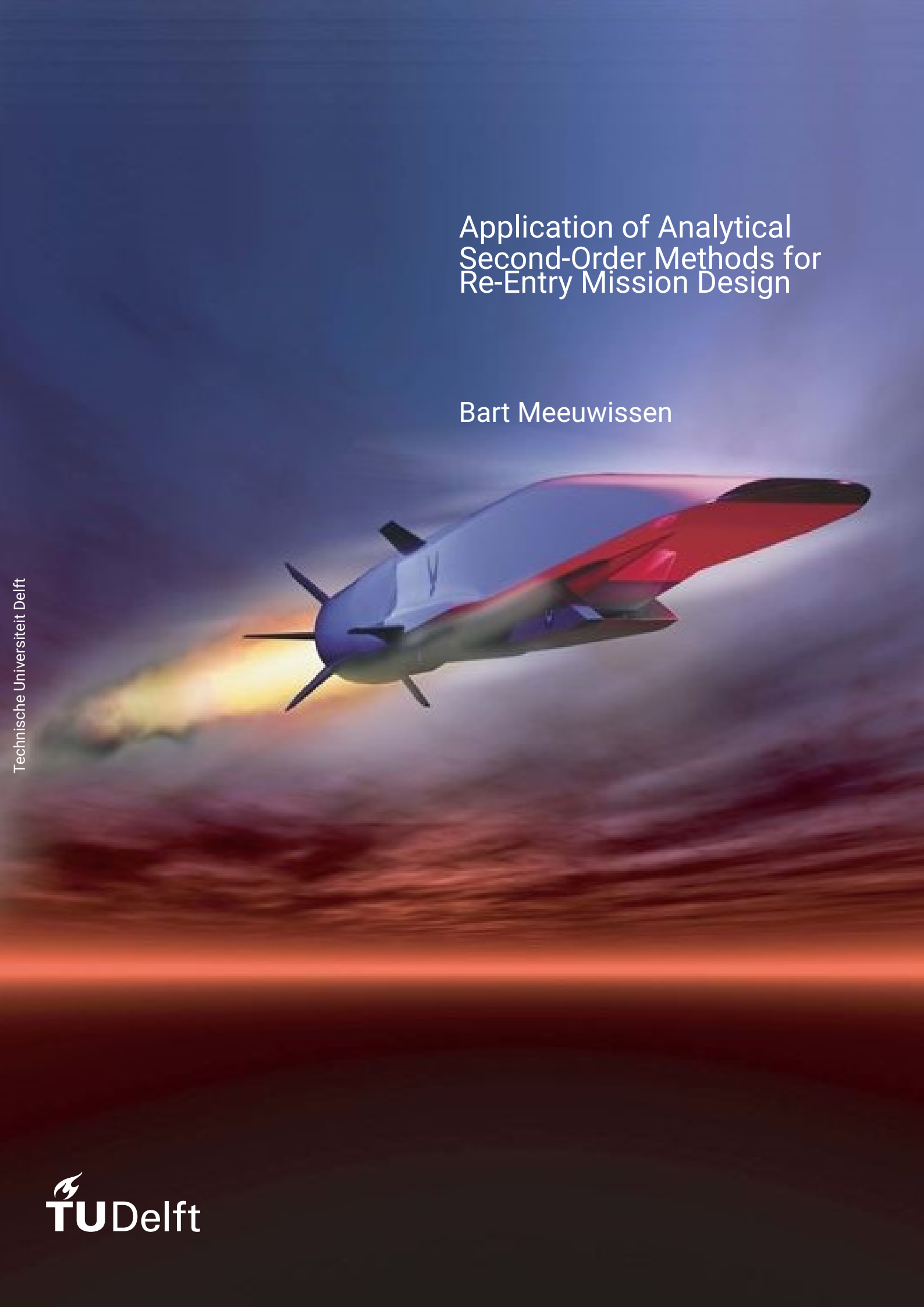
Application of Analytical Second-Order Methods for Re-Entry Mission Design

Bart Meeuwissen



Technische Universiteit Delft

TU Delft

Application of Analytical Second-Order Methods for Re-Entry Mission Design

by

Bart Meeuwissen

to obtain the degree of Master of Science at the Delft University of Technology
to be defended publicly on Friday February 26, 2021 at 13:30 AM.

Student number:		4366557
Thesis committee:	Prof. dr. ir. P.N.A.M. Visser	TU Delft, chair
	Dr. ir. E. Mooij	TU Delft, supervisor
	Dr. ir. E. van Kampen	TU Delft

Cover image credit: Artist rendition of the Boeing X-51A Waverider
Available at:https://www.afmc.af.mil/News/Photos/igphoto/2000361080/mediaid/75347/

TU Delft

Preface

The past months have led up to this point where I conclude my time as a student. The research that I conducted in order to obtain my Master of Science degree is presented in the pages ahead. I look back at the past years with great joy and I am grateful for the many good memories and friends that I made. Now that I conclude this phase of my life, I would like to thank a few people.

First of all, I would like to thank Dr. ir. Erwin Mooij, who supervised my thesis work through our weekly meetings. Having had the pleasure of his supervision for both my Bachelor and Master theses has shown that his sense of humor made working with him very enjoyable, Aside from that, he has made me a better engineer through his critical questions and his infamous red pen, that he used to provide pages full of feedback for every deliverable. Finally, although the past year required most meetings to take place online due to the current regulations, it was always a surprise to find the Skype background he used for a meeting, oftentimes being one from a movie we both enjoy.

Next, I would like to thank my friends for making my student time much better. As most of us were living in Delft, there were always people around to study with, go to lectures with or simply spend our free time together. Even in the past months, we regularly met up online to spend time together during pub quizzes and other online activities.

Also, special thanks for my colleagues at Albert Heijn, where I worked in the past year next to working on my thesis project. Especially during this year, where restrictions took away many of the things one would otherwise do outside of studying, the work at Albert Heijn proved a welcome change to the long days working on my thesis. Their enthusiasm and professionality have taught me how to work in a large company and the trust from my manager Vincent van Dijk has given me the opportunity to work on a number of impactful projects that can now be seen in Albert Heijn and Gall&Gall stores throughout the country.

I would like to thank my parents and brother for their continuous support throughout the years and for always being there for me. Finally, I would like to thank my girlfriend for bearing with me through the past years and ensuring that I would not get lost in my engineering world.

Bart Meeuwissen
Delft, January 2020

Abstract

Atmospheric re-entry remains an active field of study as it is defined by extremes and a requirement for safe return from space missions. During re-entry, a vehicle experiences high deceleration and heating loads as it tries to shed its velocity before reaching the Earth. Mission design still plays an important role in these problems, as the vehicle has to safely find its path to the Earth while satisfying its operational constraints. Presently, focus has shifted towards numerical methods for mission design, which are known to be accurate and flexible. However, these methods are computationally intensive and less suitable for on-board optimization of the trajectory. The focus of this thesis lies on testing second order analytical methods for their application in re-entry mission design. The equations have been proven to provide accurate results, although have not found wide application in literature. The objectives of the optimization are the maximization of flight range and minimization of the integrated heat load over the course of the trajectory, while remaining within the operational constraints for heat flux and g-load. The research goal of this thesis is formulated as: To what extent can optimal two-dimensional re-entry trajectories be developed using a second-order analytical method with the objectives of maximizing the flight range and minimizing the heat load, while adhering to operational constraints?

To answer this question both a numerical and analytical simulator are developed and compared. The main goal for both simulators is the flexibility and ability to test various guidance profiles that are given by the optimization algorithms. In addition, two new guidance algorithms that have not been previously applied to re-entry are tested, namely a newly developed variable chromosome length algorithm, SCEA, and the Multi-Objective Hypervolume Ant Colony Optimizer.

Results have shown that the second order solutions have significant drawbacks in their implementation for mission design. The method for gliding flight prescribes the entry state of the vehicle, thus allowing very little flexibility. Additionally, the entry velocity that is prescribed becomes so high for the winged HORUS-2B reference vehicle that numerical simulations results in skipping flight, whereas the analytical method still finds monotonically decreasing altitudes. For tests with a sample return capsule, accurate trajectories are found, given the limitations in the entry state.

The skipping method provides a different problem, where returned values might have imaginary components. Still, the obtained trajectories provide good results and would be applicable to re-entry mission design.

Optimization results show that the limitation in the entry state for the analytical methods for gliding flight provide a range of solutions that converges towards a single optimal point. Without an adaptation of the method, these results are thus not directly applicable for re-entry mission design.

The results for numerical simulations against a reference trajectory of HORUS-2b were still performed to test the performance of the SCEA and MHACO algorithms. SCEA proved to provide competitive results against the MOEA/D algorithm, although for a slightly worse convergence rate and a larger computational requirement in terms of function evaluations.

MHACO required an adaptation of the objectives and constraints of the method to ensure that the hypervolumes would not be dominated by one of the objectives or constraints. After this adaptation, MHACO proved to be more efficient for an equal population size compared to MOEA/D, although the Pareto fronts that were found were slightly less optimal.

List of Symbols

Greek

α	Angle of attack	[rad]
γ	Flight-path angle	[rad]
δ	Geographic latitude	[rad]
θ	Range angle	[rad]
λ	Non-dimensional lift over drag (Skip)	[-]
μ	Gravitational parameter	$[m^3/s^{-2}]$
ρ	Air Density	$[kg/m^3]$
σ	Bank angle	[rad]
τ	Geographic longitude	[rad]
τ	Non-dimensional range angle	[-]
ϕ	Non-dimensional flight path angle	[-]
χ	Heading angle	[rad]
ω	Rotational rate	[rad/s]

Roman

a	acceleration	$[m/s^2]$
B, E^*, k	Non-dimensional parameters	[-]
C_D	Drag coefficient	[-]
C_L	Lift coefficient	[-]
C_S	Side force coefficient	[-]
D	Drag force	[N]
E	Energy	[J]
E	Non-dimensional lift over drag (Glide)	[-]
F	Force	[N]
g	Gravitational acceleration	$[m/s^2]$
h	Altitude	[m]
H_s	Scale height	[m]
L	Lift force	[N]
m	Mass	[kg]
M	Molecular mass	[]kg/kmole]
M	Mach number	[-]
n_g	G-load	[-]
p	Pressure	[Pa]
p	Constraint violation	[-]
$\bar{q}$	Dynamic Pressure	[Pa]
q	Heat flux	$[W/m^2]$
Q	Heat load	$[J/m^2]$
r	Relative position	[m]
R^*	Universal gas constant	[8314.32 J/kmole K]
R	Radius	[m]
S	Area	$[m^2]$
S	Side force	[N]
T	Temperature	[K]
t	Time	[s]
T_s	Sampling time	[s]
u	Non-dimensional velocity	[-]
v	Non-dimensional velocity	[-]
V	Velocity	[m/s]
x	Non-dimensional altitude	[-]

Subscripts

0	Sea-level conditions
cb	Central body
E	Entry conditions

List of Abbreviations

LEO	Low Earth Orbit.
MHACO	Multi-Objective Hypervolume-Based Ant Colony Optimizer.
MOEA/D	Multi-Objective Evolutionary Algorithm with Decomposition.
NUIN	Non-uniform Independent Node Control.
SCEA	Structured Chromosome Evolutionary Algorithm.
SRM	Sample Return Mission.
TBD	To Be Determined.
USSA	United States Standard Atmosphere.

Contents

1 Introduction

Since its early days, humankind has wondered how life on Earth originated. This curiosity has compelled humankind to explore and expand its boundaries throughout its history, according to Olson et al. (2012). This led to the discovery of new continents and the development of many of the modes of transportation we think common today. The most recent of these developments is the exploration of space with the goal of helping us understand how our planet, the Solar System and the universe work.

Currently, most space missions are executed by satellites and robots, which return the data so that it can be studied back on Earth. However, recent initiatives have returned the focus to human exploration of the 'nearby' region of space. A prime example of this is NASA's Artemis program, which aims to return to the Moon and build a sustainable base (Smith et al., 2020).

An inherent design problem for human space exploration arises as the astronauts will have to return to Earth at some point. This atmospheric entry process carries extreme requirements for the vehicle carrying the astronauts. As the vehicle typically re-enters the Earth's atmosphere at velocities of over 7 km/s it experiences immense deceleration loads and aerodynamic heating. Of course, the main goal of re-entry missions is to safely return the vehicle and its crew to the surface of the Earth, which can only be guaranteed by controlling the vehicle throughout its flight.

This leads to three main elements in the design of the vehicle and its software: the guidance, navigation and control. The guidance algorithm prescribes the planned path of the vehicle. The navigation system determines the exact state of the vehicle over the course of its trajectory. Lastly, the control system steers the vehicle in an effort to make it match the planned path from the guidance.

This thesis will focus on the guidance algorithm for re-entry. Over the years, a lot of research has been performed in this field. Initially, analytical equations such as the ones developed by Chapman (1960) were used to guide the vehicles, due to their low computational effort. Up until the Space Shuttle , this was the preferred method (Graves and Harpold, 1979). In more recent years, the focus has shifted towards numerical methods for guidance, such as the method proposed by Lu et al. (2017) for the Orion crew capsule. Although these numerical trajectories are more accurate compared to analytical results, they also impose a more stringent requirement on the computational power that is required.

The numerical methods also perform well for heuristic optimization of the re-entry mission as was proven by among others Papp (2014). From the methods that were selected by Papp (2014), the MOEA/D algorithm was shown to provide the best results. However, new algorithms have been introduced since, such as the Multi-Objective Hypervolume-based Ant Colony Optimizer, which was shown to perform well for several problems by Acciarini (2020). Another interesting development is the use of a variable length chromosome as applied to interplanetary trajectory planning by Nyew et al. (2012). As optimization of re-entry missions often requires the selection of the number of guidance node, it might be possible to circumvent this design step by letting the variable chromosome length algorithm determine the number of guidance node on its own.

Both the analytical and numerical methods for trajectory design can be used to generate a reference trajectory for re-entry vehicles. These trajectories have to adhere to a number of constraints that are defined based on either the vehicle capabilities or the payload or crew. Still, within these constraints many trajectories remain with different properties. Typically, a trade-off has to be made between the flight range of the

vehicle and the heat load that is built up over the course of the trajectory. As one would expect, these two objectives are conflicting, i.e., when one improves, the other deteriorates. Numerical mission design has been studied widely in combination with global optimization methods to find Pareto optimal trajectories, within the given constraints, to find the possible trade-offs. However, analytical methods have barely been studied in this context, although they have the benefit of requiring much less computer power. One of the most accurate analytical methods is found in Vinh et al. (1993). The focus of this thesis lies on testing this analytical method for trajectory optimization and the following research goal can be formulated:

To what extent can optimal two-dimensional re-entry trajectories be developed using an analytical second-order method with the objectives of maximizing the flight range and minimizing the heat load, while adhering to operational constraints?

To answer this question, a number of sub-questions can be formulated that support the research goal and further elaborate on it.

1. *To what extent can the second order analytical solutions be used to generate an optimal guidance profile for re-entry?*
2. *How do results of the second-order methods for skipping and gliding flight compare for use in an optimization problem?*
3. *What differences are present in the Pareto optimal set of numerical and second order solutions?*
4. *How does Ant Colony Optimization compare to the previously used MOEA/D algorithm for re-entry trajectory optimization?*
5. *How does a variable chromosome length algorithm compare to the traditional fixed chromosome length optimization algorithms?*
6. *How does the increased complexity of a variable chromosome length weigh against the circumvention of node selection?*

The combination of these sub-questions will provide an answer to the main question of this thesis. First of all, the analytical method should be studied to get an idea of the applicability of this method for re-entry flight. Secondly, two new optimization methods were identified that provide interesting results in literature. These are the MHACO algorithm as found in Acciarini et al. (2020) and a variable chromosome length algorithm based on the MOEA/D algorithm, which is developed in this thesis and based on Dasgupta and McGregor (1992b). This last algorithm is of particular interest as it might allow circumvention of the node selection that is typically included in the design of the optimization problem.

To answer these questions, a re-entry simulator was developed with the goal of optimization of numerical and analytical trajectories. Two reference vehicles are selected for this simulator. Firstly, HORUS-2B was selected from Mooij (1995). HORUS is a winged vehicle similar in geometry to the Space Shuttle capable of both skipping and gliding flight. The availability of detailed information on its aerodynamics was the main reason for selecting this vehicle. The aerodynamic data can be found in Mooij (1995). A verified trajectory is available from Mooij (1998) and was used for the verification of the re-entry simulator.

Secondly, a capsule was selected based on the Apollo Command Module. The capsule was obtained from Engelsma (2019) and selected for its performance alongside the second-order analytical trajectories.

The vehicles perform an unpowered re-entry into the atmosphere and thus their trajectories are only influenced by the external forces as a consequence of their instantaneous environment. For re-entry, the dominating forces are the aerodynamic and gravitational forces. The developed simulator uses the United States Standard Atmosphere 1976 model and a gravitational model that includes only the J_2 term in addition to the central gravity field. Lastly, the shape of the Earth is considered to be a rotating sphere.

The trajectories in the simulator are influenced by the vehicle's attitude that is defined by two aerodynamic angles, the angle of attack and the bank angle. The third attitude angle, the angle of sideslip, is assumed to be zero and its effects neglected. The angle of attack influences the lift and drag coefficients of the vehicle and the bank angle influences the direction of the lift coefficient. The guidance defines these angles throughout the trajectory with respect to the normalized specific energy of the vehicle. This normalized energy takes a value of 1 at entry conditions and 0 when the vehicle reaches a standstill on the surface of the

Earth. The main benefit of using the normalized specific energy is that its limits are found at the ends of the trajectory, but also that its value is monotonically decreasing. It is also much less sensitive to disturbances and more flexible than time.

The analytical methods on the other hand take a simplified approach to the re-entry problem. They simply use the relation between the maximum available lift force and the instantaneous lift force to formulate a trajectory. The analytical simulator is capable of setting a value for the angle of attack at the start of a trajectory segment, to still allow for an angle of attack profile along the bank angle profile which can be deduced from the ratio between the maximum lift compared to the instantaneous vertical lift.

The optimization problem then tries to find a guidance profile for the simulators that optimize the trajectory for its flight range and heat load, while adhering to the operational constraints. The goal is to compare the guidance profiles that are obtained through the analytical simulator to the Pareto front that is obtained from the numerical simulator.

This document provides a background in the theory that is required for the optimization of the trajectories, specifies the software design and discusses the results that were obtained. First, a background on re-entry is given in Chapter 2, followed by a description of the flight dynamics models that were used in the simulator in Chapter 3. The selected optimization methods are discussed in Chapter 4. The numerical methods that were used are described in Chapter 5 and the software that was used can be found in Chapter 6. A study of the analytical methods is found in Chapter 8. The optimization of the trajectories is discussed in Chapter 9. Finally, the conclusions and recommendations can be found in Chapter 10.

I

Part I: Background

2

Mission Heritage

This chapter will discuss previous missions in the field of re-entry flight. It serves to provide a background on re-entry missions and its constraints, as well as providing the necessary background of the reference vehicles that are used.

2.1. The Re-Entry Problem

A definition of the re-entry problem assists in understanding the problem that is posed in this research. Atmospheric entry missions seek to transfer an object from the space environment, with typically high velocities, to the planet's surface taking into account high aerodynamic loads and heating over its trajectory. This poses the main problem for re-entry flight. The conditions are harsh, while safety is paramount for the (often human) cargo. This thesis will seek to improve on the safety based on the trajectory design for these missions and thus the re-entry problem will be described in this context.

2.1.1. Trajectory types

Typically, three types of re-entry trajectories can be distinguished. First, ballistic entry will be considered, then gliding entry and finally skipping entry. The trajectories are visualized in Figure 2.1.

Ballistic Entry

The simplest form of entry flight is ballistic entry. In this case, the vehicle enters the atmosphere at a steep angle and maintains this angle for the largest part of the flight. This results in a typically short flight with a lift force negligible compared to the drag force. In reality, capsules fly semi-ballistic trajectories as they still produce a small lift force to control the trajectory. Due to the steep entry angle, the vehicle maintains a large part of its velocity when entering the denser layers of the atmosphere and thus experiences high heating and g-loads.

Gliding Entry

In contrast to the ballistic entry, gliding entry maintains a small entry angle. To maintain this small angle, the vehicle is required to have a high lift-over-drag ratio, making winged entry vehicles well suited for this kind of mission. This results in lower velocities when entering the denser layers of the atmosphere and thus lower heating and g-loads. Additionally, the flight range is much larger and can be more accurately controlled.

Skipping Entry

The last type of entry flight is skipping entry. For this type of entry flight, the altitude does not monotonically decrease over time, resulting in 'skips'. If these skips do not cause the vehicle to leave the atmosphere, the trajectory can be more accurately called 'lofting'. This type of trajectory is possible for both high and low values for the lift-to-drag, given that the velocity is high enough. The main benefit of this trajectory type is the large increase in flight range and the possibility to shed some of its velocity during the skips, again resulting in lower velocities when the vehicle enters the dense layers of the atmosphere. By performing these skipping maneuvers and thus shedding velocity in the upper regions of the atmosphere, the thermo-mechanical loads on the vehicle are also reduced with the additional benefit of increasing the flight range.

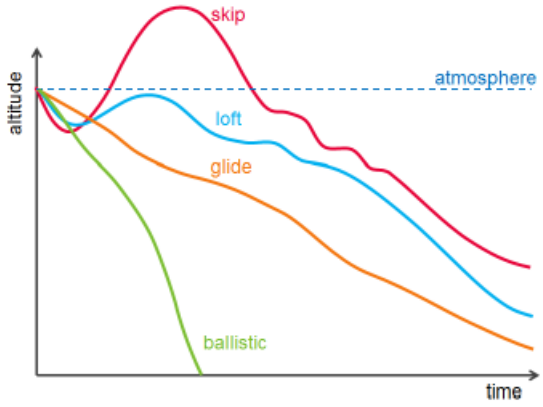


Figure 2.1: Schematic representation of the four types of trajectories available to re-entry vehicles (Papp, 2014).

2.1.2. Constraints

A mission's requirements are largely defined by its constraints. These constraints are formulated based on the limitations of both vehicle and crew, as well as its intended landing point. These constraints are the aerodynamic load, aerodynamic heating, control of the landing point and the capture withing the atmosphere (Graves and Harpold, 1970). It should be noted that the constraint for landing point control will not be further discussed here, as the objective of the optimization problem is a trade-off between flight range and heat load and thus targeting is not applicable.

Load Constraints

Upon entering the atmosphere, the main influence on the motion of the vehicle shifts from gravitational forces to aerodynamic forces. The load on the vehicle is a consequence of the lift, drag and side force, the latter of which is much smaller than the other two. The resultant of the lift and drag force results in an acceleration that is proportional to the mass of the vehicle. The g-load that is experienced by the vehicle then only needs to be normalized against the sea-level gravitational acceleration:

$$n_g = \frac{1}{mg_0}\sqrt{L^2 + D^2} \tag{2.1}$$

Heating Constraints

The heating constraints on the vehicle are present in two manners: the total heat load and the maximum heating rate. Heating occurs due to the friction between the vehicle and the atmosphere and the phenomenon is used to shed the high levels of kinetic energy that the vehicle possesses at the start of the trajectory. This friction results in the heating of the vehicle and in its deceleration. The total heat load equals the integral of the heating rate over the trajectory and for optimization problems it is typically a problem objective. The heating rate on the other hand is the instantaneous rate at which heat is transferred to the vehicle and has a limit value based on the vehicle design. Determining the heating rate is complicated as it depends on the hypersonic properties of the vehicle. Chapman (1960) provides a useful approximation to determine the heating rate q_c:

$$q_c = c^* \frac{1}{R_N^n}\left(\frac{\rho}{\rho_0}\right)^{1-n}\left(\frac{V}{V_c}\right)^m \tag{2.2}$$

where R_N is the nose radius of the vehicle, ρ the local atmospheric density with subscript zero denoting the value at sea-level, V is the velocity and V_c the circular velocity, defined as 7905.4m/s. The constants are defined as follows: $c^* = 1.1097 \cdot 10^8$, $n = 0.5$, $m = 3.15$. The exact determination of these constants goes beyond the scope of this document and it should suffice to say that these constants describe the airflow characteristics. Note that the above relation only holds under the assumption of a cold wall.

Atmospheric Capture Constraints

The atmospheric capture constraint is typically used for gliding flight, where it implies that no exit from the atmosphere may be present. As this research includes methods for skipping flight, the capture constraint

does not hold for all elements of the flight. It should be included for the leg which was determined to be the final entry of the vehicle into the atmosphere. It can be applied simply by applying a penalty to the problem constraints when the vehicle leaves the atmosphere, which is defined to occur above 100 km in the implemented atmosphere models.

2.2. Reference Mission

The selected reference mission was simulated for HORUS-2B using a high fidelity simulator and selected for its good comparability to previous research. The mission was previously used as a reference for Papp (2014) and developed by Mooij (1998). The guidance nodes are defined as:

$$\Gamma_{\text{HORUS, nom}} = \begin{bmatrix} t_1 & \alpha_1 & \sigma_1 \\ t_2 & \alpha_2 & \sigma_2 \\ t_3 & \alpha_3 & \sigma_3 \\ t_4 & \alpha_4 & \sigma_4 \\ t_5 & \alpha_5 & \sigma_5 \\ t_6 & \alpha_6 & \sigma_6 \\ t_7 & \alpha_7 & \sigma_7 \end{bmatrix} = \begin{bmatrix} 0s & 40^\circ & 0^\circ \\ 264s & 40^\circ & 0^\circ \\ 290s & 40^\circ & 79.6^\circ \\ 554s & 40^\circ & 59.8^\circ \\ 686s & 40^\circ & 59.8^\circ \\ 924s & 40^\circ & 59.8^\circ \\ 1319s & 11.5^\circ & 54^\circ \end{bmatrix} \tag{2.3}$$

and the mission parameters are given in Table 2.1.

Table 2.1: Reference mission parameters for HORUS-2B

	Value	**Unit**
Mass	26029.0	[kg]
Reference area	110.0	[m^2]
Nose radius	0.8	[m]
Entry interface		
Altitude	122.0	[km]
Longitude	-106.7	[deg]
Latitude	-22.3	[deg]
Entry velocity	7435.5	[m/s]
Flight path angle	-1.43	[deg]
Heading Angle	70.75	[deg]
Constraints		
Maximum heat flux	500	[kW/m^2]
Maximum g-load	3.0	[-]
Maximum g-load rate	0.5	[s^{-1}]
Maximum alpha rate	5.0	[deg/s]
Maximum sigma rate	15.0	[deg/s]

2.3. Reference Vehicle

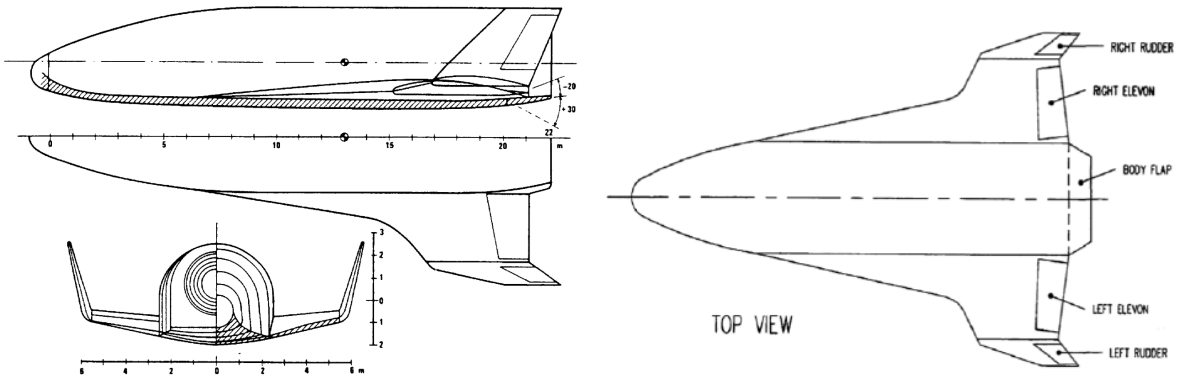
The following section presents the reference vehicles that are used in this thesis. First HORUS-2B will be described, followed by an SRM capsule.

