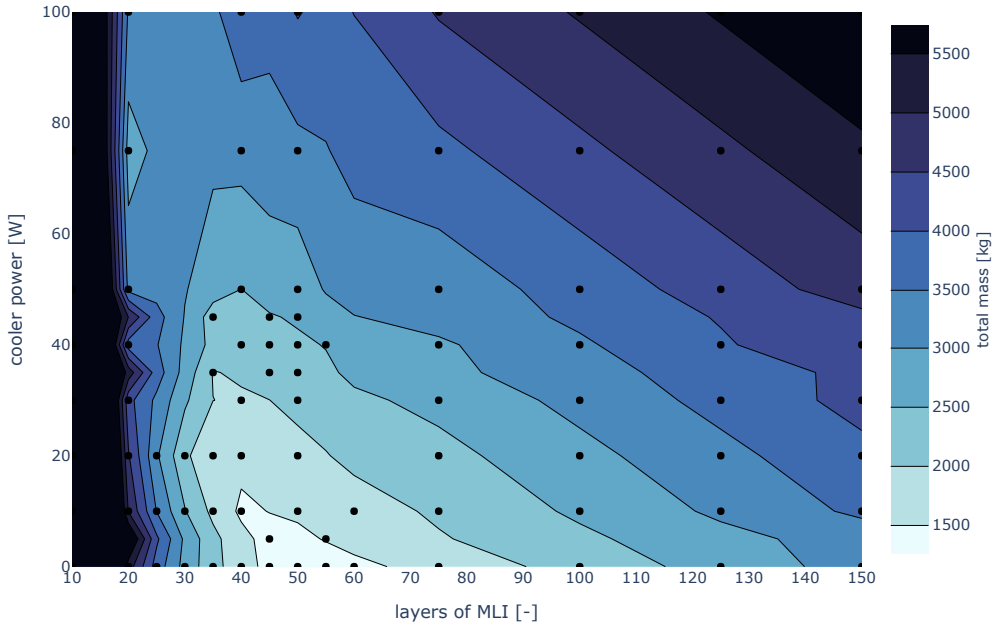

Figure 5.7: Total mass in kilogram for a mission time of 12 months with SOFI thickness of 0.01 [m].

Table 5.3: 10 lightest design options propellant depot design for a mission duration of 12 months.

#	t_{SOFI} [m]	M_{SOFI} [kg]	MLI layers [-]	M_{MLI} [kg]	Q_{cooler} [W]	M_{cooler} [kg]	$M_{boil-off}$ [kg]	Boil-off rate [%/month]	M_{total} [kg]
1	0.01	165	50	1013	0	0	0	0.00	1179
2	0.01	165	55	1115	0	0	0	0.00	1280
3	0.01	165	45	912	5	246	0	0.00	1323
4	0.02	332	50	1018	0	0	0	0.00	1350
5	0.01	165	45	912	0	0	282	0.07	1360
6	0.01	165	60	1216	0	0	0	0.00	1382
7	0.01	165	40	811	10	392	40	0.01	1409
8	0.02	332	55	1120	0	0	0	0.00	1451
9	0.03	498	50	1023	0	0	0	0.00	1521
10	0.01	165	55	1115	5	246	0	0.00	1526

5.2.3. 24 month simulation

With the results of the 6 and 12-month missions given, the mission duration is extended to 24 months. In this section, the results of a propellant depot orbiting the Earth in GSO for 24 months are given.

As for the previous sections, the first figure given in the section is the average boil-off rate in per cent per month as a function of SOFI thickness on the vertical axis and the number of MLI layers on the horizontal axis for a spacecraft without cooling power and with MLI of layer density 16 [layers/cm]. The vertical bar at the right side of the figure gives the boil-off rate.
Figure 5.8 demonstrates equal trends to figure 5.4. Overall the boil-off rate has increased compared to the 12-month mission. A similar pattern as in figure 5.4 is observed. From 40 layers of MLI up to 100 layers of MLI, it is seen that there is boil-off whereas there was no boil-off at 12 months. With more than 125 layers of MLI, no boil-off is observed. From the figure it is observed that the more layers of MLI are added, the less the boil-off rate decreases with adding an equal number of MLI layers.

From figure 5.9 it is seen that adding a cryocooler to the design mitigates boil-off to less than 0.4% per month boil-off if the cooling power is 100 [W] and the number of MLI layers is 40 or more. In this figure, the number of MLI layers is presented on the horizontal axis, while the cooling power is presented on the vertical axis. It is observed from the figure that adding cooling power is reducing the average boil-off

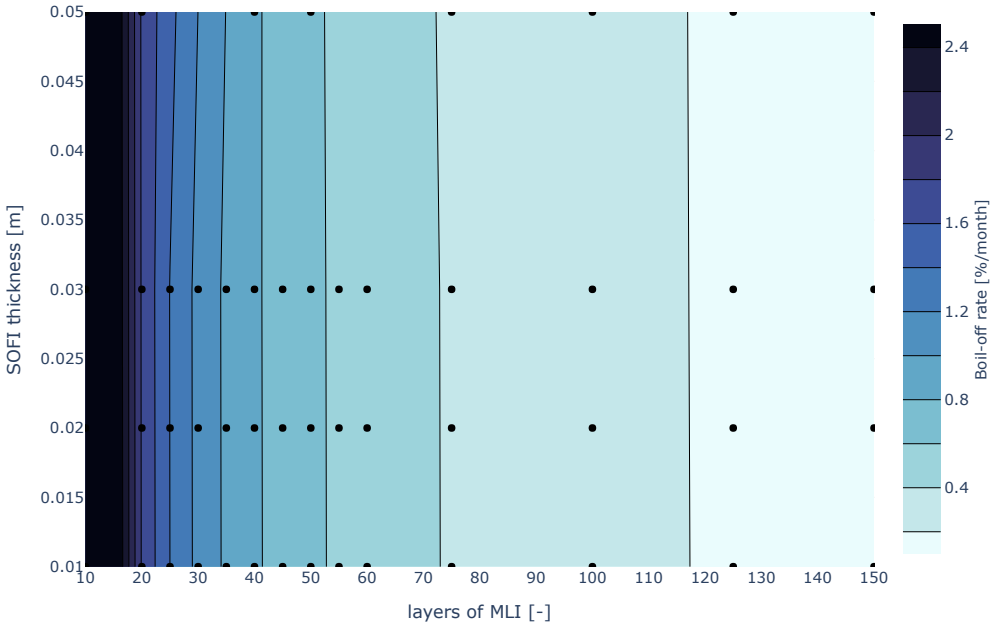

Figure 5.8: Average boil-off rate in percent per month for a mission time of 24 months with passive insulation only.

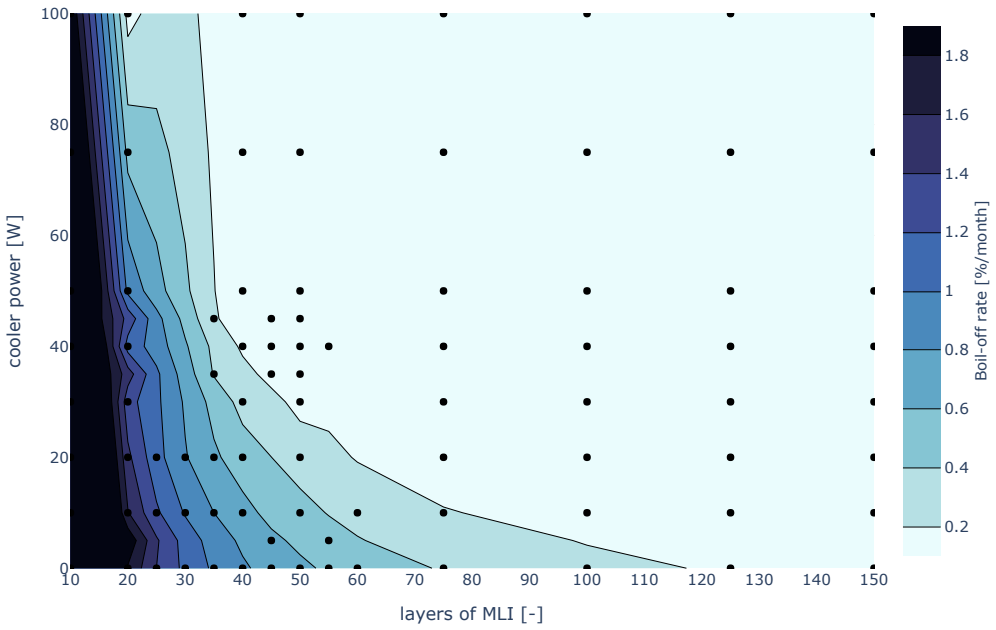

Figure 5.9: Average boil-off rate in percent per month for a mission time of 24 months with SOFI thickness of 0.01 [m].

rate. With more than 60 layers of MLI, the boil-off rate is less than 0.2% per month if the cooling power is 20 [W] or more.

Figure 5.10 gives the total mass as a function of the number of layers of MLI on the horizontal axis and the SOFI thickness on the vertical axis. The right-hand sidebar gives the mass in [kg]. From the figure, it is seen that adding more SOFI does slightly increase the weight of the design. This is indicated by the small positive slope the contour plot has. For the MLI, it is observed that up to 150 layers of MLI, mass is reduced by adding more layers if there is no active component present.
Figure 5.11 gives the total mass of the 24-month mission as a function of the layers of MLI on the horizontal axis and the cooling power on the vertical axis. In the figure, a clear saddle point is observed

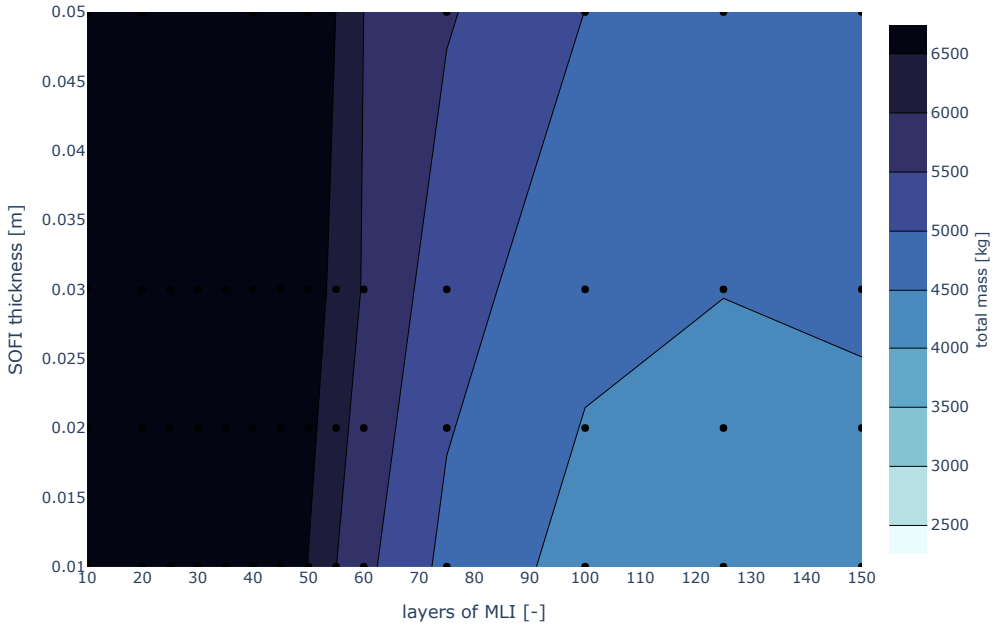

Figure 5.10: Total mass in kilogram for a mission time of 24 months with passive insulation only.

at 50 layers of MLI and 40 [W] of cooling power. A vertical pattern around the saddle point is visible, indicating that either adding MLI or adding cooling power reduces the mass, but adding both increases the mass.

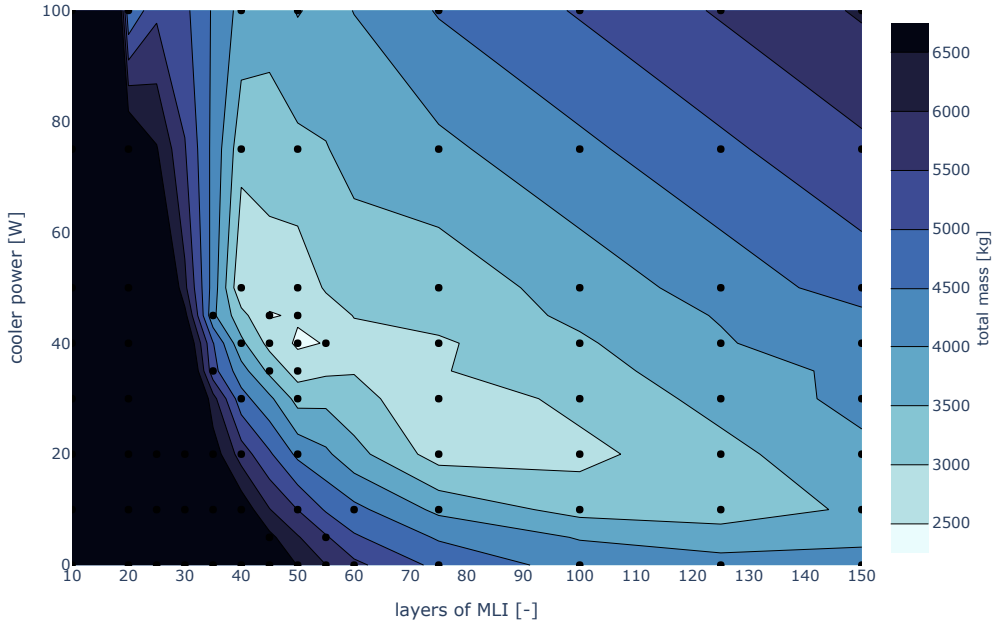

Figure 5.11: Total mass in kilogram for a mission time of 24 months with SOFI thickness of 0.025 [m].

As for the 12-month mission, in table 5.4 the 10 designs which give the smallest total mass are shown. All options have a cryocooler installed, but it is noted that the cooling power is limited to a maximum of 50 [W]. All but 2 designs are zero boil-off designs. The number of MLI layers is focused around 50 layers of MLI, as for the 12-month mission duration.

Table 5.4: 10 lightest design options propellant depot design for a mission duration of 24 months.

#	t_{SOFI} [m]	M_{SOFI} [kg]	MLI layers [-]	M_{MLI} [kg]	Q_{cooler} [W]	M_{cooler} [kg]	$M_{boil-off}$ [kg]	Boil-off rate [%/month]	M_{total} [kg]
1	0.01	165	50	1013	40	1241	0	0.000	2419
2	0.01	165	45	912	45	1381	0	0.000	2458
3	0.01	165	55	1115	40	1241	0	0.000	2521
4	0.01	165	40	811	50	1520	47.14297761	0.006	2543
5	0.01	165	50	1013	45	1381	0	0.000	2560
6	0.02	332	50	1018	40	1241	0	0.000	2590
7	0.01	165	75	1520	30	955	0	0.000	2641
8	0.01	165	50	1013	50	1520	0	0.000	2699
9	0.02	332	40	814	50	1520	39.33436893	0.005	2705
10	0.03	498	50	1023	40	1241	0	0.000	2762

5.2.4. Performance over time

The results for the 6, 12 and 24-month missions suggest that the best design varies over time. As figure 2.7 indicates, there should be a point in time in which passive only designs cross in mass with active designs. In figure 5.12 the total mass $M_{SOFI} + M_{MLI} + M_{cooler} + M_{boil-off}$ of four different designs, given in5.5, is given over time. Two of the designs were passive only and 2 designs contain a cooler. One of the cooler lines is what is possible with current technology, according to Chai and Wilhite [33], while the other is sized on ZBO.

Table 5.5: Four design options investigated over time.

ID	layers of MLI [-]	Q_{cooler} [W]
passive-light	20	0
passive-heavy	50	0
active-state of art	50	10
active-ZBO	50	40

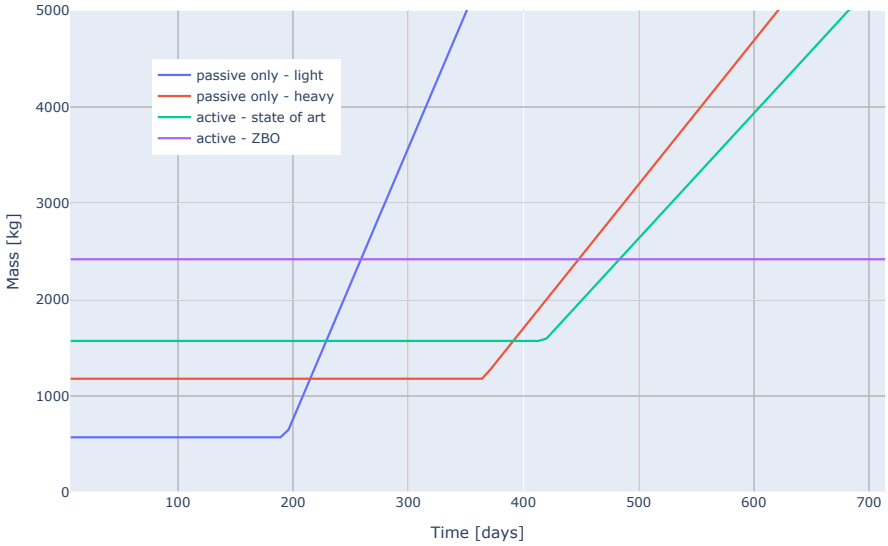

Figure 5.12: Change of best design over time for a propellant depot.

In figure 5.12 the horizontal axis indicates the time in days and the vertical axis indicates the mass. The legend gives which of the lines belongs to which design. For mission duration up to about a year, the passive only insulation options are lighter than the options investigated with a cryocooler. It is

also observed from the figure, that adding a small amount of cooling power extends the period without boil-off, but does not remove sufficient heat to have a zero boil-off propellant tank for all missions. A ZBO propellant tank is the heaviest option for a long time, but the advantage of having ZBO is that the mission duration is not important for the mass.

In comparison with the trade-off lines developed by the design program, Chai and Wilhite give figure 5.13. A qualitative comparison is not possible since it is not provided what the exact designs are. However, from the figure, similar trends are seen for the mass of the passive option, the current state of the art option and the future state of the art (ZBO) option. Initially, the mass difference is not too much, but over time the advantage of having ZBO grows.

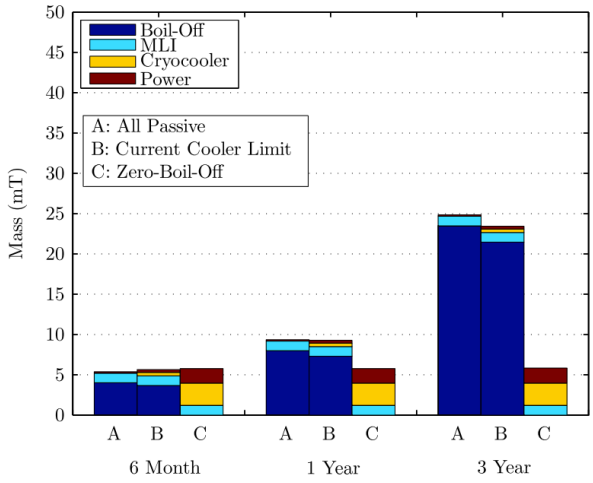

Figure 5.13: Passive insulation only (A), current cryocooler limit (B) and ZBO (C) mass performance for 6, 12 and 36 month missions [33]

5.3. Discussion of the propellant depot results

The results for a propellant depot orbiting the Earth in GSO are presented in the previous section. In this section, the results are discussed and observations are linked to theory from the previous chapters.

From the results it can be seen that the amount of SOFI insulation does not influence the boil-off rate, indicated by the vertical lines in both figure 5.4 and figure 5.8. This is explained by the low effectiveness of reducing the heat flow by conduction. The statement made by Plachta [24] is confirmed: MLI is necessary to effectively mitigate boil-off for a cryogenic propellant tank. Mitigating boil-off by radiative measures is more effective than mitigating boil-off by conductive measures. The boil-off rate is determined for the biggest part by the combination of MLI layers and cryocooler cooling power.

From the results over time, it shows that the lightest design option changes over time. This is due to boil-off. A cryogenic propellant tank designed for a 12-month mission only has to limit the heat flow to the liquid as much as the difference between the initial temperature of the liquid and the saturation temperature. If this capacity is divided by a shorter mission duration, less insulation material is required to meet the heat flow limit to the liquid. Therefore, for a shorter mission duration, a passive insulation only design can sufficient to mitigate boil-off. For longer missions with an equal temperature buffer, the amount of passive insulation required to prevent boil-off will exceed the additional mass of using a cryocooler in the design.

For a 24-month mission duration, a cryocooler is necessary to obtain ZBO. The cryocooler power used for the best design is 40 [W]. This cooling capacity exceeds the current limit of cryocoolers of cooling at 20 [K]. A 40 to 100 [W] cryocooler is still to be developed. Therefore, for the short term, it is possible cryocoolers are not used in designs using cryogenic liquids for long space missions.

6

Centaur case

In this chapter, it is investigated how the Centaur upper stage can be improved with respect to boil-off performance and mass of the propellant tank system. The Centaur upper stage is a cryogenic stage using liquid hydrogen and liquid oxygen, its design is however not suitable for long-duration space missions. The conventional Centaur upper stage uses 0.0381 [m] of SOFI and has a 3-layer radiation shield. In section 6.1 it is given what the input parameters of this case are and which options are investigated to improve the design. Section 6.2 gives the results of the Centaur simulations and in section 6.3, the results are discussed.

6.1. Centaur case description

The second case study is improving the Centaur G upper stage insulation to extend the possible mission duration. The following data is found for the input parameters. The SOFI material used is similar to the foam used on the Atlas rocket: NCFI 24-124 foam. The properties of this foam are given in the previous case study. The Centaur G upper stage was insulated by a standard layer of $t_{SOFI} = 0.0381[m]$ of SOFI [17].

The outer coating used is the same as for the Centaur, which is Teflon with Vacuum Deposited Aluminium (VDA). Teflon-VDA gives an emissivity of $\epsilon_{ext} = 0.66[-]$ and an absorptivity of $\alpha_{ext} = 0.08[-]$[64]. The shell is made of type 301 Stainless Steel and the original shell thickness is $t_{shell} = 0.0005[m]$ [89]. This is as thick as the thickest part of the shell of the Centaur. However, using a shell thickness of $t_{shell} = 0.0005[m]$ gives numerical errors due to the mass of the node becoming too small. A shell thickness for the simulation of $t_{shell} = 0.005[m]$ is used, which does not give numerical errors. The influence of this assumption is analysed in the sensitivity analysis and from the analysis, it is concluded that the shell thickness is of little influence on the result. The coefficient of conductivity and the density of the shell are $k_{shell} = 16.2[W/m/K]$ and $\rho_{shell} = 7880[kg/m^3]$ respectively. The specific heat coefficient of type 301 Stainless Steel is $C = 500[J/kg/K]$.

The maximum propellant tank radius is 1.4 meter. The outer diameter of the Centaur G is $D_{outer} = 3.05[m]$ [40], thus a maximum radius of $r_{max} = 1.4[m]$ fits in the spacecraft. With a propellant tank radius of 1.4 meter, a propellant mass of $M_{propellant} = 21,105[kg]$ [13] resulting in $M_{fuel} = 3,518[kg]$ with the oxidiser over fuel ratio of 5:1 for the Centaur [90], a fill level of 0.9 and an initial propellant density of $\rho_{H_2} = 71.5[kg/m^3]$, this results in a cylindrical propellant tank with spherical endcaps with the dimensions of $h_{total} = 9.8[m]$ total height.

To improve the insulation performance of the Centaur upper stage, MLI is added to the propellant tank. The layer density used is equal to the density used in the propellant depot case, $\tilde{N}_{MLI} = 16[layers/cm]$. The case is analysed with 0, 3, and 5 to 70 with a step size of 5 layers of MLI. The properties of the MLI are further equal to the properties used in the previous case.
Next to the variable number of layers of MLI, also an active cryocooler is added to the design. This cryocooler provides 0, 5, 10, 15 or 20 [W] of cooling power. The mass of the cryocooler is determined

by equation 2.28 found in section 2.4.2.

The initial pressure and temperature in the propellant tank are 1.86 [bar], as given in [91] as maximum pressure in the LH2 tank of the Centaur, and 20 [K]. The 20 [K] is the rounded saturation temperature of liquid hydrogen at one-atmosphere pressure. The propellant tank ullage level is assumed to be 0.1 [-], which gives a fill level of 0.9 [-].

The environment of the mission varies over time. The mission is a 9-month mission to Mars, giving a maximum distance to the Sun of 1.4 AU. The heat-flux from the Sun is $q_{sun} = 1350/AU^2[W/m^2]$ in which AU is calculated by $AU = 1 + 0.4 * t/t_{mission}$. Next to the solar radiation, also albedo heat and infrared radiation from the Earth are taken into account. The altitude from Earth is used and by using equation 2.15 and 2.16 with $\rho_{b_{Earth}} = 0.28[-]$ and $q_{earth} = 237[W/m^2]$ the additional heat is calculated every time step.
The surrounding temperature of the spacecraft is $T_{space} = 3[K]$. The spacecraft will act in microgravity $g = 10^{-6} * g_0[m/s^2]$. In the list given after this, all the inputs are summarised:

Table 6.1: List of input parameters used for the Centaur simulation.

- $Q_{cooler} =$ See text
- $M_{fuel} = 3,418[kg]$
- $T_{initial} = 20[K]$
- $p_{initial} = 186000[Pa]$
- $r_{tank} = 1.4[m]$
- $dt = 2[s]$
- $t_{mission} = 9[months]$
- $fill\_level = 0.9[-]$
- $t_{SOFI} = 0.0381[m]$
- $\rho_{SOFI} = 38.44[kg/m^3]$
- $k_{SOFI} = 0.02[W/m/K]$
- $C_{SOFI} = 1300[J/kg/K]$
- $\epsilon_{SOFI} = 0.3[-]$
- $C_{shell} = 500[J/kg/K]$
- $k_{shell} = 16.2[W/m/K]$
- $\rho_{shell} = 7880[kg/m^3]$
- $t_{shell} = 0.005[m]$
- $N_{MLI} =$ See text
- $\tilde{N}_{MLI} = 16[layers/cm]$
- $\rho_{MLI} = 0.047[kg/m^2/layer]$
- $k_{MLI} = 0.24[W/m/K]$
- $C_{MLI} = 1170[J/kg/K]$
- $\epsilon_{MLI} = 0.03[-]$
- $p_{interstitial} = 1.33 * 10^{-5}[Pa]$
- $\epsilon_{coating} = 0.66[-]$
- $\alpha_{coating} = 0.08[-]$
- solar incidence angle = $180\,\mathrm{deg}$
- $g = 10^{-6} * g_0[m/s^2]$
- $T_{space} = 3[K]$
- $q_{Sun} = 1350/AU^2[W/m^2]$
- $AU = 1 + 0.4 * \frac{t}{t_{mission}}$
- $h_{Earth} = 0.4 * 149,597,871 * \frac{t}{t_{mission}}$
- $q_{Earth-IR} = 237 * \frac{r_{Earth}}{r_{Earth}+h_{Earth}} 2[W/m^2]$
- $q_{albedo} = 0.38 * q_{Sun} * \frac{r_{Earth}}{r_{Earth}+h_{Earth}} 2[W/m^2]$

6.2. Results Centaur simulation

Next to the propellant depot case, in section 6.1 also the Centaur case is described. In this section, the results of the simulation performed with the input parameters of the Centaur case are given. The first figure of this section is figure 6.1. In this figure, the heat flux over time is given for the albedo heat, the IR heat and the solar radiation flux.