2.3.1. Horus-2B

The first reference vehicle is HORUS-2B. This vehicle was selected for its performance, the availability of its data and its comparability to other winged re-entry vehicles. A full overview of the vehicle characteristics and a database of its aerodynamic characteristics can be found in Mooij (1995). Especially the availability of reference mission data obtained from Mooij (1998) is useful for the verification of the simulator. This reference mission was discussed in Section 2.2.

Geometry

The geometry of HORUS can be observed in Fig. 2.2(a). The winged shape of HORUS allows it to control the trajectory through angle of attack and bank angle modulation. Additionally, skipping flight is possible for re-entry even from LEO.



(a) HORUS-2B geometry.

(b) HORUS control surfaces

Figure 2.2: Lay-out of the HORUS-2B reference vehicle (Mooij, 1995).

HORUS has engines available to provide thrust when used as a second stage for the Sänger vehicle, although for re-entry only the control through the aerodynamic attitude is used.

A description of various parameters is found in Table 2.2.

Control Surfaces

The control surfaces that are incorporated in HORUS-2B are two rudders, two elevons and one body flap. Its locations along the vehicle are shown in Figure 2.2(b). As for the current research only the pitching moment will be taken into account, the effects of the rudders and ailerons can be neglected as they affect the roll and yaw, respectively. This leaves the body flap that is used for trimmed flight in pitch. For low Mach numbers, trimmed flight is no longer possible and one would have to include the rudders and ailerons in the simulation. As this region of flight lies outside of the scope of this thesis, the effects can be safely neglected.

Aerodynamics

The aerodynamic database of HORUS-2B as described in Mooij (1995) gives tabulated values describing the aerodynamic coefficients as a function of the angle of attack and Mach number. Increments of these coefficients are added based on the control surface deflections, which are tabulated with respect to the angle of attack, the Mach number and the deflection angle of the respective control surface.

Some important assumptions and simplifications were made to obtain the tabulated data. These are listed as follows:

- No aeroelastic effects are included.
- The influence of the Reynolds number on the skin-friction drag is included in the drag coefficient rather

Table 2.2: Vehicle dimensions of HORUS-2B

Parameter	**Value**
Length	25 m
Height	4.5 m
Wingspan	13.0 m
Reference wing area	110 m^2
Nose Radius	0.8 m
Empty weight	18394 kg
Gross weight	56196 kg
Main propulsion (HM60)	1055 kN
Orbital Manoeuvre	2*4kN
RCS	14*400 kN

then being explicitly given. This is done through averaged skin friction drag along a standard trajectory to an altitude of 20 km.

- The interference effects of the flaps are not included.

The altitude dependent drag increment is not included as simulations of the vehicle trajectory are terminated around an altitude of 25 km.

The following relations can now be used for the aerodynamic force coefficients of HORUS:

$$\begin{aligned} C_D &= C_{D_0} + \Delta C_{D_b} \\ C_S &= 0 \\ C_L &= C_{L_0} + \Delta C_{L_b} \end{aligned} \tag{2.4}$$

and the aerodynamic moment coefficients are found from:

$$\begin{aligned} C_l &= 0 \\ C_m &= C_{m_0} + \Delta C_{m_b} \\ C_n &= 0 \end{aligned} \tag{2.5}$$

Trim for the HORUS vehicle is achieved by balancing the pitch moment of the clean configuration with a deflection angle of the body flap. The pitch moment of the vehicle is found from:

$$C_m = C_{m_0} + \Delta C_{m_b} \tag{2.6}$$

To maintain trimmed flight it must hold that $C_m = 0$ and thus:

$$\Delta C_{m_b} = -C_{m_0} \tag{2.7}$$

An inverse interpolation is thus required to find the body flap deflection angle that satisfies the trim condition.

2.3.2. SRM Capsule

As a second reference vehicle, a sample return mission (SRM) capsule was selected. The benefit of this second vehicle is that it also allows the study of a smaller and, more importantly, lighter vehicle, while still maintaining a high lift-to-drag ratio.

The selected SRM capsule was taken from Engelsma (2019), as this carries the benefit of comparability of the results for the second order analytical methods. The vehicle is a scaled down version of the Apollo Command Module based on a comparison with the Stardust mission as reported in Boyd et al. (2010) and the Hayabusa mission as described by Davies and Arcadi (2006). This results in a vehicle that is scaled down to 19% of the Apollo Command Module. The SRM vehicle then has a diameter of 0.75 m and from this an aerodynamic reference area of 0.441 m^2 is derived. The ratio between the diameter and nose radius is preserved to find a nose radius of 0.9 m. Lastly, the mass is derived by using a linear relation between the volume and mass of the vehicle, leading to a mass of 38.77 kg. The parameters of this vehicle are summarized in Table 2.3. The geometry of the capsule Apollo capsule that was used as a baseline for the SRM vehicle is presented in Figure 2.3.

Table 2.3: Physical parameters of the SRM capsule reference vehicle.

Vehicle	**Diameter [m]**	**Reference area [m^2]**	**Nose radius [m]**	**Mass [kg]**
SRM	0.75	0.441	0.90	38.77

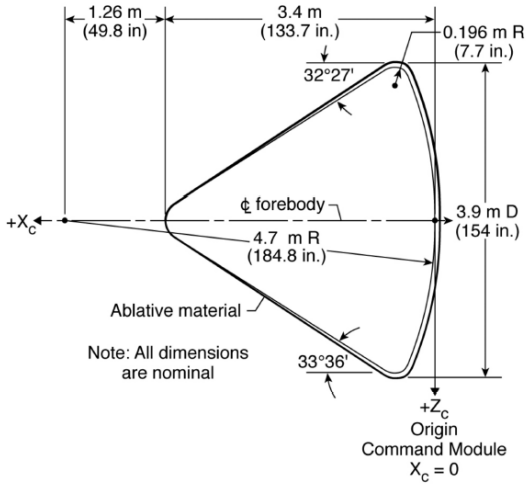


Figure 2.3: Geometery of the Apollo capsule as used for the baseline of the SRM capsule (Robinson et al., 2009).

3

Flight Dynamics

The goal of this research is to study a skipping re-entry into Earth's atmosphere. To conduct this study, the motion and dynamics of the vehicle during the re-entry need to be understood. This chapter will discuss the background required to understand the dynamics that are applied in the simulations used for this research.

As this research will only focus on the study of trajectories, it suffices to determine the translational motion of the center-of-gravity of the vehicle. Thus, the change in attitude during the flight can be ignored, resulting in a three degree of freedom simulation.

To evaluate the motion in this system, the forces acting on the body need to be known. Both the magnitude and direction of the forces need to be known to accurately describe the motion of the vehicle.

This chapter will first describe the required reference frames in Section 3.1, followed by the transformations to describe these frames relative to each other in Section 3.2. The state variables that are required to describe the vehicle state are described in Section 3.3. The equations of motion in their general form are described in Section 3.4 and their analytical form is discussed in Section 3.5. Lastly, the environment models that are used in this research are described in Section 3.6.

3.1. Reference Frames

The reference frames form the basis of the description of a vehicle's motion. The reference frames that are required to study the motion of HORUS are described below.

3.1.1. Inertial Planetocentric Reference Frame (Index I)

The inertial planetocentric reference frame is typically used to create a framework for other reference frames. It is also a useful method to describe the absolute motion of a vehicle in space. The origin of the frame lies at the center of mass of the central body. The effect of the central body moving through space is marginal for entry flight. The $X_I Y_I$-plane coincides with the equatorial plane of the body and the Z_I-axis points towards the north. The reference meridian determines the direction of the X_I-axis and is determined as the zero-longitude meridian at zero time. The Y-axis completes the right-handed system. For Earth, the X_I-axis points to a fixed point in space at a predetermined time. In this research, the J2000 definition will be used, where the X_I-axis points towards the Vernal Equinox, a point in the Aries star sign on 12:00 on January 1, 2000 (Mooij, 2016).

3.1.2. Rotating Planetocentric Reference Frame Index R

The rotating planetocentric reference frame is similar to the inertial frame, except it follows the rotation of the Earth. At $t = 0$ the rotating planetocentric and inertial planetocentric reference frames coincide. Afterwards, the rotating frame follows the Earth rotation with the same angular rate. Again, the Z_R-axis points north, the X_R-axis intersects the equator at zero longitude and the Y_R-axis completes the right handed system.

3.1.3. Body Fixed Reference Frame (Index B)

The body fixed reference frame can be used to describe the forces acting on the body. It also helps to describe other reference frames that are related to the vehicle. The origin of this frame lies at the center of mass of the

vehicle. The $X_B Z_B$-plane coincides with the plane of symmetry of the vehicle. The Z_B-axis points down and the X_B-axis points forward. The Y_B-axis completes the right-handed system.

3.1.4. Vertical Reference Frame (Index V)

The vertical reference frame also originates in the center of mass of the vehicle. The Z_V-axis point towards the center of mass of the central body. The X_V-axis lies in the meridian plane, is perpendicular to the Z-axis and points north. The Y-axis completes the right-handed system. The $X_V Y_V$-plane is also called the local horizontal plane. Strictly speaking, this only holds for perfect spheres and for more accurate Earth models a small error would be introduced.

3.1.5. Trajectory Reference Frame (Index T)

The trajectory of the vehicle can be described using the trajectory reference frame. The X_T-axis coincides with the velocity relative to the atmosphere, the Z_T-axis lies in the vertical plane and the Y_T-axis completes the right-handed system.

3.1.6. Aerodynamic Reference Frame (Index A)

The aerodynamic reference frame is used to describe the attitude of the vehicle and calculate the aerodynamic forces. The X_A-axis lies along the velocity vector relative to the atmosphere, the Z_A-axis is collinear with the aerodynamic lift force and opposite in direction and the Y_A-axis completes the right-handed system.

3.2. Frame Transformations

Together, the reference frames described above are sufficient to describe the state and motion of the vehicle. The required information on the state and trajectory is not available in a single reference frame. To transfer the information between the frames, transformations are required. The following section will describe these transformations.

3.2.1. Euler Angles

The relative orientation of one reference frame to another can be described by three angles. By describing these angles, one can apply these rotations one-by-one and move from any reference frame to any other. These angles are also called Euler angles. The rotations take place along axes of a Cartesian reference frame and can thus be described by unit rotation matrices. The rotation matrices for rotations around the X, Y-, and Z-axes are defined as:

$$\mathbf{R}_X(a_1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha_1 & \sin\alpha_1 \\ 0 & -\sin\alpha_1 & \cos\alpha_1 \end{bmatrix}$$

$$\mathbf{R}_Y(a_2) = \begin{bmatrix} \cos\alpha_2 & 0 & -\sin\alpha_2 \\ 0 & 1 & 0 \\ \sin\alpha_2 & 0 & \cos\alpha_2 \end{bmatrix} \tag{3.1}$$

$$\mathbf{R}_Z(\alpha_3) = \begin{bmatrix} \cos\alpha_3 & \sin\alpha_3 & 0 \\ -\sin\alpha_3 & \cos\alpha_3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3.3. State Variables

Several different sets of state variables are available for this research. The variable sets that are used will be shortly discussed in the following section.

3.3.1. Cartesian Coordinates

The Cartesian coordinate system is one of the simplest, consisting of x-, y- and z-coordinates to describe a three dimensional position. Additionally, to describe the change in position the variables V_x, V_y and V_Z are used. This thus results in the following state vector:

$$\boldsymbol{x} = \{x, y, z, \dot{x}, \dot{y}, \dot{z}\}^T \tag{3.2}$$

To determine the change in the state, the derivatives of these variables are required:

$$\dot{\boldsymbol{x}} = \{\dot{x}, \dot{y}, \dot{z}, \ddot{x}, \ddot{y}, \ddot{z}\}^T \tag{3.3}$$

Although this method is quite simple, it often fails to provide a strong insight into the actual state of the trajectory as it remains abstract.

3.3.2. Spherical Coordinates

A common state variable set for re-entry equations of motion is that of the spherical components. Here, it forms the basis for the analytical equations that will be used. In this set, the position and velocity are defined as two vectors. The position vector has a magnitude r, which is the distance between the centers of mass of the vehicle and the central body. The direction is defined by the latitude, δ, and the longitude, τ. The latitude angle is defined with respect to the equatorial plane, positive in northern direction. The longitude is defined with respect to a reference meridian and positive in eastward direction.

The velocity vector is defined by its magnitude, V, the flight-path angle, γ, and the heading angle, χ. The state derivative vector is now simply the derivative of the state variables. The state and state derivative can thus be defined as:

$$\mathbf{x} = \{r, \tau, \delta, V, \gamma, \chi\} \tag{3.4}$$

$$\dot{\mathbf{x}} = \{\dot{r}, \dot{\tau}, \dot{\delta}, \dot{V}, \dot{\gamma}, \dot{\chi}\} \tag{3.5}$$

Finally, it is important to know the transformation between the spherical components and the cartesian components, which can be found as follows:

$$r = \sqrt{x^2 + y^2 + z^2} \tag{3.6}$$

$$\tau = \text{atan2}(y, x) \tag{3.7}$$

$$\delta = \text{atan2}(z, \sqrt{x^2 + y^2}) \tag{3.8}$$

$$V = \sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2} \tag{3.9}$$

$$\gamma = \text{atan2}(\dot{z}, \sqrt{\dot{x}^2 + \dot{y}^2}) \tag{3.10}$$

$$\chi = \text{atan2}(\dot{y}, \dot{x}) \tag{3.11}$$

3.4. General Formulation of the Equations of Motion

This section describes the equations of motion that are used in the numerical integration of the trajectory of the entry vehicle.

3.4.1. Equations of Motion in Cartesian Coordinates

The equations of motion in the numerical simulator will be propagated using Cartesian coordinates. The equations of motion in this system are simple and are governed by the following equation:

$$\boldsymbol{F} = m\boldsymbol{a} \tag{3.12}$$

where $\boldsymbol{F}$ is the vector sum of the forces and $\boldsymbol{a}$ the resulting acceleration vector.

3.4.2. Equations of Motion in Spherical Coordinates

Although the equations of motion will be used in their Cartesian form for the numerical integration, they also need to be studied in their spherical form, as this forms the basis of the subsequent analytical equations. The definition as given in Mooij (2016) will be used for a rotating Earth including Coriolis, centripetal and centrifugal terms:

$$\begin{aligned}
\dot{r} &= \dot{h} = V\sin\gamma \\
\dot{\tau} &= \frac{V\sin\chi\cos\gamma}{r\cos\delta} \\
\dot{\delta} &= \frac{V\cos\chi\cos\gamma}{r} \\
\dot{V} &= -\frac{D}{m} - g\sin\gamma + \omega_{cb}^2 r\cos\delta(\sin\gamma\cos\delta - \cos\gamma\sin\delta\cos\chi) \\
V\dot{\gamma} &= \frac{L\cos\sigma}{m} - g\cos\gamma + 2\omega_{cb}V\cos\delta\sin\chi + \frac{V^2}{r}\cos\gamma + \\
&\quad + \omega_{cb}^2 R\cos\delta(\cos\delta\cos\gamma + \sin\gamma\sin\delta\cos\chi) \\
V\cos\gamma\dot{\chi} &= \frac{L\sin\sigma}{m} + 2\omega_{cb}V(\sin\delta\cos\gamma - \cos\delta\sin\gamma\cos\chi) + \frac{V^2}{r}\cos^2\gamma\tan\delta\sin\chi + \\
&\quad + \omega_{cb}^2 r\cos\delta\sin\delta\sin\chi
\end{aligned} \tag{3.13}$$

Typically these equations are simplified one step further, which makes them less computationally intensive. This is done by neglecting the terms related to the rotation of the Earth. It should be noted that this affects the accuracy of the equations, especially for re-entry trajectories with a long flight time, such as skipping missions. However, as a baseline for analytical approximations, this is a necessary step. The simplified equations are:

$$\begin{aligned}
\dot{r} &= \dot{h} = V\sin\gamma \\
\dot{\tau} &= \frac{V\sin\chi\cos\gamma}{r\cos\delta} \\
\dot{\delta} &= \frac{V\cos\chi\cos\gamma}{r} \\
\dot{V} &= -\frac{D}{m} - g\sin\gamma \\
V\dot{\gamma} &= \frac{L\cos\sigma}{m} - g\cos\gamma + \frac{V^2}{R}\cos\gamma \\
V\dot{\chi} &= \frac{V^2}{r}\cos^2\gamma\tan\delta\sin\chi
\end{aligned} \tag{3.14}$$

This set of equations provides a clear benefit over the Cartesian representation of the equations of motion, as they are much more intuitive. The equations yield direct changes in their respective state variables, whereas a conversion would be required when the Cartesian state is propagated.

3.5. Analytical Second-Order Method

A more efficient method for solving the equations of motion can be found by implementing an analytical formulation. Here, the methods will be separated for the atmospheric and ballistic phases.

3.5.1. Atmospheric Skip Phase

The analytical method for skipping flight has been selected as developed by Vinh et al. (1993) for its accurate results compared to numerical models, while still providing the benefits of low computational speed that come with analytical methods. The goal is to obtain insights into the trajectory, without the need of computationally intensive numerical models. The selected analytical model shows good results for the (peak) heat load and terminal conditions of the skipping trajectory. The method starts with a first-order derivation, after which terms are added to more accurately describe the trajectory through the second order method.

Second-Order Approximation

The first order approximations was shown to be inaccurate by (Engelsma, 2019). Vinh et al. (1993) continues by formulating a more accurate second order theory based on these first order equations.

First, a new independent variable is introduced in the form of the range angle, which is a measure for the ground distance traveled by the vehicle. The range angle θ can be expressed as:

$$\frac{d\theta}{dt} = \frac{V\cos\gamma}{r} \tag{3.15}$$

After dividing the spherical equations of motion for r, V and γ by this equation, the independent variable is now the range angle instead of time.

The radial position and velocity are now non-dimensionalized. First, the radial position is written as the penetration depth h by subtracting the entry radius and subsequently dividing by the entry radius:

$$h = \frac{r - r_E}{r_E} \tag{3.16}$$

where r is again the instantaneous distance to the center of the Earth and r_E is the distance at the point of entry. This leads to a value of $h = 0$ at entry interface and decreases until the lowest point in the trajectory is reached, after which it increases back to zero until the end of the atmosphere is reached.

The velocity is non-dimensionalized by taking its squared value and dividing it by the square of the circular velocity at the entry interface. Thus:

$$u = \frac{V^2}{V_{c,E}^2} = \frac{V^2}{g_E r_E} \tag{3.17}$$

were $V_{c,E}$ is the circular velocity at the point of entry and g_E is the gravitational acceleration at this point.

The equations of motion are further simplified by introducing three constants. One should take note here that these constants do indeed simplify the equations themselves, but also increase the complexity of understanding the equations themselves. First, the constant E^* is introduced to express the maximum lift-to-drag ratio:

$$E^* = \frac{1}{2\sqrt{KC_{D_0}}} \tag{3.18}$$

where K and C_{D_0} are related by the parabolic drag polar:

$$C_D = C_{D_0} + KC_L^2 \tag{3.19}$$

The second constant, B, is used to describe the entry conditions and vehicle characteristics and is defined as

$$B = \frac{\rho_E S r_E}{2m}\sqrt{\frac{C_{D_0}}{K}} \tag{3.20}$$

where rho_E is the density at the point of entry, S the aerodynamic reference area and m the vehicle mass.

Thirdly, the constant for lift-control is introduced, which allows for lift modulation in the entry problem. This variable, λ is defined as:

$$\lambda = \frac{C_L}{\sqrt{C_{D_0}/K}} \tag{3.21}$$

which relates the lift-to-drag ratio to its maximum value. This leads to a value of λ of 1 for maximum L/D and 0 for ballistic entry. Indirectly, the profile of λ over the course of the flight also describes the angle of attack and bank angle control of a vehicle. It should be noted that this does not hold entirely, as the side force that is induced by the bank angle is not modeled by this method and thus only planar flight is considered.

Now, the independent variable θ is also non-dimensionalized by using the entry radius r_E and scale height H_s:

$$\tau = \sqrt{\frac{r_E}{H_s}}\theta \tag{3.22}$$

where both τ and θ hold a value of zero at the beginning of the trajectory. Another variable, y is introduced to describe the altitude as a function of atmospheric density:

$$y = e^{-\frac{r_E h}{H_s}} = \frac{\rho}{\rho_E} \tag{3.23}$$

It is clear that $y = 1$ at the entry interface and increases to $y_{\max}$ at the lowest point in the skip trajectory. Note that this relation inherently assumes that the density is not zero at the entry interface. For the second order approach, the velocity is non-dimensionalized as:

$$v = \frac{1}{\eta} \ln \frac{V_E^2}{V^2} \tag{3.24}$$

where $v = 0$ holds at the entry interface and v becomes negative as the velocity decreases during the flight. In the equation, V_E is the velocity at the entry interface and V again the instantaneous velocity and the constant η is found to be:

$$\eta = \frac{B}{E^* \sqrt{\frac{r_E}{H_s}}} = \frac{\rho_E S C_{D_0}}{m} \sqrt{H_s r_E} \tag{3.25}$$

Lastly, the flight path angle is non-dimensionalized as ϕ:

$$\phi = -\sqrt{\frac{r_E}{H_s}} \sin\gamma \tag{3.26}$$

Two more constants are introduced by Vinh et al. (1993) to further simplify the equations:

$$k = \frac{2E^*}{\sqrt{\frac{r_E}{H_s}} B} \tag{3.27}$$

$$\alpha = \frac{g_E r_E}{V_E^2} \tag{3.28}$$

The full derivation of the new non-dimensional equations will not be repeated here and the interested reader is referred to Vinh et al. (1993). The final result leads to three exact, non-dimensional equations of motion that match the results of the regular equations of motion. One should note that this system of equations is less intuitive than the regular system due to the introduction of the new variables and constants discussed above. The non-dimensionalized equations for altitude, velocity and flight path angle are, respectively:

$$\frac{dy}{d\tau} = \frac{1+h}{\cos\gamma} y\phi \tag{3.29}$$

$$\frac{dv}{d\tau} = \frac{1+h}{\cos\gamma} \left(1+\lambda^2\right) y - \frac{k\alpha\phi e^{v\eta}}{(1+h)\cos\gamma} \tag{3.30}$$

$$\frac{d\phi}{d\tau} = -(1+h) B\lambda y + \frac{\cos\gamma}{1+h} \alpha e^{\eta v} - \cos\gamma \tag{3.31}$$

The above equations are only a rewritten form of the original equations of motion. The result of these equations still yields the exact solution. Two simplifications are now introduced to be able to obtain analytical equations. First, the flight path angle is assumed to be small and thus $cos(\gamma) \approx 1$. Secondly, it is known that the radius of the Earth is much larger than the change of the altitude during the skipping flight and thus $r - r_E \approx 0$ and $h \approx 0$ and $1 + h \approx 1$. The new parameter k_1 is now introduced as:

$$k_1 = B\lambda \tag{3.32}$$

These simplifications and the constant k_1 leads to a simplification of the above exact equations:

$$\frac{dy}{d\tau} = y\phi \tag{3.33}$$

$$\frac{dv}{d\tau} = (1+\lambda^2) y - k\alpha\phi e^{\eta v} \tag{3.34}$$

$$\frac{d\phi}{d\tau} = -k_1 y - (1 - \alpha e^{\eta v}) \tag{3.35}$$

To eventually find the second order solutions, first-order solutions are first developed. To do so, the variation of the speed in the term $\exp \eta v$ is ignored. This can be done by setting the velocity in this term equal to the entry velocity at all times, leading to $\exp \eta v = 1$ at all points in the trajectory. This process is the same as taking the first order term of a Taylor series expansion with respect to the range angle. The resulting equations are:

$$\frac{dy}{d\tau} = y\phi \tag{3.36}$$

$$\frac{dv}{d\tau} = (1 + \lambda^2) y - k\alpha\phi \tag{3.37}$$

$$\frac{d\phi}{d\tau} = -k_1 y - (1 - \alpha) \tag{3.38}$$

The third equation can now be divided by the first and by using the product rule an explicit equation is obtained. Integrating this equation between $y(0) = 1$ and $\phi(0) = c$, which is an arbitrary point in the trajectory, an expression for ϕ^2 can be obtained against a newly introduced variable x:

$$\phi^2 = c^2 - 2k_1 (e^x - 1) - 2(1 - \alpha) x \tag{3.39}$$

with

$$x = \log y = \frac{r_E h}{H_s} = \frac{r_E - r}{r_E} \tag{3.40}$$

The newly obtained variable x is yet another measure of the altitude in non-dimensional form. The value of x is zero at the entry interface and increases to its maximum value x_{max} at the deepest point in the trajectory. After reaching this point x decreases back to zero until it reaches the entry altitude again. The constant $c = \phi_E$.

Replacement of y in Equation (3.38) by the new variable x and by using Equation (3.39) leads to:

$$d\tau = \frac{dx}{\phi} = \frac{dx}{\pm\sqrt{c^2 - 2k_1 (e^x - 1) - 2(1 - a) x}} \tag{3.41}$$

from which it is possible to obtain an equation that relates x to τ. To do so, an approximation of ϕ^2 as a second order polynomial is required. This simply is:

$$\phi_a^2 = a_1 x^2 + a_2 x + a_3 \tag{3.42}$$

where the subscript a denotes the approximation of ϕ. The expression for x can now be obtained:

$$x = -\frac{a_2}{a_1} - \frac{\sqrt{a_2^2 - 4a_1 a_3}}{2a_2} \sin\left(\sqrt{-a_1}\tau - \beta\right) \tag{3.43}$$

where β is introduced to simplify the equation and is defined as:

$$\beta = \frac{a_2}{\sqrt{a_2^2 - 4a_1 a_3}} \tag{3.44}$$

Vinh et al. (1993) initially state that the coefficients a_1, a_2 and a_3 are to be determined through an approximation scheme. However, eventually an exact solution for these three coefficients is also given:

$$a_1 = \frac{3}{x_1}\left(2(1 - \alpha) - \frac{c^2 + 4k_1}{x_1} + \frac{4k_1 (e^{x_1} - 1)}{x_1^2}\right) \tag{3.45}$$

$$a_1 = -6(1 - \alpha) + \frac{2\left(c^2 + 6k_1\right)}{x_1} - \frac{12k_1 (e^{x_1} - 1)}{x_1^2} \tag{3.46}$$

$$a_3 = c^2 \tag{3.47}$$

where x_1 is the value of x at the point of deepest penetration into the atmosphere. The value of x_1 can be determined by solving Equation (3.39) for $\phi = 0$. Although Vinh et al. (1993) does not propose a specific method to find x_1, Engelsma (2019) proposes a numerical root-finding scheme to find the value of x_1.

Lastly, a first-order approximation of the velocity is required to fully describe the two-dimensional state of the vehicle. This equation is obtained after inverting Equation (3.38), which gives an expression for y. This inverted equation is then substituted in Equation (3.37). This is followed by substituting Equation (3.41) in this intermediate equation and finally integrating the result to an arbitrary point in the trajectory. The first-order equation for the velocity is now obtained as:

$$v = \frac{\left(1+\lambda^2\right)}{k_1}(c-\phi) - k\alpha x - \frac{(1-\alpha)\left(1+\lambda^2\right)}{k)1}\tau \tag{3.48}$$

An important note that goes with this equation is that ϕ is required instead of ϕ^2 and until now only a relation for ϕ^2 was obtained. The solution for ϕ can thus have both positive and negative values based on Equation (3.39). Given the method in which ϕ is normalized, it is the negative sine of the flight-path angle, the sign of ϕ can be determined to be positive for the descending leg of the re-entry flight and positive for the ascending leg. It should be noted that this sign has to be externally added to a propagation of the equations to ensure that the sign changes for both legs of the skip.

The accuracy of the equations can now be improved by taking into account the term $\exp \eta v$. Initially, this is not possible in an analytical manner, but Vinh et al. (1993) propose to use the first two terms of the Taylor series expansion. The term $\exp \eta v$ is then found as $\exp \eta v \approx 1+\eta v$, which allows the formulation of the second-order non-dimensionalized equations of motion:

$$\begin{aligned} \tfrac{dy}{d\tau} &= y\phi \\ \tfrac{dv}{d\tau} &= \left(1+\lambda^2\right) y - k\alpha\phi - \eta k\alpha\phi v \\ \tfrac{d\phi}{d\tau} &= -k_1 y - (1-\alpha) + \eta\alpha v \end{aligned} \tag{3.49}$$

A relation between x and ϕ can now be obtained, after replacing y by x, and integrated to an arbitrary point in the trajectory. The second-order analytical solution for ϕ now is:

$$\phi^2 = c^2 - 2k_1\left(e^x - 1\right) - 2(1-\alpha)x + 2\eta\alpha I(x) \tag{3.50}$$

where the constant $I(x)$ is defined as:

$$I = \frac{\left(1+\lambda^2\right)}{k_1}cx - \frac{1}{2}k\alpha x^2 - \frac{(1-\alpha)\left(1+\lambda^2\right)}{k_1}\tau x - \frac{\left(1+\lambda^2\right)}{k_1}I_1(x) + \frac{(1-\alpha)\left(1+\lambda^2\right)}{k_1}I_2(x) \tag{3.51}$$

here, the additional constants I_1 and I_2 are introduced, which results in:

$$I_1 = \frac{1}{2}\phi_a x - \frac{a_2}{4a_1}\left(c-\phi_a\right) + \frac{\left(4a_1a_3 - a_2^2\right)}{8a_1}\tau \tag{3.52}$$

$$I_2 = -\frac{1}{a_1}\left(c-\phi_a\right) - \frac{a_2}{2a_1}\tau \tag{3.53}$$

The coefficients a_1, a_2 and a_3 return in this relation and are the same as previously defined for the first-order solutions. The second-order solution for the velocity can now be obtained in its non-dimensionalized form following the same procedure that was used for the first order relations. After integration to an arbitrary point in the trajectory, the relation for the non-dimensional velocity v is obtained as a function of the range angle:

$$v = \frac{\left(1+\lambda^2\right)}{k_1}(c-\phi) - k\alpha x - \frac{(1-\alpha)\left(1+\lambda^2\right)}{k_1}\tau - \eta k\alpha I(x) + \frac{\eta\alpha\left(1+\lambda^2\right)}{k_1}J(x) \tag{3.54}$$

where the term $I(x)$ again returns. The new variable $J(x)$ is also introduced and defined as:

$$J(x) = \frac{\left(1+\lambda^2\right)}{k_1}c\tau - \frac{\left(1+\lambda^2\right)}{k_1}x - \frac{(1-\alpha)\left(1+\lambda^2\right)}{2k_1}\tau^2 - k\alpha I_2(x) \tag{3.55}$$

where $I_2(x)$ is also defined as before.

As the non-dimensional equation for the altitude does not have any terms that include $e^{\eta v}$, the same relation as before can be used. Given the second-order relation for ϕ^2, it is possible to find a more accurate value of the maximum atmospheric penetration x_1. Again, this should be done by using a root-finding scheme and solving for $\phi^2(x) = 0$.

An important note is made by Vinh et al. (1993) for vehicles with low L/D for this new approximation. The first order approximation of x_1 will be smaller than the second-order approximation. The root-finding algorithm will thus test values of x that are in excess of x_1. The proposed solution by Vinh et al. (1993) is to maintain a constant value of x in the following term: $2\eta\alpha I(x)$. This constant value of x is the same as the deepest penetration point for the first-order approximation of x_1.

To be able to perform constrained optimization, the aerodynamic load and heat load have to be modeled as well. Engelsma (2019) has developed equations that express these parameters as a function of τ such that they can be propagated along the equations of motion. They are given as:

$$\frac{D}{g_E}(\tau,\delta) = \frac{\left(1+\lambda^2\right)Be^{x(\tau)-\eta v(\tau,\delta)}}{2\alpha E^*} \tag{3.56}$$

$$\frac{L}{g_E}(\tau,\delta) = \frac{\lambda Be^{x(\tau)-\eta v(\eta,\delta)}}{\alpha} \tag{3.57}$$

$$q_c(\tau,\delta,C,a,b,M) = CR_n^a\rho_E^b V_E^M e^{bx(\tau)-\frac{M\eta v(\tau,\delta)}{2}} \tag{3.58}$$

The resulting analytical equations are thus dependent of the entry state, the non-dimensional range angle τ and the sign, defined as δ. As mentioned before, the sign is used to indicate whether the flight is in the downward of upward leg.

Finally, the equations need to be dimensionalized to obtain results that can be more easily understood. The values for the aerodynamic load and heat flux are already in their dimensional form and thus only the altitude, velocity and flight-path angle need to be converted to their spherical coordinates:

$$r(\tau) = -H_s x(\tau) + r_E \tag{3.59}$$

$$V(\tau,\delta) = \sqrt{\frac{V_E^2}{e^{\eta v(\tau,\delta)}}} \tag{3.60}$$

$$\gamma(\tau,\delta) = \arcsin\left(\frac{\phi(\tau,\delta)}{-\sqrt{\frac{r_E}{H_s}}}\right) \tag{3.61}$$

3.5.2. Atmospheric Glide Phase

The method for the gliding phase is also based on the derivation as presented in Vinh et al. (1993).

First, the system Equation (3.33), Equation (3.34) and Equation (3.35) are used in combination with $k_1 = B\lambda$ and the arc length is eliminated by the dividing the second and third equation by the first.

$$\frac{\mathrm{d\,v}}{dy = \frac{\left(1+\lambda^2\right)}{\phi} - \frac{k\alpha \mathrm{e}^{\eta v}}{y}} \tag{3.62}$$

$$\frac{d\phi}{dy} = -\frac{k_1}{\phi} + \frac{(\alpha \mathrm{e}^{\eta v} - 1)}{y\phi} \tag{3.63}$$

Then, the relation $\alpha u = e^{-\eta v}$ is used to rewrite the equation for speed as

$$\frac{\eta\left(1+\lambda^2\right)u}{\phi} = -\frac{1}{y'} + \frac{\eta k}{y} \tag{3.64}$$

In this equation, the prime denotes the derivative with respect to u. Using the same principle, the equation for the flight path angle can be rewritten by noticing that $d\phi/ = \phi'/y'$:

$$y\left[k_1 - \eta\left(1+\lambda^2\right)u\phi'\right] = \frac{1-u}{u} - \eta k\phi\phi' \tag{3.65}$$

These last two equations form a non-linear system of equations for y and ϕ with respect to the independent variable u. Vinh et al. (1993) applies the iterative technique to solve these equations.

This begins with setting the derivative of the flight path angle to zero, $\phi' \approx 0$. This leads to the following relation for the non-dimensional altitude as a function of the non-dimensional speed, within the assumption of equilibrium glide:

$$y_0 = \frac{(1-u)}{k_1 u} \tag{3.66}$$

which forms the first order solution for y. the derivative of this function is:

$$y_0' = \frac{-1}{k_1 u^2} \tag{3.67}$$

Next, these two equations can be substituted into Equation (3.64) instead of y and y', resulting in the first order solution for the flight path angle.

$$\phi_0 = \frac{2(1-u)}{E\sqrt{\beta r_E}\left[\frac{2}{\beta r_E} + u(1-u)\right]} \tag{3.68}$$

E is the lift-to-drag ratio that is used for the glide, thus forming the control parameter that can be used when applying this method.

Similarly, ϕ_0 and ϕ_0' and substituted in Equation (3.65) to obtain the second-order solution for y:

$$y_1 = y_0 \frac{1+AB}{1+B} \tag{3.69}$$

where

$$w = \frac{2}{\beta r_E u(1-u)} \tag{3.70}$$

$$A = \frac{w}{1+w} \tag{3.71}$$

$$B = \frac{2w(1-(1-w)u]}{E^2(1+w)^2} \tag{3.72}$$

Now, the second-order solution for the flight path angle can be obtained by substituting y_1 and y_1' into Equation (3.64), which leads to:

$$\phi_1 = \phi_0 \frac{(1+AB)}{(1+B)} \frac{(1+w)}{D} \tag{3.73}$$

with

$$D = w + \frac{1}{1 + \frac{B(1-2u)(3A+AB-2) + \frac{2A(A-1)}{E^2}(1-u)(3u-1-wu)}{(1+B)(1+AB)(1+w)}} \tag{3.74}$$

Lastly, the range angle remains to be determined. To do so, the equation for speed is considered:

$$\frac{du}{d\tau} = -\eta\left(1+\lambda^2\right) yu + \eta k\phi \tag{3.75}$$

When substituting the first order solutions for y and ϕ, this gives

$$\frac{du}{d\tau} = -\frac{2u(1-u)^2}{E\sqrt{\beta r_E[\eta k + u(1-u)]}} \tag{3.76}$$

The relation for τ and subsequently the range angle are found by quadrature:

$$\theta = \frac{1}{2}E\left[\begin{array}{l} \log\frac{(1-u)}{(1-u_e)} + \frac{2}{\beta r_E}\log\frac{u_e}{(1-u_e)}\frac{(1-u)}{u} \\ +\frac{2}{\beta r_E}\left(\frac{1}{1-u_E} - \frac{1}{1-u}\right) \end{array}\right] \tag{3.77}$$

Finally, Vinh et al. (1993) presents two equations that give the appropriate entry velocity and flight path angle for a given altitude.

$$u_E = \frac{1}{1+k_1} \tag{3.78}$$

and

$$\sin\gamma_E = \frac{2k_1(1+k_1)}{E[2(1+k_1)^2 + \frac{r_E}{H_s}k_1]} \tag{3.79}$$

The altitude is implemented in the variable k_1 in Section 8.1.1. The initial state for planar flight in equilibrium glide is thus obtained as input for a numerical simulation to verify the results.

3.5.3. Ballistic Phase

The previously described method is only capable of describing the atmospheric phase of the entry and thus a separate analytical method needs to be defined to describe the ballistic phase. The method as presented in Bate et al. (1971) for two-dimensional flight has been adopted for this research. This method was developed to model the motion of a missile from its burn-out point until the re-entry point and can be similarly applied for this research.

The method introduces a variable S_0, which is defined as follows:

$$S_0 = \frac{V_0}{V_{c,0}} \tag{3.80}$$

where V_0 is the velocity at the atmospheric exit point and $V_{c,0}$ is the local circular velocity at this point. The relation between the distance from the center of the Earth and the radius of the Earth can be defined as:

$$\frac{r}{R_e} = \frac{S_0^2 cos^2\gamma_0}{1+\sqrt{1-S_0^2 cos^2\gamma_0(2-S_0^2)cos\theta}} \tag{3.81}$$

The velocity can now be found by applying the law of energy:

$$V = \sqrt{\left(-\frac{\mu}{2a}+\frac{\mu}{r}\right)\cdot 2} \tag{3.82}$$

3.6. Forces and Environment

For the analysis of re-entry flight, two forces need to be considered to be acting on the body, the gravitational and aerodynamic forces. Other perturbing forces can be neglected, as they are several orders of magnitude smaller, especially during the aerodynamic phase of the flight. The aerodynamic and gravitational forces are considered external forces, as they are a direct result of the environment in which the vehicle is modeled. It is thus required to define the environment models that are used for the simulations, which will be done in the following section. First, an atmosphere model is required to be able to model the aerodynamic forces and heating that are acting on the vehicle. A central body shape will have to be determined to obtain the altitude of the vehicle, which in turn also influences the atmosphere. To obtain the gravitational forces, a gravity model needs to be defined. These models need to be separately defined for the numerical simulations and the analytical simulations, which were discussed in Section 3.4 and Section 3.5, respectively.

3.6.1. Aerodynamics

The aerodynamics of a re-entry flight depend on the atmospheric density ρ and the airspeed V as well as the vehicle attitude, described by α, β and σ. Two important variables can be derived as well, the Mach number M and the dynamic pressure q:

$$M = \frac{V}{a} \tag{3.83}$$

$$\bar{q} = \frac{1}{2}\rho V^2 \tag{3.84}$$

where a is the local speed of sound.

Aerodynamic Forces
In the aerodynamic reference frame, the aerodynamic forces can be defined as:

$$\mathbf{F_{A,A}} = \begin{pmatrix} -D \\ -S \\ -L \end{pmatrix} = \begin{pmatrix} -C_D \overline{q} S_{ref} \\ -C_S \overline{q} S_{ref} \\ -C_L \overline{q} S_{ref} \end{pmatrix} \tag{3.85}$$

where D is the drag, S the side force and L is lift. The force vector has been defined in the aerodynamic reference frame (A) and need to be converted to the inertial frame. The coefficients C_D, C_L and C_S are specific to a vehicle and are available in Mooij (1995) for HORUS. These coefficients depend on the Mach number (M), angle of attack (α) and angle of side-slip (β).

3.6.2. Atmosphere Model

This research will employ two different kinds of atmosphere models. The exponential atmosphere model is used in the analytical solutions for the entry trajectory and the US76 model will be used for the numerical simulator.

Exponential Atmosphere
The exponential atmosphere model is the simplest model available. It is selected for the need of a simple analytical representation for the atmosphere to represent the equations of motion analytically.

The first important simplification that is made in the model is the assumption that the atmosphere is an ideal gas. An ideal gas inherently assumes that the gas consists of randomly-moving, non-interacting point particles. This leads to the ideal gas law:

$$p = \rho RT = \rho \frac{R^*}{M} T \tag{3.86}$$

From this equation, the relation between the atmospheric pressure, the atmospheric density and the temperature can be obtained. Relating the density to altitude can be done by assuming the atmosphere is in hydrostatic equilibrium. The hydrostatic equation for the atmosphere is:

$$dp = -\rho g dh \tag{3.87}$$

Integration of Equation (3.87) requires knowledge of ρ and g in relation to the altitude h. This can be avoided by introducing the geopotential altitude z, which is related to h by:

$$g_0 dz = g dh \tag{3.88}$$

where g_0 is the gravitational acceleration at the Earth's surface. Using Newton's law of gravitation and after integration, the following solution is found, under the assumption of constant temperature and $g = g_0$:

$$\frac{\rho}{\rho_0} = e^{-\frac{h}{H_s}} \tag{3.89}$$

US 1976 Standard Atmosphere
Static atmosphere models use the assumptions for exponential atmospheres to provide a number of basic relations that describe the vertical temperature, density and pressure distribution in an averaged atmosphere. One of the most commonly used models is the US 1976 Standard Atmosphere (National Oceanic and Atmospheric Administration, 1976).

One of the main assumptions is that the air is dry and homogeneously mixed at altitudes below 86 km. When also assuming the air is a perfect gas, the equation of state can be used to relate the pressure, p, and temperature, T:

$$p = \frac{\rho R_g T}{M_m} \tag{3.90}$$

Above 86 km, this last assumption gradually breaks down and the individual gas species become a larger influence. Until the altitude exceeds 86 km, the atmospheric pressure is obtained using the following linear relationship:

$$T_M = T_{M,b} + L_{M,b} \cdot (H - H_b) \tag{3.91}$$

The variables that allow the use of this relation are shown in Table 3.1.

Table 3.1: Reference levels and gradients determined for the U.S. 1976 Standard Atmosphere (National Oceanic and Atmospheric Administration, 1976).

Subscript b	**Geopotential height H_b (km)**	**Molecular scale temperature gradient $L_{m,b}$ (K/km)**	**Form of function**
0	0	-6.5	Linear
1	11	0.0	Constant
2	20	1.0	Linear
3	32	2.8	Linear
4	47	0.0	Constant
5	51	-2.8	Linear
6	71	-2.0	Linear
7	84.8520		Linear

Heights above 86 km are defined using four functions using different input arguments. These four layers succeed each other in the following order:

1. 86 to 91 km: Isothermal Layer
2. 91 to 110 km: T(Z) follows an elliptical shape.
3. 110 to 120 km: Constant positive gradient.
4. 120 to 1000 km: T increases exponentially towards an asymptote.

The second section is defined by the following equation:

$$T = T_c + A \cdot \left(1 - \left(\frac{Z - Z_s}{a}\right)^2\right)^{1/2} \tag{3.92}$$

where $T_c = 263.1905K$, $A = -76.3232$ and $a = -19.9429$.

The third region follows the profile for its temperature change against altitude as:

$$\frac{dT}{dZ} = \frac{-A}{a}\left(\frac{Z - Z_s}{a}\right)\left[1 - \left(\frac{Z - Z_s}{a}\right)^2\right]^{-1/2} \tag{3.93}$$

The last section can be described as:

$$T = T_\infty - (T_\infty - T_{10}) \cdot \exp(-\lambda\xi) \tag{3.94}$$

with

$$\begin{aligned} &\lambda = L_{K,a} / (T_\infty - T_{10}) = 0.01875, \text{ and} \\ &\xi = \xi(Z) = (Z - Z_{10}) \cdot (r_0 + Z_{10}) / (r_0 + Z) \end{aligned} \tag{3.95}$$

The temperature profile for the exponential and US76 models can be found in Figure 3.1. The difference between both models is clearly observed for the altitude, where the exponential atmosphere assumes a constant temperature, while the US76 atmosphere follows the actual distribution more closely. It starts at sea-level temperature and follows the decreases, phases of constant temperature and increases as found in the actual atmosphere.

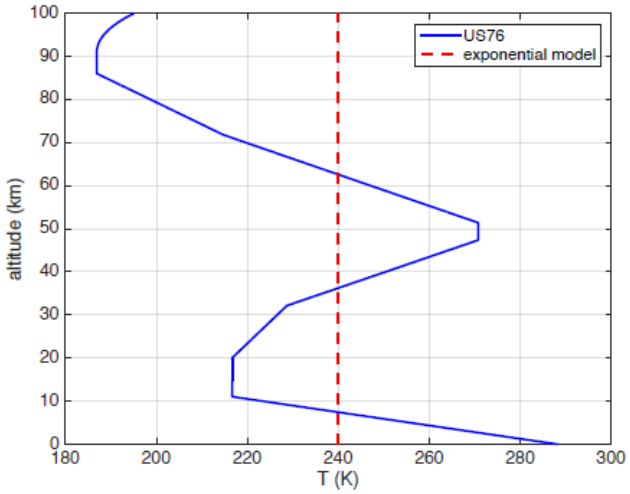


Figure 3.1: Temperature profile against altitude for the exponential and US76 atmosphere models.

3.6.3. Aerodynamic Heating

The re-entry flight is subject to constraints for the aerodynamic heating in two manners: the total heat load that is built up over the flight and the maximum local heating rate (Eggers et al., 1957). The aerodynamic heating originates from the conversion of the high kinetic and potential energy to heat, through friction with the atmosphere. This in turn also causes the deceleration of the vehicle when it travels through the atmosphere. The heating rate depends on the aerodynamic properties of the vehicle and requires intensive computation. An approximation is developed by Chapman (1960):

$$q_e = c^* \frac{1}{R_N^n} \left(\frac{\rho}{\rho_0} \right)^{1-n} \left(\frac{V}{V_c} \right)^m \tag{3.96}$$

Here, R_N is the nose radius of the vehicle, ρ the density, V the velocity and V_c the circular entry velocity. The constants are defined as : $c^* = 1.1097 \cdot 10^8$, $n = 0.5$ and $m = 3.15$. These constants approximate the airflow around the vehicle.

The total heat load can simply be found by taking the integral of the heat rate over the trajectory.

3.6.4. Central Body Shape

Different types of body shape models are available, depending on the required accuracy. For re-entry flight, the effects of the body shape are limited as the simulation stops before the vehicle comes close to the surface of the body. The main effect of the body shape is related to the atmosphere, which is defined with respect to the reference surface of the body.

A trade-off between accuracy and computational complexity for the models already excludes all models except the spherical and oblate spheroid shape models for this research. Previous studies into HORUS entry trajectories as presented in Mooij (1998) and Papp (2014) have used the spherical model. For consistency, this model has also been selected for this research. Although the simulator has the capacity to model an oblate spheroid Earth, the spherical shape approximation was applied in this research to simplify the comparison of the results with the previous works.

3.6.5. Gravitational Model

The gravitational force is the external force that draws the re-entry vehicle towards the Earth. According to Newton's Law of Gravitation the force between two point masses can be described as

$$\boldsymbol{F_G} = \frac{GMm}{r^2} \boldsymbol{r} \tag{3.97}$$

where F_G is the force vector, G is the universal gravity constant, M and m the masses of the central and orbiting body, respectively, r the norm of the distance vector and r the distance vector. The product GM is defined as the gravitational parameter μ, which in practice can be more accurately measured. The gravitational potential can now also be described as

$$U = -\frac{\mu}{r} \tag{3.98}$$

Eq. (3.98) is known as the central field model, which is the simplest form to approximate a body's gravitational potential.

Spherical Harmonics Model
The central gravity model as previously described is not sufficiently accurate when describing the motion of a vehicle re-entering the Earth's atmosphere as the Earth is not a perfect sphere, both in shape and in gravity. the main disturbance for Earth is a 'ring' of mass around the equator. The effect of this disturbance is also called the J_2-effect. When considering the gravity to be defined as:

$$\mathbf{g}_R = \begin{bmatrix} g_\delta \\ g_\tau \\ g_r \end{bmatrix} \tag{3.99}$$

the three components can be found from:

$$\begin{aligned} g_r &= \tfrac{\mu}{r^2}\left[1 - \tfrac{3}{2}J_2\left(\tfrac{R_E}{r}\right)^2\left(3\sin^2\delta - 1\right)\right] \\ g_\delta &= -3\tfrac{\mu}{r^2}J_2\left(\tfrac{R_E}{r}\right)^2 \sin\delta\cos\delta \end{aligned} \tag{3.100}$$

while the third, longitudinal component is negligible if the Earth is assumed to be symmetrical around its rotational axis. In these equations, r is the distance between the centers of mass of both bodies, μ is the standard gravitational parameter, δ is the latitude and R_E is the radius of the Earth. Finally, J_2 is the coefficient of oblateness of the Earth, given as (Wertz, 2001):

$$J_2 = 0.00108263 \tag{3.101}$$

3.7. Guidance

In addition to the previously described models, a desired state has to be given to the system in the form of guidance. The guidance will be defined by a set of nodes based on an independent variable with respective values for the decision variables. Intermediate values can be found during the flight by interpolation between the two nearest neighboring nodes. This method is called node control and lends itself well to optimization problems (Mooij and Hänninen (2009); Dijkstra et al. (2013); Papp (2014)). The geometry of this method can be seen in Figure 3.2 (Papp, 2014). A node is defined as:

$$\mathcal{N}_\mathrm{i} = \left\{v_\mathrm{t}, \alpha_{C,\mathrm{i}}, \sigma_{C,\mathrm{i}}\right\} \quad \text{with } i = 1,\ldots,n \tag{3.102}$$

where n is the number of nodes, v_i is the value of the independent variable at the node and α_C and σ_C are the commanded angle of attack and bank angle, respectively. The guidance matrix Γ contains the full set of these commands for the entire trajectory.

$$\Gamma = \begin{pmatrix} \mathcal{N}_1 \\ \mathcal{N}_2 \\ \vdots \\ \mathcal{N}_{n-1} \\ \mathcal{N}_n \end{pmatrix} = \begin{bmatrix} v_1 & \alpha_{C,1} & \sigma_{C,1} \\ v_2 & \alpha_{C,2} & \sigma_{C,2} \\ \vdots & \vdots & \vdots \\ v_{n-1} & \alpha_{C,n-1} & \sigma_{C,n-1} \\ v_n & \alpha_{C,n} & \sigma_{C,n} \end{bmatrix} \tag{3.103}$$

It should be noted that node control has a number of downsides which need to be taken into account. A clear disadvantage is the difficulty that comes with selecting the number of nodes. Ideally, the number of nodes would be large to improve the flexibility of the system. This would also increase the computational load and thus an optimum needs to be found. A variable-length optimization method is introduced in this research to circumvent this design step and optimize the number of nodes. This method will be discussed in Section 4.1.5.

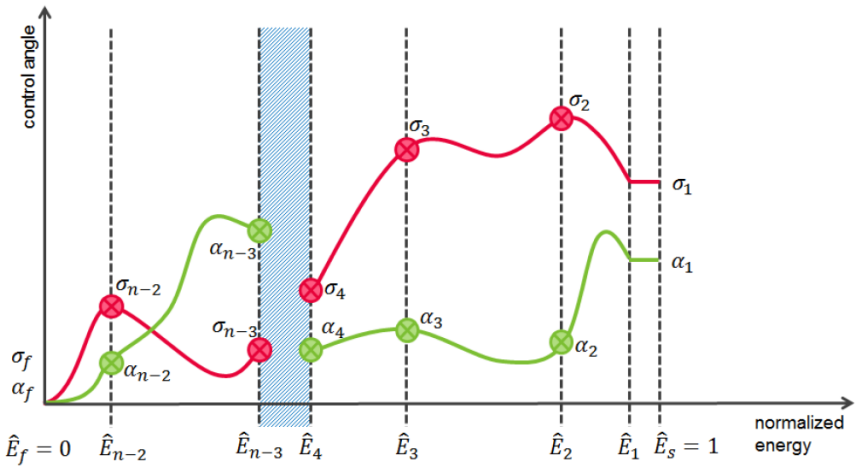


Figure 3.2: Guidance profile as defined by nodes(Papp, 2014).

Independent variable

The guidance logic requires an independent variable to link the commands to a point in the re-entry trajectory. A common independent variable is the total specific energy of the vehicle, which is monotonically decreasing during unpowered re-entry flight, since only energy is lost due to friction, while no new energy (i.e. by thrust) is added:

$$E = gh + \frac{1}{2}V^2 \tag{3.104}$$

Note that this relation holds for a flat Earth approximation of the specific energy. Especially for skipping flight, this relation does not hold and the energy should be more accurately described as:

$$E = \frac{V^2}{2} - \frac{\mu}{r} \tag{3.105}$$

where the first term describes the kinetic energy and the second term the potential energy of the vehicle. The variables V and r are the velocity and distance from the center of the Earth, respectively and μ is the standard gravitational parameter of the Earth.