From figure 6.1 it is seen that the solar radiation flux is by far the highest radiation source. The IR flux and the albedo flux start at magnitudes of $10^{-2}[W/m2]$, but decrease after 50 days already to values lower than $5 * 10^{-4}[W/m2]$, whereas the solar flux is still above $1000[W/m2]$.

In figure 6.2 the boil-off rate is given as a function of the cooler power on the vertical axis and number of layers of MLI on the horizontal axis. The boil-off rate in per cent per month is given on the vertical

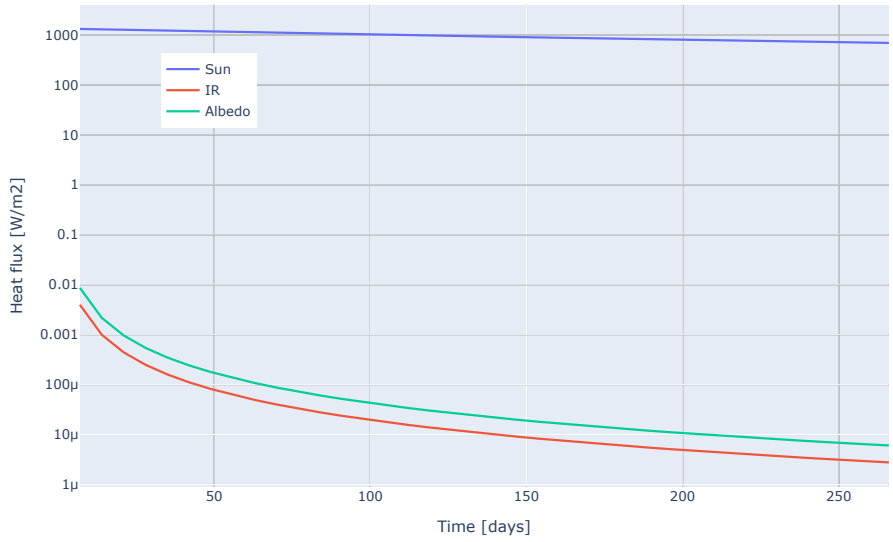

Figure 6.1: Heat flux over the mission time for the Centaur mission.

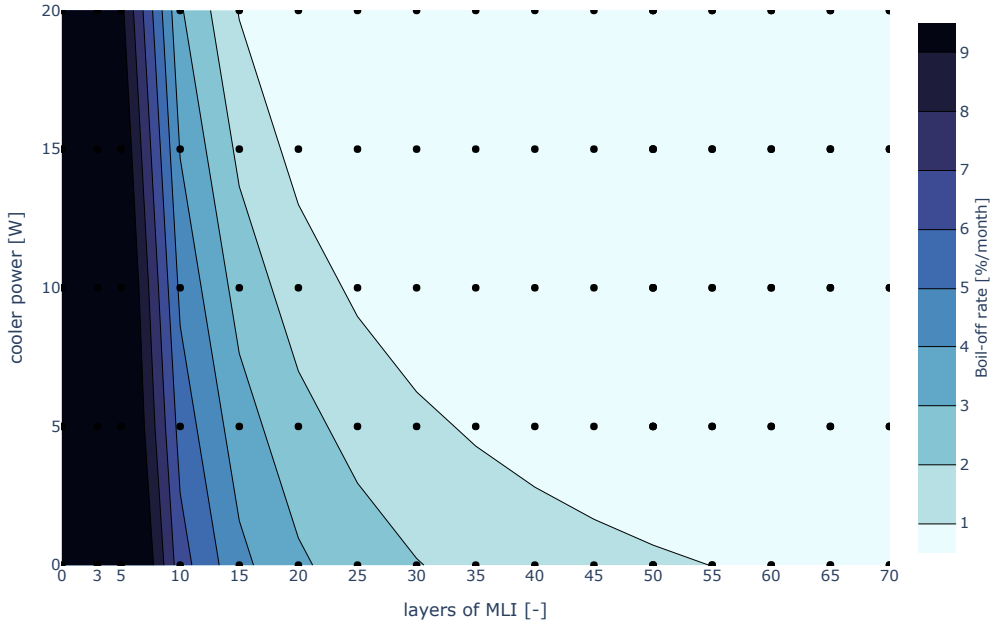

Figure 6.2: Boil-off rate as a function of MLI layers and cooling power for the 9 month mission of the Centaur.

bar next to the main figure. The figure shows that the boil-off rate reduces with more layers of MLI and with more cooling power. The contour shows curved diagonal lines for the boil-off rate, with less boil-off in the top right corner of the figure and the maximum amount of boil-off on the bottom left. The pattern seen is explained by that with adding more layers of MLI, the gradient with which the boil-off reduces becomes less. The boil-off rate of the option without MLI and cooling power is more than 9% month and is of the scale of the graph. The actual performance of this option is given in table 6.2. It is seen that after 21 hours, the temperature buffer of the liquid ($23.1 - 20.0 = 3.1$[K]) is used and the boil-off starts. In 307 hours the whole propellant tank boils-off.

Figure 6.3 gives the boil-off, MLI and cooler mass summed as a function of the cooler power on the vertical axis and the number of layers of MLI on the horizontal axis. The vertical bar to the right of the figure provides the legend for the mass in kilograms. In the figure, two saddle points are observed. The first is the with 35 to 40 layers of MLI and cooling power of 10 [W]. The second saddle point is

Table 6.2: Results of the Centaur-G upper stage design with only SOFI.

t_{SOFI} [m]	M_{SOFI} [kg]	$M_{\text{boil-off}}$ [kg]	Boil-off rate [%/month]	M_{total} [kg]	$t_{\text{start boil-off}}$ [hours]	$t_{100\%\text{ boil-off}}$ [hours]
0.0381	129	3506	220	3635	21	328

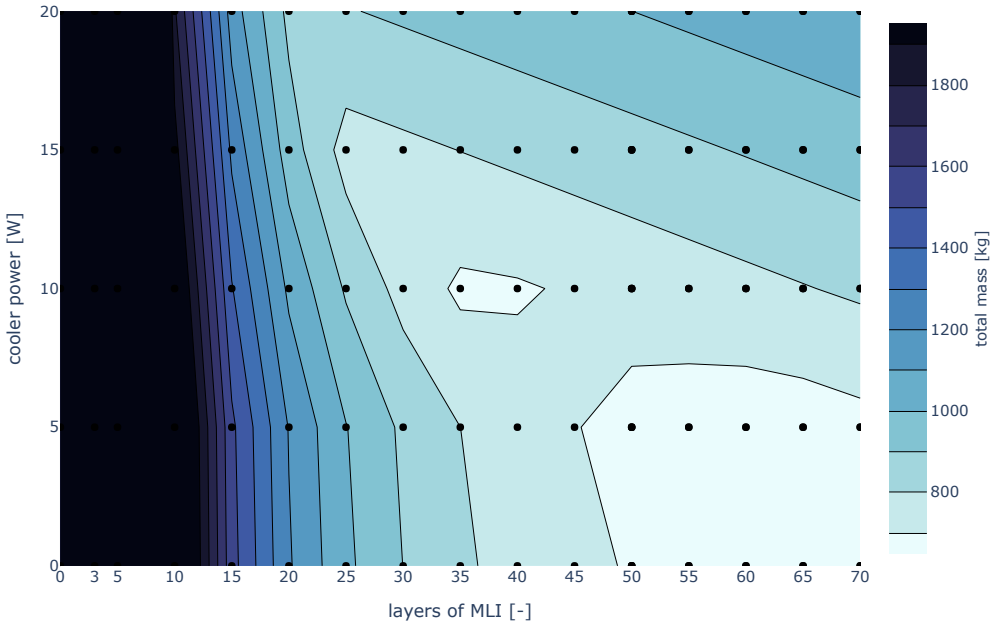

Figure 6.3: Total mass as a function of MLI layers and cooling power for the 9 month mission of the Centaur.

with 0 to 5 [W] of cooling power and 50 to 70 layers of MLI. If zoomed in to that particular area, see table 6.3, it is seen that the lightest option is with 60 layers of MLI and 5 [W] of cooling power. The mass with more cooling power or more layers of MLI increases. From the figure it is seen that the total mass reduces with more layers of MLI and depending on the number of MLI layers, the mass increases or decreases by adding more cooling power. In table 6.3 the results are given of the simulation. The constant properties as the SOFI thickness and mass, shell thickness and mass and MLI layer density are omitted from the table.

Table 6.3: Performance of different MLI and cryocooler settings for the Centaur stage with a mission duration of 9 months.

#	MLI layers [-]	M_{MLI} [kg]	Q_{cooler} [W]	M_{cooler} [kg]	$M_{boil-off}$ [kg]	Boil-off rate [%/month]	M_{total} [kg]	$t_{\text{start boil-off}}$ [days]
1	60	253	5	246	15	0.047303613	642	254
2	65	274	5	246	0	0	648	0
3	70	295	0	0	226	0.715088714	649	143
4	65	274	0	0	250	0.793824875	653	135
5	55	232	5	246	49	0.155883708	655	224
6	60	253	0	0	279	0.885182019	661	128
7	70	295	5	246	0	0	669	0
8	55	232	0	0	313	0.992516995	674	120
9	50	211	5	246	90	0.285018197	6745	196
10	35	147	10	392	13	0.041185836	682	254

Table 6.3 shows that the design options which have the lowest mass do allow some boil-off. Only the 2 out of the 10 best options are zero boil-off options. Table 6.4 gives the steady-state heat flows to the liquid for the number of MLI layers without a cryocooler. It is seen, that the heat flow reduces with every additional layer of MLI. However, the rate with which the heat flow reduces becomes less with more layers of MLI. Eventually, the remaining heat flow to the liquid is due to the heat leaks accounted for.

Table 6.4: Steady state heat flow to the LH2 without cryocooler and standard thickness SOFI $t_{SOFI} = 0.0381[m]$.

layers of MLI [-]	5	10	15	20	25	30	35	40	45	50	55	60	65	70
Heat flow to liquid [W]	77	40	28	22	18	16	14	13	12	11	10	10	9	9

Figure 6.4 gives four designs and the total mass over time. The three designs are labelled ”low rate”, ”medium rate”, ”high rate” and ”complete boil-off” and have 35, 25, 15 and 5 layers of MLI respectively and all a cooling power of 5 [W]. This figure is used to see whether the boil-off rate changes over time. From the slopes in the figure, it is seen that the boil-off rates do not vary over time, except for when the whole tank is almost boiled-off.

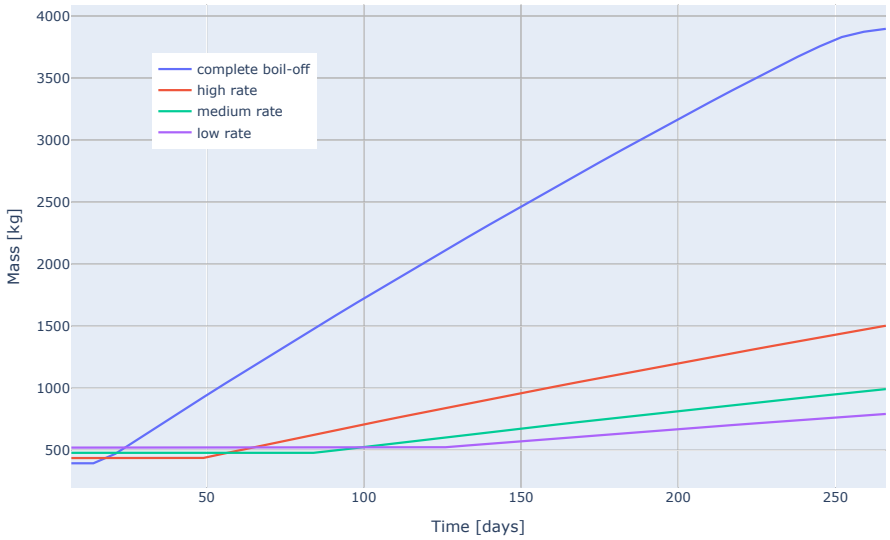

Figure 6.4: Total mass over time for three different Centaur designs.

6.3. Discussion of the Centaur results

The original Centaur upper stage was not designed for long space missions. It was not deemed necessary by the designers to include more insulation than 0.038 [m] of SOFI and a three-layer MLI shield. Kruif and Kutter [25] demonstrated that with slight modifications of the design, by adding up to 20 layers of MLI, the boil-off rate of the stage could be reduced to around 1 % per day compared to the 4.1 % per day for the original design [25]. In this chapter, more design options are investigated.

The results for the Centaur stage support the conclusions by Kruif and Kutter, although the results presented are different from the performance predictions by Kruif and Kutter. The option from the results with 20 layers of MLI has an average boil-off rate of 3.2% per month, which is 0.1% per day. Also, the option with three-layer MLI performs better (152 days before complete boil-off compared to 110 days (only 1 % propellant left)) than the original design with three-layer MLI. The explanation of the difference can be due to the ground stage. In the results presented, the ground stage of the mission is not taken into account. In reality, the ground stage is the most severe environment the spacecraft will face due to the convection of the air to the propellant tank. Boil-off will already start before the launch.

The results give the options which allow some boil-off as most mass efficient. If the first ZBO propellant tank is compared with the first LBO propellant tank, the mass difference is $M_{ZBO} - M_{LBO} = 690 - 675 = 15[kg]$. With a boil-off rate of 0.29 [%/month] and the initial propellant mass of 3518, within 1.5 months the difference is vaporised. Analysing the option with 25 layers of MLI and the same cryocooler, it is seen from table 6.4 that the steady-state flow is 18 [W] without the cryocooler. A cry-

ocooler can keep the LH2 below boiling point for any mission duration given that the cryocooler can provide 18 [W] of cooling power.

From figure 6.1 it is observed that the heat flux of the sources does reduce over time. Figure 6.4 gives the total mass over time and from the figure, it is seen that the boil-off rate is constant. Together these figures indicate that the boil-off rate would increase if the heat flux would have been constant.

7

Conclusion and recommendations

In this chapter, the conclusions drawn from the work performed will be given and the answers to the research questions. Next to the conclusions, also recommendations will be given to improve the work performed in the future. After this chapter, the reader will know the findings from this thesis.

7.1. Conclusion

The study conducted focused on finding the mass optimal configuration for long-duration space missions. For this study, a simulation tool called the Boil-off Monte Carlo program is developed, which ,through a Monte Carlo system, aims to find the most optimal configuration with layers of MLI, SOFI thickness and cooling power as output.

The amount of passive insulation used to obtain zero boil-off is depending on the mission duration. It is possible, theoretically, to produce a passive only zero boil-off propellant tank for any mission duration. In section 5.2 it is demonstrated that for a mission duration up to a year it is more mass efficient to use passive insulation only. For missions longer than a year, adding an active component can reduce the total mass. Comparing the results in section 5.2 of the loiter period of 12 months with the loiter period of 24 months, it is observed that the mass efficient designs for the 24 months include a cryocooler, whereas 3 out of 10 do for the 12 month loiter period. It is possible to have a propellant depot orbiting the Earth in GSO with only passive insulation, as long as the onboard fuel is used within 12 months.
From the results presented in section 5.2 and section 6.2 it is concluded that it is not always mass efficient to stick to passive only insulation. However, if the mass penalty of using only passive insulation is limited compared to using active systems, the added complexity of using an active system is not worth the little mass savings.
From the sensitivity analysis in chapter 4 it can be concluded that the amount of insulation required to have zero boil-off is also depending on the initial temperature of the liquid and is related to the heat capacity of the liquid up to boiling temperature. The cooler the liquid is stored, the better the performance to reduce boil-off. This is if the propellant tank size and propellant tank fill level are maintained from the original design.

The results from section 5.2 indicate by the vertical pattern of the boil-off rate that the most effective passive insulation measure is using MLI. This is seen from the vertical regions in the figures presenting the boil-off rate as a function of the amount of MLI layers and the SOFI thickness. The boil-off rate varies per layer of MLI and not so much with the SOFI thickness. From this observation, it is concluded that adding MLI is more effective than adding SOFI to a propellant tank.

The liquid fill level is important since the liquid fill level determines the propellant tank size required for a certain amount of liquid, as identified in section 4.2.3. A lower fill level results in a larger propellant tank, which results in a larger surface area to receive heat flux. A higher fill level results in a smaller propellant tank, which is beneficial for the performance concerning the boil-off as well as the initial mass, since a smaller propellant tank is lighter.

Pressurising the propellant in advance of the mission has little effect on the performance since heating of the propellant will raise the pressure to the maximum allowable pressure anyways. However, it is beneficial to increase the maximum allowable pressure from a saturation temperature perspective, as shown in the sensitivity analysis. If cryogenic propellant boils-off, the vapour pressure increases. If this increase in pressure is allowed, the saturation temperature increases. A higher saturation temperature results in less boil-off of liquid propellant.

From the results in chapter 6 it is demonstrated that a mission duration of nine months is achievable by adding MLI to the original Centaur design. Zero boil-off for the Centaur upper stage is only mass efficient if an active cryocooler is added to the system. This raises the complexity of the system, but decreases the performance degrading due to environmental uncertainty since fluctuations in heat can be compensated for by the cryocooler.

The main research question was what the most mass efficient configuration of propellant and structure would be for a given propellant tank. The results in this thesis demonstrate that for a propellant depot orbiting the Earth in GSO, it would a propellant tank with 40 layers of MLI, a cryocooler delivering 50 [W] of cooling power and a layer of 0.01 [m] of SOFI. However, the results over time demonstrate that the most mass efficient configuration is very dependent on the mission duration. Also, from the results of the Centaur, it can be added to the conclusion that the optimal configuration is also depending on the propellant tank size. A smaller propellant tank has a smaller mass penalty if more layers of MLI are added compared to a larger propellant tank, whereas a 20 [W] cooler adds the same weight for both cases. For a smaller propellant tank, adding a cryocooler gives a relatively large mass penalty compared to passive insulation.
The Centaur upper stage propellant tank design can be expanded with 60 layers of MLI and a cryocooler providing 5 [W] of cooling power to make the stage mass efficient for a 9-month loiter period to Mars.

7.2. Recommendations

In the previous section, the answers to the research questions were given. This section gives recommendations on how future research can be given form and which topics are important to invest time in.

In section 4.2 a sensitivity analysis was done to numerical, design & material and liquid properties to identify which parameters are important for either design or which parameters have to be known accurately. For this research, the maximum amount of sections was 12. It is recommended for further research that the same methodology used here is applied to more than 12 sections. Although it is expected that adding more sections will eventually not change the results anymore while the computation will increase, it is not concluded by this research that the maximum detail has been reached with using 12 sections.

The assumption that the fluid in the propellant tank is homogeneous, is valid as long as it is stated that a fluid mixing device is installed. Part of future research should be on how this assumption holds without a mixing device. It could be that natural convection mixes the fluid well enough. The long timescale of the problem gives the liquid sufficient time to mix, although fluid velocities are low.

From the research conducted it is concluded that the issue of propellant boil-off of cryogenic liquid is complex due to the many factors and parameters influencing the process. This research focused on the insulation, active and passive, of the spacecraft propellant tank. It is recommended for further studies to create a multi-disciplined exercise. The thermal environment over time should be a focus of research, as should the behaviour of the propellant in the tank. For preliminary sizing, the latter is not important because it is concluded that to mitigate boil-off, a zero boil-off solution is recommended. However, if detailed design it is important to understand the behaviour of the fluid, although there won't be boil-off.

This project could be the start of something much bigger. With the desire of mankind to travel fur-

ther into space than ever before, cryogenic propellants will be more important in the future. Much research is done currently by NASA, but the research to boil-off in Europe seems not to progress.
This is a big opportunity for the Delft University of Technology since a leap can be taken in progressing research to using cryogenic liquids on long-duration space mission, by using the multi-disciplined knowledge of different departments of the university. As mentioned, the thermal environment is one of the topics which requires more attention. The department of Space Exploration could invest some research on this topic. The department of Aerodynamics could research the behaviour of cryogenic liquid under heating conditions with a CFD analysis. These are two examples which indicate how different disciplines can work together on one problem.

It should be mentioned that the results of this thesis are based upon the assumption that the mass of the spacecraft is the most important factor in the design of the spacecraft. In reality, the total mass of the spacecraft is not always the main focus. In case of relatively short duration missions, it could be that spacecraft including cryocoolers are lighter with respect to mass than the spacecraft with only passive insulation. However, adding a cryocooler increases the complexity of manufacturing and design a lot as well as increases the price of the spacecraft. Although the spacecraft using active cooling is the most mass efficient option, operators could still choose the passive only option for its minor complexity compared to the active option. Also, high cooling power cryocoolers are still to be developed and the relationships used are based on existing coolers. The high cooling power regime does not have data points, thus the validity of the relationships at this regime is unknown.

List of References

[1] J. R. Wertz, D. F. Everett, and J. J. Puschell, Eds., *Space mission engineering: the new SMAD*, ser. Space technology library v. 28. Hawthorne, CA: Microcosm Press : Sold and distributed worldwide by Microcosm Astronautics Books, 2011, OCLC: ocn747731146, ISBN: 9781881883166.

[2] J. R. Wertz, D. F. Everett, and J. J. Pischell, "Liquid Rocket Engines", in *Space Mission Engineering: The New SMAD*, Microcosm Press, 2011, ch. 18.3.2.2, p. 538, ISBN: 9781600864001.

[3] R. Radebaugh, *Cryogenics*, 2002.

[4] G. Arnold and J. Wolf, "Liquid Hydrogen for Automotive Application Next Generation Fuel for FC and ICE Vehicles", *Journal of the Cryogenic Society of Japan*, vol. 40, no. 6, pp. 221–230, 2005, ISSN: 0389-2441. DOI: 10.2221/jcsj.40.221.

[5] D. B. Marron, "Alternatives for future U.S. Space-Launch Capabilities", no. October, pp. 1–81, 2006.

[6] United Launch Alliance, "Delta IV Launch Services User's Guide", *June*, no. June, 2013.

[7] B. Collaudin and N. Rando, "Cryogenics in space: A review of the missions and of the technologies", *Cryogenics*, vol. 40, no. 12, pp. 797–819, 2000, ISSN: 00112275. DOI: 10.1016/S0011-2275(01)00035-2.

[8] R. K. Plebuch and J. S. Martinez, "NUCLEAR ROCKETS", *Annals of the New York Academy of Sciences*, vol. 140, no. 1, pp. 380–392, 1966, ISSN: 17496632. DOI: 10.1111/j.1749-6632.1966.tb50973.x.

[9] B. Kutter, F. Zegler, S. Lucas, L. Hines, M. Ragab, I. Spradley, and J. Hopkins, "Atlas Centaur Extensibility to Long-Duration In-Space Applications", pp. 1–13, 2012. DOI: 10.2514/6.2005-6738.

[10] D. J. Chate and M. P. Doherty, *NASA Perspectives on Cryo H2 Storage*, 2011.

[11] D. Glaister, J. Schmidt, C. McLean, and G. Mills, "Long Term Cryogenic Storage Technologies Overview for NASA Exploration Applications", no. June, 2012. DOI: 10.2514/6.2011-3774.

[12] C. B. Muratov, "Issues of Long-Term Cryogenic Propellant Storage in Microgravity October 2011", no. October, 2011.

[13] S. S. Pietrobon, "Analysis of Propellant Tank Masses", no. November 2008, pp. 1–30, 2009.

[14] D. Plachta, J. Stephens, W. Johnson, and M. Zagarola, "NASA cryocooler technology developments and goals to achieve zero boil-off and to liquefy cryogenic propellants for space exploration", *Cryogenics*, vol. 94, pp. 95–102, 2018, ISSN: 00112275. DOI: 10.1016/j.cryogenics.2018.07.005.

[15] R. D. Braun, "Investment in Our Future: Exploring Space Through Innovation and Technology", 2011.

[16] G. R. Smolak, R. H. Knoll, and L. E. Wallner, "Analysis of thermal-protection systems for space-vehicle cryogenic-propellant tanks", NASA Lewis Research Center, Cleveland, Ohio, Tech. Rep., 1962, p. 39.

[17] P. N. MacNeil, J. E. England, and R. H. Knoll, "Design, development, and test of shuttle/centaur g-prime cryogenic tankage thermal protection systems", *Advances in cryogenic engineering*, vol. 33, no. New York, NY, United States, pp. 341–348, 1988, ISSN: 00652482.

[18] L. J. Hastings, D. W. Plachta, L. Salerno, and P. Kittel, "An overview of NASA efforts on zero boiloff storage of cryogenic propellants", *Cryogenics*, vol. 41, no. 11-12, pp. 833–839, 2001, ISSN: 00112275. DOI: 10.1016/S0011-2275(01)00176-X.

[19] A. Hedayat, L. J. Hastings, J. Sims, and D. W. Plachta, "Cryogenic Propellant Long-term Storage With Zero Boil-off", *NASA Report NAS800187*, 2001.

[20] D. Plachta and P. Kittel, “An updated zero boil-off cryogenic propellant storage analysis applied to upper stages or depots in an LEO environment”, *38th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, no. June 2003, 2002. DOI: 10.2514/6.2002-3589.

[21] C. Panzarella and M. Kassemi, “Simulations of zero boil-off in a cryogenic storage tank”, *41st Aerospace Sciences Meeting and Exhibit*, no. January, 2003. DOI: 10.2514/6.2003-1159.

[22] D. Plachta, “Results of an Advanced Development Zero Boil-Off Cryogenic Propellant Storage Test”, no. November 2004, 2004. DOI: 10.2514/6.2004-3837.

[23] C. Guernsey, R. Baker, D. Plachta, and P. Kittel, “Cryogenic Propulsion with Zero Boil-Off Storage Applied to Outer Planetary Exploration”, *41st AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit*, no. July, pp. 1–13, 2005. DOI: 10.2514/6.2005-3559.

[24] D. W. Plachta, R. J. Christie, J. M. Jurns, and P. Kittel, “Passive ZBO storage of liquid hydrogen and liquid oxygen applied to space science mission concepts”, *Cryogenics*, vol. 46, no. 2-3, pp. 89–97, 2006, ISSN: 00112275. DOI: 10.1016/j.cryogenics.2005.11.012.

[25] J. S. D. Kruif, B. F. Kutter, U. L. Alliance, V. Density, S. Engineer, S. Analysis, and A. Evolution, “Centaur Upperstage Applicability for Several-Day Mission Durations with Minor Insulation Modifications”, *Review Literature And Arts Of The Americas*, no. July, pp. 1–8, 2007.

[26] D. W. Plachta, R. J. Christie, E. Carlberg, and J. R. Feller, “Cryogenic propellant boil-off reduction system”, *AIP Conference Proceedings*, vol. 985, no. March 2008, pp. 1457–1466, 2008, ISSN: 0094243X. DOI: 10.1063/1.2908506.

[27] S. H. Ho and M. M. Rahman, “Three-dimensional analysis for liquid hydrogen in a cryogenic storage tank with heat pipe-pump system”, *Cryogenics*, vol. 48, no. 1-2, pp. 31–41, 2008, ISSN: 00112275. DOI: 10.1016/j.cryogenics.2007.09.005.