The specific energy has the benefit that it does not depend on the exact trajectory when determining the end value, unlike the time. Additionally, for skipping trajectories it has the benefit of being constant during the skip-out phase, meaning that no change in guidance commands is made during this phase. The only case where problems would still occur is when powered flight is considered and the specific energy changes due to the thrust and decrease in mass. This could potentially be solved by re-calculating the trajectory after the powered phase. Commonly the values for E are normalized with respect to the energy at the entry interface, which is its highest value. This allows for mapping of the nodes in the normalized interval $\hat{E} \in [0,1]$ with

$$\hat{E} = \frac{E}{E_{\text{entry}}} \tag{3.106}$$

Node Positioning

A number of methods regarding node positioning have been analysed in Dijkstra et al. (2013) for atmospheric entry analysis. His conclusion was that non-uniform independent node control (NUIN) was the most suitable method. It was compared against PID-control, UIN and NUDN control. The latter two of these methods were considered unsuitable for entry control as they were found to not be robust. The PID method proved to converge faster when the search space is well-defined. Additionally, it is drawn towards local optima. The NUIN control was found to be the most flexible and to converge towards the global optimum. It is relatively slow, but can be considered preferable due to its better ability to reach the global optimum.

NUIN uses two nodes at the ends of the interval which are fixed and randomly generates the other nodes locations. This method costs relatively little computational power when combined with optimisation. The

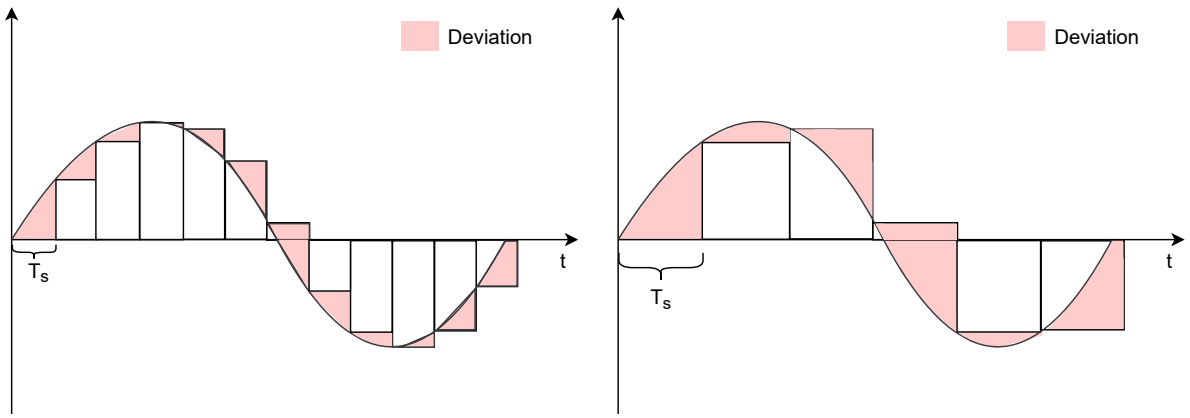
addition of Papp (2014) to fix the values of the final nodes is also useful to improve the efficiency of the mission planner. The reasons described by her are:

- The number of dimensions is greatly decreased as two less values need to be optimised for each variable.
- The attitude of the vehicle can be assumed to be neutral at $\hat{E} = 0$ with $\alpha_c = \sigma_c = 0$.

Zero-Order Hold

Realistically, a guidance algorithm will not sample the current commanded angle at every available time due to computation constraints. This means that in between time steps the value of the commanded angle is fixed and a discrete pattern is generated for the guidance. This is introduced in the simulator through zero-order hold, which is visualized in Figure 3.3. Instead of following a continuous pattern, the guidance commands are discretized and only sampled at given time steps. When comparing a smaller step size, such as Figure 3.3(a) to a large time step, such as Figure 3.3(b), it becomes clear that for larger sample times the deviation is more significant when applying the zero order hold. It is thus important to find the most appropriate time step that limits the deviations with respect to a continuous input, while also keeping in mind the required computation time that is required to take more samples.

In order to limit the computation time that is required for the simulations a time step of 2.0 s is selected. It should be noted that this value is large and thus not the most accurate. A more typical value would be a sampling of 10 Hz or 0.1 s, but this was found to put too stringent constraints on the computation time.



(a) Small time step

(b) Large time step

Figure 3.3: Effect of the zero-order hold time step on the accuracy of the input.

4

Optimization Methods

The following chapter provides background on the selected optimization methods. It will provide a description of the functioning of the methods and the purpose they hold in this research. The MOEA/D algorithm will be described in Section 4.1.2, followed by the ACO algorithm in Section 4.1.4 and a variable chromosome length algorithm in Section 4.1.5. The chapter is concluded with a description of the optimization problem in Section 4.2.

4.1. Methodology

The optimization methodology can be split into multiple parts. First, the principles of multi-objective optimizations are discussed. Then, the algorithms that are used to find the optimal solutions will be discussed.

4.1.1. Multi-Objective Optimization

Multi-Objective optimization carries the inherent problem that a trade-off between two objectives has to be made. For such problems, a shape exists where no improvement can be made for an objective without deteriorating another objective (Coello Coello, 1999). This principle is based on domination of solutions. For two solutions $\mathbf{u}$ and $\mathbf{v}$ with respective objective function values $\mathbf{f}(\mathbf{u})$ and $\mathbf{f}(\mathbf{v})$, $\mathbf{u}$ dominates $\mathbf{v}$ if and only if $f_i(\mathbf{u}) \leq f_i((v))$ for every $i \in 1,\ldots,m$, while also satisfying $f_i(\mathbf{u}) < f_i((v))$ for at least one $i \in 1,\ldots,m$ and also no worse values for the other objectives.

Eventually, a set of non-dominated solutions will emerge from the problem, which is called the Pareto set. The shape that is created by this solution set is called the Pareto front. An illustration of the Pareto front is found in Figure 4.1.

The Pareto front can be used by a human decision maker to choose a final solution to the problem.

4.1.2. MOEA/D

The Multi-Objective Evolutionary Algorithm with Decomposition (MOEA/D) was originally developed by Zhang and Li (2007). The method is based on the decomposition of a multi-objective problem into several scalar, single-objective optimization subproblems, which are simultaneously optimized. This method uses the fact that a Pareto optimal solution of a multi-objective problem can be defined as a single-objective problem where the objective is found by aggregating all objectives of the MOP. The subproblems are optimized by using information of only the nearest neighbors.

Evolutionary Algorithms

An evolutionary algorithm bases its optimization approach on genetics. It uses the concept of chromosomes and genes, where the decision vector of the optimization problem is the chromosome and the individual variables in the vector are the genes. First, a parent population is set up randomly. Then, for each generation, parents are randomly selected to generate an offspring individual based on crossover between two parent chromosomes. During this crossover, each gen for the offspring is selected based on the parent chromosomes. Afterwards, mutation of the offspring chromosome is performed, based on the mutation rate of the algorithm. Finally, the offspring is compared against the parent chromosome and the best individual is carried over to the next generation.

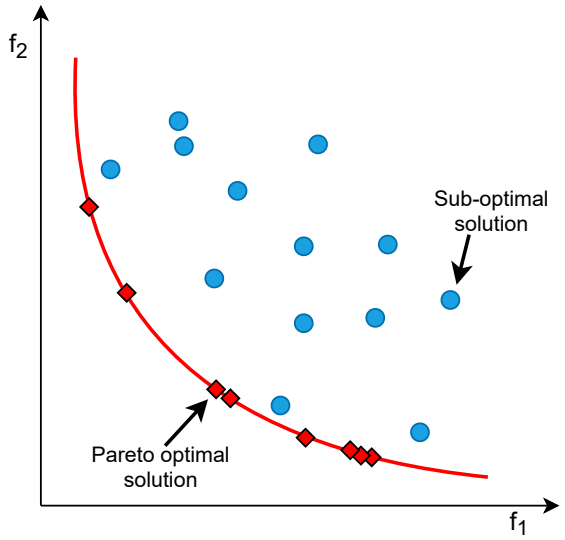


Figure 4.1: Schematic representation of a Pareto front for a two-dimensional problem.

Decomposition

Although other alternatives are available in literature, the method of aggregating the objectives is based on the Tchebycheff method, which is the main available method in PaGMO for the MOEA/D algorithm. For further elaboration on PaGMO the reader is referred to Chapter 6.

The Tchebycheff method defines a weight vector $\boldsymbol{\lambda} = [\lambda_1, ..., \lambda_m]$, where m is the number of objectives and the components of the vector add up to 1. Mathematically this is defined as (Zuiani and Vasile, 2013):

$$\min_{x\in\Omega}\left\{g\left(\mathbf{f}(\mathbf{x}) \mid \boldsymbol{\lambda}, \mathbf{z}^*\right)\right\} = \min_{\mathbf{x}\in\Omega}\left(\max_{i=1,...,m}\left\{\lambda_i \mid f_i(\mathbf{x}) - z_i^*\right\}\right) \tag{4.1}$$

where the reference objective vector $\mathbf{z}^*$ is:

$$z_i^* = \min\left\{f_1(x) \mid x \in \Omega\right\} \text{ with } i = 1, \ldots, m \tag{4.2}$$

Each Pareto optimal point $\boldsymbol{x}^*$ has a related weight vector $\boldsymbol{\lambda}$ for which the point is the optimal solution of Equation (4.1). By finding the different solutions that are optimal in this sense, a Pareto front can be constructed.

4.1.3. Constraint Handling

The MOEA/D algorithm as implemented in PaGMO does not include a method to handle constraints. However, for this research a method has been added to the algorithm that allows one to handle constraints as described by Lampinen (2002). This method applies a logic based on comparing the objective values and feasibility of candidates. An individual is compared to the corresponding member in the parent population and selected if:

1. Both individuals are feasible and the new individual has a lower objective function value.

2. The new individual is feasible and the parent individual is infeasible.

3. Both individuals are infeasible and the new individual has a lower or equal constraint violation value.

The method is reported to perform especially well on fast convergence to the feasible solution space. Also, by comparing infeasible solutions based on their violation values instead of their function values, no selective pressure towards low function values is exerted and the algorithm is less likely to be forced towards local optima due to the constraint handling.

4.1.4. MHACO

The Multi-Objective Hypervolume Ant Colony Optimizer (MHACO) is a novel optimization algorithm presented in Acciarini et al. (2020). It uses that logic of ant colony optimization combined with hypervolumes to allow for multi-objective optimization. Tests against the MOEA/D and NSGA-II algorithm have shown that MHACO is competitive with these optimization algorithms and capable of finding better Pareto fronts in terms of hypervolume values for the presented solar sailing mission in Acciarini (2020).

Ant Colony Optimization

Ant Colony Optimization was first introduced by Colorni et al. (1991) and based on a fundamentally different approach compared to genetic or evolutionary algorithms. It belongs to a class of nature-based optimization algorithms and as the name implies is a mathematical representation of a natural phenomenon. For ACO this is the method in which real ant colonies forage food. Ants explore paths starting at their nest to find food and leave pheromones behind so other ants can follow their path. As the amount of pheromone is proportional to the quantity and quality of the food that was found by the ant, other ants can find the most optimal path to follow.

The mathematical model consists of a graph $G = (V, E)$ where V represents two nodes and E represents two paths. The nest of the ants is represented by v_s and the food source is v_d. The first path, e_1, with length l_1 and the second path, e_2 with length l_2 are assumed to be $l_2 > l_1$. A pheromone value τ_i can now be given to each path, where the magnitude of the pheromone value corresponds to the performance of each path. The probability that an ant follows one of the paths from the nest is now given as:

$$p_i = \frac{\tau_i}{\sum_{i=1}^{i=2} \tau_i} \quad i = 1,2 \tag{4.3}$$

Since the path e_1 was found to be shorter it now holds that $p_1 > p_2$ and the ants are more likely to follow the first path. Having chosen one of the paths, the ants change the pheromone value that is associated with the path with an amount:

$$\Delta\tau_i = \frac{Q}{l_i} \tag{4.4}$$

A single flaw remains in this definition, which is that the amount of pheromone would be maintained on a path, even if a shorter path is found. Statistically, the ants would thus still follow a previously found short path, even if a better path has already been found. This problem is avoided by adding an evaporation parameter, which causes a path to lose its pheromone value over time. The evaporation is included mathematically as:

$$\tau_i \Rightarrow (1-\rho) \cdot \tau_i \quad i = 1,2 \tag{4.5}$$

where $\rho \in (o, 1]$ is the evaporation parameter used to control the evaporation.

For the exact implementation of the algorithm, the reader is referred to Acciarini et al. (2020). One last important step is to identify the input parameters of the algorithm, which can be used for tuning of the algorithm. These are defined as follows:

1. *ker*: The dimension of the solution archive where pheromones from previous generations are stored. It should be noted that this parameter can never be larger than the population size that is used in the problem.

2. *q*: The convergence speed parameter. The standard value of this parameter is 1.0. For this research, no changes will be made to this parameter.

3. *threshold*: The threshold parameter. This determines the number of generations after which the value of *q* is set to 0.01 and thus the solutions will start converging towards the best found value.

4. *NGenMark*: The parameter controls the convergence speed of the standard deviation values in the solution archive.

5. *evalstop*: This integer variable counts the runs for which no improvement is made for the hypervolume value, stops the algorithm when the integer criterion is met and returns the current population.

6. *focus*: The parameter is used to make the search greedier towards the optimum. A greedier search would make the algorithm more likely to get stuck in a local optimum.

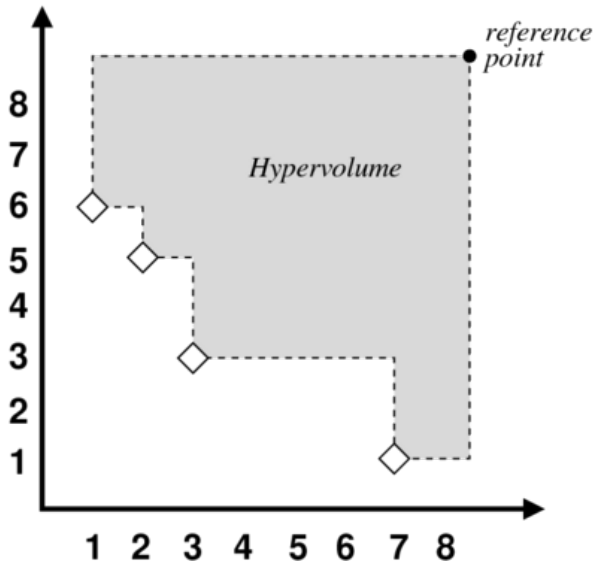


Figure 4.2: Representation of a hypervolume for a two dimensional optimization problem.

Hypervolumes
The hypervolume metric is an alternative performance metric to measure the performance of individuals in MO optimization problems. Being first introduced in Zitzler et al. (2003), it has become widely applied in MO optimization as it evaluates convergence and diversity in a single variable with the additional benefit that it is strictly Pareto compliant. This expresses itself as a more Pareto optimal individual will by definition have a lower hypervolume value.

The hypervolume can now be mathematically represented as:

$$I_H\left(B, \mathbf{y}_{ref}\right) = \Lambda\left(\bigcup_{\mathbf{y}\in B}\left\{\mathbf{y}' \mid \mathbf{y} < \mathbf{y}' < \mathbf{y}_{ref}\right\}\right), \quad B \subset \mathbb{R}^m \tag{4.6}$$

where Λ is the Lebesgue measure, which is commonly used to measure subsets of Euclidean space in n dimensions. Furthermore, m is the objective space dimension. $\mathbf{y}, \mathbf{y}' \in B$ belong to a subset of the overall objective function vectors and $\mathbf{y_{ref}} \in \mathbb{R}^m$ is the reference point vector that should be dominated by all Pareto optimal solutions. A graphical representation of the hypervolume for a two dimensional problem is given in Figure 4.2.

Acciarini et al. (2020) have proposed a number of algorithms to compute the hypervolume resulting from the optimization problem. Exact algorithms are available for low dimensions, but for higher dimension problems only numerical algorithms are found to be suitable. Given that the MHACO algorithm is not capable of handling constraints and treats these as additional objectives, the objective space has a dimension of 5. The hypervolume calculation algorithm that is included in MHACO for these problem dimensions is the Walking Fish Group algorithm as presented in While et al. (2011).

Constraint handling
The MHACO algorithm implemented in PaGMO also does not include a method to handle constraints. Instead, Acciarini et al. (2020) reports that constraints can be included as objectives instead, thus creating an unconstrained optimization problem.

Due to the method that is used for MHACO, the constraint handling as proposed by Lampinen (2002) and used for the MOEA/D algorithm cannot be simply applied to the MHACO algorithm and the choice was made to treat the problem as unconstrained for MHACO instead. It should be noted that the use of hypervolumes increases the need for proper normalization of the problem objectives, which will be further elaborated in Chapter 9.

4.1.5. Variable Chromosome Length

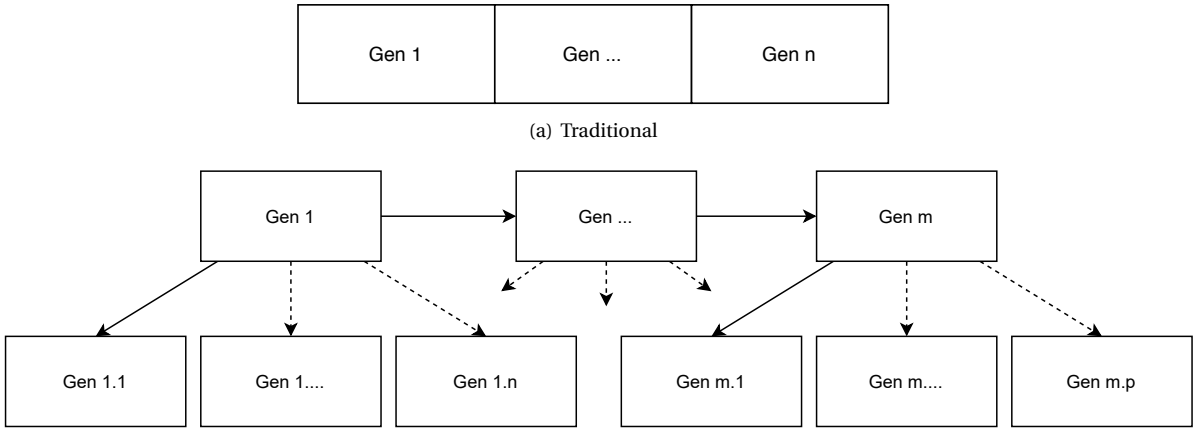
In addition to the more traditional optimization methods, a method was developed that employs a variable chromosome length and structure. The concept for this method was first presented by Dasgupta and McGre-

gor (1992b) and involves the dimensions of the chromosome as an optimization parameter. The method was proven to successfully find the global optimum in engineering problems by Dasgupta and McGregor (1992a).

Although the concept of variable chromosomes can be applied to any genetic algorithm, the algorithm that was developed for this research build on the MOEA/D algorithm that was previously discussed as a baseline, the method developed here will thus be called the Structured Chromosome Evolutionary Algorithm (SCEA). The existing algorithm is extended with a function capable of evolving variable size individuals. To comply with the PaGMO environment it is necessary to maintain the same length for each chromosome. To prevent excessive changes to the PaGMO environment, the unused elements of the chromosome will be left in place, but otherwise disregarded by the software. As a result, no improved performance will be observed in memory usage compared to other algorithms in PaGMO.

To understand the SCEA, the difference between the regular and structured chromosome needs to be defined. Figure 4.3 presents the lay-out of the traditional chromosome compared to the structured chromosome. The traditional chromosome simply exists of an array of decision variables. In case of multiple decision variables, such as the angle-of-attack and bank angle in re-entry trajectories, these are concatenated to ensure a single, longer vector.

The structured chromosome uses a different, more flexible approach as seen in Figure 4.3(b). Instead of predefining the length of the array in advance, it takes the length of the vector as a decision variable. This also allows for a varying number of decision variables for different parts of the optimization problem. In re-entry the benefit becomes evident when considering a flight consisting of multiple legs. By use of the structured chromosome, the legs that require little guidance will only take a very limited number of nodes, whereas the legs that strongly influence the optimization result will use more nodes. Usually, this process of node design requires manual labor, which might be unnecessary by the application of a structured chromosome.



(a) Traditional

(b) Structured

Figure 4.3: Lay-out of a traditional chromosome compared to a structured chromosome.

The main adaptation for structured chromosome algorithms is related to the evolutionary operator, the crossover and the mutation. Where traditional optimization simply goes through the vector describing the chromosome step by step, the structured algorithms requires a more specific order in which the operations take place and requires adaptation of the chromosome based on these operations.

The logic that is applied is presented in Algorithm 1. The logic is applicable to both crossover and mutation operations. First, the integer genes for the new chromosome are created by evolutionary operations. It is important to note that the algorithm should always start these operations at the highest level to ensure that all levels are passed during evolution. A check is performed whether these are linked to dependent genes, which are then either appended to or deleted from the chromosome to ensure that the chromosome length is correct again. The algorithm then moves to the real variables in the chromosome and performs evolutionary operations on these as long as the parent chromosome real variables, n_{real} are not exceeded.

The conceptual structure of the chromosome applied to skipping re-entry flight can be found in Fig. 4.4. The highest level of the chromosome determines the number of legs of the skipping flight. For each leg, the number of nodes is then determined. Lastly, the different parameters that are optimized at each node are described. In this three-level structure, both the number of legs and the number of nodes per leg are

Algorithm 1 Logic describing the crossover operations for the structured chromosome algorithm.

input: chromosomeToMutate, LB, UB, n_{int}, n_{real}
for $i \in n_{\text{int}}$ **do**
    Perform evolutionary operator on integer element
    if *newChromosome*(i) > *chromosomeToMutate*(i) **then**
        if *newChromosome*(i) has dependent genes **then**
            Append random gens equal to difference.
        else if *newChromosome*(i) < *chromosomeToMutate*(i) **then**
            Delete gens equal to difference.
        end if
    end if
end for
for $j \in n_{\text{real}}$ **do**
    Perform traditional evolutionary operations.
end for
return *newChromosome*

taken into account as an optimization parameter and might thus change for each individual. Although this process could also achieved by muting certain elements of the chromosome as a consequence of the selected number of legs, the approach taken by the SCGA minimizes computational effort by completing leaving out the unused parameters.

4.2. Optimization Problem

The following section defines the optimization problem of a re-entry trajectory. The values of the decision variables, which are the input to find an optimal result are determined by the algorithms. Then, the problem objectives are the variables that are to be optimized. Finally, the constraints are imposed on the solution

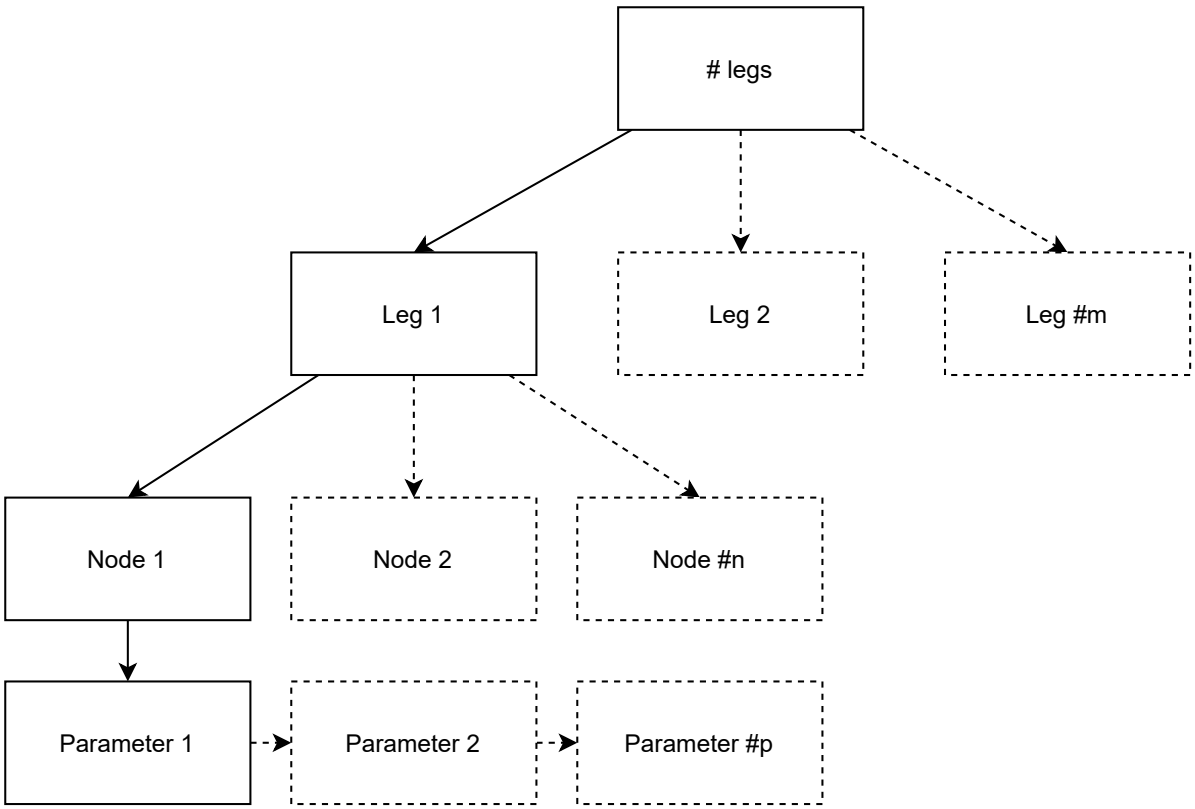


Figure 4.4: Conceptual structure of the structured chromosome GA for re-entry flight.

space to ensure that the solutions do not violate the limits of vehicle or crew.

4.2.1. Decision Variables

Re-entry trajectories are influenced through the guidance commands. These guidance commands can be defined by specifying the value of the guidance parameters at certain nodes. The values in between those nodes are determined by interpolation as described in Chapter 5 at each moment in which the guidance system is called.

The nodes and their values can then be combined into the guidance matrix Γ, which defines the node locations in terms of energy $\hat{E}$ and the attitude commands at these locations ($\bar{\alpha}(\hat{\mathbf{E}})$ and $\bar{\sigma}(\hat{\mathbf{E}})$). This can be represented mathematically as:

$$\mathbf{\Gamma} = \begin{bmatrix} \mathscr{N}_s \\ \mathscr{N}_1 \\ \vdots \\ \mathscr{N}_{n-1} \\ \mathscr{N}_f \end{bmatrix} = \begin{bmatrix} \hat{E}_s & \alpha_s & \sigma_s \\ \hat{E}_0 & \alpha_0 & \sigma_0 \\ \hat{E}_1 & \alpha_1 & \sigma_1 \\ \vdots & \vdots & \vdots \\ \hat{E}_{n-1} & \alpha_{n-1} & \sigma_{n-1} \\ \hat{E}_f & \alpha_f & \sigma_f \end{bmatrix} \tag{4.7}$$

where n is the number of nodes.

It should be noted that the first and last nodes of the matrix are not included in the optimization, as these represent the initial state and desired final state. The number of nodes that lies in between is typically a part of the design of the optimization problem. However, the use of the variable length chromosome algorithms takes the number of nodes as a decision variable.

Note that for the SRM vehicle, no data of the C_L with respect to α is available, and thus only the bank angle can be used as an input, which influences the vertical L/D ratio.

Generally, a larger number of guidance nodes leads to more optimal trajectories, as the trajectory can be more accurately controlled. A larger number of nodes does however increase the number of iterations that is required by the optimization algorithm as the dimension of the optimization problem increases. The scaling of the dimension of the solution space Ω_n is found as:

$$\dim \Omega_n = 3(n-2) \tag{4.8}$$

Here, the benefit of the variable length chromosomes of the SCEA algorithm becomes apparent. In addition to the previously mentioned decision variables, extra variables are added to represent the number of legs of the trajectory and the number of nodes for each of these legs. An additional task is thus passed to the optimization problem, where it automatically determines the number of nodes that is used by the algorithm and thus might find a solution in the Pareto front with less nodes faster, improving the convergence rate for the optimization problem.