[28] C. H. McLean, G. L. Mills, M. E. Riesco, M. L. Meyer, D. W. Plachta, and E. A. Hurlbert, “Long term space storage and delivery of cryogenic propellants for exploration”, *44th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, no. July, 2008. DOI: 10.2514/6.2008-4853.

[29] P. Perczyński, “Thermal protection system and tank design for in-orbit cryogenic propellant storage”, Master of Science Thesis Report, Delft University of Technology, 2009, pp. 1–102.

[30] C. H. Panzarella and M. Kassemi, “Comparison of Several Zero-Boil-Off Pressure Control Strategies for Cryogenic Fluid Storage in Microgravity”, *Journal of Propulsion and Power*, vol. 25, no. 2, pp. 424–434, 2009, ISSN: 02100010. DOI: 10.2514/1.35611.

[31] T. Nast, D. Frank, and K. Burns, “Cryogenic Propellant Boil-Off Reduction Approaches”, no. January, 2011. DOI: 10.2514/6.2011-806.

[32] J. E. Fesmire, B. E. Coffman, B. J. Meneghelli, and K. W. Heckle, “Spray-on foam insulations for launch vehicle cryogenic tanks”, *Cryogenics*, vol. 52, no. 4-6, pp. 251–261, 2012, ISSN: 00112275. DOI: 10.1016/j.cryogenics.2012.01.018.

[33] P. R. Chai and A. W. Wilhite, “Cryogenic thermal system analysis for orbital propellant depot”, *Acta Astronautica*, vol. 102, pp. 35–46, 2014, ISSN: 00945765. DOI: 10.1016/j.actaastro.2014.05.013. [Online]. Available: http://dx.doi.org/10.1016/j.actaastro.2014.05.013.

[34] X. W. Sun, Z. Y. Guo, and W. Huang, “Passive zero-boil-off storage of liquid hydrogen for long-time space missions”, *International Journal of Hydrogen Energy*, vol. 40, no. 30, pp. 9347–9351, 2015, ISSN: 03603199. [Online]. Available: http://dx.doi.org/10.1016/j.ijhydene.2015.05.184.

[35] K. J. Jaya Kumar, “Heat transfer analysis of light weight cryogenic tank for space vehicles”, *Indian Journal of Science and Technology*, vol. 8, no. 4, pp. 314–319, 2015, ISSN: 09745645. DOI: 10.17485/ijst/2015/v8i4/62291.

[36] B. Taylor, J. Caffrey, A. Hedayat, J. Stephens, and R. Polsgrove, “Cryogenic Fluid Management Technology Development for Nuclear Thermal Propulsion”, pp. 1–8, 2015. DOI: 10.2514/6.2015-3957. [Online]. Available: https://ntrs.nasa.gov/search.jsp?R=20150016562%7B%5C&%7Dqs=N%7B%5C%%7D3D4294928011%7B%5C%%7D2B4294928192%7B%5C%%7D2B4294965818.

[37] Y. W. Liu, X. Liu, X. Z. Yuan, and X. J. Wang, “Optimizing design of a new zero boil off cryogenic storage tank in microgravity”, *Applied Energy*, vol. 162, pp. 1678–1686, 2016, ISSN: 03062619. [Online]. Available: http://dx.doi.org/10.1016/j.apenergy.2015.01.104.

[38] W. U. Notardonato, A. M. Swanger, J. E. Fesmire, K. M. Jumper, W. L. Johnson, and T. M. Tomsik, “Zero boil-off methods for large-scale liquid hydrogen tanks using integrated refrigeration and storage”, *IOP Conference Series: Materials Science and Engineering*, vol. 278, no. 1, 2017, ISSN: 1757899X. DOI: 10.1088/1757-899X/278/1/012012.

[39] S. M. Motil, *Cryogenic Propellant Storage & Transfer (CPST) Project Technology Advancement*, 2013.

[40] United Launch Alliance, *Atlas V Launch Services User’s Guide*, March. 2010, pp. 1–420, ISBN: 3039775960. [Online]. Available: http://www.ulalaunch.com/uploads/docs/AtlasVUsersGuide2010.pdf.

[41] J. J. Martin and L. Hastings, “Large-Scale of a Variable With a Foam Liquid Hydrogen Testing Insulation Density Multilayer Substrate”, 2001.

[42] A. K. Majumdar, T. E. Steadman, J. L. Maroney, J. P. Sass, and J. E. Fesmire, “Numerical modeling of propellant boil-off in a cryogenic storage tank”, *AIP Conference Proceedings*, vol. 985, pp. 1507–1514, 2008, ISSN: 0094243X. DOI: 10.1063/1.2908513.

[43] A. Majumdar, J. Valenzuela, A. Leclair, and J. Moder, “Numerical modeling of self-pressurization and pressure control by a thermodynamic vent system in a cryogenic tank”, *Cryogenics*, vol. 74, pp. 113–122, 2016, ISSN: 00112275. DOI: 10.1016/j.cryogenics.2015.12.001. [Online]. Available: http://dx.doi.org/10.1016/j.cryogenics.2015.12.001.

[44] Z. Liu, G. Zhou, Y. Li, and P. Gao, “Thermal performance of liquid hydrogen tank in reduced gravity”, *Advances in Space Research*, vol. 62, no. 5, pp. 957–966, 2018, ISSN: 18791948. DOI: 10.1016/j.asr.2018.06.006. [Online]. Available: https://doi.org/10.1016/j.asr.2018.06.006.

[45] M. S. Haberbusch, R. J. Stochl, and A. J. Culler, “Thermally optimized zero boil-off densified cryogen storage system for space”, *Cryogenics*, vol. 44, no. 6-8, pp. 485–491, 2004, ISSN: 00112275. DOI: 10.1016/j.cryogenics.2004.02.016.

[46] W. Jing, T. Tao, L. Xiyuan, and P. Yifei, “Analysis on Influence of Gravity on Convection Heat Transfer in Manned Spacecraft during Terrestrial Test”, vol. 6, no. 9, pp. 723–728, 2012.

[47] F. P. Incropera, D. P. DeWitt, T. L. Bergman, and A. S. Lavine, sixth edition. John WIley & Sons, Inc, 2007, pp. 3–13, ISBN: 9780471457282.

[48] S. Bewick, R. Parsons, T. Forsythe, S. Robinson, and J. Dupon. (2019). Types of radioactivity: Alpha, beta, and gamma decay, [Online]. Available: https://chem.libretexts.org/Courses/Eastern_Wyoming_College/EWC%5C%3A_Introductory_Chemistry_(Budhi)/17%5C%3A_Radioactivity_and_Nuclear_Chemistry/17.03%5C%3A_Types_of_Radioactivity%5C%3A_Alpha%5C%2C_Beta%5C%2C_and_Gamma_Decay (visited on 12/30/2019).

[49] A. Aueron and D. Thomas, “Trades on Densified Propellant for Nuclear Thermal Propulsion”, *Nuclear and Emerging Technologies for Space*, pp. 1–5, 2019. [Online]. Available: http://anstd.ans.org/.

[50] R. Byron Bird, W. E. Stewart, and E. N. Lightfoot, “Thermal Conductivity and the Mechanisms of Energy Transport”, in *Transport Phenomena*, 2nd ed., New York, NY: John WIley & Sons, Inc, 2002, p. 268, ISBN: 0471410772.

[51] P. J. Linstrom and W. G. Mallard, *The NIST Chemistry WebBook: A Chemical Data Resource on the Internet*. DOI: https://doi.org/10.18434/T4D303.

[52] H. Q. Yang and J. West, “CFD extraction of heat transfer coefficient in cryogenic propellant tanks”, *51st AIAA/SAE/ASEE Joint Propulsion Conference*, 2015. DOI: 10.2514/6.2015-3856.

[53] D. B. Go, *Lecture slides: Free Convection Overview*, 2018.

[54] S. W. Churchill and H. H. Chu, “Correlating equations for laminar and turbulent free convection from a vertical plate”, *International Journal of Heat and Mass Transfer*, vol. 18, no. 11, pp. 1323–1329, 1975, ISSN: 00179310. DOI: 10.1016/0017-9310(75)90243-4.

[55] J. E. Keesee, “Spacecraft Thermal Control Systems • Purpose of thermal control systems”, 2003.

[56] Steven L. Rickman, “Introduction to On-Orbital Thermal Environments”, *Thermal and Fluids Analysis Workshop*, no. August, 2014.

[57] I. de Pater and J. J. Lissauer, *Fundamental Planetary Science: physics, chemistry and habitability*, 4th ed. New York: Cambridge University Press, 2013, ch. E.11, p. 522, ISBN: 978-0-521-61855-7.

[58] H. Baltes, “I on the validity of kirchhoff’s law of heat radiation for a body in a nonequilibrium environment**this report was written under sponsorship of the national bureau of standards, washington, d. c. 20234, and is therefore not subject to copyright.”, in, ser. Progress in Optics, E. Wolf, Ed., vol. 13, Elsevier, 1976, pp. 1–25. DOI: https://doi.org/10.1016/S0079-6638(08)70017-9.

[59] R. G. Ross, “Quantifying MLI thermal conduction in cryogenic applications from experimental data”, *IOP Conference Series: Materials Science and Engineering*, vol. 101, no. 1, 2015, ISSN: 1757899X. DOI: 10.1088/1757-899X/101/1/012017.

[60] A. Hedayat, “Analytical modeling of variable density multilayer insulation for cryogenic storage”, no. May 2004, pp. 1557–1564, 2003. DOI: 10.1063/1.1472190.

[61] G. E. McIntosh, “Layer by layer mli calculation using a separated mode equation”, in *Advances in Cryogenic Engineering*, P. Kittel, Ed. Boston, MA: Springer US, 1994, pp. 1683–1690, ISBN: 978-1-4615-2522-6. DOI: 10.1007/978-1-4615-2522-6_206. [Online]. Available: https://doi.org/10.1007/978-1-4615-2522-6_206.

[62] W. L. Johnson, “Optimization of layer densities for multilayered insulation systems”, *AIP Conference Proceedings*, vol. 1218, no. 2010, pp. 804–811, 2010, ISSN: 0094243X. DOI: 10.1063/1.3422434.

[63] D. G. Gilmore, *Spacecraft thermal control handbook. Volume I, Volume I*. 2002. [Online]. Available: http://app.knovel.com/hotlink/toc/id:kpSTCHVFT2/spacecraft-thermal-control.

[64] L. Kauder, “Spacecraft Thermal Control Coatings References - NASA-TP-2005-212792”, *Spacecraft Thermal Control Coatings References*, no. December, p. 130, 2005.

[65] M. Barrios, M. Vanderlaan, and S. W. Van Sciver, “Thermal conductivity of spray-on foam insulations for aerospace applications”, *AIP Conference Proceedings*, vol. 1434, no. 57, pp. 1319–1326, 2012, ISSN: 0094243X. DOI: 10.1063/1.4707057.

[66] M. J. Gruszczynski, R. R. Wronski, V. L. Thorp, and T. W. Waiters, “Development of a foam insulation system for pressure stabilized liquid hydrogen propellant tanks”, *AIAA 23rd Thermophysics, Plasmadynamics and Lasers Conference, 1988*, no. September, 1998. DOI: 10.2514/6.1988-2737.

[67] P. J. Shirron, “Cooling capabilities of adiabatic demagnetization refrigerators”, *Journal of Low Temperature Physics*, vol. 148, no. 5-6, pp. 915–920, 2007, ISSN: 00222291. DOI: 10.1007/s10909-007-9441-7.

[68] L. Duband, “Space cryocooler developments”, *Physics Procedia*, vol. 67, pp. 1–10, 2015, ISSN: 18753892. DOI: 10.1016/j.phpro.2015.06.003. [Online]. Available: http://dx.doi.org/10.1016/j.phpro.2015.06.003.

[69] R. Radebaugh, “Cryocoolers: The state of the art and recent developments”, *Journal of Physics Condensed Matter*, vol. 21, no. 16, 2009, ISSN: 09538984. DOI: 10.1088/0953-8984/21/16/164219.

[70] (May 2019). Cryocoolers for space applications. Department of Engineering Sciences, University of Oxford, [Online]. Available: http://www2.eng.ox.ac.uk/cryogenics/research/cryocoolers-for-space-applications.

[71] P. Kittel, “Cryocooler Performance Estimator”, *Cryocoolers*, vol. 14, pp. 563–572, 2007.

[72] H. J. Ter Brake and G. F. Wiegerinck, “Low-power cryocooler survey”, *Cryogenics*, vol. 42, no. 11, pp. 705–718, 2002, ISSN: 00112275. DOI: 10.1016/S0011-2275(02)00143-1.

[73] T. Strobridge, “Cryogenic Refrigerators – an Updated Survey”, *NBS Tech Note 655*, 1974.

[74] D. S. Glaister and D. G. T. Curran, “Spacecraft Cryocooler System Integration Trades and Optimization”, *Cryocoolers 9*, pp. 873–884, 1997. DOI: 10.1007/978-1-4615-5869-9_98.

[75] C. H. Panzarella and M. Kassemi, “On the validity of purely thermodynamic descriptions of two-phase cryogenic fluid storage”, *Journal of Fluid Mechanics*, vol. 484, no. 484, pp. 41–68, 2003, ISSN: 00221120. DOI: 10.1017/S0022112003004002.

[76] A. Isselhorst, D. Suslov, and J. Riccius, “TRL 5 Tests of a Sub-Scale LH2 Feed Line Evaporation Cooler Armin”, pp. 2–12, 2017. DOI: 10.13009/EUCASS2017-59.

[77] B. Benthem, *Lecture slides ae4s20 satellite thermal control lecture 6*, 2015.

[78] R. Klees and R. Dwight, *Applied Numerical Analysis Reader AE2220 - Part 1*. 2014, ch. 8. Numerical methods for solving ordinary differential equations, pp. 125–137.

[79] J. Leijtens, “Verifying and Validating”, in *AE4S12 Space Systems Engineering*, Delft, 2017.

[80] J. E. Fesmire and W. L. Johnson, “Cylindrical cryogenic calorimeter testing of six types of multilayer insulation systems”, *Cryogenics*, vol. 89, no. July 2017, pp. 58–75, 2018, ISSN: 00112275. DOI: 10.1016/j.cryogenics.2017.11.004. [Online]. Available: https://doi.org/10.1016/j.cryogenics.2017.11.004.

[81] M. Kassemi, “The Zero Boil-off Tank (ZBOT) Experiment”, NCSER, Tech. Rep. 3.9, 2008.

[82] R. J. G. Hermsen, “Cryogenic propellant tank pressurization Practical investigation on the tank collapse factor for small, high-pressure, cryogenic rocket propellant tanks”, pp. 1–191, 2017.

[83] RMI Titanium, “Titanium Alloy Guide”, p. 28, 2000.

[84] NASA, “External Tank Thermal Protection System”, *NASA Facts*, vol. Pub 8-4039, pp. 1–4, 2005.

[85] M. M. Finckenor and D. Dooling, “Multilayer Insulation Material Guidelines”, *NASA Technical Paper*, vol. 209263, no. April, pp. 1–44, 1999. DOI: NASA/TP-1999-209263.

[86] W. L. Johnson and J. E. Fesmire, “Thermal performance of low layer density multilayer insulation using liquid nitrogen”, pp. 1–9, 2012.

[87] DuPont, “Mylar® Product Information”, 2003. [Online]. Available: http://usa.dupontteijinfilms.com/wp-content/uploads/2017/01/Mylar%7B%5C_%7DPhysical%7B%5C_%7DProperties.pdf.

[88] J. R. Feller and W. L. Johnson, “Dependence of multi-layer insulation thermal performance on interstitial gas pressure”, *AIP Conference Proceedings*, vol. 1434, no. 57, pp. 47–54, 2012, ISSN: 0094243X. DOI: 10.1063/1.4706904.

[89] S. V. Szabo, “Centaur space vehicle pressurized propellant feed system tests”, Lewis Research Center, Tech. Rep., 1972, p. 206.

[90] United Launch Alliance, “History of the Titan Centaur Launch Vehicle”, no. April, pp. 1–10, 2004.

[91] Lewis Staff, “Flight Performance of Atlas-Centaur AC-13, AC-14, AC-15 in Support of the Surveyor Lunar Landing Program”, Lewis Research Center, Tech. Rep., 1970, pp. 1–246.

A

Navier-Stokes equations

The Navier-Stokes equations are relations describing the conservation of mass (equation A.1), momentum (equation A.2) and energy (equation A.3) in a liquid or gas. The equations are valid for Newtonian fluids, which is a fluid of constant viscosity [1]. In Navier-Stokes equations the pressure, density, internal energy and velocity of the fluid and the gas are used. With the internal energy, the temperature of the fluid can be calculated by $e = c_v T$ [2]. Alternative to the Navier-Stokes equations are the Euler equations [3].

$$\frac{\partial \rho}{\partial t} + \vec{\nabla} \cdot (\rho \vec{u}) = 0 \tag{A.1}$$

$$\frac{\partial (\rho \vec{u})}{\partial t} + \vec{\nabla} \cdot [\rho \vec{u} \vec{u}] = -\overrightarrow{\nabla p} + \vec{\nabla} \cdot \vec{\tau} + \rho \vec{f} \tag{A.2}$$

$$\frac{\partial (\rho E)}{\partial t} + \vec{\nabla} \cdot ((\rho E + p)\vec{u}) = \vec{\nabla} \cdot (\bar{\bar{\tau}} \cdot \vec{u}) + \rho \vec{f} \vec{u} + \vec{\nabla} \cdot (\vec{q}) + r \tag{A.3}$$

For most numerical models created, the Boussinesq approximation is used. This approximation omits all the differences in the fluid concerning density, except the terms which are multiplied by the gravity [4] (also called buoyancy terms). This results in fluid behaviour in which gravity and buoyancy are modelled, but inertia differences are not. Equation A.1 reduces with this approximation to equation A.4.

$$\vec{\nabla} \cdot \vec{u} = 0 \tag{A.4}$$

B

Simulation flow diagram

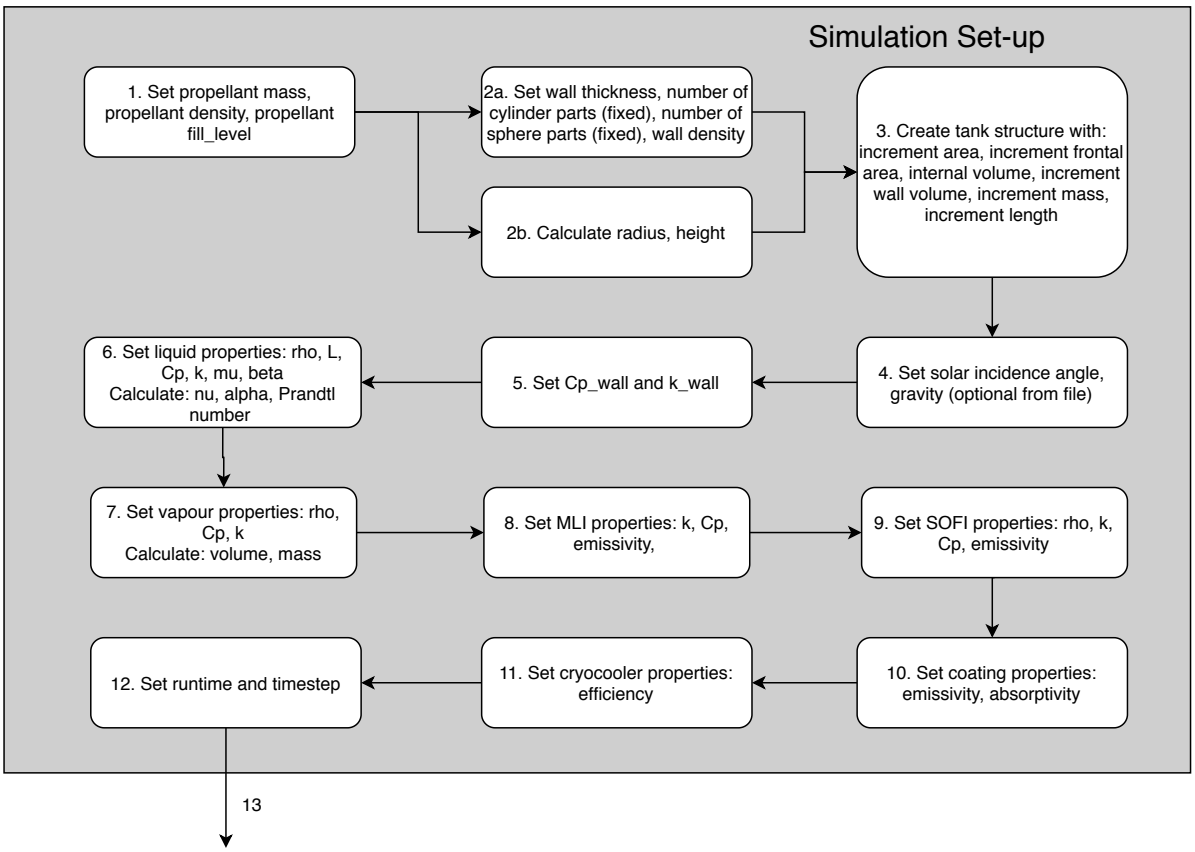

Figure B.1: Simulation set-up module of the software tool developed.

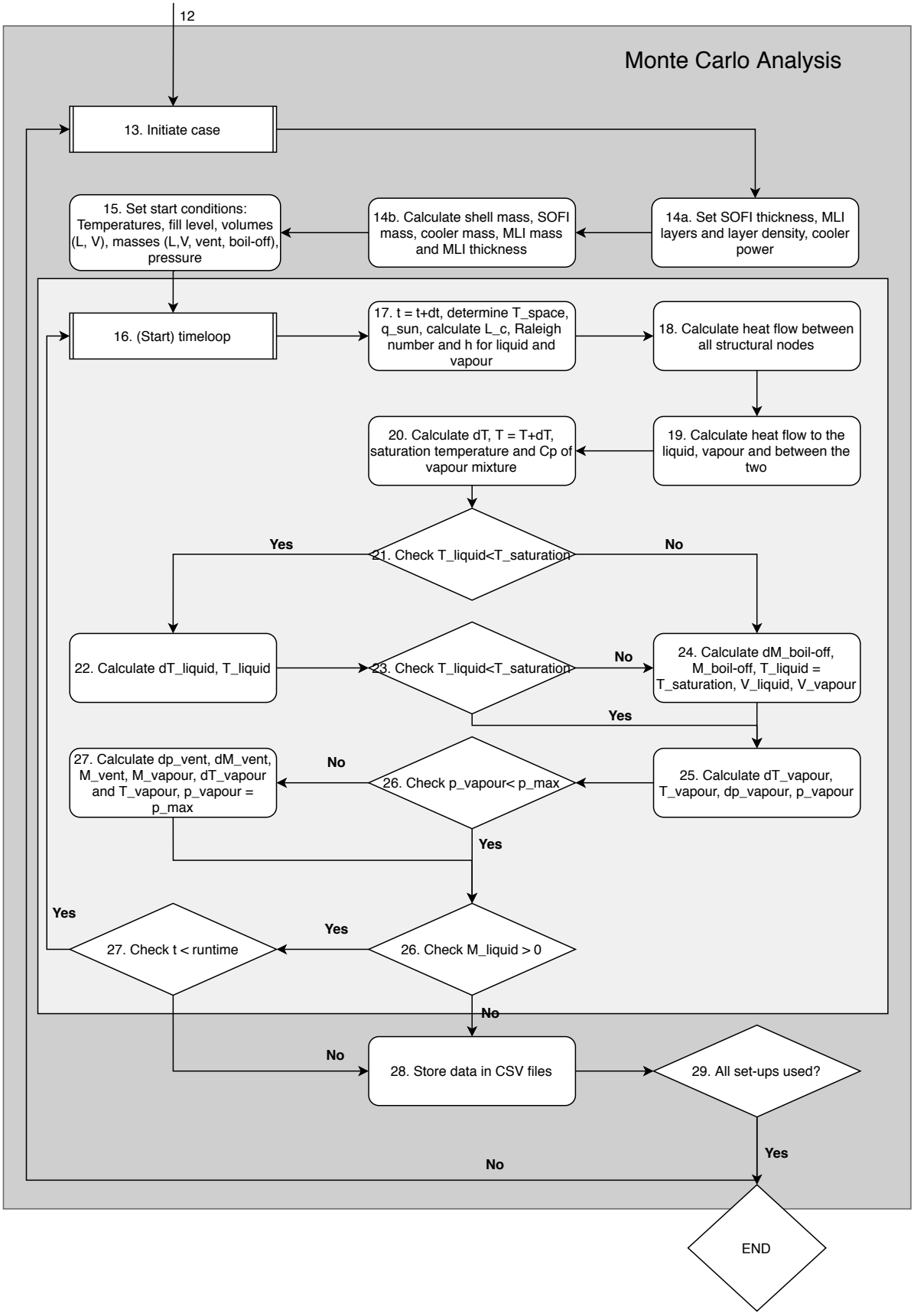

Figure B.2: Flow diagram of the Monte Carlo Analysis module of the software tool developed.

C

Boil-off tool instructions

In this appendix, instructions are presented for others to use the Boil-off Monte Carlo program developed.

C.1. Python v2.7 and required packages

In section 3.5 it has been explained that the program is developed in Python. It is possible to create an executable of the program. However, an executable is operating system dependant: an executable created on a macOS system is only runnable on a macOS system. Therefore, users are not recommended to use an executable but use Python. Python is easy to install from `https://www.python.org/downloads/` or by installing Anaconda Python from `https://www.anaconda.com/distribution/`. The program is written in Python version 2.7. This version is not supported anymore from January 2020 onwards. New users are therefore recommended to install Python version 3.7.

With Python installed, there are a few more requirements. The Python environment allows the user to manually install packages. The following packages need to be installed:

- Pandas
- tqdm
- Numpy
- pathlib
- Cython

The other packages used in the program are installed along with the installation of Python.

C.2. Retrieve liquid and vapour properties from NIST

The program uses liquid and vapour properties. These properties are retrieved from the NIST database. To obtain the properties, visit `https://webbook.nist.gov/chemistry/` and follow the next instruction:

- Click on ”Formula” and search for either ”He” or ”H2”;
- Click on the desired name;
- Under ”Other Data Available:”, click on ”Fluid Properties”;
- Set the units to [K], [bar], [kg/m3], [kJ/kg], [m/s], [Pa*s] and [N/m], select the ”Isobaric” option and click ”Press to Continue”;
- Enter the maximum allowable pressure in [bar] and set the minimum and maximum expected fluid/vapour temperature. Use an increment of 0.1 [K] and click ”Press for Data”;

- Under ”Other Data Available:” click on ”Viem data in HTML table.”;
- Copy the HTML table from the bottom to the top and paste the data in an .xlsx file, remove double entries;
- Store the .xlsx file with the name ”H2_properties.xlsx” for LH2 and ”He_properties.xlsx” for Helium, store the files in the same folder as where the program is located.

C.3. Run Boil-off program

With Python and the packages installed, it is required to ”cythonize”[1] the program. To do so, open a terminal window on your operating system. In this terminal window, direct to the folder in which all the files of the program are located. Then, type in the terminal ”python setup.py build_ext –inplace”, after which the terminal starts to ”cythonize” the code. After the process, a .so and a .c file should be present in the program folder. Before the program can be used, make sure the following files are located in one folder:

- Boiloff_Monte_Carlo_program_and_parameters.py
- H2_properties.xlsx
- He_properties.xlsx
- setup.py
- Tank_v15.c
- Tank_v15.pyx
- Tank_v15.so

It is optional to include an environment file with the name ”Environment_properties.xlsx”.

The program is ready to be used. To set the design options, the ”Boiloff_Monte_Carlo_program_and_parameters.py” file should be opened and in the file, the design options should be given on the places allocated. When this is done, run the file and the program starts calculating.

C.4. Runtime statistics

The calculation time per design option is depending on a few factors. Two factors are input parameters: the mission time and the time step used. The other factors are computer characteristics. It is not researched what the influence is of separate computer characteristics. However, it is known that Python runs programs on a single core and that the number of calculations a core is able to perform is depending on the processor speed. Two different processors have been used to perform calculations, the first was a Apple Macbook Air. The second was a Google Cloud server.

- 1.6 GHz Dual-Core Intel Core I5 Processor (Macbook Air): 40 minutes per design option for a mission time of 24 months and 34,600 kg tank size.
- 3.6 GHz Intel Xeon W-2223 Processor (Google Cloud server): 20 minutes per design option for a mission time of 24 months and 34,600 kg tank size.

It should be noted that the times logged are depending on the exact settings of the program and also how often parameters are stored in files.

Table C.1: Overview tasks at which Boil-off Monte Carlo program lines

Lines	Task
1-10	import Python packages
12-46	create .csv files
48-191	define Functions and Classes
193-199	create parameters
201-208	(optional) import environment file
210-219	import LH2 properties
221-230	import He properties
232-776	create variables and constants
779-782	import constant environment
784-787	create different design options from import
789	start Monte Carlo analysis
790-878	(re)set initial parameter values
880	enter time (while) loop
886-908	update LH2 and He properties every $\Delta T = 0.5[K]$
910-918	(optional) update non-constant environment from file
919-920	determine characteristic length
922-940	determine average shell temperature
942-945	update Rayleigh and coefficient of convection
948-1625	determine heat flow between nodes
1627-1670	determine heat flow liquid and vapour
1681-1812	determine temperature
1814-1823	determine saturation temperature and vapour mix properties
1826-1859	if $T_{liquid} < T_{saturation}$ determine temperature, pressure, volume of liquid and vapour
1860-1892	if $T_{liquid} > T_{saturation}$ determine boil-off, pressure, fill level, volume and vent mass
1894-1937	store data in .csv files
1939-2731	same procedure lines 880-1937 but without MLI

C.5. Raw code

The program code is only presented in the digital version, available in the TU Delft repository. The part in which the MLI is omitted, is not included in the code. The reason is because that part of the code is the same as with MLI, except for that the nodes with suffix A and i are set to zero. In table C.1 is given which lines of code perform what.

```
1  import csv
2  import itertools as it
3  from math import *
4  import pathlib
5  import numpy as np
6  import pandas as pd
7  import Boiloff_Monte_Carlo_program_and_parameters as pf
8  from tqdm.auto import tqdm
9  import os
10 cdef int months = pf.months
11
12 with open('mass_and_boil-off_v15-%sm.csv' %months, 'w') as csvfile:
13     filewriter = csv.writer(csvfile, delimiter= ',', quotechar = '|', quoting=csv.
           QUOTE_MINIMAL)
14     filewriter.writerow(['t_SOFI [m]', 'SOFI mass [kg]', 'layer density MLI [layers/cm]', '
           layers of MLI', 'MLI mass [kg]', 'cooler power [W]', 'Cooler mass [kg]', 't_shell [m]
           ', 'Shell mass [kg]', 'Boil-off mass [kg]', 'Vent mass [kg]', 'Total mass [kg]', '
           Boil-off rate [%/month]', 'Time start boil [days]', 'Time to boil-off [days]', '
           T_liquid', 'T_vapour', 'T_1a', 'T_1i', 'T_1b', 'T_1c', 'T_1d', 'T_2a', 'T_2i', 'T_2b'
           , 'T_2c', 'T_2d', 'T_3a', 'T_3i', 'T_3b', 'T_3c', 'T_3d', 'T_4a', 'T_4i', 'T_4b', '
           T_4c', 'T_4d', 'T_5a', 'T_5i', 'T_5b', 'T_5c', 'T_5d', 'T_6a', 'T_6i', 'T_6b', 'T_6c'
           , 'T_6d',  'T_7a', 'T_7i', 'T_7b', 'T_7c', 'T_7d', 'T_8a', 'T_8i', 'T_8b', 'T_8c', '
           T_8d', 'T_9a', 'T_9i', 'T_9b', 'T_9c', 'T_9d', 'T_10a', 'T_10i', 'T_10b', 'T_10c', '
```

[1]"cythonizing" is the process of converting the Python code to a C code.

```
        T_10d', 'T_11a', 'T_11i', 'T_11b', 'T_11c', 'T_11d', 'T_12a', 'T_12i', 'T_12b', '
        T_12c', 'T_12d', 'dP_liquid_net', 'dP_vapour_net', 'dP_1a_net', 'dP_1i_net', '
        dP_1b_net', 'dP_1c_net', 'dP_1d_net', 'dP_2a_net', 'dP_2i_net', 'dP_2b_net', '
        dP_2c_net', 'dP_2d_net', 'dP_3a_net', 'dP_3i_net', 'dP_3b_net', 'dP_3c_net', '
        dP_3d_net', 'dP_4a_net', 'dP_4i_net', 'dP_4b_net', 'dP_4c_net', 'dP_4d_net', '
        dP_5a_net', 'dP_5i_net', 'dP_5b_net', 'dP_5c_net', 'dP_5d_net', 'dP_6a_net', '
        dP_6i_net', 'dP_6b_net', 'dP_6c_net', 'dP_6d_net', 'dP_7a_net', 'dP_7i_net', '
        dP_7b_net', 'dP_7c_net', 'dP_7d_net', 'dP_8a_net', 'dP_8i_net', 'dP_8b_net', '
        dP_8c_net', 'dP_8d_net', 'dP_9a_net', 'dP_9i_net', 'dP_9b_net', 'dP_9c_net', '
        dP_9d_net', 'dP_10a_net', 'dP_10i_net', 'dP_10b_net', 'dP_10c_net', 'dP_10d_net', '
        dP_11a_net', 'dP_11i_net', 'dP_11b_net', 'dP_11c_net', 'dP_11d_net', 'dP_12a_net', '
        dP_12i_net', 'dP_12b_net', 'dP_12c_net', 'dP_12d_net'])

csvfile.close()

if months > 6:
    with open('mass_and_boil-off_v15-6m.csv', 'w') as csvfile:
        filewriter = csv.writer(csvfile, delimiter= ',', quotechar = '|', quoting=csv.
            QUOTE_MINIMAL)
        filewriter.writerow(['t_SOFI [m]', 'SOFI mass [kg]', 'layer density MLI [layers/cm]',
             'layers of MLI', 'MLI mass [kg]', 'cooler power [W]', 'Cooler mass [kg]', '
            t_shell [m]', 'Shell mass [kg]', 'Boil-off mass [kg]', 'Vent mass [kg]', 'Total
            mass [kg]', 'Boil-off rate [%/month]', 'Time start boil [days]', 'Time to boil-
            off [days]', 'T_liquid', 'T_vapour', 'T_1a', 'T_1i', 'T_1b', 'T_1c', 'T_1d', '
            T_2a', 'T_2i', 'T_2b', 'T_2c', 'T_2d', 'T_3a', 'T_3i', 'T_3b', 'T_3c', 'T_3d', '
            T_4a', 'T_4i', 'T_4b', 'T_4c', 'T_4d', 'T_5a', 'T_5i', 'T_5b', 'T_5c', 'T_5d', '
            T_6a', 'T_6i', 'T_6b', 'T_6c', 'T_6d',  'T_7a', 'T_7i', 'T_7b', 'T_7c', 'T_7d', '
            T_8a', 'T_8i', 'T_8b', 'T_8c', 'T_8d', 'T_9a', 'T_9i', 'T_9b', 'T_9c', 'T_9d', '
            T_10a', 'T_10i', 'T_10b', 'T_10c', 'T_10d', 'T_11a', 'T_11i', 'T_11b', 'T_11c', '
            T_11d', 'T_12a', 'T_12i', 'T_12b', 'T_12c', 'T_12d', 'dP_liquid_net', '
            dP_vapour_net', 'dP_1a_net', 'dP_1i_net', 'dP_1b_net', 'dP_1c_net', 'dP_1d_net',
            'dP_2a_net', 'dP_2i_net', 'dP_2b_net', 'dP_2c_net', 'dP_2d_net', 'dP_3a_net', '
            dP_3i_net', 'dP_3b_net', 'dP_3c_net', 'dP_3d_net', 'dP_4a_net', 'dP_4i_net', '
            dP_4b_net', 'dP_4c_net', 'dP_4d_net', 'dP_5a_net', 'dP_5i_net', 'dP_5b_net', '
            dP_5c_net', 'dP_5d_net', 'dP_6a_net', 'dP_6i_net', 'dP_6b_net', 'dP_6c_net', '
            dP_6d_net', 'dP_7a_net', 'dP_7i_net', 'dP_7b_net', 'dP_7c_net', 'dP_7d_net', '
            dP_8a_net', 'dP_8i_net', 'dP_8b_net', 'dP_8c_net', 'dP_8d_net', 'dP_9a_net', '
            dP_9i_net', 'dP_9b_net', 'dP_9c_net', 'dP_9d_net', 'dP_10a_net', 'dP_10i_net', '
            dP_10b_net', 'dP_10c_net', 'dP_10d_net', 'dP_11a_net', 'dP_11i_net', 'dP_11b_net'
            , 'dP_11c_net', 'dP_11d_net', 'dP_12a_net', 'dP_12i_net', 'dP_12b_net', '
            dP_12c_net', 'dP_12d_net'])
    csvfile.close()

if months > 9:
    with open('mass_and_boil-off_v15-9m.csv', 'w') as csvfile:
        filewriter = csv.writer(csvfile, delimiter= ',', quotechar = '|', quoting=csv.
            QUOTE_MINIMAL)
        filewriter.writerow(['t_SOFI [m]', 'SOFI mass [kg]', 'layer density MLI [layers/cm]',
             'layers of MLI', 'MLI mass [kg]', 'cooler power [W]', 'Cooler mass [kg]', '
            t_shell [m]', 'Shell mass [kg]', 'Boil-off mass [kg]', 'Vent mass [kg]', 'Total
            mass [kg]', 'Boil-off rate [%/month]', 'Time start boil [days]', 'Time to boil-
            off [days]', 'T_liquid', 'T_vapour', 'T_1a', 'T_1i', 'T_1b', 'T_1c', 'T_1d', '
            T_2a', 'T_2i', 'T_2b', 'T_2c', 'T_2d', 'T_3a', 'T_3i', 'T_3b', 'T_3c', 'T_3d', '
            T_4a', 'T_4i', 'T_4b', 'T_4c', 'T_4d', 'T_5a', 'T_5i', 'T_5b', 'T_5c', 'T_5d', '
            T_6a', 'T_6i', 'T_6b', 'T_6c', 'T_6d',  'T_7a', 'T_7i', 'T_7b', 'T_7c', 'T_7d', '
            T_8a', 'T_8i', 'T_8b', 'T_8c', 'T_8d', 'T_9a', 'T_9i', 'T_9b', 'T_9c', 'T_9d', '
            T_10a', 'T_10i', 'T_10b', 'T_10c', 'T_10d', 'T_11a', 'T_11i', 'T_11b', 'T_11c', '
            T_11d', 'T_12a', 'T_12i', 'T_12b', 'T_12c', 'T_12d', 'dP_liquid_net', '
            dP_vapour_net', 'dP_1a_net', 'dP_1i_net', 'dP_1b_net', 'dP_1c_net', 'dP_1d_net',
            'dP_2a_net', 'dP_2i_net', 'dP_2b_net', 'dP_2c_net', 'dP_2d_net', 'dP_3a_net', '
            dP_3i_net', 'dP_3b_net', 'dP_3c_net', 'dP_3d_net', 'dP_4a_net', 'dP_4i_net', '
            dP_4b_net', 'dP_4c_net', 'dP_4d_net', 'dP_5a_net', 'dP_5i_net', 'dP_5b_net', '
            dP_5c_net', 'dP_5d_net', 'dP_6a_net', 'dP_6i_net', 'dP_6b_net', 'dP_6c_net', '
            dP_6d_net', 'dP_7a_net', 'dP_7i_net', 'dP_7b_net', 'dP_7c_net', 'dP_7d_net', '
            dP_8a_net', 'dP_8i_net', 'dP_8b_net', 'dP_8c_net', 'dP_8d_net', 'dP_9a_net', '
            dP_9i_net', 'dP_9b_net', 'dP_9c_net', 'dP_9d_net', 'dP_10a_net', 'dP_10i_net', '
            dP_10b_net', 'dP_10c_net', 'dP_10d_net', 'dP_11a_net', 'dP_11i_net', 'dP_11b_net'
            , 'dP_11c_net', 'dP_11d_net', 'dP_12a_net', 'dP_12i_net', 'dP_12b_net', '
            dP_12c_net', 'dP_12d_net'])
    csvfile.close()