In this thesis, the number of nodes for a fixed-length guidance algorithm is set equal to 8 at all times to ensure that relatively high quality optimal solutions are found. This selection was made as a larger number of nodes provides more optimal solutions, whereas the maximum number of 8 nodes, in combination with the fixed first and final nodes, is equal to the maximum guidance vector size investigated by Papp (2014). The variable length guidance is investigated for a range of guidance nodes, which lies between 4 and 8 nodes, thus being capable of finding the same quality solutions as the fixed-length problem, but allowing for faster convergence if a lower number of nodes suffices to find (a region of) the Pareto front.

Finally, the upper and lower bounds of the decision variables should be noted. The guidance and bank angle values are limited by the vehicle and are $0° - 45°$ and $0° - 90°$, respectively. The energy values are set to a range between 0.1 and 0.95, to ensure that the algorithm will not find solutions that are too close to the extremities of the re-entry trajectory. Finally, the number of nodes lies between 4 and 8 and the number of legs is fixed to a value of 1 for the gliding flight.

4.2.2. Problem Objectives

Re-entry problems have two primary objectives. The flight range should be maximized and the total heat load minimized. The complexity of re-entry trajectory design becomes clear when one realizes that these objectives are conflicting.

Minimum Heat Load
When a trajectory is only optimized for the heat load, it will generally follow a smooth gliding trajectory. The heat load is then obtained by simply integrating the heat flux over the time-of-flight. The heat load is then minimized by performing a trajectory that is as short as possible, without violating the operational constraints. Such trajectories follow the heat flux constraint closely throughout and would violate the constraint when shorter, while increasing the total heat load when flying a longer trajectory.

Maximum Flight Range
Contrary to minimum heat load trajectories, a trajectory that is optimized for its flight range will try to increase the duration of entry for as long as possible. The heat flux profile will thus not lie close to its constraint value throughout the entire trajectory, while also a lofting flight is more likely. The lofts in this trajectory allow the vehicle to trade its energy while still in the upper regions of the atmosphere, where the air is less dense and thus less heat load will be built up. However, for this trajectory type, the fluctuations in the trajectory often cause the g-load and g-load rate constraints to be more stringent. Eventually, more heat load will experienced by the vehicle due to the longer flight time, thus causing a conflict with the previous objective.

4.2.3. Constraints

The constraints of the re-entry trajectory have been previously described. However, their implementation is implemented in different parts of the simulation. First of all, the constraints for the angle of attack and bank angle are included in the decision vector bounds. Their rate constraints directly implemented in the guidance algorithm and the change of these angles will thus never exceed their limit values.

Three constraints now remain that are implemented in the optimization problem. These are the heat flux, g-load and g-load rate. The optimization algorithms will handle them based on the previously described constraint handling methods. The exact implementation is given below:

1. **Maximum heat flux** The maximum heat flux $q_{c,max}$ can vary throughout the trajectory, although it should always remain below its limit value of 500 kW/m^2 to find a feasible trajectory. A constraint violation is thus implemented to let the optimization algorithms know that the trajectory is unfeasible. Mathematically, this is implemented as follows:

$$p_q = \begin{cases} 0, & \text{if } q_{max} \leq q_{c,max} \\ \frac{q_{max}}{q_{c,max}}, & \text{if } q_{max} > q_{c,max} \end{cases} \tag{4.9}$$

2. **Maximum g-load** The maximum allowable g-load, $g_{c,max}$ is determined by either the structural integrity of the vehicle or the limits of its passengers. Its implementation is the same as the heat load constraint and given as:

$$p_g = \begin{cases} 0, & \text{if } g_{max} \leq g_{c,max} \\ \frac{g_{max}}{g_{c,max}}, & \text{if } g_{max} > g_{c,max} \end{cases} \tag{4.10}$$

3. **G-load rate** The maximum allowable g-load rate $\dot{g}_{c,max}$ is also limited by the structural integrity of the vehicle and the limits of the passengers. It is implemented as:

$$p_{\dot{g}} = \begin{cases} 0, & \text{if } \dot{g}_{max} \leq \dot{g}_{c,max} \\ \frac{\dot{g}_{max}}{\dot{g}_{c,max}}, & \text{if } \dot{g}_{max} > \dot{g}_{c,max} \end{cases} \tag{4.11}$$

Finally, also constraint for the rate of change of the attitude angles are implemented. These are not implemented as constraints for the optimization problem, but rather implemented directly in the guidance. They are:

1. **Maximum** $\dot{\alpha}$: The maximum allowable rate of change of the angle of attack is set to 5.0°/s, as is determined for the HORUS-2B entry vehicle. This is implemented directly into the guidance by limiting the change of the angle of attack to its maximum value when the interpolation would cause it to exceed this value.

2. **Maximum** $\dot{\sigma}$: The maximum allowable rate of change of the bank angle is set to 15.0°/s, again as determined for the HORUS-2B entry vehicle. Its implementation is similar to the angle of attack.

II

Part II: Software

5

Numerical Methods

This chapter describes the various numerical methods that are applied in this research. It presents an overview of the functionality of the method and its purpose within the research. The methods that are discussed in this chapter are the root-finding in Section 5.1, interpolation in Section 5.2 and integration in Section 5.3.

5.1. Root-Finding Method

The second order analytical method for skipping flight as described in Section 3.5 requires a root finding method to determine the first iteration of x_0, the lowest point in the trajectory. As the equation that needs to be solved is simple, a bisection root finding method will suffice without any significant loss of accuracy or computation time.

5.1.1. Bisection Root-Finding

The bisection root finding method is the simplest available. As only one-dimensional root finding is required for this research, multi-dimensional methods are not discussed here, although the bisection method can also be applied to those problems.

The bisection method can be applied for any problem where the zero-axis must be crossed. Although functions with multiple zeroes cannot be accurately described using this method, it suffices for the problems in this research. The function requires two values around the zero, thus one positive value and one negative value and converges towards the zero by iteration. The method evaluates the function value at the center between two nodes. If the value at the center is positive, the node that corresponded to the previous positive value is replaced and vice versa. In mathematical terms, this can be illustrated as follows:

$$
\begin{aligned}
x_0 &< x_1 \\
f(x_0) &< 0 \\
f(x_1) &> 0 \\
x_2 &= \frac{x_1 - x_0}{2} \\
x_3 &= \begin{cases} \frac{x_2 - x_0}{2} & \text{If: } f(x_2) > 0 \\ \frac{x_1 - x_2}{2} & \text{If: } f(x_2) < 0 \end{cases}
\end{aligned}
$$

The steps as described above are repeated, resulting in a continuously smaller interval, until the interval becomes smaller than a predefined thresh hold. The absolute error in the estimate of the root, ϵ, can be determined to be:

$$\epsilon = \frac{x_1 - x_0}{2^n} \tag{5.1}$$

where x_0 and x_1 are the initial nodes and n is the number of iterations.

The main benefit of the bisection method is its simplicity in implementation. It requires only the function and an if-statement that replaces the bounds as appropriate. Secondly, the method does not require

knowledge on the derivative of the function, as this derivative is not known and would be very complex in this research.

5.2. Interpolation Method

In this research, two important data sets are only available in tabulated form; the aerodynamic data of HORUS and the guidance commands. Still, values between those nodes are often required, which can be found by an interpolation method.

5.2.1. Linear Interpolation

One of the simplest forms of interpolation is linear interpolation. This method determines the value between two nodes by a linear relation between the two nearest nodes. For a one-dimensional problem, the linear interpolation estimate of the function is found from:

$$\phi(x) = f(x_i) + (f(x_{i-1}) - f(x_i))\frac{x - x_i}{x_{i+1} - x_i} \tag{5.2}$$

5.2.2. Cubic Spline Interpolation

The linear interpolation method can be extended to ensure smooth behaviour of the first and second derivatives by cubic spline interpolation. Equation (5.2) then becomes:

$$y = Ay_j + By_{j+1} + C\ddot{y}_j + D\ddot{y}_{j+1} \tag{5.3}$$

and the new terms C an D are given by:

$$\begin{aligned} C &= \frac{1}{6}\left(A^3 - A\right)\left(x_{j+1} - x_j\right)^2 \\ D &= \frac{1}{6}\left(B^3 - B\right)\left(x_{j+1} - x_j\right)^2 \end{aligned} \tag{5.4}$$

The second derivatives of the function are obtained from:

$$\frac{x_j - x_{j-1}}{6}\ddot{y}_{j-1} + \frac{x_{j+1} - x_{j-1}}{3}\ddot{y}_j + \frac{x_{j+1} - x_j}{6}\ddot{y}_{j+1} = \frac{y_{j+i} - y_j}{x_{j+1} - x_j} + \frac{y_j - y_{j-1}}{x_j - x_{j-1}} \tag{5.5}$$

Equation (5.5) gives $N-2$ equations for N unknown values of $\ddot{y}_i$. To solve Equation (5.3) the boundary conditions of $\ddot{y}_0$ and $\ddot{y}_{N-1}$ are required. When these are set to zero, the so-called natural cubic spline is found. From these assumptions a tridiagonal matrix can be derived:

$$\begin{bmatrix} 2(h_0+h_1) & h_1 & & \\ h_1 & 2(h_1+h_2) & \ddots & \\ & \ddots & \ddots & h_{n-3} \\ & & h_{n-3} & 2(h_{n-3}+h_{n-2}) \end{bmatrix} \cdot \begin{bmatrix} \ddot{y}_1 \\ \ddot{y}_2 \\ \vdots \\ \ddot{y}_{n-2} \end{bmatrix} = 6 \begin{bmatrix} \frac{y2-y_1}{h_1} - \frac{y_1-y_0}{h_0} \\ \vdots \\ \frac{y_{n-1}-y_{n-2}}{h_{n-2}} - \frac{y_{n-2}-y_{n-3}}{h_{n-3}} \end{bmatrix} \tag{5.6}$$

5.2.3. Hermite Spline Interpolation

The Hermite spline interpolation adds another layer compared to the cubic splines that were previously described. It uses third-degree polynomials to describe both the value and first derivatives at the given given points. For a unit interval of $(0,1)$ with starting point p_0 and endpoint p_1, with their respective tangents m_0 and m_1, the polynomial is found as:

$$p(t) = h_{00}(t)p_0 + h_{10}(t)m_0 + h_{01}(t)p_1 + h_{11}(t)m_1 \tag{5.7}$$

where the functions $h(t)$ are defined as:

$$\begin{aligned} h_{00}(t) &= 2t^3 - 3t^2 + 1 \\ h_{10}(t) &= t^3 - 2t^2 + 1 \\ h_{01}(t) &= -2t^3 + 3t^2 \\ h_{11}(t) &= t^3 - t^2 \end{aligned} \tag{5.8}$$

and $t \in (0,1)$. When interpolating over an arbitrary interval, this interval is converted to the interval by changing the independent variable:

$$t = \frac{x - x_j}{x_{j+1} - x_j} \tag{5.9}$$

with the new polynomial defined as:

$$p(x) = h_{00}(t)p_j + h_{10}(t)\left(x_{j+1} - x_j\right)m_j + h_{01}(t)p_{j+1} + h_{11}(t)\left(x_{j+1} - x_j\right)m_{j+1} \tag{5.10}$$

Lastly, the tangents need to be chosen. For this research the method of cardinal splines was selected, which selects the tangents from:

$$m_j = (1-c)\frac{p_{j+1} - p_{j-1}}{t_{j+1} - t_{j-1}} \tag{5.11}$$

where $0 \leq c \leq 1$.

5.3. Integrator

An integrator is required to be able to numerically solve the differential equations that describe the motion of the entry vehicle. In specific, integrators are used to solve these differential equations based on their initial values. The need for these numerical methods arises from the oftentimes high complexity of boundary equations, while some might not have an analytical solution at all.

For this research, two different sets of equations need to be integrated. One method is required to numerically solve the differential equations to obtain an accurate trajectory. The other method is needed to integrate the analytical methods.

5.3.1. Runge-Kutta Integrator

The Runge-Kutta type of integrators provides a fairly simple method to accurately determine the state of the vehicle. In this research, the variable step size Runge-Kutta will be applied. The step size is varied based on multiple approximations of the state by comparing a lower order Runge-Kutta method to a method that is one order higher. Below, the method will be discussed in detail.

A simple fourth-order fixed step-size method (RK4) requires a predefined step-size and will thus require closer tuning of this step-size. This stems from the fact that integration methods have a limited order and are thus by definition only capable of approximating the solution. If the selected time step is too large, this will introduce an error in the result with each step that is made, especially when the order of the problem is higher than the order of the selected integrator.

This is often solved by using a method of order p along with a $p+1$-order method to estimate the integration error that is made. The methods then vary the step size based on the accuracy that is found by the comparison of the p and $p+1$ order methods. Mathematically, the two methods can be defined as:

$$\begin{aligned} f_0 &= f\left(x_0, y_0\right) \\ f_k &= f\left(x_0 + \alpha_k h, y_0 + h\sum_{\lambda=0}^{k-1}\beta_{k\lambda} f_\lambda\right) \end{aligned} \tag{5.12}$$

and:

$$\begin{aligned} y &= y_0 + h\sum_{0}^{k-1} c_k f_k + O\left(h^p\right) \\ \hat{y} &= y_0 + h\sum_{0}^{k-1} \hat{c}_k f_k + O\left(h^{p+1}\right) \end{aligned} \tag{5.13}$$

Where α, β, c and $\hat{c}$ are coefficients specific for the integration method and:

$$\begin{aligned} f &: \text{ function to calculate the derivative} \\ x &: \text{ independent variable} \\ y &: \text{ state variable} \\ k &: \text{ number of stages } -1 \end{aligned} \tag{5.14}$$

Given the variable size time step and the Runge-Kutta method, still several integrators are available, depending on the order of the integrator. A trade has to be made between the accuracy of the higher order methods compared to the increase in computation time. Previous work on re-entry GNC by Papp (2014), Rijnsdorp (2017) and Lucassen (2019) has shown that the lowest available variable step size method, the RKF4(5) integrator, already provides sufficiently accurate results compared to high fidelity simulations. Given that this method also has the lowest computation time it will thus also be appropriate for this research.

For the selected RKF4(5) method, the next step for x can be formulated for the fifth order as:

$$x_{n+1} = x_n + c_1 k_1 + c_2 k_2 + c_3 k_3 + c_4 k_4 + c_5 k_5 + c_6 k_6 + O\left(\Delta t^6\right) \tag{5.15}$$

and for the fourth order as:

$$x^*_{n+i} = x_n + c^*_1 k_1 + c^*_2 k_2 + c^*_3 k_3 + c^*_4 k_4 + c^*_5 k_5 + O\left(\Delta t^5\right) \tag{5.16}$$

where the following intermediate values per step are used for the determination of the accuracy:

$$\begin{aligned}
k_1 &= hf\left(t_n, x_n\right) \\
k_2 &= hf\left(t_n + a_2 h, x_n + b21 k_1\right) \\
k_3 &= hf\left(t_n + a_3 h, x_n + b_{31} k_1 + b_{32} k_2\right) \\
k_4 &= hf\left(t_n + a_4 h, x_n + b_{41} k_1 + b_{42} k_2 + b_{43} k_3\right) \\
k_5 &= hf\left(t_n + a_5 h, x_n + b_{51} k_1 + b_{52} k_2 + b_{53} k_3 + b_{54} k_4\right) \\
k_6 &= hf\left(t_n + a_6 h, x_n + b_{61} k_1 + b_{62} k_2 + b_{63} k_3 + b_{64} k_4 + b_{65} k_5\right)
\end{aligned} \tag{5.17}$$

where a, b, c and c^* are defined differently in literature, but were defined originally by Fehlberg as seen in Table 5.1.

The truncation error is:

$$\varepsilon \equiv x_{n+1} - x^*_{n+1} = \sum_{i=1}^{6} \left(c_i - c^*_i\right) k_i \tag{5.18}$$

If a time step is found to produce an error ε, the preferred new time step is found from:

$$\Delta t^* = \Delta t \left| \frac{\varepsilon^*}{\varepsilon} \right|^{\frac{1}{4}} \tag{5.19}$$

Table 5.1: RKF45 constants as defined by Fehlberg.

i	a_i	b_{ij}					c_i	c^*_i
1							$\frac{16}{135}$	$\frac{25}{216}$
2	$\frac{1}{4}$	$\frac{1}{4}$					0	0
3	$\frac{3}{8}$	$\frac{9}{32}$	$\frac{9}{40}$				$\frac{6656}{12825}$	$\frac{1408}{2565}$
4	$\frac{12}{13}$	$\frac{1932}{2197}$	$-\frac{7200}{2197}$	$\frac{7296}{2197}$			$\frac{28561}{56430}$	$\frac{2197}{4104}$
5	1	$\frac{439}{216}$	-8	$\frac{3680}{513}$	$-\frac{845}{4104}$		$-\frac{9}{50}$	$-\frac{1}{5}$
6	$\frac{1}{2}$	$-\frac{8}{27}$	2	$-\frac{3544}{2565}$	$\frac{1859}{4104}$	$-\frac{11}{40}$	$\frac{2}{55}$	0
j =		1	2	3	4	5		

5.4. Pseudo-Random Number Generation

Computers are incapable of selecting a truly random number. Algorithms that approximate random number generation are called Pseudo-Random Number Generators, which generate sequences of numbers which resemble actual random generation. The two elements that are required to determine the random number generation are the seed, which is used to ensure that sequences are fixed between different calls to the number generator, and the generator itself. In this research, the Mersenne Twister will be used according to its implementation in the C++ Boost library, mt19937 (Watanabe, 2010).

6

Software

In the following chapter, the software that will be used to perform the research is described. It can be divided into two sections: the required external software packages are described in Section 6.1 and the architecture of the software that was developed is described in Section 6.2.

6.1. External Software

The software that is used for this research is largely available in pre-existing software packages. The two packages that are described below offer a complete environment in which the separate parts of the software need only be linked together.

6.1.1. TU Delft Astrodynamics Toolbox

The TU Delft Astrodynamics Toolbox, Tudat for short, provides a collection of C++ libraries that can be used to create astrodynamic simulations. It involves a range of numerical methods and interfaces that are commonly used in the field, as well as more specialized tools that have been developed by students and staff of the Astrodynamics & Space Missions group of the faculty of Aerospace Engineering. For extensive documentation on the toolbox, the reader is referred to the official website of Tudat, which features tutorials and documentation[1].

Importantly, all features that are included in Tudat come with their own unit tests that can easily be used to verify the available features and models. After installation of the Tudat libraries, these units tests can be run to verify their correct installation and function.

With the exception of the optimization algorithms, all numerical methods as described in Chapter 5 are readily available in Tudat. Due to the available unit tests in Tudat, these methods can all be assumed to be verified after succesful installation. The optimization algorithms will be discussed along with their respective libraries in the following section.

Similarly, all environment models that are included in this research are available in the base version of Tudat and are verified after installation. This also holds for the aerodynamic coefficient interface that is available in Tudat, which reads the aerodynamic data from files and interpolates between the available nodes.

6.1.2. Parallel Global Multi-Objective Optimizer

The Parallel Global Multi-Objective Optimizer, or PaGMO, is a C++ library for parallel optimization. Similar to Tudat, it is constantly being extended based on the work performed in the field of optimization. The library offers a framework in which optimization can be carried out, as well as a selection of algorithms for both single and multi-objective optimization problems. All elements of the PaGMO library can be verified by built-in unit tests.

The PaGMO algorithms are focused on global optimization. However, algorithms from the NLOPT and IPOPT local optimization libraries are included in the installation of PaGMO. The PaGMO installation that can be obtained as a part of Tudat also features a number of example applications for space mission optimization problems.

[1] TU Delft Astrodynamics Toolbox. Available at: `http://tudat.tudelft.nl/`

Similar to Tudat, the PaGMO installation can be verified by running the built-in unit tests after the installation. If no errors are obtained, the algorithms, problems and other elements available in PaGMO are also proven to be successfully verified.

In this research the MOEA/D and MHACO algorithms from PaGMO were used. Additionally, the interface and lower level links were used for the development and implementation of the SCEA algorithm. A high-level interface from PaGMO was used to set up the optimization problem and allows for simple specification of the population size, the number of generations, the search space boundaries and the constraint values. Lastly, it should be noted that for all utilized PaGMO features, unit tests were run and successfully completed.

6.2. Software Architecture

As described before, the software that was developed for this research is largely created from links between existing Tudat and PaGMO elements. To identify the required areas of development, a software architecture was made as seen in Figure 6.1.

Three main modules can be identified in the software architecture. First, the environment includes all environment models that were described in Chapter 3 and provides their information to the propagation algorithm. Secondly, the guidance module takes a vector of nodes as input and is called at each point of integration to interpolate the current guidance values. Lastly, the state propagation module combines the input of the environment and guidance and calculates the state of the vehicle.

6.2.1. Environment Module

The environment module of the simulator is responsible for obtaining the required parameters of the environment for the instantaneous vehicle state. A schematic overview can be found in Figure 6.2. The inputs to the system are the vehicle state and time. The value of r_i is used to determine the gravity, the local air density and the local temperature. These are then used to calculate the Mach number, before everything is sent to the output. The velocity is used to determine the local dynamic pressure of the vehicle.

6.2.2. External Forces

The external forces determine the motion of the vehicle through the atmosphere. The module of the simulator that calculates these forces is presented in Figure 6.3. The input is again the vehicle state, but also the vehicle mass, reference area and its aerodynamic coefficients. It is now important to note that within the external forces module a large number of reference frame conversions takes place. Eventually, all forces have to be converted to the inertial frame $\mathbf{F_I}$. However, the aerodynamics are defined in the aerodynamic reference frame $\mathbf{F_A}$ and the gravity in the vertical reference frame $\mathbf{F_V}$. The conversions are performed through the Euler angles that were described in Chapter 3. Note that the simulator is capable of including thrust and thus the thrust reference frame $\mathbf{F_T}$ and the commanded thrust T_c are also included.

6.2.3. Propagation Module

Finally, the propagation module is used to generate a trajectory based on the commanded guidance and the environment. The module takes the current state as inputs. It then finds the commanded attitude and trim conditions from the guidance algorithm, from which the aerodynamic coefficients can be determined. These are given to the previously described external forces module, which generates the instantaneous aerodynamic and gravity forces. The acceleration of the vehicle can be then determined by dividing the resultant of these accelerations by the vehicle mass. The state derivative is then determined through and passed on as the output of the propagation module. This forms the input of the integrator.

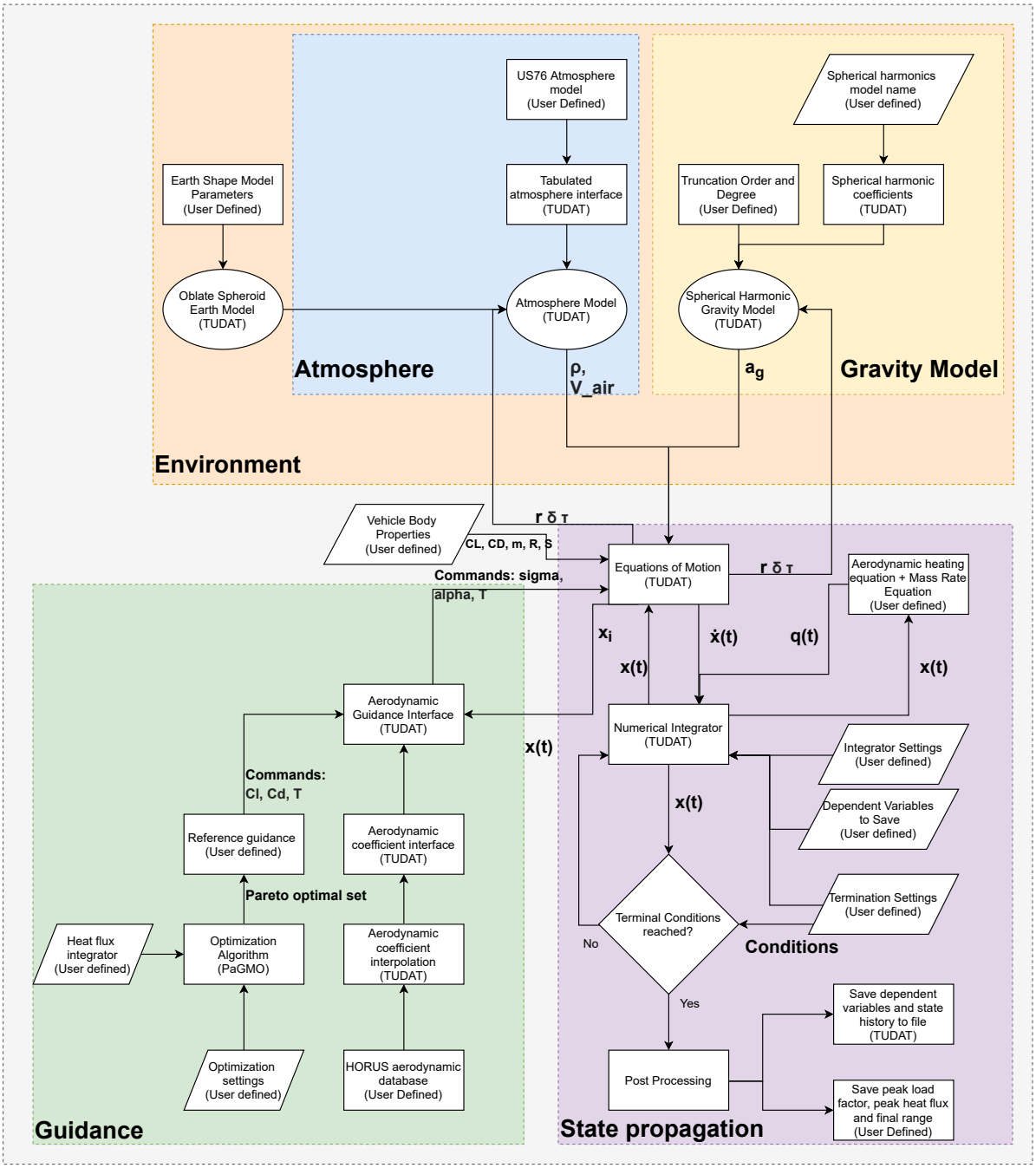


Figure 6.1: Top level overview of the simulator.

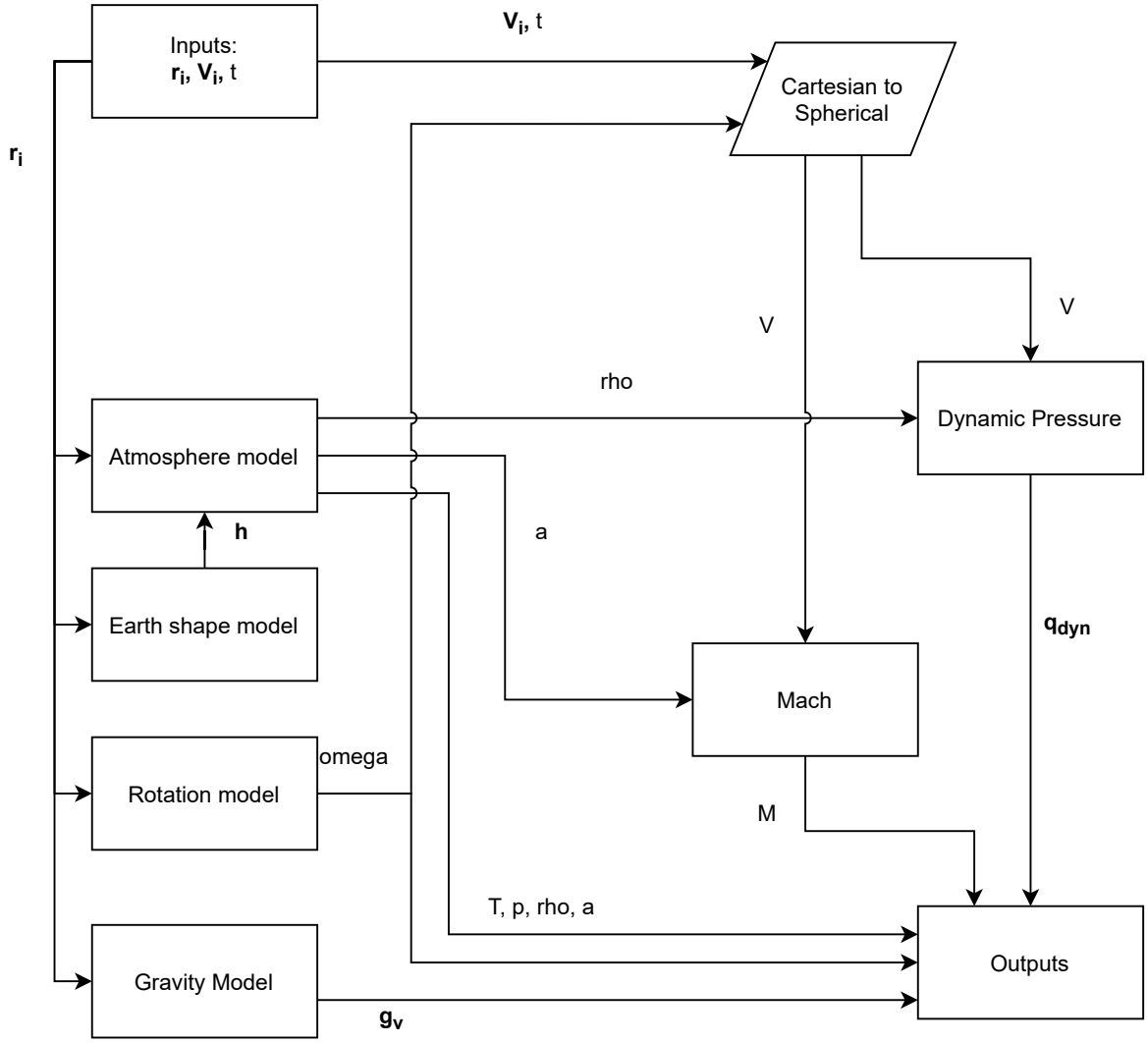


Figure 6.2: Layout of the environment module of the simulator.

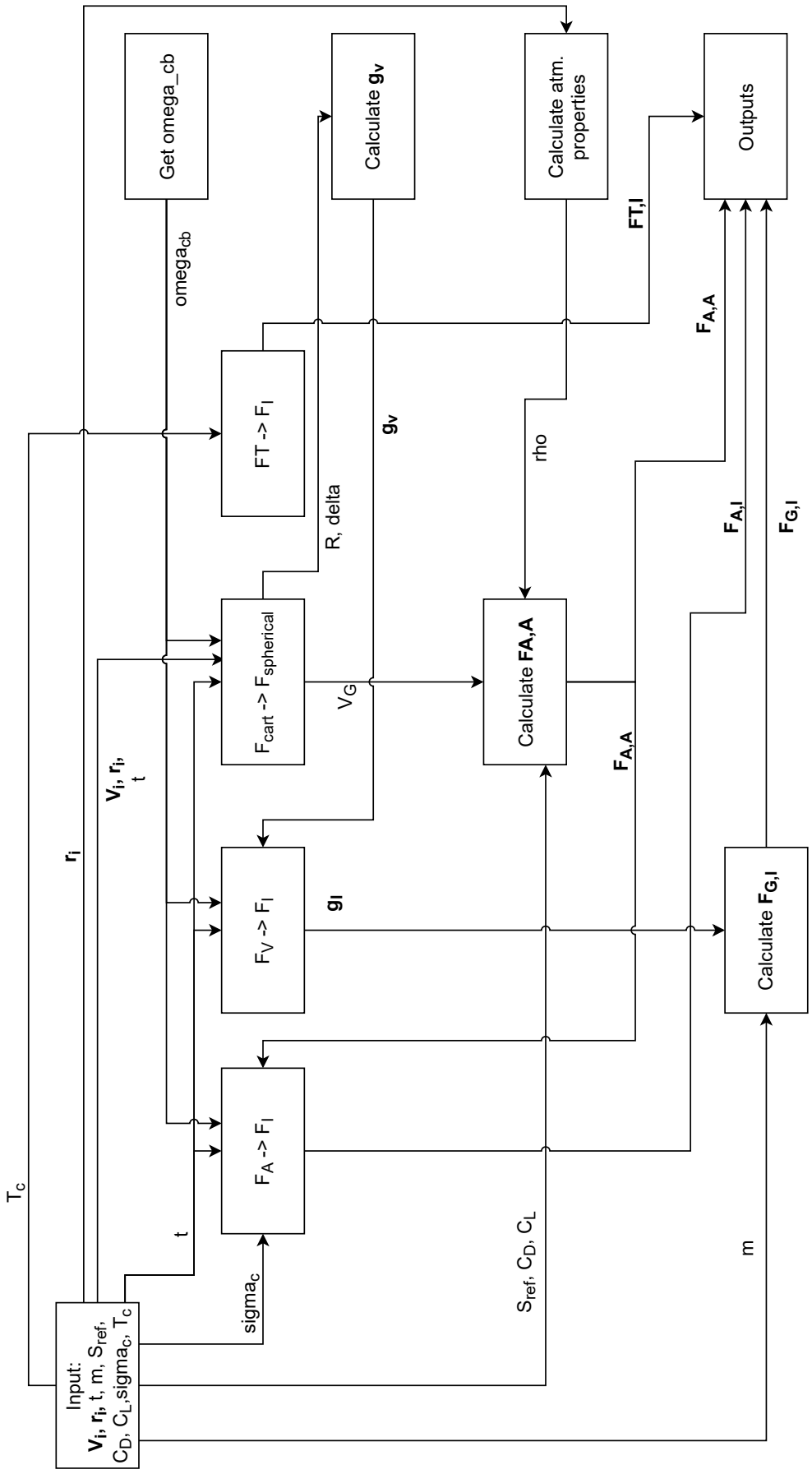


Figure 6.3: Lay-out of the external force module in the simulator.

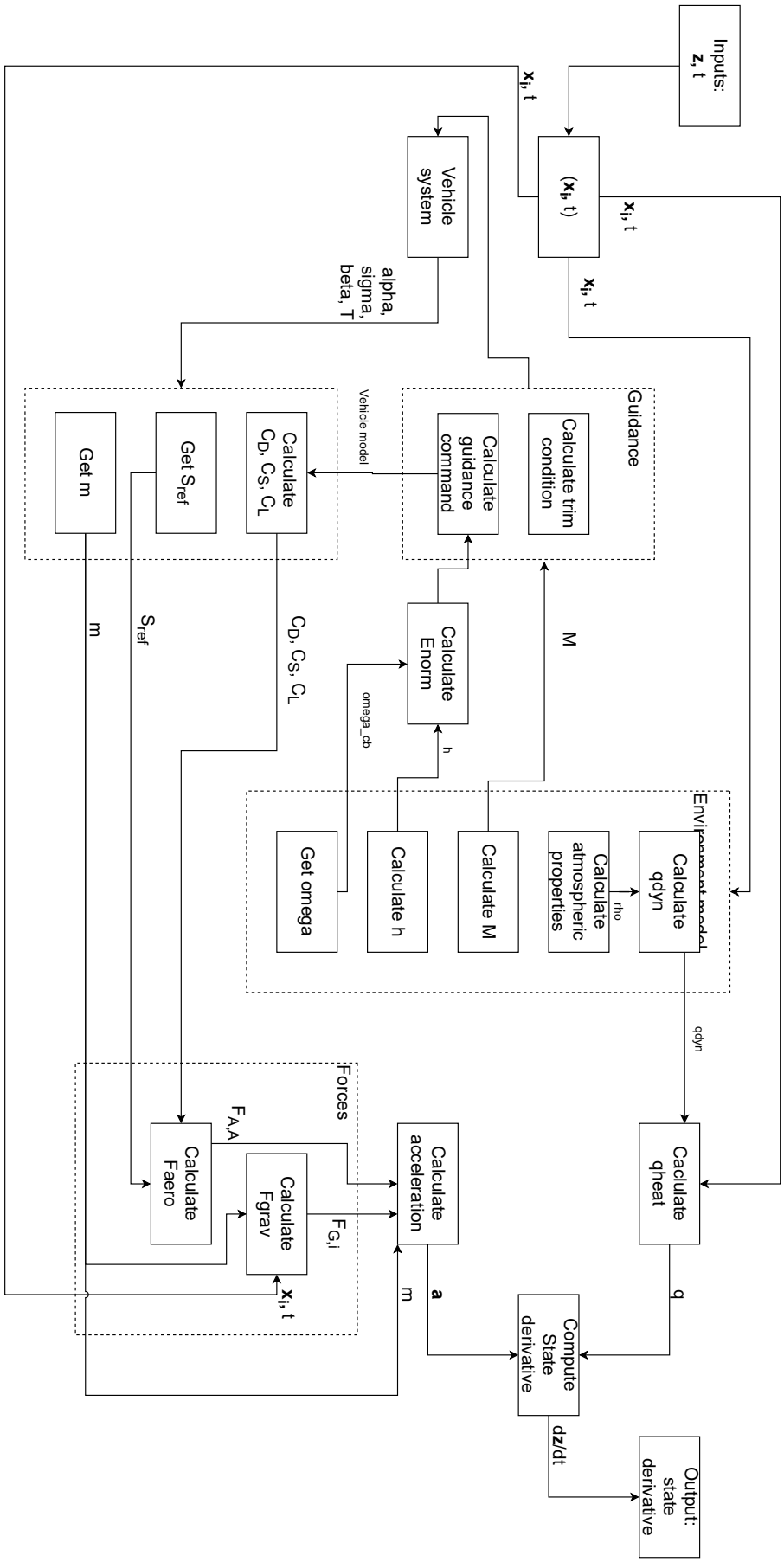


Figure 6.4: Lay-out of the propagation module.

7

Verification and Validation

The following chapter discusses the verification of the software that was developed for this research. First, the numerical methods and models will be shortly discussed in Section 7.1 and Section 7.2, respectively. The structured chromosome evolutionary algorithm will be discussed in Section 7.3. Lastly, the simulator verification will be presented in Section 7.4.

7.1. Numerical Methods

The numerical methods that are used in this research are mainly available in the Tudat and PaGMO bundles. For these methods, built-in unit tests are available.

Root finding
The bisection root finding method that has been applied in this research can be tested simply by letting it find the root of the function $y = 1 - x$. The method successfully finds the root at $x = 1$.

Interpolation
The interpolation methods that are used are all successfully tested against the unit tests that are available in Tudat.

Integration
The Runge-Kutta 4(5) integration method is available in Tudat. The unit tests compare results of five numerical integrations against tabulated reference data presented in Burden and Faires (2001). The unit test was run successfully and thus the method is determined to function correctly.

MOEA/D
The MOEA/D algorithm is included in PaGMO, along with its unit tests. These unit tests are successfully run and completed, thus verifying the proper functioning of the MOEA/D algorithm.

MHACO
The MHACO algorithm is tested similarly to the MOEA/D algorithm. Unit tests on the specific member functions are executed and the algorithm is verified by testing it on well-known optimization problems. All tests were passed successfully, thus verifying correct implementation of the MHACO algorithm.

7.2. Physical Models

Next, the different models that are implemented in the off-the-shelf software need to be verified. These methods also have built-in unit tests.

Atmosphere Models
Both the exponential and US76 atmosphere models are verified by comparing the pressure, atmospheric density and temperature against tabulated verification data.

Earth Shape Model
The shape model is verified by determining the altitude from a Cartesian position and by verifying that the correct equatorial radius is returned by the function.

Gravity Model
The spherical harmonics gravity model is verified by comparing the gravitational parameter to a reference value and by obtaining the gradient of the gravitational potential at a test position.

Aeroheating Model
The heat transfer model is tested against the more accurate Fay Riddell Heat Flux model, both of which are implemented in Tudat. The Fay Riddell model is in turn verified against a pre-computed value that is known to be correct. Another test is implemented that verifies that the heat flux is zero for a value of zero for either the air density or airspeed. All tests were passed without complications.

7.3. Structured Chromosome Evolutionary Algorithm

As the structured chromosome method was developed separately for this research, no standard verification methods are available in Tudat or PaGMO. Below, the selected verification method will be described, followed by the results that were obtained when verifying the algorithm.

7.3.1. Test Problem Description

The selected set of problems was selected from Li and Deb (2017) for its simplicity and suitability for variable chromosome length algorithms. The focus of these problems lies on multi-objective optimization for algorithms with a chromosome length up to N. A visual representation of the problems that will be described can be found in Figure 7.1. To obtain a well-defined Pareto front shape, a shape function is selected as follows:

$$\begin{pmatrix} \alpha_1(x) \\ \alpha_2(x) \end{pmatrix} = \begin{pmatrix} x \\ 1-x \end{pmatrix} \tag{7.1}$$

which describes a line segment between $(0,1)$ and $(1,0)$. A distance function is added which offsets the solution of an individual from the line segment. This function is defined as:

$$g_i(x,y,L(x)) = \sum_{j=1}^{L(x)} \left[y_j - \sin\left(\frac{L(x)}{2N} \times \pi \right) \right]^2 \tag{7.2}$$

where

$$L(x) = 1 + H(x) \tag{7.3}$$

It should be noted that $H(x) = \in \{0,\ldots,N-1\}$. $H(x)$ is defined such that a test problem with variable length $(m-1+L(x))$:

$$H(x) = \left\lfloor \frac{\theta(x)}{\theta_{\max}} \times (N-1) \right\rfloor \tag{7.4}$$

Two angles are required in this equation, $\theta(x)$ and θ_{max}. Li and Deb (2017) report that small values of θ are easier in convergence compared to larger values. Based on the definition of θ, three test problems are formulated:

- For formulation 1, let $\alpha(x) = (\alpha_1(x), \alpha_2(x))$, $v = (0,1)$ and $\theta_{max} = \frac{\pi}{2}$, then

$$\begin{aligned} \theta_1(x) &= \arccos\left(\frac{\langle \alpha(x), v \rangle}{\|\alpha(x)\| \cdot \|v\|} \right) \\ &= \arccos\left(\frac{1}{\sqrt{\alpha_1(x)^2 + \alpha_2(x)^2}} \right) \end{aligned} \tag{7.5}$$

The visual representation of the test problem is found in Figure 7.2(a). The optimal solutions near a value of $f_2 = 1$ are easier to find for the algorithm compared to the solutions near $f_1 = 1$. This can be explained by the definition of the variable $L(x)$, representing the number of distance functions, as more distance functions are present in the solution near $f_1 = 1$.

- For formulation 2, let $\alpha(x) = (\alpha_1(x), \alpha_2(x))$, $v = (1,1)$ and $\theta_{max} = \frac{\pi}{4}$, then

$$\begin{aligned}\theta_2(x) &= \arccos\left(\frac{\langle \alpha(x), v >}{\|\alpha(x)\| \cdot \|v\|}\right) \\ &= \arccos\left(\frac{1}{\sqrt{\alpha_1(x)^2 + \alpha_2(x)^2}\sqrt{2}}\right)\end{aligned} \tag{7.6}$$

The second test problem is visualized in Figure 7.2(b). Due to the changed definition of θ_{max}, the solutions with the most distance functions are now found in the middle of the Pareto front, where $v = (1,1)$, thus making these solutions harder to find for the algorithm.

- For formulation 3, let $\alpha(x) = (\alpha_1(x), \alpha_2(x))$ and $\theta_{max} = \frac{\pi}{8}$ and

$$\begin{aligned}v_a &= \left(\frac{\sqrt{2-\sqrt{2}}}{4}, \frac{\sqrt{2+\sqrt{2}}}{4}\right) \\ v_b &= \left(\frac{\sqrt{2+\sqrt{2}}}{4}, \frac{\sqrt{2-\sqrt{2}}}{4}\right)\end{aligned} \tag{7.7}$$

with

$$\theta(x) = \min\{\theta_1(x), \theta_2(x)\} \tag{7.8}$$

and

$$\theta_{3,i}(x) = \arccos\left(\frac{\langle \alpha(x), v_i \rangle}{\|\alpha(x)\| \cdot \|v_i\|}\right), i = 1,2 \tag{7.9}$$

The visualization of test problem 3 is found in Figure 7.2(c). The angles θ_1 and θ_2 are taken with respect to the vectors v_1 and v_2. The easy solutions are found around these vectors, while the harder solutions are found at the parts of the Pareto front that lie the furthest away from these angles, thus the middle of the Pareto front and the two extremities. Again, in this region the value of L reaches its maximum.

In addition to these problems, also two triple objective problems have been defined in (Li and Deb, 2017). For these problems, the shape function is:

$$\begin{bmatrix} \alpha_1(x) \\ \alpha_2(x) \\ \alpha_3(x) \end{bmatrix} = \begin{bmatrix} x_1(1-x_2) \\ x_1 x_2 \\ 1 - x_1 \end{bmatrix} \tag{7.10}$$

which describes a unit simplex in $[0,1]^3$. The distance functions are maintained as in Equation (7.2). The following two triple objective problems are defined based on two examples of $L(x)$ and again kept similar to the two objective problems by using a method based on angles.

- For problem formulation 4 let $\alpha(x) = (\alpha_1(x), \alpha_2(x), \alpha_3(x))$ and $\theta_{max} = \arcsin\frac{\sqrt{3}}{3}$ after which the angles are computed as:

$$\begin{aligned}\theta_1(x) &= \arcsin\left(\frac{a_1(x)}{\sqrt{\alpha_1^2 + a_2^2 + \alpha_3^2}}\right) \\ \theta_2(x) &= \arcsin\left(\frac{\alpha_2(x)}{\sqrt{\alpha_1^2 + \alpha_2^2 + \alpha_3^2}}\right) \\ \theta_3(x) &= \arcsin\left(\frac{\alpha_3(x)}{\sqrt{\alpha_1^2 + a_2^2 + \alpha_2^2}}\right)\end{aligned} \tag{7.11}$$

The minimal angle $\alpha(x)$ is given by:

$$\theta(x) = \min\{\theta_1(x), \theta_2(x), \theta_3(x)\} \tag{7.12}$$

The visualization of the test problem is given in Figure 7.2(d). It can be be proven that $H(x)$ has a minimal value of zero, when $\theta(x)$ reaches zero. For this case, $\alpha(x)$ is located on the boundary of the simplex Pareto front. When $\theta(x)$ reaches its maximum value, $H(x)$ reaches its maximum value of $N-1$ and $\alpha(x)$ is located at the center of the pareto front. Thus, the solutions at the center of the Pareto front are harder to find compared to the solutions that lie on the boundary.

- For problem formulation 5, let $v_1 = (\frac{2}{3}, \frac{1}{6}, \frac{1}{6})$, $v_2 = (\frac{1}{6}, \frac{2}{3}, \frac{1}{6})$, $v_3 = (\frac{1}{6}, \frac{1}{6}, \frac{2}{3})$ and $\theta_{max} = \frac{\sqrt{6}}{3}$ with the angles θ defined as:

$$\begin{aligned} \theta_1 &= \arccos\left(\frac{\langle \alpha, v_1 \rangle}{\|\alpha\| \|v_1\|}\right) \\ \theta_2 &= \arccos\left(\frac{<\alpha, v_2\rangle}{\|\alpha\| \|v_2\|}\right) \\ \theta_3 &= \arccos\left(\frac{\langle \alpha, v_3 \rangle}{\|\alpha\| \|v_3\|}\right) \end{aligned} \tag{7.13}$$

The angle angle $\theta(x)$ is again found using Equation (7.12).

The visualization of this last test problem can be found in Figure 7.1(e).

The original set of problems does not include constraints. To be able to fully verify the method, constraints have to be added to the problem. The angle θ was selected to be constrained to limit the solution space of the problems. For all problems a simple definition of the constraint is applied to the values of $\alpha_1(x)$ by stating the following constraint:

$$\alpha_1(x) \leq 0.5 \tag{7.14}$$

7.3.2. Verification Results

The SCEA algorithm was tested against the first four test problems. The results are presented in Figure 7.2. As can be observed, the SCEA algorithm is able to find the solutions along the Pareto front for all problems. For the two objective problems, this is achieved in 10 generations. The triple objective test problem converges slightly slower, although the solution lie on the Pareto front after 20 generations. Strong convergence is thus shown for this set of test problems.

Given that the algorithm is able to find the solutions along the Pareto front for all cases, it is proven that the algorithm functions correctly. To determine its exact performance, more effort has to be invested into its performance for other problems, specifically into the convergence.

7.4. Simulator Verification

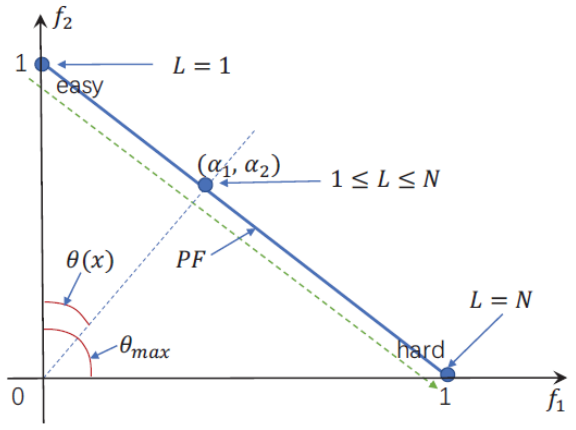
With all modules successfully verified, the combined system still needs to be verified to ensure that the implementation and combination of the individual modules are successful. As no actual flight data is available for HORUS, output data from a high-accuracy re-entry simulator developed by E. Mooij in FORTRAN was used as reference. As the guidance algorithm in the FORTRAN simulator was different from the one applied in this research, the attitude angles were commanded with respect to time instead of the normalized energy.

The following specifications were used to generate the verification data:

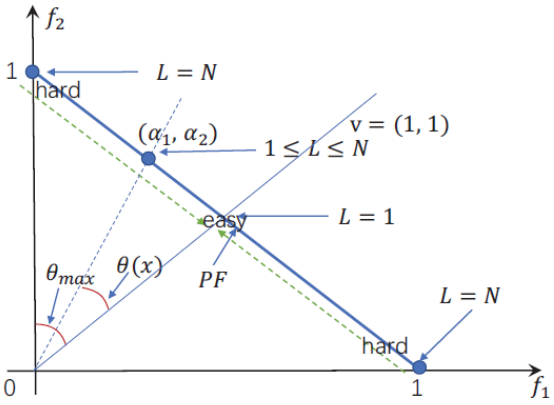
- Atmosphere model: US76
- Earth model: Spherical, rotating Earth
- Gravity Field: Central gravity with J_2 term.
- Initial State: $h = 122$ km, $\tau = -106.7°$, $\delta = -22.3°$, $V_{G,0} = 7.4355$ km/s, $\gamma_0 = -1.43°$, $\chi = 70.75°$
- Initial Attitude: $\alpha_{A,0} = 40°$, $\beta_{A,0} = 0°$, $\sigma_{A,0} = 0°$

The entry simulator that was developed for this research uses the same specification with a few exceptions:

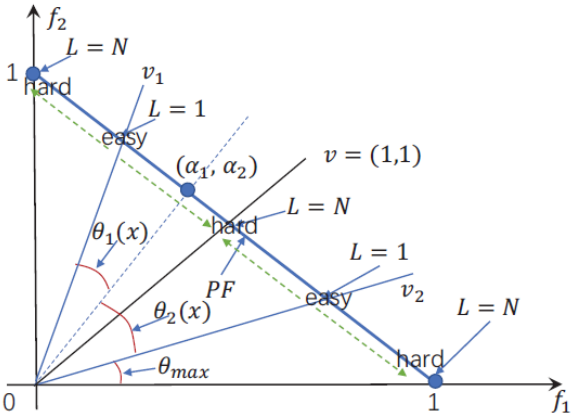
- Rotational motion from the vehicle is not simulated, which has no effect on the results as the attitude of the vehicle is directly prescribed from the guidance and assumed to be directly applied as no control system is present.
- The elevator deflection δ_e is not taken into account. The effect of this discrepancy is minimal as the elevators are only used near the end of the trajectory where trimmed flight is not possible.
- The altitude-dependent drag increment was not included in the computation of the aerodynamic coefficients. This effect can be neglected as it would only become relevant at altitude below 20 km.