```

```
30 if months > 12:
31     with open('mass_and_boil-off_v15-12m.csv', 'w') as csvfile:
32         filewriter = csv.writer(csvfile, delimiter= ',', quotechar = '|', quoting=csv.
               QUOTE_MINIMAL)
33         filewriter.writerow(['t_SOFI [m]', 'SOFI mass [kg]', 'layer density MLI [layers/cm]',
                'layers of MLI', 'MLI mass [kg]', 'cooler power [W]', 'Cooler mass [kg]', '
               t_shell [m]', 'Shell mass [kg]', 'Boil-off mass [kg]', 'Vent mass [kg]', 'Total
               mass [kg]', 'Boil-off rate [%/month]', 'Time start boil [days]', 'Time to boil-
               off [days]', 'T_liquid', 'T_vapour', 'T_1a', 'T_1i', 'T_1b', 'T_1c', 'T_1d', '
               T_2a', 'T_2i', 'T_2b', 'T_2c', 'T_2d', 'T_3a', 'T_3i', 'T_3b', 'T_3c', 'T_3d', '
               T_4a', 'T_4i', 'T_4b', 'T_4c', 'T_4d', 'T_5a', 'T_5i', 'T_5b', 'T_5c', 'T_5d', '
               T_6a', 'T_6i', 'T_6b', 'T_6c', 'T_6d',  'T_7a', 'T_7i', 'T_7b', 'T_7c', 'T_7d', '
               T_8a', 'T_8i', 'T_8b', 'T_8c', 'T_8d', 'T_9a', 'T_9i', 'T_9b', 'T_9c', 'T_9d', '
               T_10a', 'T_10i', 'T_10b', 'T_10c', 'T_10d', 'T_11a', 'T_11i', 'T_11b', 'T_11c', '
               T_11d', 'T_12a', 'T_12i', 'T_12b', 'T_12c', 'T_12d', 'dP_liquid_net', '
               dP_vapour_net', 'dP_1a_net', 'dP_1i_net', 'dP_1b_net', 'dP_1c_net', 'dP_1d_net',
               'dP_2a_net', 'dP_2i_net', 'dP_2b_net', 'dP_2c_net', 'dP_2d_net', 'dP_3a_net', '
               dP_3i_net', 'dP_3b_net', 'dP_3c_net', 'dP_3d_net', 'dP_4a_net', 'dP_4i_net', '
               dP_4b_net', 'dP_4c_net', 'dP_4d_net', 'dP_5a_net', 'dP_5i_net', 'dP_5b_net', '
               dP_5c_net', 'dP_5d_net', 'dP_6a_net', 'dP_6i_net', 'dP_6b_net', 'dP_6c_net', '
               dP_6d_net', 'dP_7a_net', 'dP_7i_net', 'dP_7b_net', 'dP_7c_net', 'dP_7d_net', '
               dP_8a_net', 'dP_8i_net', 'dP_8b_net', 'dP_8c_net', 'dP_8d_net', 'dP_9a_net', '
               dP_9i_net', 'dP_9b_net', 'dP_9c_net', 'dP_9d_net', 'dP_10a_net', 'dP_10i_net', '
               dP_10b_net', 'dP_10c_net', 'dP_10d_net', 'dP_11a_net', 'dP_11i_net', 'dP_11b_net'
               , 'dP_11c_net', 'dP_11d_net', 'dP_12a_net', 'dP_12i_net', 'dP_12b_net', '
               dP_12c_net', 'dP_12d_net'])
34     csvfile.close()
35
36 if months > 18:
37     with open('mass_and_boil-off_v15-18m.csv', 'w') as csvfile:
38         filewriter = csv.writer(csvfile, delimiter= ',', quotechar = '|', quoting=csv.
               QUOTE_MINIMAL)
39         filewriter.writerow(['t_SOFI [m]', 'SOFI mass [kg]', 'layer density MLI [layers/cm]',
                'layers of MLI', 'MLI mass [kg]', 'cooler power [W]', 'Cooler mass [kg]', '
               t_shell [m]', 'Shell mass [kg]', 'Boil-off mass [kg]', 'Vent mass [kg]', 'Total
               mass [kg]', 'Boil-off rate [%/month]', 'Time start boil [days]', 'Time to boil-
               off [days]', 'T_liquid', 'T_vapour', 'T_1a', 'T_1i', 'T_1b', 'T_1c', 'T_1d', '
               T_2a', 'T_2i', 'T_2b', 'T_2c', 'T_2d', 'T_3a', 'T_3i', 'T_3b', 'T_3c', 'T_3d', '
               T_4a', 'T_4i', 'T_4b', 'T_4c', 'T_4d', 'T_5a', 'T_5i', 'T_5b', 'T_5c', 'T_5d', '
               T_6a', 'T_6i', 'T_6b', 'T_6c', 'T_6d',  'T_7a', 'T_7i', 'T_7b', 'T_7c', 'T_7d', '
               T_8a', 'T_8i', 'T_8b', 'T_8c', 'T_8d', 'T_9a', 'T_9i', 'T_9b', 'T_9c', 'T_9d', '
               T_10a', 'T_10i', 'T_10b', 'T_10c', 'T_10d', 'T_11a', 'T_11i', 'T_11b', 'T_11c', '
               T_11d', 'T_12a', 'T_12i', 'T_12b', 'T_12c', 'T_12d', 'dP_liquid_net', '
               dP_vapour_net', 'dP_1a_net', 'dP_1i_net', 'dP_1b_net', 'dP_1c_net', 'dP_1d_net',
               'dP_2a_net', 'dP_2i_net', 'dP_2b_net', 'dP_2c_net', 'dP_2d_net', 'dP_3a_net', '
               dP_3i_net', 'dP_3b_net', 'dP_3c_net', 'dP_3d_net', 'dP_4a_net', 'dP_4i_net', '
               dP_4b_net', 'dP_4c_net', 'dP_4d_net', 'dP_5a_net', 'dP_5i_net', 'dP_5b_net', '
               dP_5c_net', 'dP_5d_net', 'dP_6a_net', 'dP_6i_net', 'dP_6b_net', 'dP_6c_net', '
               dP_6d_net', 'dP_7a_net', 'dP_7i_net', 'dP_7b_net', 'dP_7c_net', 'dP_7d_net', '
               dP_8a_net', 'dP_8i_net', 'dP_8b_net', 'dP_8c_net', 'dP_8d_net', 'dP_9a_net', '
               dP_9i_net', 'dP_9b_net', 'dP_9c_net', 'dP_9d_net', 'dP_10a_net', 'dP_10i_net', '
               dP_10b_net', 'dP_10c_net', 'dP_10d_net', 'dP_11a_net', 'dP_11i_net', 'dP_11b_net'
               , 'dP_11c_net', 'dP_11d_net', 'dP_12a_net', 'dP_12i_net', 'dP_12b_net', '
               dP_12c_net', 'dP_12d_net'])
40     csvfile.close()
41
42 if months > 24:
43     with open('mass_and_boil-off_v15-24m.csv', 'w') as csvfile:
44         filewriter = csv.writer(csvfile, delimiter= ',', quotechar = '|', quoting=csv.
               QUOTE_MINIMAL)
45         filewriter.writerow(['t_SOFI [m]', 'SOFI mass [kg]', 'layer density MLI [layers/cm]',
                'layers of MLI', 'MLI mass [kg]', 'cooler power [W]', 'Cooler mass [kg]', '
               t_shell [m]', 'Shell mass [kg]', 'Boil-off mass [kg]', 'Vent mass [kg]', 'Total
               mass [kg]', 'Boil-off rate [%/month]', 'Time start boil [days]', 'Time to boil-
               off [days]', 'T_liquid', 'T_vapour', 'T_1a', 'T_1i', 'T_1b', 'T_1c', 'T_1d', '
               T_2a', 'T_2i', 'T_2b', 'T_2c', 'T_2d', 'T_3a', 'T_3i', 'T_3b', 'T_3c', 'T_3d', '
               T_4a', 'T_4i', 'T_4b', 'T_4c', 'T_4d', 'T_5a', 'T_5i', 'T_5b', 'T_5c', 'T_5d', '
               T_6a', 'T_6i', 'T_6b', 'T_6c', 'T_6d',  'T_7a', 'T_7i', 'T_7b', 'T_7c', 'T_7d', '
               T_8a', 'T_8i', 'T_8b', 'T_8c', 'T_8d', 'T_9a', 'T_9i', 'T_9b', 'T_9c', 'T_9d', '
               T_10a', 'T_10i', 'T_10b', 'T_10c', 'T_10d', 'T_11a', 'T_11i', 'T_11b', 'T_11c', '
               T_11d', 'T_12a', 'T_12i', 'T_12b', 'T_12c', 'T_12d', 'dP_liquid_net', '
```

```
            dP_vapour_net', 'dP_1a_net', 'dP_1i_net', 'dP_1b_net', 'dP_1c_net', 'dP_1d_net',
            'dP_2a_net', 'dP_2i_net', 'dP_2b_net', 'dP_2c_net', 'dP_2d_net', 'dP_3a_net', '
            dP_3i_net', 'dP_3b_net', 'dP_3c_net', 'dP_3d_net', 'dP_4a_net', 'dP_4i_net', '
            dP_4b_net', 'dP_4c_net', 'dP_4d_net', 'dP_5a_net', 'dP_5i_net', 'dP_5b_net', '
            dP_5c_net', 'dP_5d_net', 'dP_6a_net', 'dP_6i_net', 'dP_6b_net', 'dP_6c_net', '
            dP_6d_net', 'dP_7a_net', 'dP_7i_net', 'dP_7b_net', 'dP_7c_net', 'dP_7d_net', '
            dP_8a_net', 'dP_8i_net', 'dP_8b_net', 'dP_8c_net', 'dP_8d_net', 'dP_9a_net', '
            dP_9i_net', 'dP_9b_net', 'dP_9c_net', 'dP_9d_net', 'dP_10a_net', 'dP_10i_net', '
            dP_10b_net', 'dP_10c_net', 'dP_10d_net', 'dP_11a_net', 'dP_11i_net', 'dP_11b_net'
            , 'dP_11c_net', 'dP_11d_net', 'dP_12a_net', 'dP_12i_net', 'dP_12b_net', '
            dP_12c_net', 'dP_12d_net'])
    csvfile.close()

cdef conduction(k,DT,L):
    cdef dq = -k*DT/L
    return dq

cdef radiation(epsilon, T1, T2):
    cdef sigma = 5.670373e-8
    cdef dq = epsilon*sigma*-(T1**4.-T2**4.)
    return dq

cdef pressure_increase(T_s,L,m,V,Q):
    cdef R_G = 8.314469848  # J/K/mol (universal gas constant)
    cdef dpv = R_G*T_s/(L*m*V)*Q
    return dpv

cdef saturation_temperature(p):
    cdef p_0 = 101325.  # Pa
    cdef R_G = 8.314469848 #J/K/mol
    cdef T_0 = 20.369  # K (NIST)
    cdef dH_vap_mol = 0.8992e3  # J/mol (NIST)
    cdef T_saturation = (1 / T_0 - (R_G * log(p/p_0) / dH_vap_mol))** (-1)
    return T_saturation

cdef Rayleigh(L_c, g, beta, alpha, kinematic_viscosity, T_s, T_inf):
    cdef Ra = (g*beta)/(kinematic_viscosity*alpha)*abs(((T_s+T_inf)/2.-T_inf))*L_c**3.
    return Ra

cdef h(k,L_c,Ra):
    cdef h
    cdef Nu
    if Ra < 0:
        Ra = -Ra
        if Ra <= 1.e7:
            Nu = 0.642*Ra**(1./6.)
        elif 1.e7 < Ra <= 1.e10:
            Nu = 0.167*Ra**(1./4.)
        elif 1.e10 < Ra <= 5.*1.e13:
            Nu = 0.00053*Ra**(1./2.)
        else:
            Nu = 0.00053*Ra**(1./2.)
        h = -k/L_c*Nu
    else:
        if Ra <= 1.e7:
            Nu = 0.642*Ra**(1./6.)
        elif 1.e7 < Ra <= 1.e10:
            Nu = 0.167*Ra**(1./4.)
        elif 1.e10 < Ra <= 5.*1.e13:
            Nu = 0.00053*Ra**(1./2.)
        else:
            Nu = 0.00053*Ra**(1./2.)
        h = k/L_c*Nu

    return h

cdef h_vap(k,L_c,Ra):
    cdef h
    cdef Nu
    if Ra <= 1.e7:
        Nu = 4.5
```

```
106     elif 1.e7 < Ra <= 1.e12:
107         Nu = 0.08*Ra**(1./4.)
108     else:
109         Nu = 0.08*Ra**(1./4.)
110 
111     h = k/L_c*Nu
112     return h
113 
114 cdef Lockheed(T_H, T_C, emissivity, N_star, N_s):
115     ### NON MODIFIED
116     # cdef A = 7.30e-8
117     # cdef n = 2.63
118     # cdef T_m = (T_H+T_C)/2.
119     # cdef B = 7.07e-10
120     ### MODIFIED
121     cdef A = 2.4*10.**(-4)
122     cdef n = 2.63
123     cdef T_m = (0.017+7.*10.**(-6)*(800.-(T_H+T_C)/2.)+0.0228*log((T_H+T_C)/2.))
124     cdef B = 4.944*10.**(-10)
125     cdef C = 1.46e4
126     cdef p_star = 1.33e-5
127     cdef m = -0.48
128     cdef dq_solidconduction = A*N_star**n*T_m*-(T_H-T_C)/N_s
129     cdef dq_radiation = B*emissivity*-(T_H**4.67-T_C**4.67)/N_s
130     cdef dq_gasconduction = C*p_star*-(T_H**(m+1.)-T_C**(m+1.))/N_s
131     cdef dq_total = dq_solidconduction+dq_radiation+dq_gasconduction
132     return dq_total, dq_solidconduction, dq_radiation, dq_gasconduction
133 
134 cdef Cryocooler_mass(Q_cooler, T_C, T_H, eta):
135     cdef carnot = T_C/(T_H-T_C)
136     cdef P_in_cooler = Q_cooler/eta*(1./carnot)
137     cdef mass_cooler = 0.1422*P_in_cooler**(0.905)+0.325*P_in_cooler
138     return mass_cooler
139 
140 cdef efficiency(Q_cooler):
141     cdef eta = 10. ** (min(
142     (-1.26281 + 0.45936 * log10(Q_cooler) - 0.08743 * log10(Q_cooler) ** 2.),
143     (-0.92237 + 0.07763 * log10(1 + Q_cooler)))) #FROM CHAI AND WILHITE
144     return eta
145 
146 class structure:
147     def __init__(self, height, radius, thickness, sphere_parts, cylinder_parts, wall_density,
          SOFI=0.):
148         self.height = height
149         self.radius = radius
150         self.thickness = thickness
151         self.sphere_parts = float(sphere_parts)
152         self.cylinder_parts = float(cylinder_parts)
153         self.wall_density = wall_density
154         self.wall_mass = (pi*((radius+thickness)**2.-radius**2.)*height+4./3.*pi*((radius+
              thickness)**3.-radius**3.))*wall_density
155 
156         self.sphere_increment_area = 4.*pi*self.radius**2./self.sphere_parts
157         self.sphere_increment_frontalarea = pi*(self.radius)**2./(0.5*self.sphere_parts)
158         self.sphere_internalvolume = (4/3.)*pi*(self.radius)**3.
159         self.sphere_increment_wallvolume = (4 / 3.) * pi * ((self.radius + self.thickness) **
               3. - (self.radius) ** 3.)/self.sphere_parts
160         self.sphere_increment_mass = self.sphere_increment_wallvolume * self.wall_density
161         self.sphere_increment_length = 2.*pi*self.radius/self.sphere_parts
162 
163         self.increment_area = (2*pi*self.radius)*self.height/self.cylinder_parts
164         self.increment_frontalarea = 2*(self.radius)*self.height/(0.5*self.cylinder_parts)
165         self.internalvolume = (pi*(self.radius)**2.)*self.height
166         self.increment_wallvolume = pi*((self.radius+self.thickness)**2. - (self.radius)**2.)
              *self.height/self.cylinder_parts
167         self.increment_mass = self.increment_wallvolume * self.wall_density
168         self.increment_length = height/self.cylinder_parts
169 
170 cdef volume_calc(mass = 0., fill_ratio = 0., max_radius = 0., propellant_type = 'LH2',
      propellant_density = 71.541, volume = 0.):
171     cdef double radius
```

```
172     cdef double height
173
174     if volume == 0.:
175         volume = mass/propellant_density
176
177     volume_tank = volume/fill_ratio
178     #IF THERE IS A MAX RADIUS, THIS RADIUS IS ALSO USED. FOR THE HIGHER THE RADIUS, THE
            BIGGER VOLUME OVER AREA
179     if max_radius != 0.:
180         volume_sphere = (4/3.)*pi*max_radius**3.
181         if volume_sphere > volume_tank:
182             radius = (volume_tank/(4./3.*pi))**(1./3.)
183             height = 0.
184         else:
185             volume_cylinder = volume_tank-volume_sphere
186             height = volume_cylinder/(pi*max_radius**2.)
187             radius = max_radius
188     else:
189         radius = (volume_tank/(4./3.*pi))**(1./3.)
190         height = 0.
191     return radius, height
192
193 cdef double R_G = 8.314469848  # J/K/mol (universal gas constant)
194
195 cdef double height_calc
196 cdef double radius_calc
197 cdef double fill_level_initial = pf.fill_level_initial
198
199 cdef time_dict = 0.
200
201 try:
202     time_dict = pf.environment_df.to_dict('index')
203 except:
204     try:
205         df = pd.read_excel(pathlib.Path.cwd() / 'Boil-off_Monte_Carlo_program' / '
                Environment_properties.xlsx').set_index('time [min]')
206         time_dict = df.to_dict('index')
207     except:
208         print 'Static environment used'
209
210 cdef df2
211 try:
212     df2 = pd.read_excel('./H2_properties.xlsx').set_index('Temperature (K)')
213 except:
214     try:
215         df2 = pd.read_excel(pathlib.Path.cwd() / 'Boil-off_Monte_Carlo_program' / '
                H2_properties.xlsx').set_index('Temperature (K)')
216     except:
217         print 'Give H2 properties excel'
218
219 cdef H2_dict = df2.to_dict('index')
220
221 cdef df3
222 try:
223     df3 = pd.read_excel('./He_properties.xlsx').set_index('Temperature (K)')
224 except:
225     try:
226         df2 = pd.read_excel(pathlib.Path.cwd() / 'Boil-off_Monte_Carlo_program' / '
                He_properties.xlsx').set_index('Temperature (K)')
227     except:
228         print 'Give He properties excel'
229
230 cdef He_dict = df3.to_dict('index')
231
232 #heat input
233 cdef double solar_incidence_angle = pf.solar_incidence_angle
234 cdef double grav = pf.gravity
235
236 #wall properties
237 cdef double Cp_wall = pf.Cp_wall #570. #J/kg*K
238 cdef double k_wall = pf.k_wall #7.3
```

```
239 cdef double density_wall = pf.density_wall
240 cdef double t_wall = pf.t_wall
241
242 #liquid properties
243 cdef density_liquid
244 cdef double L_H2 = 446.1 #J/g
245 cdef Cp_H2
246 cdef k_H2
247 cdef mu_H2
248 cdef nu_H2
249 cdef alpha_H2
250 cdef beta_H2
251 cdef double V_total
252 cdef fill_level
253 cdef double V_liquid
254 cdef m_liquid_initial
255
256 cdef Pr
257
258 #vapour properties
259 cdef density_vapour
260 cdef Cp_He_vapour
261 cdef double Cp_H2_vapour = 11.7e3
262 cdef Cp_mix
263 cdef k_He_vapour
264 cdef mu_He
265 cdef nu_He
266 cdef alpha_He
267 cdef beta_He
268 cdef double V_vapour
269 cdef m_vapour_initial
270 cdef double p_vapour_initial = pf.initial_pressure  #Pa
271
272 #SOFI properties NCFI 24-124 foam
273 cdef list t_SOFI_list = pf.t_SOFI
274 cdef double density_SOFI = pf.density_SOFI
275 cdef double k_SOFI = pf.k_SOFI
276 cdef double Cp_SOFI = pf.Cp_SOFI
277 cdef emissivity_SOFI = pf.emissivity_SOFI
278 cdef double t_SOFI
279
280 cdef layers
281 cpdef list MLI_layers = pf.MLI_layers
282
283 cdef double density_MLI = pf.density_MLI
284 cdef double k_MLI = pf.k_MLI
285 cdef double Cp_MLI = pf.Cp_MLI
286 cdef t_MLI
287 cdef double emissivity_MLI = pf.emissivity_MLI
288 cdef list layer_densities = pf.layer_densities
289 cdef layer_density
290
291
292 cdef double emissivity = pf.emissivity_coating
293 cdef double absorptivity = pf.absorptivity_coating
294
295 cdef list cooler_power = pf.cooler_power
296 cdef double Q_cooler = 0.
297
298 cdef double runtime = pf.t_run
299 cdef timestep = pf.dt
300
301 cdef t_boil = 0.
302 cdef dt
303 cdef t
304 cdef i
305
306 ### MASSES ###
307 cdef m_SOFI
308 cdef m_shell
309 cdef m_cooler
```

```
cdef m_MLI

cdef double T_start = pf.initial_temperature
cdef double T_space
cdef double T_1a, T_2a, T_3a, T_4a, T_5a, T_6a, T_7a, T_8a, T_9a, T_10a, T_11a, T_12a
cdef double T_1i, T_2i, T_3i, T_4i, T_5i, T_6i, T_7i, T_8i, T_9i, T_10i, T_11i, T_12i
cdef double T_1b, T_2b, T_3b, T_4b, T_5b, T_6b, T_7b, T_8b, T_9b, T_10b, T_11b, T_12b
cdef double T_1c, T_2c, T_3c, T_4c, T_5c, T_6c, T_7c, T_8c, T_9c, T_10c, T_11c, T_12c
cdef double T_1d, T_2d, T_3d, T_4d, T_5d, T_6d, T_7d, T_8d, T_9d, T_10d, T_11d, T_12d

cdef double T_sat
cdef double T_liquid
cdef double T_vapour

cdef dT_1a, dT_2a, dT_3a, dT_4a, dT_5a, dT_6a, dT_7a, dT_8a, dT_9a, dT_10a, dT_11a, dT_12a
cdef dT_1i, dT_2i, dT_3i, dT_4i, dT_5i, dT_6i, dT_7i, dT_8i, dT_9i, dT_10i, dT_11i, dT_12i
cdef dT_1b, dT_2b, dT_3b, dT_4b, dT_5b, dT_6b, dT_7b, dT_8b, dT_9b, dT_10b, dT_11b, dT_12b
cdef dT_1c, dT_2c, dT_3c, dT_4c, dT_5c, dT_6c, dT_7c, dT_8c, dT_9c, dT_10c, dT_11c, dT_12c
cdef dT_1d, dT_2d, dT_3d, dT_4d, dT_5d, dT_6d, dT_7d, dT_8d, dT_9d, dT_10d, dT_11d, dT_12d

cdef double dT_rest
cdef double dT_liquid
cdef double dT_vapour

cpdef list temperature_liquid
cpdef list temperature_vapour
cpdef list pressure_vapour

cdef m_vapour
cdef double m_liquid
cdef double m_boil_off
cdef double dm_boil_off

cdef dP_s1a
cdef dP_1a2a
cdef dP_1a1i
cdef dP_1a12a
cdef dP_1as
cdef dP_1a6a
cdef dP_1a_net

cdef dP_1i1a
cdef dP_1i1b
cdef dP_1i_net

cdef dP_1b2b
cdef dP_1b1c
cdef dP_1b12b
cdef dP_1b1i
cdef dP_1b6b
cdef dP_1b_net
cdef dP_1c2c
cdef dP_1c1d
cdef dP_1c12c
cdef dP_1c1b
cdef dP_1c6c
cdef dP_1c_net

cdef dP_1d2d
cdef dP_1dl
cdef dP_1dv
cdef dP_1d12d
cdef dP_1d1c
cdef dP_1d6d
cdef dP_1d_net

cdef dP_s2a
cdef dP_2a3a
cdef dP_2a2i
cdef dP_2a1a
cdef dP_2as
```

```
381 cdef dP_2a5a
382 cdef dP_2a_net
383
384 cdef dP_2i2a
385 cdef dP_2i2b
386 cdef dP_2i_net
387
388 cdef dP_2b3b
389 cdef dP_2b2c
390 cdef dP_2b1b
391 cdef dP_2b2i
392 cdef dP_2b5b
393 cdef dP_2b_net
394 cdef dP_2c3c
395 cdef dP_2c2d
396 cdef dP_2c1c
397 cdef dP_2c2b
398 cdef dP_2c5c
399 cdef dP_2c_net
400
401 cdef dP_2d3d
402 cdef dP_2dl
403 cdef dP_2dv
404 cdef dP_2d1d
405 cdef dP_2d2c
406 cdef dP_2d5d
407 cdef dP_2d_net
408
409 cdef dP_s3a
410 cdef dP_3a4a
411 cdef dP_3a3i
412 cdef dP_3a2a
413 cdef dP_3as
414 cdef dP_3a_net
415
416 cdef dP_3i3a
417 cdef dP_3i3b
418 cdef dP_3i_net
419
420 cdef dP_3b4b
421 cdef dP_3b3c
422 cdef dP_3b2b
423 cdef dP_3b3i
424 cdef dP_3b_net
425 cdef dP_3c4c
426 cdef dP_3c3d
427 cdef dP_3c2c
428 cdef dP_3c3b
429 cdef dP_3c_net
430
431 cdef dP_3d4d
432 cdef dP_3dv
433 cdef dP_3dl
434 cdef dP_3d2d
435 cdef dP_3d3c
436 cdef dP_3d_net
437
438 cdef dP_s4a
439 cdef dP_4a5a
440 cdef dP_4a4i
441 cdef dP_4a3a
442 cdef dP_4as
443 cdef dP_4a_net
444
445 cdef dP_4i4a
446 cdef dP_4i4b
447 cdef dP_4i_net
448
449 cdef dP_4b5b
450 cdef dP_4b4c
451 cdef dP_4b3b
```

```
452 cdef dP_4b4i
453 cdef dP_4b_net
454 cdef dP_4c5c
455 cdef dP_4c4d
456 cdef dP_4c3c
457 cdef dP_4c4b
458 cdef dP_4c_net
459
460 cdef dP_4d5d
461 cdef dP_4dv
462 cdef dP_4dl
463 cdef dP_4d3d
464 cdef dP_4d4c
465 cdef dP_4d_net
466
467 cdef dP_s5a
468 cdef dP_5a6a
469 cdef dP_5a5i
470 cdef dP_5a4a
471 cdef dP_5as
472 cdef dP_5a2a
473 cdef dP_5a_net
474
475 cdef dP_5i5a
476 cdef dP_5i5b
477 cdef dP_5i_net
478
479 cdef dP_5b6b
480 cdef dP_5b5c
481 cdef dP_5b4b
482 cdef dP_5b5i
483 cdef dP_5b2b
484 cdef dP_5b_net
485 cdef dP_5c6c
486 cdef dP_5c5d
487 cdef dP_5c4c
488 cdef dP_5c5b
489 cdef dP_5c2c
490 cdef dP_5c_net
491
492 cdef dP_5d6d
493 cdef dP_5dl
494 cdef dP_5dv
495 cdef dP_5d4d
496 cdef dP_5d5c
497 cdef dP_5d2d
498 cdef dP_5d_net
499
500 cdef dP_s6a
501 cdef dP_6a7a
502 cdef dP_6a6i
503 cdef dP_6a5a
504 cdef dP_6as
505 cdef dP_6a1a
506 cdef dP_6a_net
507
508 cdef dP_6i6a
509 cdef dP_6i6b
510 cdef dP_6i_net
511
512 cdef dP_6b7b
513 cdef dP_6b6c
514 cdef dP_6b5b
515 cdef dP_6b6i
516 cdef dP_6b1b
517 cdef dP_6b_net
518 cdef dP_6c7c
519 cdef dP_6c6d
520 cdef dP_6c5c
521 cdef dP_6c6b
522 cdef dP_6c1c
```

```
523 cdef dP_6c_net
524
525 cdef dP_6d7d
526 cdef dP_6dl
527 cdef dP_6dv
528 cdef dP_6d5d
529 cdef dP_6d6c
530 cdef dP_6d1d
531 cdef dP_6d_net
532
533 cdef dP_s7a
534 cdef dP_7a8a
535 cdef dP_7a7i
536 cdef dP_7a6a
537 cdef dP_7as
538 cdef dP_7a12a
539 cdef dP_7a_net
540
541 cdef dP_7i7a
542 cdef dP_7i7b
543 cdef dP_7i_net
544
545 cdef dP_7b8b
546 cdef dP_7b7c
547 cdef dP_7b6b
548 cdef dP_7b7i
549 cdef dP_7b12b
550 cdef dP_7b_net
551 cdef dP_7c8c
552 cdef dP_7c7d
553 cdef dP_7c6c
554 cdef dP_7c7b
555 cdef dP_7c12c
556 cdef dP_7c_net
557
558 cdef dP_7d8d
559 cdef dP_7dl
560 cdef dP_7dv
561 cdef dP_7d6d
562 cdef dP_7d7c
563 cdef dP_7d12d
564 cdef dP_7d_net
565
566 cdef dP_s8a
567 cdef dP_8a9a
568 cdef dP_8a8i
569 cdef dP_8a7a
570 cdef dP_8as
571 cdef dP_8a11a
572 cdef dP_8a_net
573
574 cdef dP_8i8a
575 cdef dP_8i8b
576 cdef dP_8i_net
577
578 cdef dP_8b9b
579 cdef dP_8b8c
580 cdef dP_8b7b
581 cdef dP_8b8i
582 cdef dP_8b11b
583 cdef dP_8b_net
584 cdef dP_8c9c
585 cdef dP_8c8d
586 cdef dP_8c7c
587 cdef dP_8c8b
588 cdef dP_8c11c
589 cdef dP_8c_net
590
591 cdef dP_8d9d
592 cdef dP_8dl
593 cdef dP_8dv
```

```
cdef dP_8d7d
cdef dP_8d8c
cdef dP_8d11d
cdef dP_8d_net

cdef dP_s9a
cdef dP_9a10a
cdef dP_9a9i
cdef dP_9a8a
cdef dP_9as
cdef dP_9a_net

cdef dP_9i9a
cdef dP_9i9b
cdef dP_9i_net

cdef dP_9b10b
cdef dP_9b9c
cdef dP_9b8b
cdef dP_9b9i
cdef dP_9b_net
cdef dP_9c10c
cdef dP_9c9d
cdef dP_9c8c
cdef dP_9c9b
cdef dP_9c_net

cdef dP_9d10d
cdef dP_9dl
cdef dP_9dv
cdef dP_9d8d
cdef dP_9d9c
cdef dP_9d_net

cdef dP_s10a
cdef dP_10a11a
cdef dP_10a10i
cdef dP_10a9a
cdef dP_10as
cdef dP_10a_net

cdef dP_10i10a
cdef dP_10i10b
cdef dP_10i_net

cdef dP_10b11b
cdef dP_10b10c
cdef dP_10b9b
cdef dP_10b10i
cdef dP_10b_net
cdef dP_10c11c
cdef dP_10c10d
cdef dP_10c9c
cdef dP_10c10b
cdef dP_10c_net

cdef dP_10d11d
cdef dP_10dl
cdef dP_10dv
cdef dP_10d9d
cdef dP_10d10c
cdef dP_10d_net

cdef dP_s11a
cdef dP_11a12a
cdef dP_11a11i
cdef dP_11a10a
cdef dP_11as
cdef dP_11a8a
cdef dP_11a_net