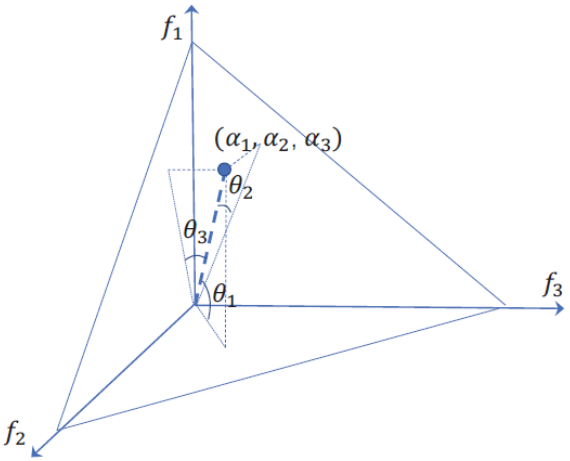
(a) Test problem 1



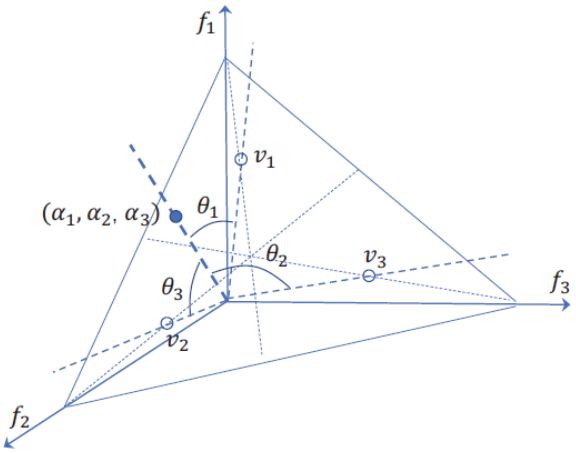
(b) Test problem 2



(c) Test problem 3



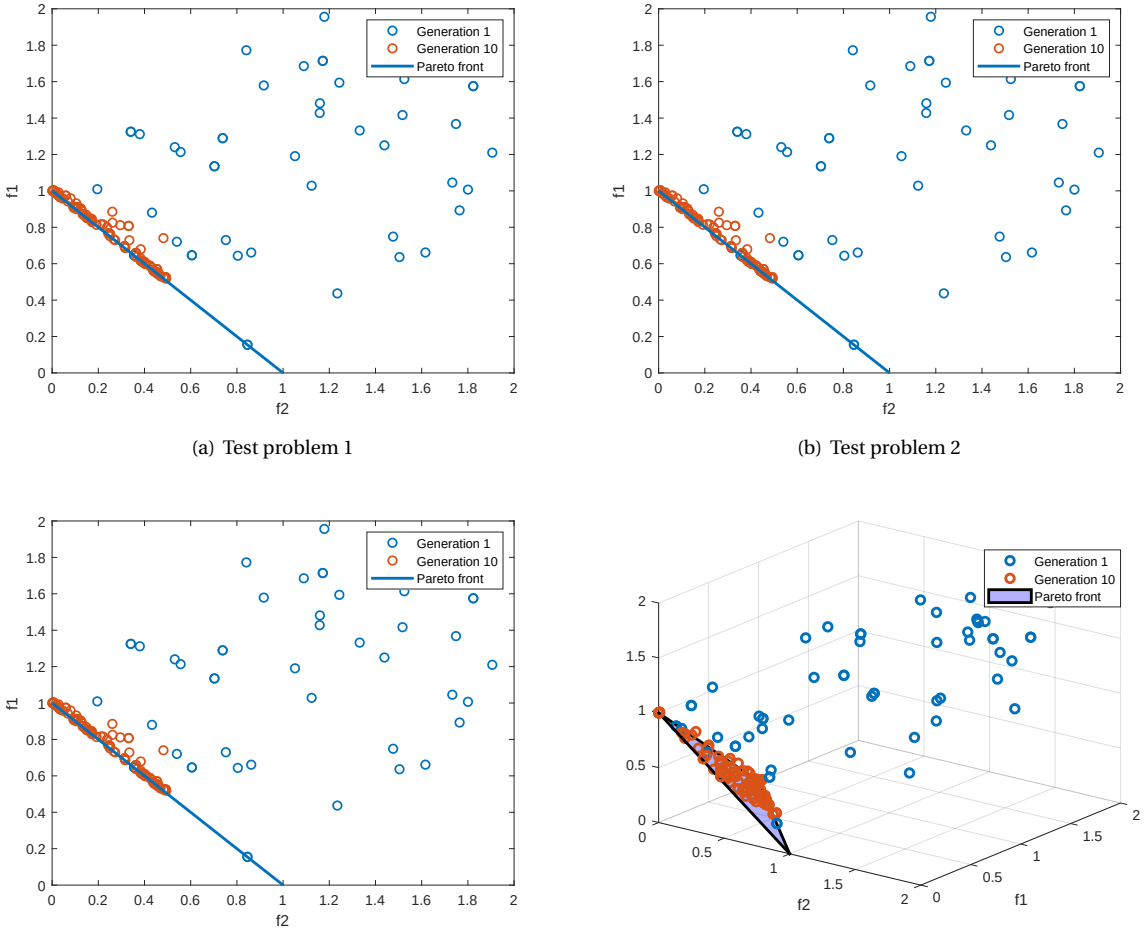
(d) Test problem 4



(e) Test problem 5

Figure 7.1: Definition of the five test problems as presented by Li and Deb (2017).

The results from the verification are presented in Figure 7.3.
Observation of figures Figure 7.3(a) and Figure 7.3(b) shows that the altitude and velocity match nearly



(a) Test problem 1

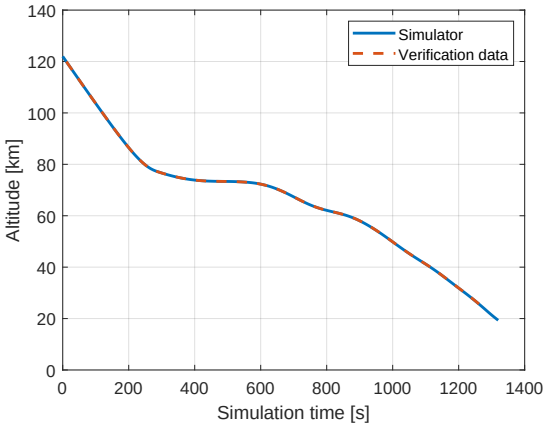
(b) Test problem 2

(c) Test problem 3

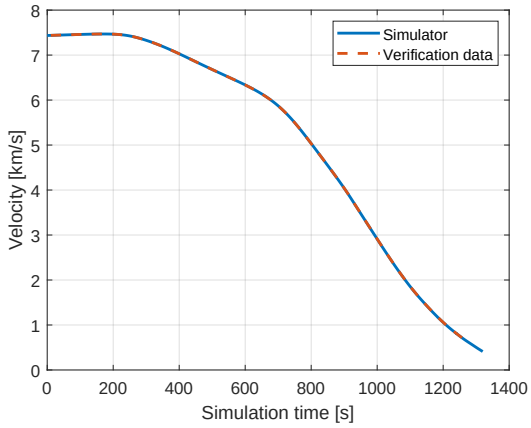
(d) Test problem 4

Figure 7.2: Optimization results for the test problems using the SCEA algorithm.

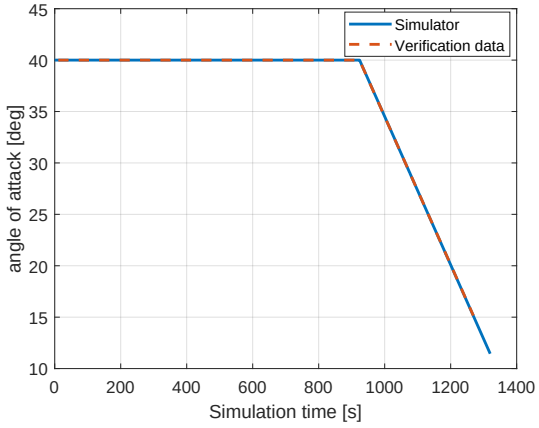
exactly between the simulation and verification data. The attitude angles also show a close match between both data sets as observed in Figure 7.3(c) and Figure 7.3(d), indicating that the guidance correctly prescribes the commanded attitude angles. It should be noted that the angle of sideslip is not shown here, since it does not deviate from zero for both sets of data. The trajectory in terms of latitude and longitude is shown in Figure 7.3(e) and Figure 7.3(f), respectively. Again, only minute deviations are observed. These can be accredited due to the small difference in the bank angle profile, caused by the difference in time steps between the verification data and the simulation data. Finally, it can be assumed that the simulator functions correctly as there are no significant discrepancies compared to the verification data.



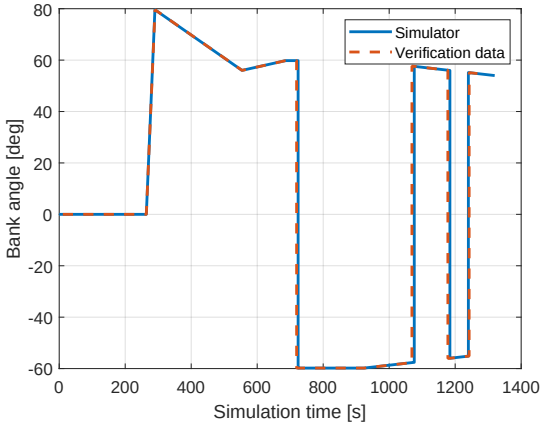
(a) Altitude profile



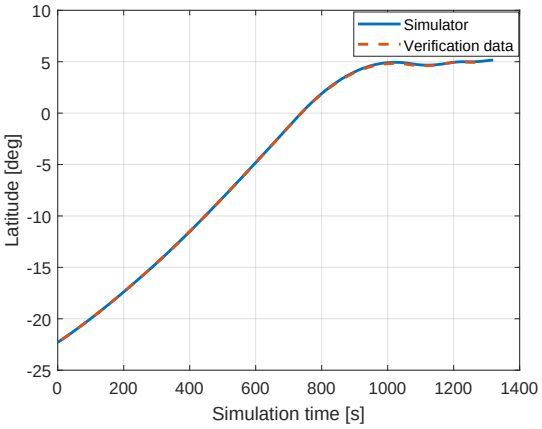
(b) Velocity profile



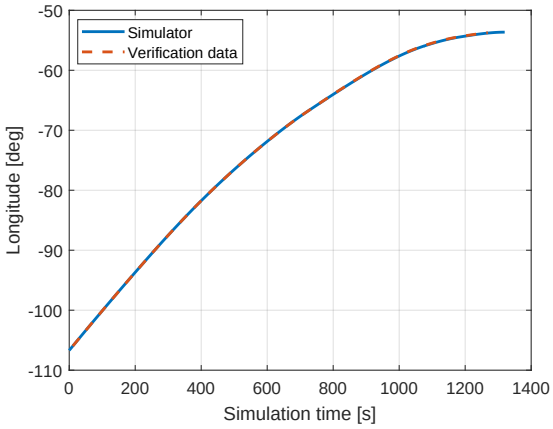
(c) Angle of attack profile



(d) Bank angle profile



(e) Latitude profile



(f) Longitude profile

Figure 7.3: Results of the HORUS simulator verification.

III

Part III: Research

8

Analytical Results

In the following chapter the analytical methods that were presented in Section 3.5 are investigated for their applicability to the optimization problem. Especially, the search space to which they can be applied is of interest, as this helps in setting the bounds of the optimization search space. Additionally, attention will be given to the errors in the results. Results for the skipping flight are presented in Section 8.2 and results for gliding flight are discussed in Section 8.1.

8.1. Gliding Flight

This sections deals with the results that were obtained by the second order equations for gliding flight. First, the applicability of the method will be tested and the verification performed. Secondly, the accuracy of the method compared to the numerical simulator will be tested for a larger range of parameters, to determine the search space that will be allowed during optimization.

8.1.1. Initial Conditions

The analytical second-order equations for gliding flight are stated to be highly accurate compared to numerical trajectory simulations. The first step in this thesis was to verify this claim. To do so, first simulations were run were the initial state for the analytical and numerical simulations were set equal. The following initial conditions were used for this simulation:

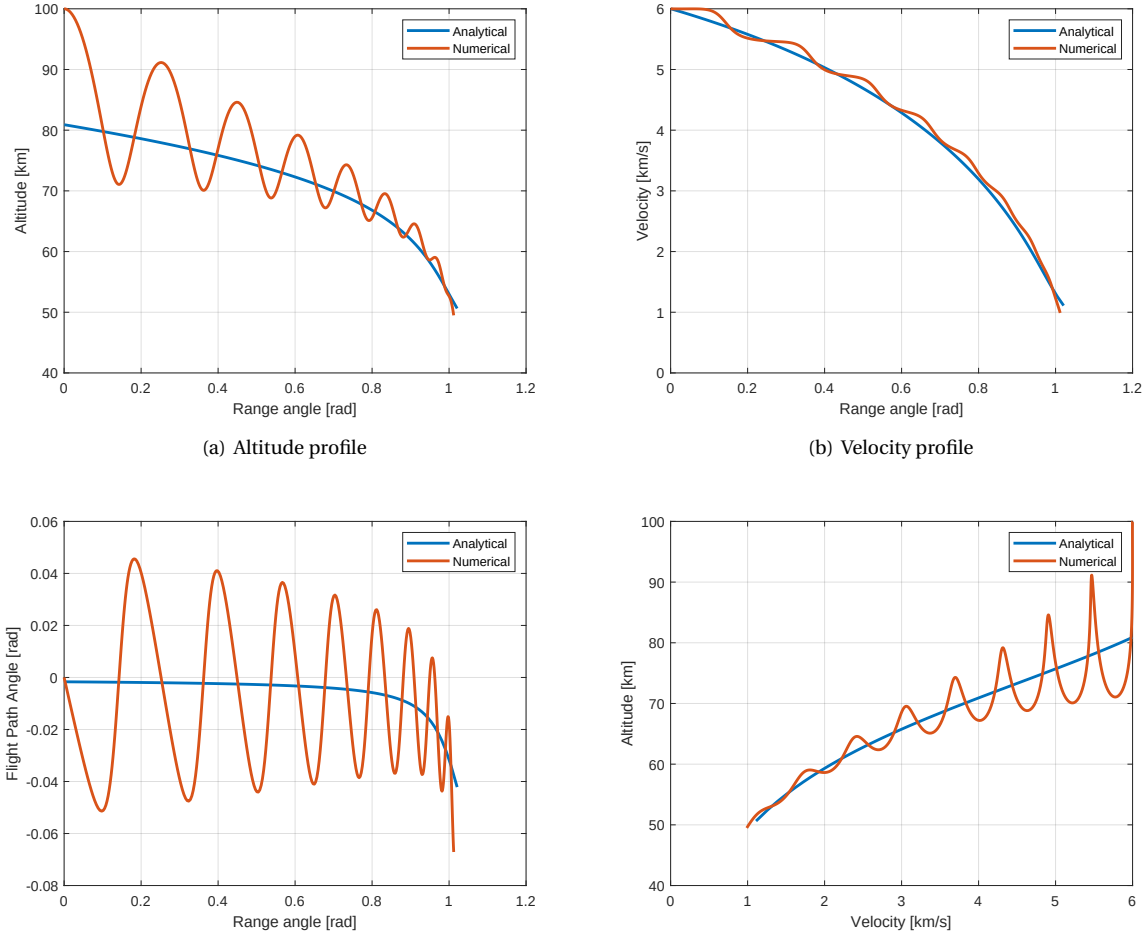
$$h = 100 \text{ km}$$
$$V = 6000 \text{ m/s}$$
$$\gamma = 0.0°$$

The results of these simulations are shown in Figure 8.1. The results show a number of discrepancies between the methods. First of all, the initial altitude of the analytical simulation is significantly lower than the numerical and input altitudes. To ensure correct implementation, all equations were manually verified to be correctly implemented and thus this change in initial altitude has to be accredited to the method. Secondly, strong oscillations of the flight-path angle are observed in the numerical method, whereas the analytical simulations follow a smooth decrease of the flight path angle.

Interestingly, the analytical method does approximate the average of the trajectory, although it should be noted that no exact average is obtained and the analytical method lies below the exact average.

The objective of this research is to use the analytical expressions for constrained optimization of re-entry flight. As stated before these trajectories are the heat flux, g-load and g-load rate. Due to the smooth trajectory that is presented by the analytical trajectory, important information on the extremes of these constraints is not known. Especially since the maximum values for these constraints are found in the peaks of the oscillation, the analytical methods can, in this form, not be used for a constrained optimization problem.

Further investigation was put into the initial conditions of the trajectory by forcing the analytical methods to meet the initial altitude that was given as an input. The results show that for such a simulation, it is possible for the analytical to start at the initial altitude, although the initial velocity that is obtained from the relations is



(a) Altitude profile

(b) Velocity profile

(c) Flight Path Angle profile

(d) Altitude-Velocity profile

Figure 8.1: Results of the second-order method for gliding entry for a strongly oscillating trajectory.

much higher compared to the numerical and input velocities. Also, the initial range angle values are negative and only become larger than zero when the analytically obtained velocity is equal to the initial conditions.

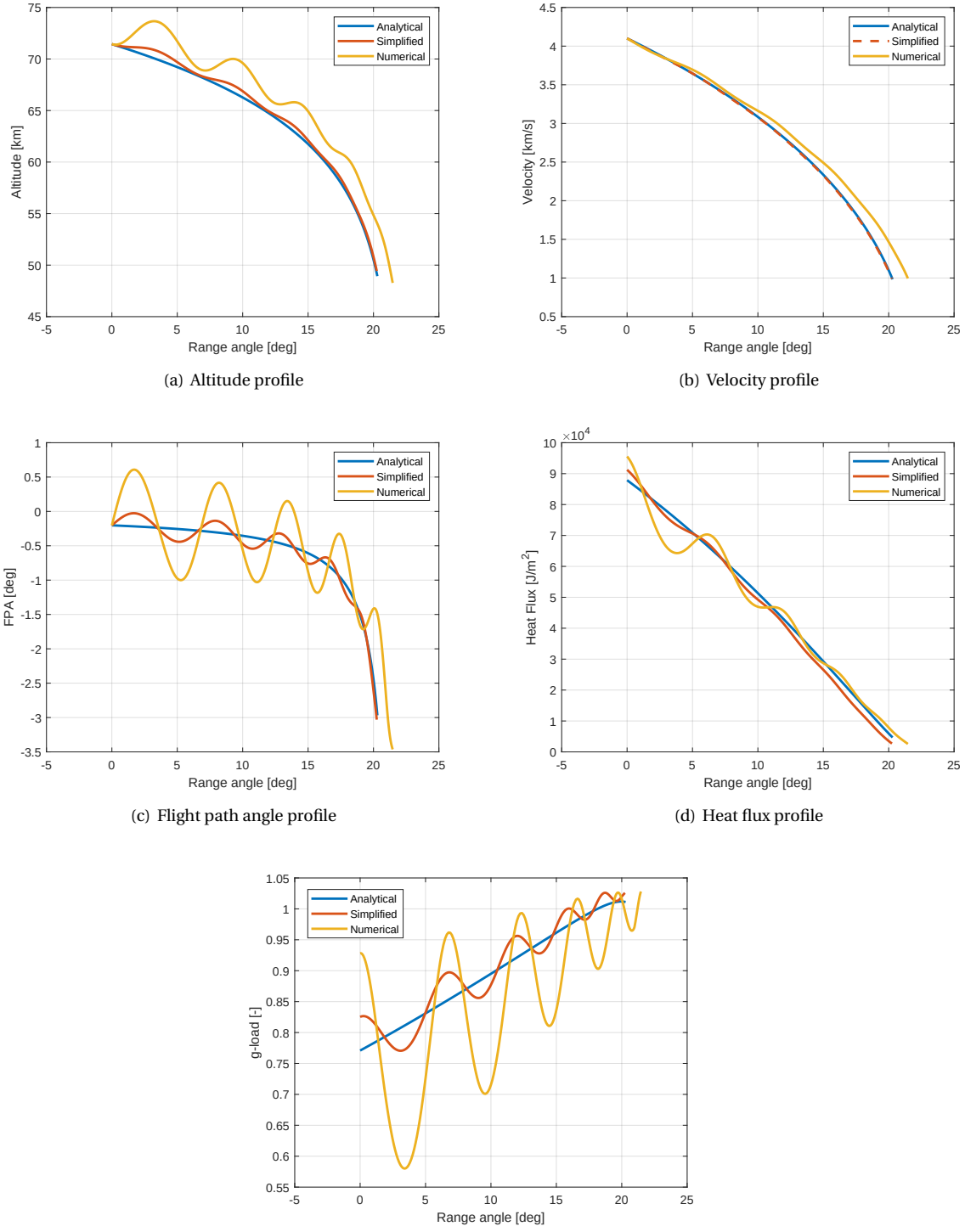
The early results thus show that the analytical equations of motion cannot be trivially applied to a re-entry mission. Vinh et al. (1993) do provide two relations that yield an initial velocity and flight path angle. For clarity they are repeated here:

$$u_E = \frac{1}{1+k_1}$$

$$\sin\gamma_E = \frac{2k_1(1+k_1)}{E[2(1+k_1)^2 + \frac{r_E}{H_s}k_1]}$$

The equations only depend on the values of E, or the current L/D, r_E, the entry altitude, and k_1, the parameter describing the vehicle parameters and entry conditions. The entry conditions that are represented in k_1 are the initial altitude r_E and the density at this altitude, ρ_E. The latter in turn also depends only on r_E, since the exponential atmosphere model is used in the derivation of the analytical equations. With only a selected altitude, it is thus possible to find the accompanying velocity and flight path angle for the analytical method.

The results for a sample trajectory in this set of simulations is found in Figure 8.2. Now, a close match between the analytical and numerical simulations is observed. The velocity profile with respect to the range angle shows excellent matching in particular, as observed in Figure 8.2(b). However, the oscillations are still



(a) Altitude profile

(b) Velocity profile

(c) Flight path angle profile

(d) Heat flux profile

(e) G-load profile

Figure 8.2: Results of trajectories with initial conditions set to the second order initial conditions.

present in both the numerical simulations with the simplified and extended environment. These oscillations have little effect on the altitude profile as observed in Figure 8.2(a). Between the simplified environment and the analytical results negligible differences are observed. The extended numerical model instead shows a lofting trajectory and as a consequence the final range angle differs more strongly from the analytical results.

The cause of this difference is clear, as the extended numerical model takes into account the rotation of the Earth, which increases the airspeed of the vehicle by a factor equal to:

$$\Delta V_E = \omega_E r_E \tag{8.1}$$

where ω_E is the rotation rate of the Earth and r_E is the range between the entry point of the vehicle and the center of the Earth.

The flight-path angle profile still shows more significant differences as the oscillations are more pronounced for this state variable. Still, the analytical method matches closely with the simplified environment model, especially near the end of the trajectory, where larger negative values of the flight-path angle are found.

The profile of the heat flux and g-load constraints are shown in Figure 8.2(d) and Figure 8.2(e), respectively. The the analytical heat flux profile again matches the simplified numerical results closely. Important to note is the undershoot of the analytical method at the start of the trajectory, which would cause the analytical method to find slightly more trajectories that meet the heat flux constraint. The numerical model again shows a larger discrepancy, caused by the early skip in the trajectory, afterwards, it also matches closely with the analytical model.

Finally, the g-load profile still shows larger differences due to the oscillations in the trajectory. The oscillations in both numerical trajectories cause even more pronounced oscillations in the g-load. The analytical method still gives a smooth profile. This leads to the conclusion that the analytical model cannot accurately measure the g-load rate constraint. The maximum obtained g-load still remains close to the numerical simulations, although again the oscillations are the cause for discrepancies. Still, the difference between the analytical and numerical simulations for the maximum g-load is in the order of <5% and still acceptable for optimization purposes.

Overall, the analytical simulations are thus capable of providing accurate results compared to numerical simulations, especially when compared to a simplified numerical environment model. A drawback that should be noted, though, is that the analytical model prescribes the initial conditions that can be used for the simulation based on the entry altitude. When applied to a realistic entry for mission planning, this would result in a required ΔV maneuver to ensure that the vehicle meets the entry conditions. As a consequence, the method might not be applicable to unpowered vehicles, which are not capable of altering their entry state and thus required larger flexibility during trajectory design.

Still, the optimization results of the method might provide useful results and should be further studied. First, the trajectory of both reference vehicles will be discussed below.

8.1.2. HORUS-2B

First, the trajectory of HORUS-2B was estimated using the second order analytical methods. The values of the variables C_{D_0} and K that are used in the analytical equations is however not given for HORUS-2B and first has to be determined. The aerodynamic data from Mooij (1995) can be used along with the lift-drag polar:

$$C_D = C_{D_0} + KC_L^2$$

One should note that the application of the lift-drag polar limits the accuracy of the obtained values for C_{D_0} and K and the obtained relation does not exactly match the data points from the aerodynamic database. Still, to describe HORUS-2B as accurately as possible in the analytical simulator, a regression model was used to obtain the values of C_{D_0} and K.

The data in Mooij (1995) is available with respect to the angle of attack and Mach number. The drag polar will describe the relation between the lift and drag coefficients for a varying angle of attack, in line with the definition given in Anderson Jr. (1999) and a given Mach number. Thus, several definitions will be available for the polar for each given Mach number. The available data for the lift and drag coefficient for HORUS is presented in Figure 8.3 for the available Mach numbers. From investigation of the data, a single polar would not suffice to describe the relation between the two coefficients. However, defining the relation for each Mach number allows for more accurate relations. The obtained results are presented in Table 8.1. For the simulations, an interpolation scheme was set up to find the values of C_{D_0} and K for Mach number values in between the tabulated values.

The results are shown in Figure 8.4 and clearly show a large discrepancy between the analytical and numerical methods. It is important to note that the initial state of both simulations was obtained using Section 8.1.1, which previously showed good results in the verification. In this case, it becomes clear that the

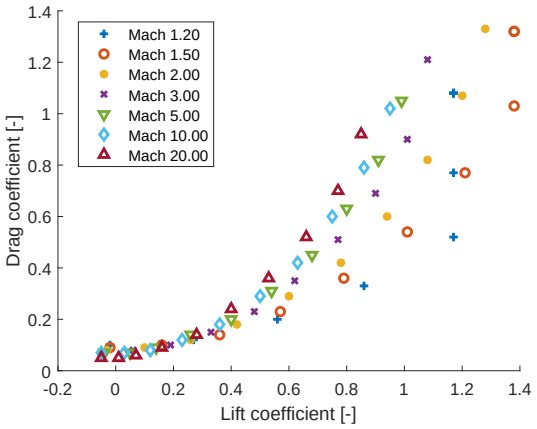


Figure 8.3: Lift and drag coefficient data of HORUS for varying Mach numbers.

analytical method does not perform well for this vehicle and set of initial state parameters. However, when comparing different values for the initial state of HORUS-2B similar or worse differences between both simulations are observed as the initial velocity obtained from the second order method remains too large in combination with the flight path angle of less than 1 degree.

An attempt was made to influence the initial state that resulted from the analytical method to prevent skipping flight for the numerical simulation based on the value of k_1 in Section 8.1.1. This equation clearly shows that as the variable k_1 increases, the value of the initial entry velocity should decrease. The vehicle parameters that are included in k_1 cannot be altered, as these are given for HORUS-2B, which leaves the entry altitude, r_E and the value of the vertical component of C_L. The entry altitude did not prove to cause improvements and the values of the vertical C_L was set to its maximum for the trajectory shown in Figure 8.4.

The main issue that was identified for the analytical method is the very large initial range angle step size. This is can be clearly observed in Figure 8.4(d), where a kink in the heat load profile is shown at a range angle value of approximately 90°. The distance between 0° and this point is covered in a single integration step, although analysis of the equation that describes the range angle in the analytical method has proven that these values are correct. No alteration to the method was found that led to an improvement of this phenomenon and it thus has to be concluded that due to the combination of limitations of the gliding flight method and the vehicle properties of HORUS-2B, the method cannot be used for optimization of the HORUS trajectory.