```

```
665 cdef dP_11i11a
666 cdef dP_11i11b
667 cdef dP_11i_net
668
669 cdef dP_11b12b
670 cdef dP_11b11c
671 cdef dP_11b10b
672 cdef dP_11b11i
673 cdef dP_11b8b
674 cdef dP_11b_net
675 cdef dP_11c12c
676 cdef dP_11c11d
677 cdef dP_11c10c
678 cdef dP_11c11b
679 cdef dP_11c8c
680 cdef dP_11c_net
681
682 cdef dP_11d12d
683 cdef dP_11dl
684 cdef dP_11dv
685 cdef dP_11d10d
686 cdef dP_11d11c
687 cdef dP_11d8d
688 cdef dP_11d_net
689
690 cdef dP_s12a
691 cdef dP_12a1a
692 cdef dP_12a12i
693 cdef dP_12a11a
694 cdef dP_12as
695 cdef dP_12a7a
696 cdef dP_12a_net
697
698 cdef dP_12i12a
699 cdef dP_12i12b
700 cdef dP_12i_net
701
702 cdef dP_12b1b
703 cdef dP_12b12c
704 cdef dP_12b11b
705 cdef dP_12b12i
706 cdef dP_12b7b
707 cdef dP_12b_net
708 cdef dP_12c1c
709 cdef dP_12c12d
710 cdef dP_12c11c
711 cdef dP_12c12b
712 cdef dP_12c7c
713 cdef dP_12c_net
714
715 cdef dP_12d1d
716 cdef dP_12dl
717 cdef dP_12dv
718 cdef dP_12d11d
719 cdef dP_12d12c
720 cdef dP_12d7d
721 cdef dP_12d_net
722
723 cdef Ra
724 cdef h_H2
725 cdef L_c_H2
726 cdef L_c_vap
727 cdef h_vapour
728 cdef Ra_vap
729 cdef Nu_H2
730 cdef Nu_vap
731
732 cdef dP_l1d, dP_l2d, dP_l3d, dP_l4d, dP_l5d, dP_l6d ,dP_l7d ,dP_l8d ,dP_l9d ,dP_l10d ,dP_l11d
        ,dP_l12d ,dP_lv ,dP_cooler
733 cdef double dP_liquid_net
734 cdef P_rest
```

```
735
736 cdef dP_v1d, dP_v2d, dP_v3d, dP_v4d, dP_v5d, dP_v6d, dP_v7d, dP_v8d, dP_v9d, dP_v10d ,dP_v11d
      ,dP_v12d, dP_v1
737 cdef double dP_vapour_net
738
739 cdef dP_1net, dP_2net, dP_3net, dP_4net, dP_5net, dP_6net, dP_7net, dP_8net, dP_9net,
    dP_10net, dP_11net, dP_12net
740
741 cdef double dP_system
742 cdef double dP_in
743 cdef double dP_out
744 cdef double dP_tot
745
746 cdef double Boil_off_monthly
747
748 cdef int t_r
749 cdef p_max = pf.p_max
750 cdef double m_vent
751 cdef double dm_vent
752 cdef double dp_vent
753 cdef double dT_vapour_vent
754 cdef double T_sl_avg
755 cdef double T_sv_avg
756 cdef double T_use
757 cdef T_reject = pf.T_rejection
758
759 cdef Q_mixer
760 cdef dP_penetration
761 cdef dP_mixer
762 cdef eta
763 cdef double T_previous
764 cdef double T_previous_vap
765 cdef m_mix
766 cdef option
767 cdef radius_max = pf.maximum_tank_radius
768 cdef q_IR
769 cdef q_albedo
770 cdef frac
771
772 cdef bint six_months = False
773 cdef bint nine_months = False
774 cdef bint twelve_months = False
775 cdef bint eighteen_months = False
776 cdef bint twentyfour_months = False
777
778 print "initiate program..."
779 T_space = pf.T_space
780 q_sun = pf.q_sun
781 q_IR = pf.q_IR
782 q_albedo = pf.q_albedo
783
784 cdef list options = list(it.product(layer_densities, MLI_layers, t_SOFI_list, cooler_power))
785 options = list(it.chain(options, pf.custom_designs))
786
787 print options
788
789 for option in tqdm(options):
790     t_boil = 0.
791
792     six_months = False
793     nine_months = False
794     twelve_months = False
795     eighteen_months = False
796     twentyfour_months = False
797
798     layer_density, layers, t_SOFI, Q_cooler = option
799
800     ### Hydrogen properties ###
801     density_liquid = H2_dict[round(T_start,1)]['Density (kg/m3)']
802     Cp_H2 = H2_dict[round(T_start,1)]['Cp (J/g*K)']*1e3 #J/kg*K
803     k_H2 = H2_dict[round(T_start,1)]['Therm. Cond. (W/m*K)']
```

```
804     mu_H2 = H2_dict[round(T_start,1)]['Viscosity (Pa*s)']
805     nu_H2 = mu_H2/density_liquid
806     alpha_H2 = k_H2/density_liquid/Cp_H2
807     beta_H2 = H2_dict[round(T_start,1)]['Therm. Exp. Coeff. (1/m3)']
808     m_liquid_initial = density_liquid*V_liquid
809
810     ### Helium properties ###
811     density_vapour = He_dict[round(T_start,1)]['Density (kg/m3)']
812     Cp_He_vapour = He_dict[round(T_start,1)]['Cp (J/g*K)']*1e3 #J/kg*K
813     k_He_vapour = He_dict[round(T_start,1)]['Therm. Cond. (W/m*K)']
814     mu_He = He_dict[round(T_start,1)]['Viscosity (Pa*s)']
815     nu_He = mu_He/density_vapour
816     alpha_He = k_He_vapour/density_vapour/Cp_He_vapour
817     beta_He = He_dict[round(T_start,1)]['Therm. Exp. Coeff. (1/m3)']
818
819     ### Structure ###
820     radius_calc, height_calc = volume_calc(mass = pf.propellant_mass, fill_ratio =
            fill_level_initial, max_radius = radius_max, propellant_type = 'LH2',
            propellant_density = density_liquid)
821     print 'The propellant tank radius is %s [m] and the height is %s [m]' %(radius_calc,
            height_calc)
822
823     prop = structure(height = height_calc, radius= radius_calc, thickness=t_wall,
            sphere_parts=4, cylinder_parts=8, wall_density=density_wall)
824
825     V_total = prop.internalvolume+prop.sphere_internalvolume
826     V_liquid = fill_level_initial*V_total
827     V_vapour = V_total-V_liquid
828     m_liquid_initial = density_liquid*V_liquid
829     m_vapour_initial = V_vapour*density_vapour
830
831     ### Masses (SOFI, MLI, Shell, Cooler) ###
832     m_SOFI = ((pi * ((prop.radius + prop.thickness + t_SOFI) ** 2. - (
833                 prop.radius + prop.thickness) ** 2.) * prop.height) + 4 / 3. * pi * (
834                           (prop.radius + prop.thickness + t_SOFI) ** 3. - (
835                               prop.radius + prop.thickness) ** 3.)) * density_SOFI
836     m_shell = prop.wall_mass
837
838     if Q_cooler == 0.:
839         eta = 0.
840         m_cooler = 0.
841     else:
842         eta = efficiency(Q_cooler)
843         m_cooler = Cryocooler_mass(Q_cooler, 20., T_reject, eta) #T_H = 263, which should be
                the spacecraft temperature according to Chai 2014.
844
845     m_MLI = density_MLI*layers*(2.*pi*(prop.radius+prop.thickness+t_SOFI)*prop.height+4.*pi*(
            prop.radius+prop.thickness+t_SOFI)**2.)
846
847     t_MLI = layers*1./(layer_density*100.)
848     Q_mixer = V_total/16.8*1.
849
850     ###### TIME ######
851     dt = 1.
852     t = 0.
853
854     fill_level = fill_level_initial
855     p_vapour = p_vapour_initial  # Pa
856
857     ###### INITIAL TEMPERATURES ######
858     T_1a, T_2a, T_3a, T_4a, T_5a, T_6a, T_7a, T_8a, T_9a, T_10a, T_11a, T_12a = T_start,
            T_start, T_start, T_start, T_start, T_start, T_start, T_start, T_start, T_start,
            T_start, T_start
859     T_1i, T_2i, T_3i, T_4i, T_5i, T_6i, T_7i, T_8i, T_9i, T_10i, T_11i, T_12i = T_start,
            T_start, T_start, T_start, T_start, T_start, T_start, T_start, T_start, T_start,
            T_start, T_start
860     T_1b, T_2b, T_3b, T_4b, T_5b, T_6b, T_7b, T_8b, T_9b, T_10b, T_11b, T_12b = T_start,
            T_start, T_start, T_start, T_start, T_start, T_start, T_start, T_start, T_start,
            T_start, T_start
861     T_1c, T_2c, T_3c, T_4c, T_5c, T_6c, T_7c, T_8c, T_9c, T_10c, T_11c, T_12c = T_start,
            T_start, T_start, T_start, T_start, T_start, T_start, T_start, T_start, T_start,
```

```
        T_start, T_start
    T_1d, T_2d, T_3d, T_4d, T_5d, T_6d, T_7d, T_8d, T_9d, T_10d, T_11d, T_12d = T_start,
        T_start, T_start, T_start, T_start, T_start, T_start, T_start, T_start, T_start,
        T_start, T_start

    T_liquid = T_start
    T_previous = T_start
    T_vapour = T_start
    T_previous_vap = T_start

    if abs(m_liquid_initial/pf.propellant_mass-1.) > 0.001:
        print 'initial propellant mass differs %s percent from target' %((m_liquid_initial/pf
            .propellant_mass-1.)*100.)

    # Masses initial
    m_vapour = m_vapour_initial
    m_liquid = m_liquid_initial
    m_vent = 0.
    dm_vent = 0.
    m_boil_off = 0.
    dm_boil_off = 0.
    if layers != 0.:
        while t <= runtime:
            if t > 60.*60.*24.*14.:
                dt = timestep

            t = t+dt

            if (T_liquid-T_previous) >= 0.5:
                print 'change liquid properties'
                T_use = min(246, int(round(T_liquid*10.,0)))
                density_liquid = H2_dict[T_use/10.]['Density (kg/m3)']
                Cp_H2 = H2_dict[T_use/10.]['Cp (J/g*K)']*1e3 #J/kg*K
                k_H2 = H2_dict[T_use/10.]['Therm. Cond. (W/m*K)']
                mu_H2 = H2_dict[T_use/10.]['Viscosity (Pa*s)']
                nu_H2 = mu_H2/density_liquid
                alpha_H2 = k_H2/density_liquid/Cp_H2
                beta_H2 = H2_dict[T_use/10.]['Therm. Exp. Coeff. (1/m3)']
                T_previous = T_liquid

            if (T_vapour-T_previous_vap) >= 1.0:
                print 'change vapour properties'
                T_use = int(round(T_vapour*10.,0))
                density_vapour = He_dict[T_use/10.]['Density (kg/m3)']
                Cp_He_vapour = He_dict[T_use/10.]['Cp (J/g*K)']*1e3 #J/kg*K
                k_He_vapour = He_dict[T_use/10.]['Therm. Cond. (W/m*K)']
                mu_He = He_dict[T_use/10.]['Viscosity (Pa*s)']
                nu_He = mu_He/density_vapour
                alpha_He = k_He_vapour/density_vapour/Cp_He_vapour
                beta_He = He_dict[T_use/10.]['Therm. Exp. Coeff. (1/m3)']
                T_previous_vap = T_vapour

            if time_dict == 0.:
                pass
            else:
                t_r = round(t/60./60.)
                T_space = time_dict[t_r]['T_env [K]']
                q_sun =  time_dict[t_r]['q_sun [W/m2]']
                grav = time_dict[t_r]['gravity [m/s2]']
                solar_incidence_angle = time_dict[t_r]['solar_incidence_angle [deg]']

            L_c_H2 = fill_level*(2.*prop.radius+prop.height) #V_liquid/(fill_level*(4.*pi*
                prop.radius**2.+2.*pi*prop.radius*prop.height))
            L_c_vap = (1-fill_level)*(2.*prop.radius+prop.height) #V_vapour/((1-fill_level)
                *(4.*pi*prop.radius**2.+2.*pi*prop.radius*prop.height))

            if fill_level > (0.5*prop.sphere_internalvolume+prop.internalvolume)/V_total:
                T_sl_avg = (T_1d+T_2d+T_3d+T_4d+T_5d+T_6d+T_7d+T_8d+T_9d+T_10d+T_11d+T_12d)
                    /12.
                T_sv_avg = (T_3d+T_4d)/2.
```

```
925             elif fill_level > (0.5*prop.sphere_internalvolume+0.75*prop.internalvolume)/
                    V_total:
926                 T_sl_avg = (T_1d+T_2d+T_5d+T_6d+T_7d+T_8d+T_9d+T_10d+T_11d+T_12d)/10.
927                 T_sv_avg = (T_3d+T_4d+T_2d+T_5d)/4.
928             elif fill_level > (0.5*prop.sphere_internalvolume+0.5*prop.internalvolume)/
                    V_total:
929                 T_sl_avg = (T_1d+T_6d+T_7d+T_8d+T_9d+T_10d+T_11d+T_12d)/8.
930                 T_sv_avg = (T_3d+T_4d+T_1d+T_2d+T_5d+T_6d)/6.
931             elif fill_level > (0.5*prop.sphere_internalvolume+0.25*prop.internalvolume)/
                    V_total:
932                 T_sl_avg = (T_7d+T_8d+T_9d+T_10d+T_11d+T_12d)/6.
933                 T_sv_avg = (T_3d+T_4d+T_12d+T_1d+T_2d+T_5d+T_6d+T_7d)/8.
934             elif fill_level > (0.5*prop.sphere_internalvolume)/V_total:
935                 T_sl_avg = (T_8d+T_9d+T_10d+T_11d)/4.
936                 T_sv_avg = (T_3d+T_4d+T_11d+T_12d+T_1d+T_2d+T_5d+T_6d+T_7d+T_8d)/8.
937             else:
938                 T_sl_avg = (T_9d+T_10d)/2.
939                 T_sv_avg = (T_1d+T_2d+T_3d+T_4d+T_5d+T_6d+T_7d+T_8d+T_9d+T_10d+T_11d+T_12d)
                        /12.
940
941             Ra = Rayleigh(L_c_H2, grav, beta_H2, alpha_H2, nu_H2, T_sl_avg, T_liquid)
942             h_H2 = h(k_H2, L_c_H2, Ra)
943
944             Ra_vap = Rayleigh(L_c_vap, grav, beta_He, alpha_He, nu_He, T_sv_avg, T_vapour)
945             h_vapour = h_vap(k_He_vapour, L_c_vap, Ra_vap)
946
947
948             ######## NODE 1 ########
949
950             # NODE 1A
951             dP_s1a = q_sun * max(round(cos(radians(solar_incidence_angle-180.)),2),0.)* prop.
                    increment_frontalarea * absorptivity \
952                      + q_IR * max(round(cos(radians(solar_incidence_angle)),2),0.)* prop.
                         increment_frontalarea * emissivity
953             dP_1a2a = conduction(k_MLI, T_1a-T_2a, prop.increment_length)*pi*(prop.radius+
                    prop.thickness+t_SOFI)*t_MLI
954             dP_1a1i = Lockheed(T_1a, T_1i, emissivity_MLI, layer_density, layers)[0]*prop.
                    increment_area
955             dP_1a12a = conduction(k_MLI, T_1a-T_12a, prop.increment_length)*pi*(prop.radius+
                    prop.thickness+t_SOFI)*t_MLI
956             dP_1as = radiation(emissivity,T_1a,T_space)*prop.increment_area
957
958             dP_1a6a = conduction(k_MLI, T_1a-T_6a, pi*(prop.radius+prop.thickness+t_SOFI+
                    t_MLI))*t_MLI*prop.increment_length
959
960             dP_1a_net = dP_s1a + dP_1a2a + dP_1a1i + dP_1a12a + dP_1as + dP_1a6a
961
962             # NODE 1i
963             dP_1i1a = -dP_1a1i
964             dP_1i1b = conduction(k_SOFI, T_1i-T_1b, t_SOFI/2.)*prop.increment_area
965
966             dP_1i_net = dP_1i1a + dP_1i1b
967
968             # NODE 1B
969             dP_1b2b = conduction(k_SOFI, T_1b-T_2b, prop.increment_length)*pi*(prop.radius+
                    prop.thickness)*t_SOFI
970             dP_1b1c = conduction(k_SOFI, T_1b-T_1c, t_SOFI/2.)*prop.increment_area
971             dP_1b12b = conduction(k_SOFI, T_1b-T_12b, prop.increment_length)*pi*(prop.radius+
                    prop.thickness)*t_SOFI
972             dP_1b1i = -dP_1i1b
973
974             dP_1b6b = conduction(k_SOFI, T_1b-T_6b, pi*(prop.radius+prop.thickness+t_SOFI))*
                    t_SOFI*prop.increment_length
975
976             dP_1b_net = dP_1b2b + dP_1b1c + dP_1b12b + dP_1b1i + dP_1b6b
977
978
979             # NODE 1C
980             dP_1c2c = conduction(k_wall, T_1c-T_2c, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
981             dP_1c1d = conduction(k_wall, T_1c-T_1d, prop.thickness)*prop.increment_area
```

```
982             dP_1c12c = conduction(k_wall, T_1c-T_12c, prop.increment_length)*pi*(prop.radius)
                    *prop.thickness/2.
983             dP_1c1b = -dP_1b1c
984
985             dP_1c6c = conduction(k_wall, T_1c-T_6c, pi*(prop.radius))*prop.thickness*prop.
                    increment_length
986
987             dP_1c_net = dP_1c2c + dP_1c1d + dP_1c12c + dP_1c1b + dP_1c6c
988
989             # NODE 1D
990             dP_1d2d = conduction(k_wall, T_1d-T_2d, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
991             dP_1d12d = conduction(k_wall, T_1d-T_12d, prop.increment_length)*pi*(prop.radius)
                    *prop.thickness/2.
992             dP_1d1c = -dP_1c1d
993
994             dP_1d6d = conduction(k_wall, T_1d-T_6d, pi*(prop.radius))*prop.thickness*prop.
                    increment_length
995
996             if fill_level > (0.5*prop.sphere_internalvolume+0.5*prop.internalvolume)/V_total:
997                 dP_1dl = h_H2*-(T_1d-T_liquid)*prop.increment_area
998                 dP_1dv = 0.
999             else:
1000                dP_1dv = h_vapour*-(T_1d-T_vapour)*prop.increment_area
1001                dP_1dl = 0.
1002
1003
1004            dP_1d_net = dP_1d2d + dP_1dl + dP_1dv + dP_1d12d + dP_1d1c + dP_1d6d
1005
1006            dP_1net = dP_1a_net+dP_1i_net+dP_1b_net+dP_1c_net+dP_1d_net
1007
1008            ########################
1009
1010            ######## NODE 2 ########
1011
1012            # NODE 2A
1013            dP_s2a = q_sun * max(round(cos(radians(solar_incidence_angle-180.)),2),0.) * prop
                    .increment_frontalarea * absorptivity \
1014                        + q_IR * max(round(cos(radians(solar_incidence_angle)),2),0.)* prop.
                            increment_frontalarea * emissivity
1015            dP_2a3a = conduction(k_MLI, T_2a-T_3a, prop.sphere_increment_length)*pi*(prop.
                    radius+prop.thickness+t_SOFI)*t_MLI
1016            dP_2a2i = Lockheed(T_2a, T_2i, emissivity_MLI, layer_density, layers)[0]*prop.
                    increment_area
1017            dP_2a1a = -dP_1a2a
1018            dP_2as = radiation(emissivity,T_2a,T_space)*prop.increment_area
1019
1020            dP_2a5a = conduction(k_SOFI, T_2a-T_5a, pi*(prop.radius+prop.thickness+t_SOFI+
                    t_MLI))*t_MLI*prop.increment_length
1021
1022            dP_2a_net = dP_s2a + dP_2a3a + dP_2a2i + dP_2a1a + dP_2as + dP_2a5a
1023
1024            # NODE 2i
1025            dP_2i2a = -dP_2a2i
1026            dP_2i2b = conduction(k_SOFI, T_2i-T_2b, t_SOFI/2.)*prop.increment_area
1027
1028            dP_2i_net = dP_2i2a + dP_2i2b
1029
1030            # NODE 2B
1031            dP_2b3b = conduction(k_SOFI, T_2b-T_3b, prop.sphere_increment_length)*pi*(prop.
                    radius+prop.thickness)*t_SOFI
1032            dP_2b2c = conduction(k_SOFI, T_2b-T_2c, t_SOFI/2.)*prop.increment_area
1033            dP_2b1b = -dP_1b2b
1034            dP_2b2i = -dP_2i2b
1035
1036            dP_2b5b = conduction(k_SOFI, T_2b-T_5b, pi*(prop.radius+prop.thickness+t_SOFI))*
                    t_SOFI*prop.increment_length
1037
1038            dP_2b_net = dP_2b3b + dP_2b2c + dP_2b1b + dP_2b2i + dP_2b5b
1039
1040            # NODE 2C
```

```
1041            dP_2c3c = conduction(k_wall, T_2c-T_3c, prop.sphere_increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
1042            dP_2c2d = conduction(k_wall, T_2c-T_2d, prop.thickness)*prop.increment_area
1043            dP_2c1c = -dP_1c2c
1044            dP_2c2b = -dP_2b2c
1045
1046            dP_2c5c = conduction(k_wall, T_2c-T_5c, pi*(prop.radius))*prop.thickness*prop.
                    increment_length
1047
1048            dP_2c_net = dP_2c3c + dP_2c2d + dP_2c1c + dP_2c2b + dP_2c5c
1049
1050            # NODE 2D
1051            dP_2d3d = conduction(k_wall, T_2d-T_3d, prop.sphere_increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
1052            dP_2d1d = -dP_1d2d
1053            dP_2d2c = -dP_2c2d
1054
1055            dP_2d5d = conduction(k_wall, T_2d-T_5d, pi*(prop.radius))*prop.thickness*prop.
                    increment_length
1056
1057            if fill_level > (0.5*prop.sphere_internalvolume+0.75*prop.internalvolume)/V_total
                    :
1058                dP_2dl = h_H2*-(T_2d-T_liquid)*prop.increment_area
1059                dP_2dv = 0.
1060            else:
1061                dP_2dv = h_vapour*-(T_2d-T_vapour)*prop.increment_area
1062                dP_2dl = 0.
1063
1064            dP_2d_net = dP_2d3d + dP_2dl + dP_2dv + dP_2d1d + dP_2d2c + dP_2d5d
1065
1066            dP_2net = dP_2a_net+dP_2i_net+dP_2b_net+dP_2c_net+dP_2d_net
1067
1068            ########################
1069
1070
1071            ######## NODE 3 ########
1072
1073            # NODE 3A
1074            dP_s3a = q_sun * max(round(cos(radians(solar_incidence_angle-135.)),2),0.) * prop
                    .sphere_increment_frontalarea * absorptivity \
1075                     + q_IR * max(round(cos(radians(solar_incidence_angle-315.)),2),0.)* prop
                            .sphere_increment_frontalarea * emissivity
1076            dP_3a4a = conduction(k_MLI, T_3a-T_4a, prop.sphere_increment_length)*pi*(prop.
                    radius+prop.thickness+t_SOFI)*t_MLI
1077            dP_3a3i = Lockheed(T_3a, T_3i, emissivity_MLI, layer_density, layers)[0]*prop.
                    sphere_increment_area
1078            dP_3a2a = conduction(k_MLI, T_3a-T_2a, prop.increment_length)*pi*(prop.radius+
                    prop.thickness+t_SOFI)*t_MLI
1079            dP_3as = radiation(emissivity,T_3a,T_space)*prop.sphere_increment_area
1080
1081            dP_3a_net = dP_s3a + dP_3a4a + dP_3a3i + dP_3a2a + dP_3as
1082
1083            # NODE 3i
1084            dP_3i3a = -dP_3a3i
1085            dP_3i3b = conduction(k_SOFI, T_3i-T_3b, t_SOFI/2.)*prop.sphere_increment_area
1086
1087            dP_3i_net = dP_3i3a + dP_3i3b
1088
1089            # NODE 3B
1090            dP_3b4b = conduction(k_SOFI, T_3b-T_4b, prop.sphere_increment_length)*pi*(prop.
                    radius+prop.thickness)*t_SOFI
1091            dP_3b3c = conduction(k_SOFI, T_3b-T_3c, t_SOFI/2.)*prop.sphere_increment_area
1092            dP_3b2b = conduction(k_SOFI, T_3b-T_2b, prop.increment_length)*pi*(prop.radius+
                    prop.thickness)*t_SOFI
1093            dP_3b3i = -dP_3i3b
1094
1095            dP_3b_net = dP_3b4b + dP_3b3c + dP_3b2b + dP_3b3i
1096
1097            # NODE 3C
1098            dP_3c4c = conduction(k_wall, T_3c-T_4c, prop.sphere_increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
```

```
1099            dP_3c3d = conduction(k_wall, T_3c-T_3d, prop.thickness)*prop.
                    sphere_increment_area
1100            dP_3c2c = conduction(k_wall, T_3c-T_2c, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
1101            dP_3c3b = -dP_3b3c
1102
1103            dP_3c_net = dP_3c4c + dP_3c3d + dP_3c2c + dP_3c3b
1104
1105            # NODE 3D
1106            dP_3d4d = conduction(k_wall, T_3d-T_4d, prop.sphere_increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
1107            dP_3d2d = conduction(k_wall, T_3d-T_2d, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
1108            dP_3d3c = -dP_3c3d
1109
1110            if fill_level > (0.5*prop.sphere_internalvolume+prop.internalvolume)/V_total:
1111                frac = max((fill_level-(0.5*prop.sphere_internalvolume+prop.internalvolume)/
                        V_total)/(0.5*prop.sphere_internalvolume/V_total),1)
1112                dP_3dl = frac*(h_H2*-(T_3d-T_liquid)*prop.sphere_increment_area)
1113                dP_3dv = (1-frac)*(h_vapour*-(T_3d-T_vapour)*prop.sphere_increment_area)
1114            else:
1115                dP_3dv = h_vapour*-(T_3d-T_vapour)*prop.sphere_increment_area
1116                dP_3dl = 0.
1117
1118            dP_3d_net = dP_3d4d + dP_3dv + dP_3dl + dP_3d2d + dP_3d3c
1119
1120            dP_3net = dP_3a_net+dP_3i_net+dP_3b_net+dP_3c_net+dP_3d_net
1121
1122            ########################
1123
1124            ######## NODE 4 ########
1125
1126            # NODE 4A
1127            dP_s4a = q_sun * max(round(cos(radians(solar_incidence_angle-45.)),2),0.) * prop.
                    sphere_increment_frontalarea * absorptivity \
1128                     + q_IR * max(round(cos(radians(solar_incidence_angle-225.)),2),0.)* prop
                            .sphere_increment_frontalarea * emissivity
1129            dP_4a5a = conduction(k_MLI, T_4a-T_5a, prop.increment_length)*pi*(prop.radius+
                    prop.thickness+t_SOFI)*t_MLI
1130            dP_4a4i = Lockheed(T_4a, T_4i, emissivity_MLI, layer_density, layers)[0]*prop.
                    sphere_increment_area
1131            dP_4a3a = -dP_3a4a
1132            dP_4as = radiation(emissivity,T_4a,T_space)*prop.sphere_increment_area
1133
1134            dP_4a_net = dP_s4a + dP_4a5a + dP_4a4i + dP_4a3a + dP_4as
1135
1136            # NODE 4i
1137            dP_4i4a = -dP_4a4i
1138            dP_4i4b = conduction(k_SOFI, T_4i-T_4b, t_SOFI/2.)*prop.sphere_increment_area
1139
1140            dP_4i_net = dP_4i4a + dP_4i4b
1141
1142            # NODE 4B
1143            dP_4b5b = conduction(k_SOFI, T_4b-T_5b, prop.increment_length)*pi*(prop.radius+
                    prop.thickness)*t_SOFI
1144            dP_4b4c = conduction(k_SOFI, T_4b-T_4c, t_SOFI/2.)*prop.sphere_increment_area
1145            dP_4b3b = -dP_3b4b
1146            dP_4b4i = -dP_4i4b
1147
1148            dP_4b_net = dP_4b5b + dP_4b4c + dP_4b3b + dP_4b4i
1149
1150            # NODE 4C
1151            dP_4c5c = conduction(k_wall, T_4c-T_5c, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
1152            dP_4c4d = conduction(k_wall, T_4c-T_4d, prop.thickness)*prop.
                    sphere_increment_area
1153            dP_4c3c = -dP_3c4c
1154            dP_4c4b = -dP_4b4c
1155
1156            dP_4c_net = dP_4c5c + dP_4c4d + dP_4c3c + dP_4c4b
1157
```

```
1158            # NODE 4D
1159            dP_4d5d = conduction(k_wall, T_4d-T_5d, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
1160            dP_4d3d = -dP_3d4d
1161            dP_4d4c = -dP_4c4d
1162
1163            if fill_level > (0.5*prop.sphere_internalvolume+prop.internalvolume)/V_total:
1164                frac = max((fill_level-(0.5*prop.sphere_internalvolume+prop.internalvolume)/
                        V_total)/(0.5*prop.sphere_internalvolume/V_total),1)
1165                dP_4dl = frac*(h_H2*-(T_4d-T_liquid)*prop.sphere_increment_area)
1166                dP_4dv = (1-frac)*(h_vapour*-(T_4d-T_vapour)*prop.sphere_increment_area)
1167            else:
1168                dP_4dv = h_vapour*-(T_4d-T_vapour)*prop.sphere_increment_area
1169                dP_4dl = 0.
1170
1171            dP_4d_net = dP_4d5d + dP_4dv + dP_4dl + dP_4d3d + dP_4d4c
1172
1173            dP_4net = dP_4a_net+dP_4i_net+dP_4b_net+dP_4c_net+dP_4d_net
1174
1175            ########################
1176
1177            ######## NODE 5 ########
1178
1179            # NODE 5A
1180            dP_s5a = q_sun * max(round(cos(radians(solar_incidence_angle)),2),0.) * prop.
                    increment_frontalarea * absorptivity \
1181                         + q_IR * max(round(cos(radians(solar_incidence_angle-180.)),2),0.)* prop
                                .increment_frontalarea * emissivity
1182            dP_5a6a = conduction(k_MLI, T_5a-T_6a, prop.increment_length)*pi*(prop.radius+
                    prop.thickness+t_SOFI)*t_MLI
1183            dP_5a5i = Lockheed(T_5a, T_5i, emissivity_MLI, layer_density, layers)[0]*prop.
                    increment_area
1184            dP_5a4a = conduction(k_MLI, T_5a-T_4a, prop.sphere_increment_length)*pi*(prop.
                    radius+prop.thickness+t_SOFI)*t_MLI
1185            dP_5as = radiation(emissivity,T_5a,T_space)*prop.increment_area
1186
1187            dP_5a2a = -dP_2a5a
1188
1189            dP_5a_net = dP_s5a + dP_5a6a + dP_5a5i + dP_5a4a + dP_5as + dP_5a2a
1190
1191            # NODE 5i
1192            dP_5i5a = -dP_5a5i
1193            dP_5i5b = conduction(k_SOFI, T_5i-T_5b, t_SOFI/2.)*prop.increment_area
1194
1195            dP_5i_net = dP_5i5a + dP_5i5b
1196
1197            # NODE 5B
1198            dP_5b6b = conduction(k_SOFI, T_5b-T_6b, prop.increment_length)*pi*(prop.radius+
                    prop.thickness)*t_SOFI
1199            dP_5b5c = conduction(k_SOFI, T_5b-T_5c, t_SOFI/2.)*prop.increment_area
1200            dP_5b4b = conduction(k_SOFI, T_5b-T_4b, prop.sphere_increment_length)*pi*(prop.
                    radius+prop.thickness)*t_SOFI
1201            dP_5b5i = -dP_5i5b
1202
1203            dP_5b2b = -dP_2b5b
1204
1205            dP_5b_net = dP_5b6b + dP_5b5c + dP_5b4b + dP_5b5i + dP_5b2b
1206
1207            # NODE 5C
1208            dP_5c6c = conduction(k_wall, T_5c-T_6c, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
1209            dP_5c5d = conduction(k_wall, T_5c-T_5d, prop.thickness)*prop.increment_area
1210            dP_5c4c = conduction(k_wall, T_5c-T_4c, prop.sphere_increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
1211            dP_5c5b = -dP_5b5c
1212
1213            dP_5c2c = -dP_2c5c
1214
1215            dP_5c_net = dP_5c6c + dP_5c5d + dP_5c4c + dP_5c5b + dP_5c2c
1216
1217            # NODE 5D
```

```
1218            dP_5d6d = conduction(k_wall, T_5d-T_6d, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
1219            dP_5d4d = conduction(k_wall, T_5d-T_4d, prop.sphere_increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
1220            dP_5d5c = -dP_5c5d
1221
1222            dP_5d2d = -dP_2d5d
1223
1224            if fill_level > (0.5*prop.sphere_internalvolume+0.75*prop.internalvolume)/V_total
                    :
1225                dP_5dl = h_H2*-(T_5d-T_liquid)*prop.increment_area
1226                dP_5dv = 0.
1227            else:
1228                dP_5dv = h_vapour*-(T_5d-T_vapour)*prop.increment_area
1229                dP_5dl = 0.
1230
1231            dP_5d_net = dP_5d6d + dP_5dl + dP_5dv + dP_5d4d + dP_5d5c + dP_5d2d
1232
1233            dP_5net = dP_5a_net+dP_5i_net+dP_5b_net+dP_5c_net+dP_5d_net
1234
1235            ########################
1236            ######## NODE 6 ########
1237
1238            # NODE 6A
1239            dP_s6a = q_sun * max(round(cos(radians(solar_incidence_angle)),2),0.) * prop.
                    increment_frontalarea * absorptivity \
1240                      + q_IR * max(round(cos(radians(solar_incidence_angle-180.)),2),0.)* prop
                            .increment_frontalarea * emissivity
1241            dP_6a7a = conduction(k_MLI, T_6a-T_7a, prop.increment_length)*pi*(prop.radius+
                    prop.thickness+t_SOFI)*t_MLI
1242            dP_6a6i = Lockheed(T_6a, T_6i, emissivity_MLI, layer_density, layers)[0]*prop.
                    increment_area
1243            dP_6a5a = -dP_5a6a
1244            dP_6as = radiation(emissivity,T_6a,T_space)*prop.increment_area
1245
1246            dP_6a1a = -dP_1a6a
1247
1248            dP_6a_net = dP_s6a + dP_6a7a + dP_6a6i + dP_6a5a + dP_6as + dP_6a1a
1249
1250            # NODE 6i
1251            dP_6i6a = -dP_6a6i
1252            dP_6i6b = conduction(k_SOFI, T_6i-T_6b, t_SOFI/2.)*prop.increment_area
1253
1254            dP_6i_net = dP_6i6a + dP_6i6b
1255
1256            # NODE 6B
1257            dP_6b7b = conduction(k_SOFI, T_6b-T_7b, prop.increment_length)*pi*(prop.radius+
                    prop.thickness)*t_SOFI
1258            dP_6b6c = conduction(k_SOFI, T_6b-T_6c, t_SOFI/2.)*prop.increment_area
1259            dP_6b5b = -dP_5b6b
1260            dP_6b6i = -dP_6i6b
1261
1262            dP_6b1b = -dP_1b6b
1263
1264            dP_6b_net = dP_6b7b + dP_6b6c + dP_6b5b + dP_6b6i + dP_6b1b
1265
1266            # NODE 6C
1267            dP_6c7c = conduction(k_wall, T_6c-T_7c, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
1268            dP_6c6d = conduction(k_wall, T_6c-T_6d, prop.thickness)*prop.increment_area
1269            dP_6c5c = -dP_5c6c
1270            dP_6c6b = -dP_6b6c
1271
1272            dP_6c1c = -dP_1c6c
1273
1274            dP_6c_net = dP_6c7c + dP_6c6d + dP_6c5c + dP_6c6b + dP_6c1c
1275
1276            # NODE 6D
1277            dP_6d7d = conduction(k_wall, T_6d-T_7d, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
1278            dP_6d5d = -dP_5d6d
```

```
1279            dP_6d6c = -dP_6c6d
1280
1281            dP_6d1d = -dP_1d6d
1282
1283            if fill_level > (0.5*prop.sphere_internalvolume+0.5*prop.internalvolume)/V_total:
1284                dP_6dl = h_H2*-(T_6d-T_liquid)*prop.increment_area
1285                dP_6dv = 0.
1286            else:
1287                dP_6dv = h_vapour*-(T_6d-T_vapour)*prop.increment_area
1288                dP_6dl = 0.
1289
1290            dP_6d_net = dP_6d7d + dP_6dl + dP_6dv + dP_6d5d + dP_6d6c + dP_6d1d
1291
1292            dP_6net = dP_6a_net+dP_6i_net+dP_6b_net+dP_6c_net+dP_6d_net
1293
1294            ########################
1295            ######## NODE 7 ########
1296
1297            # NODE 7A
1298            dP_s7a = q_sun * max(round(cos(radians(solar_incidence_angle)),2),0.) * prop.
                    increment_frontalarea * absorptivity \
1299                       + q_IR * max(round(cos(radians(solar_incidence_angle-180.)),2),0.)* prop
                           .increment_frontalarea * emissivity
1300            dP_7a8a = conduction(k_MLI, T_7a-T_8a, prop.increment_length)*pi*(prop.radius+
                    prop.thickness+t_SOFI)*t_MLI
1301            dP_7a7i = Lockheed(T_7a, T_7i, emissivity_MLI, layer_density, layers)[0]*prop.
                    increment_area
1302            dP_7a6a = -dP_6a7a
1303            dP_7as = radiation(emissivity,T_7a,T_space)*prop.increment_area
1304
1305            dP_7a12a = conduction(k_MLI, T_7a-T_12a, pi*(prop.radius+prop.thickness+t_SOFI+
                    t_MLI))*t_MLI*prop.increment_length
1306
1307            dP_7a_net = dP_s7a + dP_7a8a + dP_7a7i + dP_7a6a + dP_7as + dP_7a12a
1308
1309            # NODE 7i
1310            dP_7i7a = -dP_7a7i
1311            dP_7i7b = conduction(k_SOFI, T_7i-T_7b, t_SOFI/2.)*prop.increment_area
1312
1313            dP_7i_net = dP_7i7a + dP_7i7b
1314
1315            # NODE 7B
1316            dP_7b8b = conduction(k_SOFI, T_7b-T_8b, prop.increment_length)*pi*(prop.radius+
                    prop.thickness)*t_SOFI
1317            dP_7b7c = conduction(k_SOFI, T_7b-T_7c, t_SOFI/2.)*prop.increment_area
1318            dP_7b6b = -dP_6b7b
1319            dP_7b7i = -dP_7i7b
1320
1321            dP_7b12b = conduction(k_SOFI, T_7b-T_12b, pi*(prop.radius+prop.thickness+t_SOFI))
                    *t_SOFI*prop.increment_length
1322
1323            dP_7b_net = dP_7b8b + dP_7b7c + dP_7b6b + dP_7b7i + dP_7b12b
1324
1325            # NODE 7C
1326            dP_7c8c = conduction(k_wall, T_7c-T_8c, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
1327            dP_7c7d = conduction(k_wall, T_7c-T_7d, prop.thickness)*prop.increment_area
1328            dP_7c6c = -dP_6c7c
1329            dP_7c7b = -dP_7b7c
1330
1331            dP_7c12c = conduction(k_wall, T_7c-T_12c, pi*(prop.radius))*prop.thickness*prop.
                    increment_length
1332
1333            dP_7c_net = dP_7c8c + dP_7c7d + dP_7c6c + dP_7c7b + dP_7c12c
1334
1335            # NODE 7D
1336            dP_7d8d = conduction(k_wall, T_7d-T_8d, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
1337            dP_7d6d = -dP_6d7d
1338            dP_7d7c = -dP_7c7d
1339
```

```
1340            dP_7d12d = conduction(k_wall, T_7d-T_12d, pi*(prop.radius))*prop.thickness*prop.
                    increment_length
1341
1342            if fill_level > (0.5*prop.sphere_internalvolume+0.25*prop.internalvolume)/V_total
                    :
1343                dP_7dl = h_H2*-(T_7d-T_liquid)*prop.increment_area
1344                dP_7dv = 0.
1345            else:
1346                dP_7dv = h_vapour*-(T_7d-T_vapour)*prop.increment_area
1347                dP_7dl = 0.
1348
1349            dP_7d_net = dP_7d8d + dP_7dl + dP_7dv + dP_7d6d + dP_7d7c + dP_7d12d
1350
1351            dP_7net = dP_7a_net+dP_7i_net+dP_7b_net+dP_7c_net+dP_7d_net
1352
1353            ########################
1354            ######## NODE 8 ########
1355
1356            # NODE 8A
1357            dP_s8a = q_sun * max(round(cos(radians(solar_incidence_angle)),2),0.) * prop.
                    increment_frontalarea * absorptivity \
1358                      + q_IR * max(round(cos(radians(solar_incidence_angle-180.)),2),0.)* prop
                          .increment_frontalarea * emissivity
1359            dP_8a9a = conduction(k_MLI, T_8a-T_9a, prop.sphere_increment_length)*pi*(prop.
                    radius+prop.thickness+t_SOFI)*t_MLI
1360            dP_8a8i = Lockheed(T_8a, T_8i, emissivity_MLI, layer_density, layers)[0]*prop.
                    increment_area
1361            dP_8a7a = -dP_7a8a
1362            dP_8as = radiation(emissivity,T_8a,T_space)*prop.increment_area
1363
1364            dP_8a11a = conduction(k_MLI, T_8a-T_11a, pi*(prop.radius+prop.thickness+t_SOFI+
                    t_MLI))*t_MLI*prop.increment_length
1365
1366            dP_8a_net = dP_s8a + dP_8a9a + dP_8a8i + dP_8a7a + dP_8as + dP_8a11a
1367
1368            # NODE 8i
1369            dP_8i8a = -dP_8a8i
1370            dP_8i8b = conduction(k_SOFI, T_8i-T_8b, t_SOFI/2.)*prop.increment_area
1371
1372            dP_8i_net = dP_8i8a + dP_8i8b
1373
1374            # NODE 8B
1375            dP_8b9b = conduction(k_SOFI, T_8b-T_9b, prop.sphere_increment_length)*pi*(prop.
                    radius+prop.thickness)*t_SOFI
1376            dP_8b8c = conduction(k_SOFI, T_8b-T_8c, t_SOFI/2.)*prop.increment_area
1377            dP_8b7b = -dP_7b8b
1378            dP_8b8i = -dP_8i8b
1379
1380            dP_8b11b = conduction(k_SOFI, T_8b-T_11b, pi*(prop.radius+prop.thickness+t_SOFI))
                    *t_SOFI*prop.increment_length
1381
1382            dP_8b_net = dP_8b9b + dP_8b8c + dP_8b7b + dP_8b8i + dP_8b11b
1383
1384            # NODE 8C
1385            dP_8c9c = conduction(k_wall, T_8c-T_9c, prop.sphere_increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
1386            dP_8c8d = conduction(k_wall, T_8c-T_8d, prop.thickness)*prop.increment_area
1387            dP_8c7c = -dP_7c8c
1388            dP_8c8b = -dP_8b8c
1389
1390            dP_8c11c = conduction(k_wall, T_8c-T_11c, pi*(prop.radius))*prop.thickness*prop.
                    increment_length
1391
1392            dP_8c_net = dP_8c9c + dP_8c8d + dP_8c7c + dP_8c8b + dP_8c11c
1393
1394            # NODE 8D
1395            dP_8d9d = conduction(k_wall, T_8d-T_9d, prop.sphere_increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
1396            dP_8d7d = -dP_7d8d
1397            dP_8d8c = -dP_8c8d
1398
```

```
1399            dP_8d11d = conduction(k_wall, T_8d-T_12d, pi*(prop.radius))*prop.thickness*prop.
                    increment_length
1400
1401            if fill_level > (0.5*prop.sphere_internalvolume)/V_total:
1402                dP_8dl = h_H2*-(T_8d-T_liquid)*prop.increment_area
1403                dP_8dv = 0.
1404            else:
1405                dP_8dv = h_vapour*-(T_8d-T_vapour)*prop.increment_area
1406                dP_8dl = 0.
1407
1408            dP_8d_net = dP_8d9d + dP_8dl + dP_8dv + dP_8d7d + dP_8d8c + dP_8d11d
1409
1410            dP_8net = dP_8a_net+dP_8i_net+dP_8b_net+dP_8c_net+dP_8d_net
1411            ########################
1412
1413            ######## NODE 9 ########
1414
1415            # NODE 9A
1416            dP_s9a = q_sun * max(round(cos(radians(solar_incidence_angle+45.)),2),0.) * prop.
                    sphere_increment_frontalarea * absorptivity \
1417                     + q_IR * max(round(cos(radians(solar_incidence_angle-135.)),2),0.)* prop
                            .sphere_increment_frontalarea * emissivity
1418            dP_9a10a = conduction(k_MLI, T_9a-T_10a, prop.sphere_increment_length)*pi*(prop.
                    radius+prop.thickness+t_SOFI)*t_MLI
1419            dP_9a9i = Lockheed(T_9a, T_9i, emissivity_MLI, layer_density, layers)[0]*prop.
                    sphere_increment_area
1420            dP_9a8a = conduction(k_MLI, T_9a-T_8a, prop.increment_length)*pi*(prop.radius+
                    prop.thickness+t_SOFI)*t_MLI
1421            dP_9as = radiation(emissivity,T_9a,T_space)*prop.sphere_increment_area
1422
1423            dP_9a_net = dP_s9a + dP_9a10a + dP_9a9i + dP_9a8a + dP_9as
1424
1425            # NODE 9i
1426            dP_9i9a = -dP_9a9i
1427            dP_9i9b = conduction(k_SOFI, T_9i-T_9b, t_SOFI/2.)*prop.sphere_increment_area
1428
1429            dP_9i_net = dP_9i9a + dP_9i9b
1430
1431            # NODE 9B
1432            dP_9b10b = conduction(k_SOFI, T_9b-T_10b, prop.sphere_increment_length)*pi*(prop.
                    radius+prop.thickness)*t_SOFI
1433            dP_9b9c = conduction(k_SOFI, T_9b-T_9c, t_SOFI/2.)*prop.sphere_increment_area
1434            dP_9b8b = conduction(k_SOFI, T_9b-T_8b, prop.increment_length)*pi*(prop.radius+
                    prop.thickness)*t_SOFI
1435            dP_9b9i = -dP_9i9b
1436
1437            dP_9b_net = dP_9b10b + dP_9b9c + dP_9b8b + dP_9b9i
1438
1439            # NODE 9C
1440            dP_9c10c = conduction(k_wall, T_9c-T_10c, prop.sphere_increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
1441            dP_9c9d = conduction(k_wall, T_9c-T_9d, prop.thickness)*prop.
                    sphere_increment_area
1442            dP_9c8c = conduction(k_wall, T_9c-T_8c, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
1443            dP_9c9b = -dP_9b9c
1444
1445            dP_9c_net = dP_9c10c + dP_9c9d + dP_9c8c + dP_9c9b
1446
1447            # NODE 9D
1448            dP_9d10d = conduction(k_wall, T_9d-T_10d, prop.sphere_increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
1449            dP_9dl = h_H2*-(T_9d-T_liquid)*prop.sphere_increment_area
1450            dP_9dv = 0.
1451            dP_9d8d = conduction(k_wall, T_9d-T_8d, prop.increment_length)*pi*(prop.radius)*
                    prop.thickness/2.
1452            dP_9d9c = -dP_9c9d
1453
1454            dP_9d_net = dP_9d10d + dP_9dl +dP_9dv + dP_9d8d + dP_9d9c
1455
1456            dP_9net = dP_9a_net+dP_9i_net+dP_9b_net+dP_9c_net+dP_9d_net
```

```
1457
1458            ########################
1459
1460            ######## NODE 10 ########
1461
1462            # NODE 10A
1463            dP_s10a = q_sun * max(round(cos(radians(solar_incidence_angle+135.)),2),0.) *
                    prop.sphere_increment_frontalarea * absorptivity \
1464                      + q_IR * max(round(cos(radians(solar_incidence_angle-45.)),2),0.)* prop.
                          sphere_increment_frontalarea * emissivity
1465            dP_10a11a = conduction(k_MLI, T_10a-T_11a, prop.increment_length)*pi*(prop.radius
                    +prop.thickness+t_SOFI)*t_MLI
1466            dP_10a10i = Lockheed(T_10a, T_10i, emissivity_MLI, layer_density, layers)[0]*prop
                    .sphere_increment_area
1467            dP_10a9a = -dP_9a10a
1468            dP_10as = radiation(emissivity,T_10a,T_space)*prop.sphere_increment_area
1469
1470            dP_10a_net = dP_s10a + dP_10a11a + dP_10a10i + dP_10a9a + dP_10as
1471
1472            # NODE 10i
1473            dP_10i10a = -dP_10a10i
1474            dP_10i10b = conduction(k_SOFI, T_10i-T_10b, t_SOFI/2.)*prop.sphere_increment_area
1475
1476            dP_10i_net = dP_10i10a + dP_10i10b
1477
1478            # NODE 10B
1479            dP_10b11b = conduction(k_SOFI, T_10b-T_11b, prop.increment_length)*pi*(prop.
                    radius+prop.thickness)*t_SOFI
1480            dP_10b10c = conduction(k_SOFI, T_10b-T_10c, t_SOFI/2.)*prop.sphere_increment_area
1481            dP_10b9b = -dP_9b10b
1482            dP_10b10i = -dP_10i10b
1483
1484            dP_10b_net = dP_10b11b + dP_10b10c + dP_10b9b + dP_10b10i
1485
1486            # NODE 10C
1487            dP_10c11c = conduction(k_wall, T_10c-T_11c, prop.increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
1488            dP_10c10d = conduction(k_wall, T_10c-T_10d, prop.thickness)*prop.
                    sphere_increment_area
1489            dP_10c9c = -dP_9c10c
1490            dP_10c10b = -dP_10b10c
1491
1492            dP_10c_net = dP_10c11c + dP_10c10d + dP_10c9c + dP_10c10b
1493
1494            # NODE 10D
1495            dP_10d11d = conduction(k_wall, T_10d-T_11d, prop.increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
1496            dP_10dl = h_H2*-(T_10d-T_liquid)*prop.sphere_increment_area
1497            dP_10dv = 0.
1498            dP_10d9d = -dP_9d10d
1499            dP_10d10c = -dP_10c10d
1500
1501            dP_10d_net = dP_10d11d + dP_10dl + dP_10dv + dP_10d9d + dP_10d10c
1502
1503            dP_10net = dP_10a_net+dP_10i_net+dP_10b_net+dP_10c_net+dP_10d_net
1504
1505            ########################
1506
1507            ######## NODE 11 ########
1508
1509            # NODE 11A
1510            dP_s11a = q_sun * max(round(cos(radians(solar_incidence_angle-180.)),2),0.) *
                    prop.increment_frontalarea * absorptivity \
1511                      + q_IR * max(round(cos(radians(solar_incidence_angle)),2),0.)* prop.
                          increment_frontalarea * emissivity
1512            dP_11a12a = conduction(k_MLI, T_11a-T_12a, prop.increment_length)*pi*(prop.radius
                    +prop.thickness+t_SOFI)*t_MLI
1513            dP_11a11i = Lockheed(T_11a, T_11i, emissivity_MLI, layer_density, layers)[0]*prop
                    .increment_area
1514            dP_11a10a = conduction(k_MLI, T_11a-T_10a, prop.sphere_increment_length)*pi*(prop
                    .radius+prop.thickness+t_SOFI)*t_MLI
```

```
1515            dP_11as = radiation(emissivity,T_11a,T_space)*prop.increment_area
1516
1517            dP_11a8a = -dP_8a11a
1518
1519            dP_11a_net = dP_s11a + dP_11a12a + dP_11a11i + dP_11a10a + dP_11as + dP_11a8a
1520
1521            # NODE 11i
1522            dP_11i11a = -dP_11a11i
1523            dP_11i11b = conduction(k_SOFI, T_11i-T_11b, t_SOFI/2.)*prop.increment_area
1524
1525            dP_11i_net = dP_11i11a + dP_11i11b
1526
1527            # NODE 11B
1528            dP_11b12b = conduction(k_SOFI, T_11b-T_12b, prop.increment_length)*pi*(prop.
                    radius+prop.thickness)*t_SOFI
1529            dP_11b11c = conduction(k_SOFI, T_11b-T_11c, t_SOFI/2.)*prop.increment_area
1530            dP_11b10b = conduction(k_SOFI, T_11b-T_10b, prop.sphere_increment_length)*pi*(
                    prop.radius+prop.thickness)*t_SOFI
1531            dP_11b11i = -dP_11i11b
1532
1533            dP_11b8b = -dP_8b11b
1534
1535            dP_11b_net = dP_11b12b + dP_11b11c + dP_11b10b + dP_11b11i + dP_11b8b
1536
1537            # NODE 11C
1538            dP_11c12c = conduction(k_wall, T_11c-T_12c, prop.increment_length)*pi*(prop.
                    radius)*prop.thickness/2.
1539            dP_11c11d = conduction(k_wall, T_11c-T_11d, prop.thickness)*prop.increment_area
1540            dP_11c10c = conduction(k_wall, T_11c-T_10c, prop.sphere_increment_length)*pi*(
                    prop.radius)*prop.thickness/2.
1541            dP_11c11b = -dP_11b11c
1542
1543            dP_11c8c = -dP_8c11c
1544
1545            dP_11c_net = dP_11c12c + dP_11c11d + dP_11c10c + dP_11c11b + dP_11c8c
1546
1547            # NODE 11D
1548            dP_11d12d = conduction(k_wall, T_11d-T_12d, prop.increment_length)*pi*(prop.
                    radius)*prop.thickness
1549            dP_11d10d = conduction(k_wall, T_11d-T_10d, prop.sphere_increment_length)*pi*(
                    prop.radius)*prop.thickness
1550            dP_11d11c = -dP_11c11d
1551
1552            dP_11d8d = -dP_8d11d
1553
1554            if fill_level > (0.5*prop.sphere_internalvolume)/V_total:
1555                dP_11dl = h_H2*-(T_11d-T_liquid)*prop.increment_area
1556                dP_11dv = 0.
1557            else:
1558                dP_11dv = h_vapour*-(T_11d-T_vapour)*prop.increment_area
1559                dP_11dl = 0.
1560
1561            dP_11d_net = dP_11d12d + dP_11dl + dP_11dv + dP_11d10d + dP_11d11c + dP_11d8d
1562
1563            dP_11net = dP_11a_net+dP_11i_net+dP_11b_net+dP_11c_net+dP_11d_net
1564
1565            ########################
1566
1567            ######## NODE 12 ########
1568
1569            # NODE 12A
1570            dP_s12a = q_sun * max(round(cos(radians(solar_incidence_angle-180.)),2),0.) *
                    prop.increment_frontalarea * absorptivity \
1571                      + q_IR * max(round(cos(radians(solar_incidence_angle)),2),0.)* prop.
                          increment_frontalarea * emissivity
1572            dP_12a1a = -dP_1a12a
1573            dP_12a12i = Lockheed(T_12a, T_12i, emissivity_MLI, layer_density, layers)[0]*prop
                    .increment_area
1574            dP_12a11a = -dP_11a12a
1575            dP_12as = radiation(emissivity,T_12a,T_space)*prop.increment_area
1576
```

```
1577            dP_12a7a = -dP_7a12a
1578
1579            dP_12a_net = dP_s12a + dP_12a1a + dP_12a12i + dP_12a11a + dP_12as + dP_12a7a
1580
1581            # NODE 12i
1582            dP_12i12a = -dP_12a12i
1583            dP_12i12b = conduction(k_SOFI, T_12i-T_12b, t_SOFI/2.)*prop.increment_area
1584
1585            dP_12i_net = dP_12i12a + dP_12i12b
1586
1587            # NODE 12B
1588            dP_12b1b = -dP_1b12b
1589            dP_12b12c = conduction(k_SOFI, T_12b-T_12c, t_SOFI/2.)*prop.increment_area
1590            dP_12b11b = -dP_11b12b
1591            dP_12b12i = -dP_12i12b
1592
1593            dP_12b7b = -dP_7b12b
1594
1595            dP_12b_net = dP_12b1b + dP_12b12c + dP_12b11b + dP_12b12i + dP_12b7b
1596
1597            # NODE 12C
1598            dP_12c1c = -dP_1c12c
1599            dP_12c12d = conduction(k_wall, T_12c-T_12d, prop.thickness)*prop.increment_area
1600            dP_12c11c = -dP_11c12c
1601            dP_12c12b = -dP_12b12c
1602
1603            dP_12c7c = -dP_7c12c
1604
1605            dP_12c_net = dP_12c1c + dP_12c12d + dP_12c11c + dP_12c12b + dP_12c7c
1606
1607            # NODE 12D
1608            dP_12d1d = -dP_1d12d
1609            dP_12d11d = -dP_11d12d
1610            dP_12d12c = -dP_12c12d
1611
1612            dP_12d7d = -dP_7d12d
1613
1614            if fill_level > (0.5*prop.sphere_internalvolume+0.25*prop.internalvolume)/V_total
                    :
1615                dP_12dl = h_H2*-(T_12d-T_liquid)*prop.increment_area
1616                dP_12dv = 0.
1617            else:
1618                dP_12dv = h_vapour*-(T_12d-T_vapour)*prop.increment_area
1619                dP_12dl = 0.
1620
1621            dP_12d_net = dP_12d1d + dP_12dl + dP_12dv + dP_12d11d + dP_12d12c + dP_12d7d
1622
1623            dP_12net = dP_12a_net+dP_12i_net+dP_12b_net+dP_12c_net+dP_12d_net
1624
1625            ########################
1626
1627            ######## LIQUID ########
1628
1629            # NODE L(iquid)
1630            dP_l1d = -dP_1dl
1631            dP_l2d = -dP_2dl
1632            dP_l3d = -dP_3dl
1633            dP_l4d = -dP_4dl
1634            dP_l5d = -dP_5dl
1635            dP_l6d = -dP_6dl
1636            dP_l7d = -dP_7dl
1637            dP_l8d = -dP_8dl
1638            dP_l9d = -dP_9dl
1639            dP_l10d = -dP_10dl
1640            dP_l11d = -dP_11dl
1641            dP_l12d = -dP_12dl
1642            dP_lv = h_H2* -(T_liquid - T_vapour) * pi * prop.radius ** 2.
1643            dP_cooler = -Q_cooler
1644            dP_mixer = Q_mixer
1645            dP_penetration = 0.0025*-(T_liquid-T_space)*(V_total)**(0.5)
1646
```

```
1647            dP_liquid_net = dP_l1d + dP_l2d + dP_l3d + dP_l4d + dP_l5d + dP_l6d + dP_l7d +
                    dP_l8d + dP_l9d + dP_l10d + dP_l11d + dP_l12d + dP_lv + dP_cooler +
                    dP_penetration +dP_mixer
1648            P_rest = 0.
1649            ########################
1650
1651            ######## VAPOUR ########
1652
1653            # NODE V(apour)
1654            dP_v1d = -dP_1dv
1655            dP_v2d = -dP_2dv
1656            dP_v3d = -dP_3dv
1657            dP_v4d = -dP_4dv
1658            dP_v5d = -dP_5dv
1659            dP_v6d = -dP_6dv
1660            dP_v7d = -dP_7dv
1661            dP_v8d = -dP_8dv
1662            dP_v9d = -dP_9dv
1663            dP_v10d = -dP_10dv
1664            dP_v11d = -dP_11dv
1665            dP_v12d = -dP_12dv
1666            dP_vl = -dP_lv
1667
1668            dP_vapour_net = dP_v1d + dP_v2d + dP_v3d + dP_v4d + dP_v5d + dP_v6d + dP_v7d +
                    dP_v8d + dP_v9d + dP_v10d + dP_v11d + dP_v12d + dP_vl
1669
1670            ########################
1671
1672            # # ALL CONDUCTION IN THE SYSTEM PLUS RADIATION IN AND OUT
1673            dP_system = dP_1net + dP_2net + dP_3net + dP_4net + dP_5net + dP_6net + dP_7net +
                     dP_8net + dP_9net + dP_10net + dP_11net + dP_12net + dP_liquid_net+
                    dP_vapour_net
1674            # ALL RADIATION FROM SPACE TO THE TANK DUE TO THE SUN
1675            dP_in = dP_s1a+dP_s2a+dP_s3a+dP_s4a+dP_s5a+dP_s6a+dP_s7a+dP_s8a+dP_s9a+dP_s10a+
                    dP_s11a+dP_s12a
1676            # ALL RADIATION FROM THE TANK BACK TO SPACE DUE TO IR
1677            dP_out = dP_1as+dP_2as+dP_3as+dP_4as+dP_5as+dP_6as+dP_7as+dP_8as+dP_9as+dP_10as+
                    dP_11as+dP_12as
1678
1679            dP_tot = dP_system-dP_in-dP_out
1680
1681            #Temperature increase
1682            dT_1a = dP_1a_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.height/
                    prop.cylinder_parts)*layers*density_MLI/2.)
1683            dT_1i = dP_1i_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.height/
                    prop.cylinder_parts)*layers*density_MLI/2.)
1684            dT_1b = dP_1b_net*dt/(Cp_SOFI*(pi*(prop.thickness+prop.radius)*t_SOFI*prop.height
                    /prop.cylinder_parts)*density_SOFI)
1685            dT_1c = dP_1c_net*dt/(Cp_wall*prop.increment_mass/2.)
1686            dT_1d = dP_1d_net*dt/(Cp_wall*prop.increment_mass/2.)
1687            T_1a = T_1a + dT_1a
1688            T_1i = T_1i + dT_1i
1689            T_1b = T_1b + dT_1b
1690            T_1c = T_1c + dT_1c
1691            T_1d = T_1d + dT_1d
1692
1693            dT_2a = dP_2a_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.height/
                    prop.cylinder_parts)*layers*density_MLI/2.)
1694            dT_2i = dP_2i_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.height/
                    prop.cylinder_parts)*layers*density_MLI/2.)
1695            dT_2b = dP_2b_net*dt/(Cp_SOFI*(pi*(prop.thickness+prop.radius)*t_SOFI*prop.height
                    /prop.cylinder_parts)*density_SOFI)
1696            dT_2c = dP_2c_net*dt/(Cp_wall*prop.increment_mass/2.)
1697            dT_2d = dP_2d_net*dt/(Cp_wall*prop.increment_mass/2.)
1698            T_2a = T_2a + dT_2a
1699            T_2i = T_2i + dT_2i
1700            T_2b = T_2b + dT_2b
1701            T_2c = T_2c + dT_2c
1702            T_2d = T_2d + dT_2d
1703
```

```
1704            dT_3a = dP_3a_net*dt / (Cp_MLI * (pi * (prop.radius + prop.thickness+t_SOFI) **
                    2. *layers * density_MLI)/2.)
1705            dT_3i = dP_3i_net*dt / (Cp_MLI * (pi * (prop.radius + prop.thickness+t_SOFI) **
                    2. *layers * density_MLI)/2.)
1706            dT_3b = dP_3b_net*dt / (Cp_SOFI * ((4 / 3.) * pi * ((prop.radius + prop.thickness
                    +t_SOFI) ** 3. - (prop.radius+prop.thickness) ** 3.)/prop.sphere_parts) *
                    density_SOFI)
1707            dT_3c = dP_3c_net*dt / (Cp_wall * prop.sphere_increment_mass/2.)
1708            dT_3d = dP_3d_net*dt / (Cp_wall * prop.sphere_increment_mass/2.)
1709            T_3a = T_3a + dT_3a
1710            T_3i = T_3i + dT_3i
1711            T_3b = T_3b + dT_3b
1712            T_3c = T_3c + dT_3c
1713            T_3d = T_3d + dT_3d
1714
1715            dT_4a = dP_4a_net*dt / (Cp_MLI * (pi * (prop.radius + prop.thickness+t_SOFI) **
                    2. *layers * density_MLI)/2.)
1716            dT_4i = dP_4i_net*dt / (Cp_MLI * (pi * (prop.radius + prop.thickness+t_SOFI) **
                    2. *layers * density_MLI)/2.)
1717            dT_4b = dP_4b_net*dt / (Cp_SOFI * ((4 / 3.) * pi * ((prop.radius + prop.thickness
                    +t_SOFI) ** 3. - (prop.radius+prop.thickness) ** 3.)/prop.sphere_parts) *
                    density_SOFI)
1718            dT_4c = dP_4c_net*dt / (Cp_wall * prop.sphere_increment_mass/2.)
1719            dT_4d = dP_4d_net*dt / (Cp_wall * prop.sphere_increment_mass/2.)
1720            T_4a = T_4a + dT_4a
1721            T_4i = T_4i + dT_4i
1722            T_4b = T_4b + dT_4b
1723            T_4c = T_4c + dT_4c
1724            T_4d = T_4d + dT_4d
1725
1726            dT_5a = dP_5a_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.height/
                    prop.cylinder_parts)*layers*density_MLI/2.)
1727            dT_5i = dP_5i_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.height/
                    prop.cylinder_parts)*layers*density_MLI/2.)
1728            dT_5b = dP_5b_net*dt/(Cp_SOFI*(pi*(prop.thickness+prop.radius)*t_SOFI*prop.height
                    /prop.cylinder_parts)*density_SOFI)
1729            dT_5c = dP_5c_net*dt/(Cp_wall*prop.increment_mass/2.)
1730            dT_5d = dP_5d_net*dt/(Cp_wall*prop.increment_mass/2.)
1731            T_5a = T_5a + dT_5a
1732            T_5i = T_5i + dT_5i
1733            T_5b = T_5b + dT_5b
1734            T_5c = T_5c + dT_5c
1735            T_5d = T_5d + dT_5d
1736
1737            dT_6a = dP_6a_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.height/
                    prop.cylinder_parts)*layers*density_MLI/2.)
1738            dT_6i = dP_6i_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.height/
                    prop.cylinder_parts)*layers*density_MLI/2.)
1739            dT_6b = dP_6b_net*dt/(Cp_SOFI*(pi*(prop.thickness+prop.radius)*t_SOFI*prop.height
                    /prop.cylinder_parts)*density_SOFI)
1740            dT_6c = dP_6c_net*dt/(Cp_wall*prop.increment_mass/2.)
1741            dT_6d = dP_6d_net*dt/(Cp_wall*prop.increment_mass/2.)
1742            T_6a = T_6a + dT_6a
1743            T_6i = T_6i + dT_6i
1744            T_6b = T_6b + dT_6b
1745            T_6c = T_6c + dT_6c
1746            T_6d = T_6d + dT_6d
1747
1748            dT_7a = dP_7a_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.height/
                    prop.cylinder_parts)*layers*density_MLI/2.)
1749            dT_7i = dP_7i_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.height/
                    prop.cylinder_parts)*layers*density_MLI/2.)
1750            dT_7b = dP_7b_net*dt/(Cp_SOFI*(pi*(prop.thickness+prop.radius)*t_SOFI*prop.height
                    /prop.cylinder_parts)*density_SOFI)
1751            dT_7c = dP_7c_net*dt/(Cp_wall*prop.increment_mass/2.)
1752            dT_7d = dP_7d_net*dt/(Cp_wall*prop.increment_mass/2.)
1753            T_7a = T_7a + dT_7a
1754            T_7i = T_7i + dT_7i
1755            T_7b = T_7b + dT_7b
1756            T_7c = T_7c + dT_7c
1757            T_7d = T_7d + dT_7d
```

```
1758
1759            dT_8a = dP_8a_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.height/
                    prop.cylinder_parts)*layers*density_MLI/2.)
1760            dT_8i = dP_8i_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.height/
                    prop.cylinder_parts)*layers*density_MLI/2.)
1761            dT_8b = dP_8b_net*dt/(Cp_SOFI*(pi*(prop.thickness+prop.radius)*t_SOFI*prop.height
                    /prop.cylinder_parts)*density_SOFI)
1762            dT_8c = dP_8c_net*dt/(Cp_wall*prop.increment_mass/2.)
1763            dT_8d = dP_8d_net*dt/(Cp_wall*prop.increment_mass/2.)
1764            T_8a = T_8a + dT_8a
1765            T_8i = T_8i + dT_8i
1766            T_8b = T_8b + dT_8b
1767            T_8c = T_8c + dT_8c
1768            T_8d = T_8d + dT_8d
1769
1770            dT_9a = dP_9a_net*dt / (Cp_MLI * (pi * (prop.radius + prop.thickness+t_SOFI) **
                    2. *layers * density_MLI)/2.)
1771            dT_9i = dP_9i_net*dt / (Cp_MLI * (pi * (prop.radius + prop.thickness+t_SOFI) **
                    2. *layers * density_MLI)/2.)
1772            dT_9b = dP_9b_net*dt / (Cp_SOFI * ((4 / 3.) * pi * ((prop.radius + prop.thickness
                    +t_SOFI) ** 3. - (prop.radius+prop.thickness) ** 3.)/prop.sphere_parts) *
                    density_SOFI)
1773            dT_9c = dP_9c_net*dt / (Cp_wall * prop.sphere_increment_mass/2.)
1774            dT_9d = dP_9d_net*dt / (Cp_wall * prop.sphere_increment_mass/2.)
1775            T_9a = T_9a + dT_9a
1776            T_9i = T_9i + dT_9i
1777            T_9b = T_9b + dT_9b
1778            T_9c = T_9c + dT_9c
1779            T_9d = T_9d + dT_9d
1780
1781            dT_10a = dP_10a_net*dt / (Cp_MLI * (pi * (prop.radius + prop.thickness+t_SOFI) **
                     2. *layers * density_MLI)/2.)
1782            dT_10i = dP_10i_net*dt / (Cp_MLI * (pi * (prop.radius + prop.thickness+t_SOFI) **
                     2. *layers * density_MLI)/2.)
1783            dT_10b = dP_10b_net*dt / (Cp_SOFI * ((4 / 3.) * pi * ((prop.radius + prop.
                    thickness+t_SOFI) ** 3. - (prop.radius+prop.thickness) ** 3.)/prop.
                    sphere_parts) * density_SOFI)
1784            dT_10c = dP_10c_net*dt / (Cp_wall * prop.sphere_increment_mass/2.)
1785            dT_10d = dP_10d_net*dt / (Cp_wall * prop.sphere_increment_mass/2.)
1786            T_10a = T_10a + dT_10a
1787            T_10i = T_10i + dT_10i
1788            T_10b = T_10b + dT_10b
1789            T_10c = T_10c + dT_10c
1790            T_10d = T_10d + dT_10d
1791
1792            dT_11a = dP_11a_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.
                    height/prop.cylinder_parts)*layers*density_MLI/2.)
1793            dT_11i = dP_11i_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.
                    height/prop.cylinder_parts)*layers*density_MLI/2.)
1794            dT_11b = dP_11b_net*dt/(Cp_SOFI*(pi*(prop.thickness+prop.radius)*t_SOFI*prop.
                    height/prop.cylinder_parts)*density_SOFI)
1795            dT_11c = dP_11c_net*dt/(Cp_wall*prop.increment_mass/2.)
1796            dT_11d = dP_11d_net*dt/(Cp_wall*prop.increment_mass/2.)
1797            T_11a = T_11a + dT_11a
1798            T_11i = T_11i + dT_11i
1799            T_11b = T_11b + dT_11b
1800            T_11c = T_11c + dT_11c
1801            T_11d = T_11d + dT_11d
1802
1803            dT_12a = dP_12a_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.
                    height/prop.cylinder_parts)*layers*density_MLI/2.)
1804            dT_12i = dP_12i_net*dt/(Cp_MLI*(pi*(prop.thickness+prop.radius+t_SOFI)*prop.
                    height/prop.cylinder_parts)*layers*density_MLI/2.)
1805            dT_12b = dP_12b_net*dt/(Cp_SOFI*(pi*(prop.thickness+prop.radius)*t_SOFI*prop.
                    height/prop.cylinder_parts)*density_SOFI)
1806            dT_12c = dP_12c_net*dt/(Cp_wall*prop.increment_mass/2.)
1807            dT_12d = dP_12d_net*dt/(Cp_wall*prop.increment_mass/2.)
1808            T_12a = T_12a + dT_12a
1809            T_12i = T_12i + dT_12i
1810            T_12b = T_12b + dT_12b
1811            T_12c = T_12c + dT_12c
```

```
1812            T_12d = T_12d + dT_12d
1813
1814            T_sat = saturation_temperature(max(p_vapour_initial, min(p_max, p_vapour)))
1815
1816            # try:
1817            #     T_sat = saturation_temperature(p_vapour)
1818            # except:
1819            #     print 'error'
1820            #     T_sat = saturation_temperature(p_max)
1821
1822            Cp_mix = (m_boil_off*Cp_H2_vapour+m_vapour_initial*Cp_He_vapour)/m_vapour
1823            m_mix = (m_boil_off*2.016+m_vapour_initial*4.000)/m_vapour*1e-3
1824
1825
1826            if T_liquid < T_sat:
1827
1828                dT_liquid = dP_liquid_net*dt/(Cp_H2*m_liquid)
1829                T_liquid = T_liquid+dT_liquid
1830
1831                if T_liquid > T_sat:
1832                    if t_boil == 0.:
1833                        t_boil = t/60./60./24.
1834
1835                    dT_rest = T_liquid-T_sat
1836                    P_rest = dT_rest*Cp_H2*m_liquid
1837                    dm_boil_off = P_rest/L_H2/1000.
1838                    m_boil_off = m_boil_off+dm_boil_off
1839                    m_liquid = m_liquid-dm_boil_off
1840                    T_liquid = T_sat
1841
1842                V_liquid = m_liquid/density_liquid
1843                fill_level = V_liquid/V_total
1844                V_vapour = V_total-V_liquid
1845                dT_vapour = dP_vapour_net*dt/(Cp_mix*m_vapour)
1846                T_vapour = T_vapour+dT_vapour
1847                dp_vapour = pressure_increase(T_sat,L_H2,2.016,V_vapour,P_rest) + (R_G*
                        dP_vapour_net*dt/(m_mix*V_vapour*Cp_mix))
1848                p_vapour = p_vapour + dp_vapour
1849
1850                if p_vapour > p_max:
1851                    dp_vent = p_max-p_vapour
1852                    dm_vent = -(dp_vent * V_vapour *(2.016e-3) / (R_G * T_vapour))
1853                    m_vent = m_vent + dm_vent
1854                    m_vapour = m_vapour-dm_vent
1855                    dT_vapour_vent = V_vapour*dp_vent/(m_vapour*Cp_mix)
1856                    T_vapour = T_vapour+dT_vapour_vent
1857                    p_vapour = p_max
1858
1859
1860            elif T_liquid >= T_sat:
1861                if t_boil == 0.:
1862                    t_boil = t/60./60./24.
1863
1864                dm_boil_off = dP_liquid_net*dt/L_H2/1000.
1865
1866                m_boil_off = m_boil_off+dm_boil_off
1867                m_liquid = m_liquid - dm_boil_off
1868
1869                V_liquid = m_liquid/density_liquid
1870                fill_level = V_liquid/V_total
1871
1872                m_vapour = m_vapour + dm_boil_off
1873
1874                dT_vapour = dP_vapour_net*dt/(Cp_mix*m_vapour)
1875                T_vapour = T_vapour+dT_vapour
1876
1877                V_vapour = V_total-V_liquid
1878                dp_vapour = pressure_increase(T_sat,L_H2,2.016,V_vapour,P_rest) + (R_G*
                        dP_vapour_net*dt/(m_mix*V_vapour*Cp_mix))
1879                p_vapour = p_vapour+dp_vapour
1880
```

```
1881                if p_vapour > p_max:
1882                    dp_vent = p_max-p_vapour
1883                    dm_vent = -(dp_vent * V_vapour *m_mix / (R_G * T_vapour))
1884                    m_vent = m_vent + dm_vent
1885                    m_vapour = m_vapour-dm_vent
1886                    dT_vapour_vent = V_vapour*dp_vent/(m_vapour*Cp_mix)
1887                    T_vapour = T_vapour+dT_vapour_vent
1888                    p_vapour = p_max
1889
1890                if m_liquid <= 0. or V_liquid <= 0.:
1891                    print 'V_liquid = ', V_liquid, 'm_liquid =', m_liquid
1892                    break
1893
1894            if t >= (60.*60.*24.*30.*6.) and six_months == False:
1895                Boil_off_monthly = (m_boil_off/m_liquid_initial/t*60.*60.*24.*30.)*100.
1896                with open('mass_and_boil-off_v15-6m.csv', 'a') as filewriter:
1897                    writer = csv.writer(filewriter)
1898                    writer.writerow([t_SOFI, m_SOFI, layer_density, layers, m_MLI, Q_cooler,
                            m_cooler, prop.thickness, m_shell, m_boil_off, m_vent, m_boil_off+
                            m_SOFI+m_MLI+m_shell+m_cooler, Boil_off_monthly, t_boil, (t
                            /60./60./24.), T_liquid, T_vapour, T_1a, T_1i, T_1b, T_1c, T_1d, T_2a
                            , T_2i, T_2b, T_2c, T_2d, T_3a, T_3i, T_3b, T_3c, T_3d, T_4a, T_4i,
                            T_4b, T_4c, T_4d, T_5a, T_5i, T_5b, T_5c, T_5d, T_6a, T_6i, T_6b,
                            T_6c, T_6d,  T_7a, T_7i, T_7b, T_7c, T_7d, T_8a, T_8i, T_8b, T_8c,
                            T_8d, T_9a, T_9i, T_9b, T_9c, T_9d, T_10a, T_10i, T_10b, T_10c, T_10d
                            , T_11a, T_11i, T_11b, T_11c, T_11d, T_12a, T_12i, T_12b, T_12c,
                            T_12d, dP_liquid_net, dP_vapour_net, dP_1a_net, dP_1i_net, dP_1b_net,
                             dP_1c_net, dP_1d_net, dP_2a_net, dP_2i_net, dP_2b_net, dP_2c_net,
                            dP_2d_net, dP_3a_net, dP_3i_net, dP_3b_net, dP_3c_net, dP_3d_net,
                            dP_4a_net, dP_4i_net, dP_4b_net, dP_4c_net, dP_4d_net, dP_5a_net,
                            dP_5i_net, dP_5b_net, dP_5c_net, dP_5d_net, dP_6a_net, dP_6i_net,
                            dP_6b_net, dP_6c_net, dP_6d_net, dP_7a_net, dP_7i_net, dP_7b_net,
                            dP_7c_net, dP_7d_net, dP_8a_net, dP_8i_net, dP_8b_net, dP_8c_net,
                            dP_8d_net, dP_9a_net, dP_9i_net, dP_9b_net, dP_9c_net, dP_9d_net,
                            dP_10a_net, dP_10i_net, dP_10b_net, dP_10c_net, dP_10d_net,
                            dP_11a_net, dP_11i_net, dP_11b_net, dP_11c_net, dP_11d_net,
                            dP_12a_net, dP_12i_net, dP_12b_net, dP_12c_net, dP_12d_net])
1899                filewriter.close()
1900
1901                six_months = True
1902
1903            elif t >= (60.*60.*24.*30.*9.) and nine_months == False:
1904                Boil_off_monthly = (m_boil_off/m_liquid_initial/t*60.*60.*24.*30.)*100.
1905                with open('mass_and_boil-off_v15-9m.csv', 'a') as filewriter:
1906                    writer = csv.writer(filewriter)
1907                    writer.writerow([t_SOFI, m_SOFI, layer_density, layers, m_MLI, Q_cooler,
                            m_cooler, prop.thickness, m_shell, m_boil_off, m_vent, m_boil_off+
                            m_SOFI+m_MLI+m_shell+m_cooler, Boil_off_monthly, t_boil, (t
                            /60./60./24.), T_liquid, T_vapour, T_1a, T_1i, T_1b, T_1c, T_1d, T_2a
                            , T_2i, T_2b, T_2c, T_2d, T_3a, T_3i, T_3b, T_3c, T_3d, T_4a, T_4i,
                            T_4b, T_4c, T_4d, T_5a, T_5i, T_5b, T_5c, T_5d, T_6a, T_6i, T_6b,
                            T_6c, T_6d,  T_7a, T_7i, T_7b, T_7c, T_7d, T_8a, T_8i, T_8b, T_8c,
                            T_8d, T_9a, T_9i, T_9b, T_9c, T_9d, T_10a, T_10i, T_10b, T_10c, T_10d
                            , T_11a, T_11i, T_11b, T_11c, T_11d, T_12a, T_12i, T_12b, T_12c,
                            T_12d, dP_liquid_net, dP_vapour_net, dP_1a_net, dP_1i_net, dP_1b_net,
                             dP_1c_net, dP_1d_net, dP_2a_net, dP_2i_net, dP_2b_net, dP_2c_net,
                            dP_2d_net, dP_3a_net, dP_3i_net, dP_3b_net, dP_3c_net, dP_3d_net,
                            dP_4a_net, dP_4i_net, dP_4b_net, dP_4c_net, dP_4d_net, dP_5a_net,
                            dP_5i_net, dP_5b_net, dP_5c_net, dP_5d_net, dP_6a_net, dP_6i_net,
                            dP_6b_net, dP_6c_net, dP_6d_net, dP_7a_net, dP_7i_net, dP_7b_net,
                            dP_7c_net, dP_7d_net, dP_8a_net, dP_8i_net, dP_8b_net, dP_8c_net,
                            dP_8d_net, dP_9a_net, dP_9i_net, dP_9b_net, dP_9c_net, dP_9d_net,
                            dP_10a_net, dP_10i_net, dP_10b_net, dP_10c_net, dP_10d_net,
                            dP_11a_net, dP_11i_net, dP_11b_net, dP_11c_net, dP_11d_net,
                            dP_12a_net, dP_12i_net, dP_12b_net, dP_12c_net, dP_12d_net])
1908                filewriter.close()
1909
1910                nine_months = True
1911
1912            elif t >= (60.*60.*24.*30.*12.) and twelve_months == False:
1913                Boil_off_monthly = (m_boil_off/m_liquid_initial/t*60.*60.*24.*30.)*100.
```

```
1914                with open('mass_and_boil-off_v15-12m.csv', 'a') as filewriter:
1915                    writer = csv.writer(filewriter)
1916                    writer.writerow([t_SOFI, m_SOFI, layer_density, layers, m_MLI, Q_cooler,
                            m_cooler, prop.thickness, m_shell, m_boil_off, m_vent, m_boil_off+
                            m_SOFI+m_MLI+m_shell+m_cooler, Boil_off_monthly, t_boil, (t
                            /60./60./24.), T_liquid, T_vapour, T_1a, T_1i, T_1b, T_1c, T_1d, T_2a
                            , T_2i, T_2b, T_2c, T_2d, T_3a, T_3i, T_3b, T_3c, T_3d, T_4a, T_4i,
                            T_4b, T_4c, T_4d, T_5a, T_5i, T_5b, T_5c, T_5d, T_6a, T_6i, T_6b,
                            T_6c, T_6d,  T_7a, T_7i, T_7b, T_7c, T_7d, T_8a, T_8i, T_8b, T_8c,
                            T_8d, T_9a, T_9i, T_9b, T_9c, T_9d, T_10a, T_10i, T_10b, T_10c, T_10d
                            , T_11a, T_11i, T_11b, T_11c, T_11d, T_12a, T_12i, T_12b, T_12c,
                            T_12d, dP_liquid_net, dP_vapour_net, dP_1a_net, dP_1i_net, dP_1b_net,
                             dP_1c_net, dP_1d_net, dP_2a_net, dP_2i_net, dP_2b_net, dP_2c_net,
                            dP_2d_net, dP_3a_net, dP_3i_net, dP_3b_net, dP_3c_net, dP_3d_net,
                            dP_4a_net, dP_4i_net, dP_4b_net, dP_4c_net, dP_4d_net, dP_5a_net,
                            dP_5i_net, dP_5b_net, dP_5c_net, dP_5d_net, dP_6a_net, dP_6i_net,
                            dP_6b_net, dP_6c_net, dP_6d_net, dP_7a_net, dP_7i_net, dP_7b_net,
                            dP_7c_net, dP_7d_net, dP_8a_net, dP_8i_net, dP_8b_net, dP_8c_net,
                            dP_8d_net, dP_9a_net, dP_9i_net, dP_9b_net, dP_9c_net, dP_9d_net,
                            dP_10a_net, dP_10i_net, dP_10b_net, dP_10c_net, dP_10d_net,
                            dP_11a_net, dP_11i_net, dP_11b_net, dP_11c_net, dP_11d_net,
                            dP_12a_net, dP_12i_net, dP_12b_net, dP_12c_net, dP_12d_net])
1917                filewriter.close()
1918
1919                twelve_months = True
1920
1921            elif t >= (60.*60.*24.*30.*18.) and eighteen_months == False:
1922                Boil_off_monthly = (m_boil_off/m_liquid_initial/t*60.*60.*24.*30.)*100.
1923                with open('mass_and_boil-off_v15-18m.csv', 'a') as filewriter:
1924                    writer = csv.writer(filewriter)
1925                    writer.writerow([t_SOFI, m_SOFI, layer_density, layers, m_MLI, Q_cooler,
                            m_cooler, prop.thickness, m_shell, m_boil_off, m_vent, m_boil_off+
                            m_SOFI+m_MLI+m_shell+m_cooler, Boil_off_monthly, t_boil, (t
                            /60./60./24.), T_liquid, T_vapour, T_1a, T_1i, T_1b, T_1c, T_1d, T_2a
                            , T_2i, T_2b, T_2c, T_2d, T_3a, T_3i, T_3b, T_3c, T_3d, T_4a, T_4i,
                            T_4b, T_4c, T_4d, T_5a, T_5i, T_5b, T_5c, T_5d, T_6a, T_6i, T_6b,
                            T_6c, T_6d,  T_7a, T_7i, T_7b, T_7c, T_7d, T_8a, T_8i, T_8b, T_8c,
                            T_8d, T_9a, T_9i, T_9b, T_9c, T_9d, T_10a, T_10i, T_10b, T_10c, T_10d
                            , T_11a, T_11i, T_11b, T_11c, T_11d, T_12a, T_12i, T_12b, T_12c,
                            T_12d, dP_liquid_net, dP_vapour_net, dP_1a_net, dP_1i_net, dP_1b_net,
                             dP_1c_net, dP_1d_net, dP_2a_net, dP_2i_net, dP_2b_net, dP_2c_net,
                            dP_2d_net, dP_3a_net, dP_3i_net, dP_3b_net, dP_3c_net, dP_3d_net,
                            dP_4a_net, dP_4i_net, dP_4b_net, dP_4c_net, dP_4d_net, dP_5a_net,
                            dP_5i_net, dP_5b_net, dP_5c_net, dP_5d_net, dP_6a_net, dP_6i_net,
                            dP_6b_net, dP_6c_net, dP_6d_net, dP_7a_net, dP_7i_net, dP_7b_net,
                            dP_7c_net, dP_7d_net, dP_8a_net, dP_8i_net, dP_8b_net, dP_8c_net,
                            dP_8d_net, dP_9a_net, dP_9i_net, dP_9b_net, dP_9c_net, dP_9d_net,
                            dP_10a_net, dP_10i_net, dP_10b_net, dP_10c_net, dP_10d_net,
                            dP_11a_net, dP_11i_net, dP_11b_net, dP_11c_net, dP_11d_net,
                            dP_12a_net, dP_12i_net, dP_12b_net, dP_12c_net, dP_12d_net])
1926                filewriter.close()
1927
1928                eighteen_months = True
1929
1930            elif t >= (60.*60.*24.*30.*24.) and twentyfour_months == False:
1931                Boil_off_monthly = (m_boil_off/m_liquid_initial/t*60.*60.*24.*30.)*100.
1932                with open('mass_and_boil-off_v15-24m.csv', 'a') as filewriter:
1933                    writer = csv.writer(filewriter)
1934                    writer.writerow([t_SOFI, m_SOFI, layer_density, layers, m_MLI, Q_cooler,
                            m_cooler, prop.thickness, m_shell, m_boil_off, m_vent, m_boil_off+
                            m_SOFI+m_MLI+m_shell+m_cooler, Boil_off_monthly, t_boil, (t
                            /60./60./24.), T_liquid, T_vapour, T_1a, T_1i, T_1b, T_1c, T_1d, T_2a
                            , T_2i, T_2b, T_2c, T_2d, T_3a, T_3i, T_3b, T_3c, T_3d, T_4a, T_4i,
                            T_4b, T_4c, T_4d, T_5a, T_5i, T_5b, T_5c, T_5d, T_6a, T_6i, T_6b,
                            T_6c, T_6d,  T_7a, T_7i, T_7b, T_7c, T_7d, T_8a, T_8i, T_8b, T_8c,
                            T_8d, T_9a, T_9i, T_9b, T_9c, T_9d, T_10a, T_10i, T_10b, T_10c, T_10d
                            , T_11a, T_11i, T_11b, T_11c, T_11d, T_12a, T_12i, T_12b, T_12c,
                            T_12d, dP_liquid_net, dP_vapour_net, dP_1a_net, dP_1i_net, dP_1b_net,
                             dP_1c_net, dP_1d_net, dP_2a_net, dP_2i_net, dP_2b_net, dP_2c_net,
                            dP_2d_net, dP_3a_net, dP_3i_net, dP_3b_net, dP_3c_net, dP_3d_net,
                            dP_4a_net, dP_4i_net, dP_4b_net, dP_4c_net, dP_4d_net, dP_5a_net,
```

```
                        dP_5i_net, dP_5b_net, dP_5c_net, dP_5d_net, dP_6a_net, dP_6i_net,
                        dP_6b_net, dP_6c_net, dP_6d_net, dP_7a_net, dP_7i_net, dP_7b_net,
                        dP_7c_net, dP_7d_net, dP_8a_net, dP_8i_net, dP_8b_net, dP_8c_net,
                        dP_8d_net, dP_9a_net, dP_9i_net, dP_9b_net, dP_9c_net, dP_9d_net,
                        dP_10a_net, dP_10i_net, dP_10b_net, dP_10c_net, dP_10d_net,
                        dP_11a_net, dP_11i_net, dP_11b_net, dP_11c_net, dP_11d_net,
                        dP_12a_net, dP_12i_net, dP_12b_net, dP_12c_net, dP_12d_net])
1935                filewriter.close()
1936
1937                twentyfour_months = True
1938
1939    else:
```

Appendices list of References

[1] S. M. Richardson, “NEWTONIAN FLUIDS”, in *A-to-Z Guide to Thermodynamics, Heat and Mass Transfer, and Fluids Engineering*, Begellhouse. DOI: 10.1615/atoz.n.newtonian_fluids. [Online]. Available: https://doi.org/10.1615/atoz.n.newtonian_fluids.

[2] J. J. D. Anderson, “Governing Equations of Fluid Dynamics”, *Computational Fluid Dynamics*, pp. 15–51, 2009. DOI: 10.1007/978-3-540-85056-4_2. [Online]. Available: http://link.springer.com/content/pdf/10.1007/978-3-540-85056-4%7B%5C_%7D2.pdf.

[3] N. Hall, *Euler equations*, https://www.grc.nasa.gov/www/k-12/airplane/eulereqs.html, accessed: 2019-06-17, NASA Glenn Research Center, 2015.

[4] D. D. Gray and A. Giorgini, “The validity of the boussinesq approximation for liquids and gases”, *International Journal of Heat and Mass Transfer*, vol. 19, no. 5, pp. 545–551, 1976, ISSN: 00179310. DOI: 10.1016/0017-9310(76)90168-X.