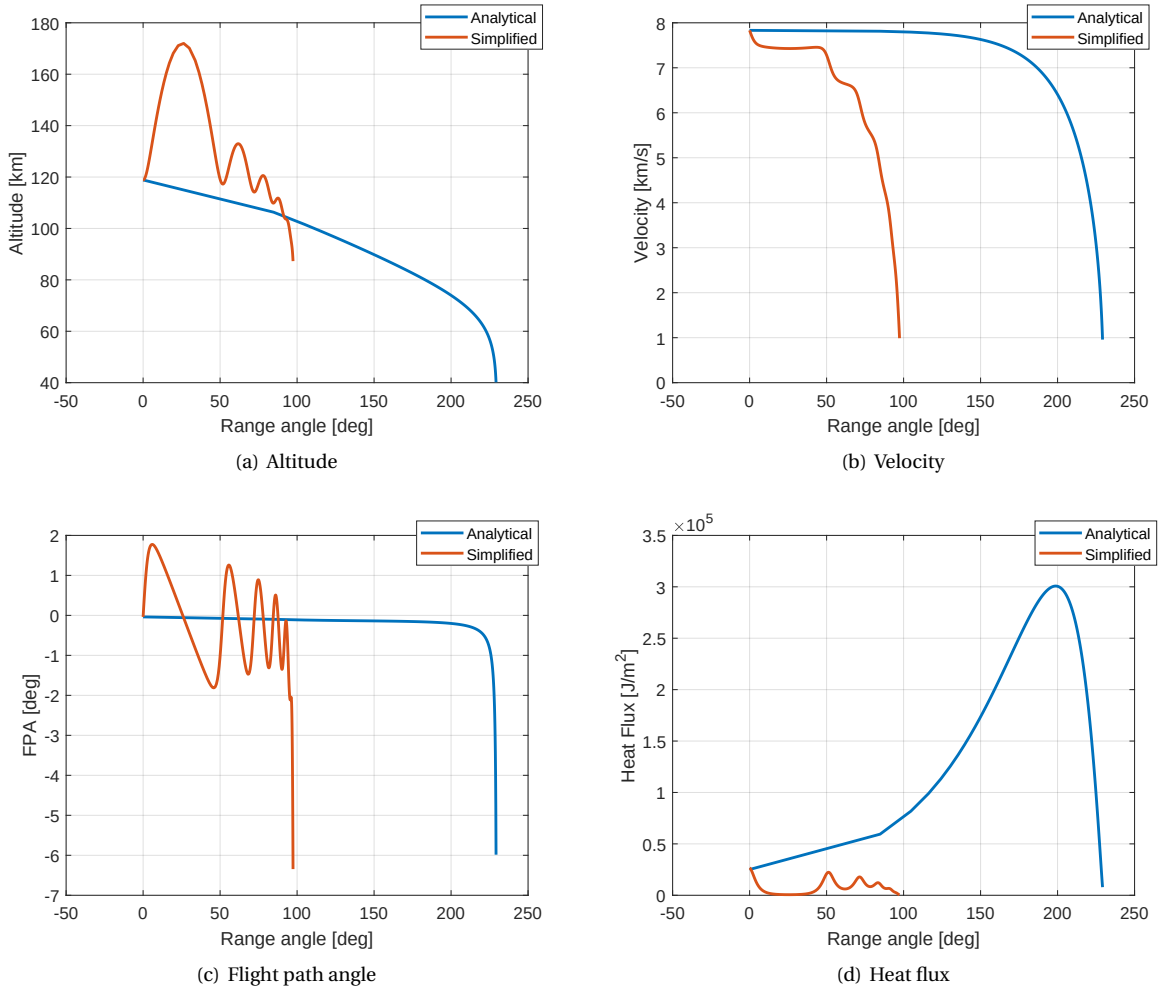
8.1.3. SRM

Figure 8.2 shows trajectories the trajectories of the SRM vehicle compared to a simplified numerical simulator and the simulator with the complete environment as discussed in Chapter 3. The simplified simulator was set to include the same environment as was used in the derivation of the analytical equations of motion and thus has a spherical, non-rotating Earth, exponential atmosphere model and only the central gravity field.

For the SRM vehicle a much closer match between the trajectories is observed and the results of the analytical method lie within an acceptable error margin from the numerical results. The most significant difference between the analytical and complete numerical model can be explained easily when considering that

Table 8.1: Obtained values of C_{D_0} and K for HORUS-2B.

Mach number	C_{D_0}	K
1.20	0.03067	0.6481
1.50	0.04055	0.5907
2.00	0.04582	0.715
3.00	0.04302	0.8864
5.00	0.05561	0.948
10.00	0.05408	1.016
20.00	0.0488	1.147



(a) Altitude (b) Velocity

(c) Flight path angle (d) Heat flux

Figure 8.4: Trajectory of HORUS-2B from the analytical and numerical simulators.

the analytical method does not include the rotation of the Earth. This results in a slightly higher initial airspeed, which results in a higher lift and explains the skip in the trajectory as seen in Figure 8.2(a). For now, this does not significantly influence the result of the research when considering the absence of Earth's rotation when comparing the optimization results. However, to create a more robust mission design tool, it should be taken into account that the Earth rotation has to be compensated for after generating an analytical trajectory.

Figure 8.2(d) and Figure 8.2(e) show the heat flux and g-load over the trajectory, respectively. Both show that the analytical method provides an approximation of the values of both variables over the trajectory, although the discrepancies remain significant. These discrepancies can be accredited to the slight oscillations that are found in the numerical trajectory, which the analytical model is incapable of representing. The results from the analytical method still provide an acceptable approximation of the actual values. It is important to note that especially trajectories that lie close to the constraint values would have to be run through a numerical simulation to verify that they actually satisfy the constraints.

8.1.4. Simplified Environment

The results that were found when comparing the second order glide model to the numerical model for a simplified simulation are presented in Figure 8.5. The results show that the matching between both models is excellent with errors in the final range angle not exceeding 3.0%.

It should be noted that the area presented in the figure is limited by the applicability of the method. Higher velocities cause the analytical model to break down. which can be explained as these velocities exceed the

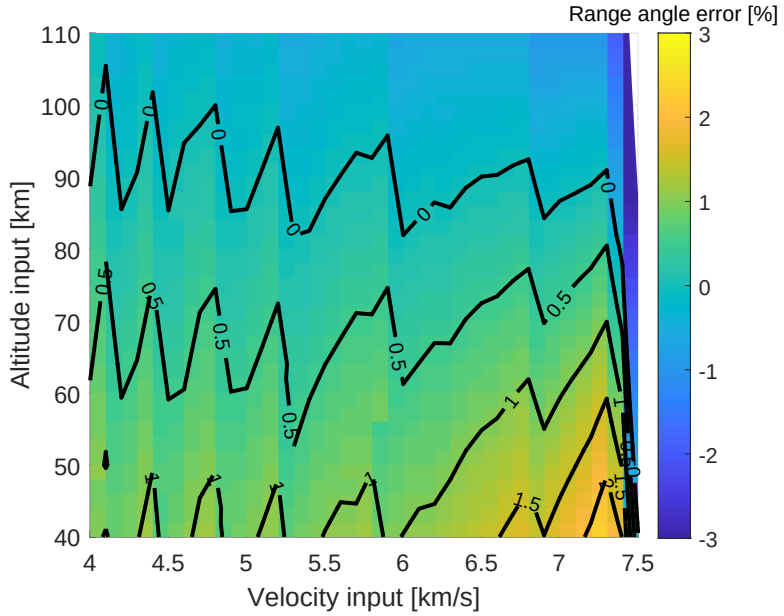


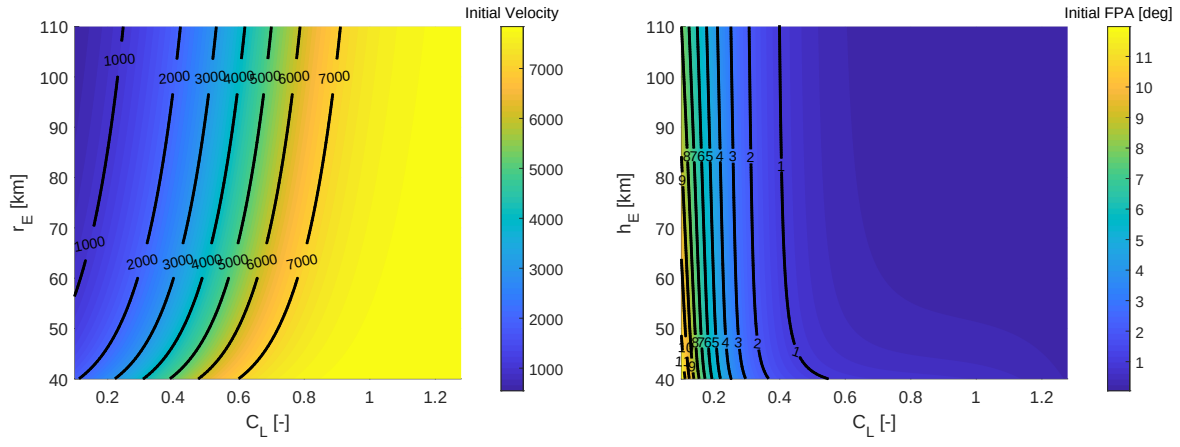
Figure 8.5: Final range angle error between the analytical model and the simplified numerical model.

local circular velocity and would thus result in parabolic trajectories. Since the analytical method does not allow freedom in the flight path angle, it is not possible to shed this velocity by means of atmospheric friction. More importantly, the mathematical definition of the method does not allow for propagation of these trajectories.

Values above the current altitude limit of 110 km cause significant deviations in the final range, but still provide a trajectory. These errors prove similar to the ones that were experienced for HORUS and a skip is observed in the numerical simulation, whereas the analytical trajectory still shows a monotonically decreasing altitude profile.

8.1.5. Initial State

As mentioned before, the initial state of the analytical trajectory is entirely prescribed based on the initial altitude, the vertical lift over drag and the vehicle parameters. To better understand how these influence the trajectory, a grid search for the initial altitude and C_L was performed for the initial velocity and initial flight



(a) Velocity

(b) Flight path angle

Figure 8.6: Grid search results for the initial velocity and initial flight path angle.

path angle. The results of this grid search are presented in Figure 8.6.

For both the velocity and the flight path angle, the effect of the altitude is shown to be small. For the lift coefficient a more pronounced effect can be observed. Still, the flight path angle remains above -1° until a C_L value of 0.5 is reached. Given the maximum C_L of the SRM capsule of 1.2786, the ratio between the vertical and maximum lift coefficient thus is 0.39, which corresponds to a bank angle of 67°.

However, for the region where the flight-path angle starts decreasing a sharp decrease in the velocity is also observed. When the flight-path angle decreases below -1° as discussed before, the initial velocity of the trajectory has already decreased to 3000 km/s, which is unrealistic for entry missions, even if one would consider a ΔV maneuver before entry takes place.

Additionally, it is not typical for re-entry vehicles to enter the atmosphere at a lower lift as this would result in a steeper dive into the atmosphere and would thus cause the vehicle to reach the denser regions of the atmosphere before sufficient energy has been shed. As a consequence, the vehicle will experience large heat load and limit itself in the achievable flight range. The obtained results to make the flight path angle more negative are thus considered to be unfeasible, as it would strongly decrease the flexibility of the vehicle.

Also, if one would divide the trajectory of the vehicle into segments to allow the optimization problem to generate a L/D profile, the L/D would have to be limited to such low values for the entire trajectory, as otherwise, the analytical method would simply jump to high velocities for the given altitude at the start of a segment, thus negating the effect of the initially decreased flight-path angle.

8.1.6. Error Investigation

The most obvious errors in the method are observed when applying the HORUS-2B vehicle data to the analytical simulation. For these values, the initial velocity that is required for an equilibrium glide is very close to the circular velocity at the given altitude. This leads to two significant issues with the analytical method:

1. The initial step in the range angle becomes unacceptably large.

2. The analytical method continues to decrease in altitude, whereas the numerical simulation finds a lofting or even skipping trajectory.

The first error leads to unrealistic trajectories, as this first step regularly takes values of more than 180 degrees. This problem is dependent on the mathematical definition of the range angle that is used for the gliding flight. The terms in this relation use the term $\frac{1-u}{1-u_E}$. It is clear that for very high values of u_E this term becomes very large as u decreases. For HORUS-2B this occurs for most initial altitudes. A smaller vehicle, such as the SRM capsule requires a lower value of u_E to satisfy the equilibrium glide assumption that the method employs and does thus provide useful results. It can thus be concluded that the second order method for gliding flight can only be employed for smaller and more importantly lighter vehicles as a varying value of L/D is required for robust optimization.

The second method also presents an inherent fallacy of the second order method, which is not capable of modeling upward behavior of a vehicle for a decreasing value of u. Again, this could be countered by selecting smaller values of L/D, but this would severely limit the design space of the optimization problem for HORUS. Again, the SRM capsule does not suffer these problems and can be robustly used for the optimization problem.

8.2. Skipping Flight

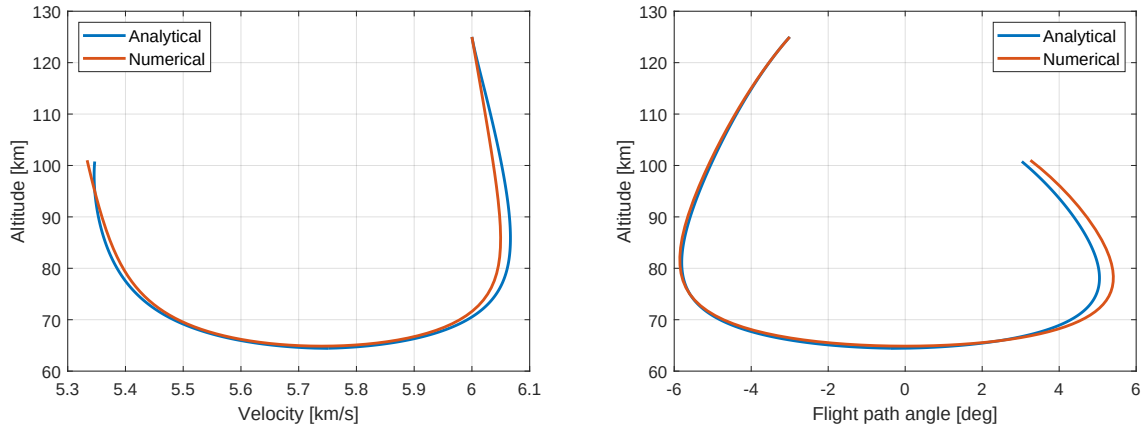
Next, the methods for a skipping trajectory will be tested. First, the trajectories are compared to numerical results, serving both to test its accuracy and as verification. This process is repeated for the ballistic phase, after which tests are performed for a combined trajectory with both skipping and ballistic phases.

8.2.1. Atmospheric Skip Phase

The analytical second-order method has been compared to the same mission in the numerical simulator. The application for the HORUS vehicle proved that again, the high mass of the vehicle caused difficulties for the analytical method. However, for the SRM vehicle results were obtained more consistently and also showed results that at least somewhat matched numerical simulations. A sample missions is presented in Figure 8.7. The figure shows the altitude-velocity profile of the two simulations. The two methods show minimal deviations for most of the trajectory. Deviations start to occur near the atmospheric exit point. This can be accredited to the fact that the analytical method does not accurately describe the motion outside of the

atmosphere, which ends at 100 km. To prevent such a discrepancy in the full simulation, it should be noted that above 100 km, the second order method for skipping flight will be stopped and stitched to a method describing ballistic flight.

Figure 8.7(b) shows the relation between the altitude and flight path angle for both simulations. Again, a close match is observed for a large part of the trajectory. In the upward leg, a deviation is observed. This deviation remains small through the rest of the trajectory and within limits for an approximate method.



(a) Altitude-Velocity Profile

(b) Altitude-Flight Path Angle profile

Figure 8.7: Results of the second order analytical simulation compared to the numerical simulator.

Limitations of the Method

During the implementation and testing of the second-order method for skipping flight, a number of limitations was identified. The most significant limitation was the occurrence of complex numbers in the results, which happens in two main cases:

- Values of x below the estimated deepest point in the trajectory.
- A negative value for C_{D_0}.
- Initial velocities above 11000 m/s.

The first case was already reported by Vinh et al. (1993) and could be solved by iterating on the estimated deepest value of the trajectory. The second case was not reported and also only occurs in rare conditions, as C_{D0} is typically positive. The third result was found consistently and also not previously listed. However, it is important to know that this phenomenon thus limits the design space of a potential optimization problem.

Although the software is capable of handling complex numbers, this does result in a loss of data when real numbers are required for the output of the state. For the second case, this also results in an output of NaN for values in the deepest point in the trajectory. As a consequence, no data is available for these cases for the state, the aerodynamic forces and the aerodynamic heating. This poses a problem for the problem, as at the lowest point of the trajectory, the forces and heating are typically highest and thus important data for the constraint handling is lost.

Although a case could be made to reject the second order method due to this inconsistency, the number of trajectories where the complex numbers occur are small and typically are found closer to the edges of the capabilities of the method. Due to the large improvement in the computation time resulting from the second order method, it is still worthwhile to study it further.

8.2.2. Ballistic Phase

The ballistic phase of the skipping flight was first separately verified. The analytical methods were compared against the numerical simulations for a flight that is purely ballistic with an initial flight path angle of +90°. As can be observed in Figure 8.8 the match between both methods is excellent. Given that the match between

analytical and numerical methods is so close, the implementation of the analytical methods can be assumed to function correctly.

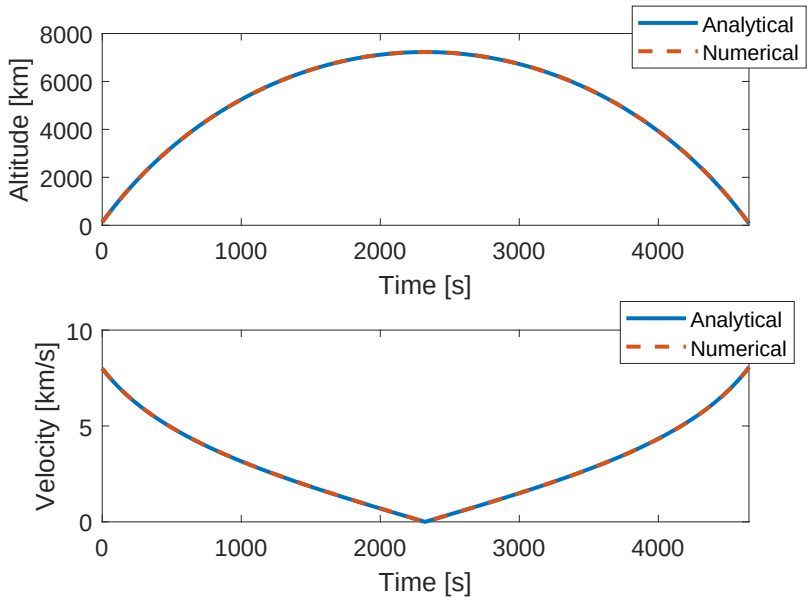


Figure 8.8: Verification results of the ballistic phase of skipping flight.

8.3. Selection for the Optimization Problem

The methods for gliding and skipping flight have been analyzed in the past sections. A trade-off now has to be made for the method that will be applied in the optimization problem.

First, the methods for gliding flight have shown to be inflexible, as the initial state of the trajectory is prescribed by the method. The accuracy of the method for the obtained trajectories is high compared to especially the simplified numerical results for the SRM mission. Although the method was incapable of providing accurate trajectories for HORUS-2b, it can still be applied to an optimization problem of the SRM mission. It should however be noted that the oscillations in the trajectory cannot be modeled by the analytical second-order method, which would still cause slight inaccuracies in the constraints.

The results for skipping flight where also showing good accuracy compared to numerical results. The main drawback of this method was that some values turned to complex numbers for a region of the solution space, causing the loss of valuable information. Especially since these regions mainly occur in the lower part of the trajectory, information on the peak values of the constraints is lost in this way, which means that for optimization purposes, the peak vales will always be lower than their actual values. Thus, trajectories would be accepted even if they do not meet the constraints.

Both methods thus show drawbacks that would make them less suitable for the optimization problem. However, given that gliding flight is accurate and does provide values for all variables over the entire trajectory makes it a strong case for optimization. Additionally, most current entry missions still make use of a gliding entry mechanic instead of skipping and thus analyzing the gliding flight provides results that are more applicable to the applied re-entry guidance.

9

Optimization Results - Gliding Flight

The following chapter presents the results that were obtained in the optimization of the previously presented numerical and analytical problems. First the results of the of the optimization of the SRM capsule trajectory are discussed. Then, the results of the HORUS-2b trajectory are presented. The optimization problems for both are based on the results that were obtained in the previous chapter. Were applicable, the optimization results are performed using the different algorithms that were presented in Chapter 4 and studied for their performance.

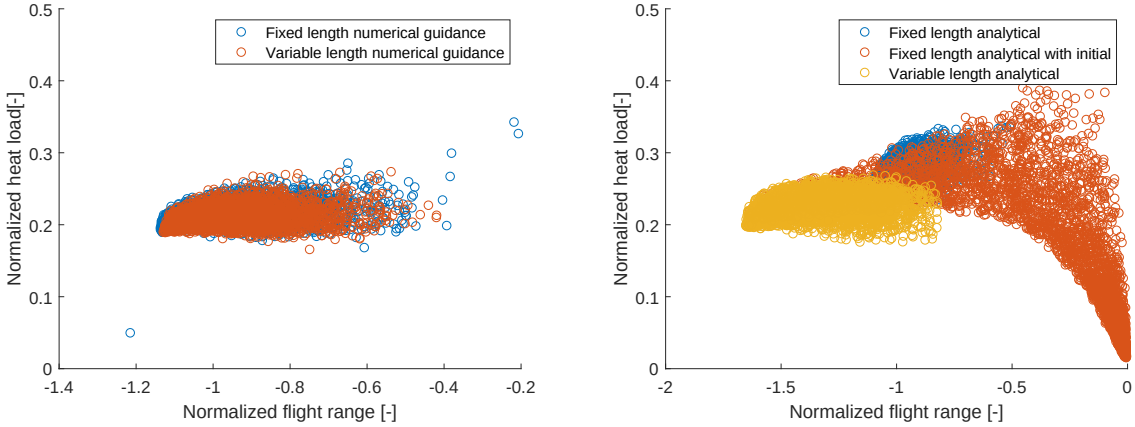
9.1. Optimization Settings

To ensure both the repeatability and the good performance of the optimization algorithms, several settings have to be determined. First of all the random seed ensures that the results for different runs are the same when repeated. For all simulations in the following sections a random seed of 123 has been used, unless stated otherwise. The MOEA/D and SCEA algorithms take the same input parameters. These are:

- **weight_generation**: The method that is selected to determine the weights of the fitness elements. Set to its standard 'grid' method.
- **decomposition**: The method by which the fitness elements are decomposed. Set to its standard method, 'tchebycheff'.
- **neighbours**: The number of neighbours that are taken into account when determining the weight of a fitness element. Set to 20.
- **CR**: The crossover parameter, which determines the likelihood that crossover occurs when generating a new individual. Set to 1.0, and thus 100% of individuals will be generated by crossover.
- **F**: The parameter for the differential evoluation operator. Set to its standard value of 0.5.
- **eta_m**: the distribution index used for the polynomial mutation. Set to the standard value of 20.0.
- **realb**: The chance that the neighborhood is considered at each generation, instead of the whole population. Set to 0.9.
- **limit**: The maximum allowable number of copies that can be introduced into the population after evolution set to 1 to preserve the diversity.
- **preserve_diversity**: A boolean variable that determines whether the **realb** and **limit** variables are considered, set to true.

A final variable remains, which determines the mutation chance. This value is set to $\frac{1.0}{\text{pop\_size}}$. This ensures that statistically a single individual will mutate for each generation.

MHACO uses a different set of input parameters. Their settings are discussed later in this chapter.



(a) Numerical Monte Carlo results

(b) Analytical Monte Carlo results

Figure 9.1: Results of the Monte Carlo simulations of the numerical and analytical problems.

9.2. SRM

The SRM vehicle was previously found to be useful for the analytical trajectory generation and will thus be further considered here for both numerical and analytical optimization. A Monte Carlo analysis of the decision vector, and thus the guidance nodes, is first performed on the different optimization problems that are available. These problems are the numerical problems with both a fixed and variable length guidance vector, the fixed-length analytical problem, the fixed length analytical problem with inclusion of the initial altitude and a variable length analytical problem, again with inclusion of the initial altitude. All Monte Carlo sets were generated using 4000 individuals.

For the three problems that take a fixed initial altitude, it is set to 100.0 km. This leads to an initial velocity of 7658.34 km/s and a flight path angle of -0.056° when applying the relations given by Vinh et al. (1993).

The results for the numerical problems are shown in Figure 9.1(a) and for the analytical problems in Figure 9.1(b). The objective space of both the numerical and analytical results show similar ranges. It should be noted that the inclusion of the initial state for the analytical problems causes the larger flight range values. An important note can be made about the solution space for both the numerical and analytical problems. Instead of showing a front of optimal solutions, the solution space shows a clear optimal point around the range angle objective value of -1.15 for the fixed entry altitude and -1.6 for the optimizations including the entry state. For the optimization, this causes issues as no Pareto front will be formed, but rather the solutions will converge towards this single point as it will dominate all others in terms of optimality.

The cause of this result is the low flight-path angle that is inherent to the analytical method. For fair comparison, the numerical trajectory has also been set to use this flight path angle and thus finds similar trajectories. An attempt was made to change the flight-path angle that is given in the method, but was not found.

In addition to the fixed entry state trajectory, the analytical problem was also tested for a variable initial state, which was included in the decision vector. The bounds of the altitude in the decision vector were set for $[50.0, 110.0]$ km. For this problem type, the solutions are found in a much wider range, starting at a normalized range angle objective value of almost 0.0. Unfortunately, the solution space still finds a single optimal point for the trajectory, thus again confirming the difficulties that arise from the low flight path angle, which is a fundamental problem with the selected analytical second-order methods.

9.2.1. Optimization Results

The results of the optimization problem for the numerical and analytical problem with a fixed entry state are presented in Figure 9.2. Both results were obtained using the MOEA/D algorithm after 12 generations where convergence to the optimal point is already achieved, thus confirming the drawbacks of the method. As expected, the results converge to a point, instead of a wider front. This leads to the conclusion that the solutions that are obtained for the analytical problem from a typical entry altitude are not useful for use in

re-entry trajectory optimization.

Numerical Flight from Analytical Guidance

Although the optimized analytical guidance was not directly applicable for the generation of a flight path, the obtained guidance nodes were applied to the numerical simulator to find how these solutions relate to the optimal solutions that were found directly from the numerical optimization.

The first results immediately showed that simulations using the simulator with an extended environment generate trajectories where atmospheric capture does not occur. Again, these issues are mainly caused by the added rotation of the Earth and its atmosphere, which effectively increases the airspeed of the vehicle. In combination with the low flight path angle that is inherent to the second order method, no solution was found for this issue.

The optimization was also performed for the numerical model with a simplified environment, which actually provided feasible trajectories.

9.3. HORUS-2B

Next, the optimization of the HORUS trajectory was performed. As mentioned in Chapter 8, analytical trajectories cannot be optimized for HORUS as the analytical model is not capable of representing these. Still, the performance of the SCEA and MACO trajectories compared to the MOEA/D algorithm can provide valuable insights when applied to the Horus mission.

The design space of the HORUS re-entry problem for both the fixed length guidance and variable length guidance is presented in Figure 9.3. The selected method was a Monte Carlo analysis of the problem using 4,000 individuals. As can be expected, the obtained solutions strongly overlap, although the results for the variable node guidance are found in a smaller range compared to the fixed 8-node guidance.

The explanation for this phenomenon becomes clear when considering that the variable length problem takes another decision vector variable in the form of the number of guidance nodes. A larger number of guidance nodes typically leads to more optimal solutions. Given that the variable length problem has to divide its solutions over a guidance vector of length 4 to 8 instead of just 8, it is thus to be expected that for a Monte Carlo analysis of the same size the variable problem will have more trouble finding the excesses of the search space.

Still, the variable length problem shows a similar shape to the fixed length problem for both the objectives as seen in Figure 9.3(a) and the constraints as seen in Figure 9.3(b). Note that the g-load rate constraint takes on large values compared to the heat flux constraint and g-load constraint. These values can be brought back to the zero-order hold that was applied, which causes the angle of attack and bank angle profiles to become discrete instead of continuous. Consequently, the g-load rate becomes large when the discrete change occurs after the next sampling time is reached in the trajectory.

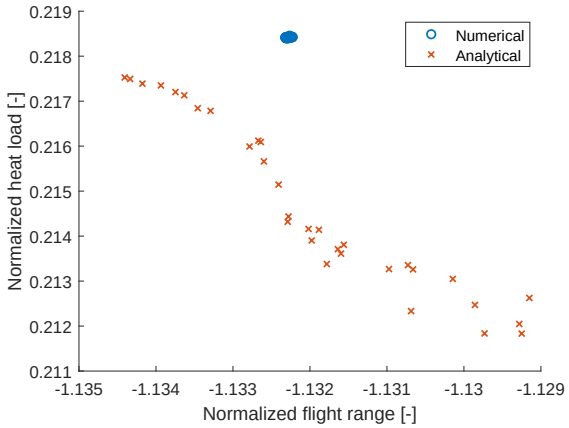
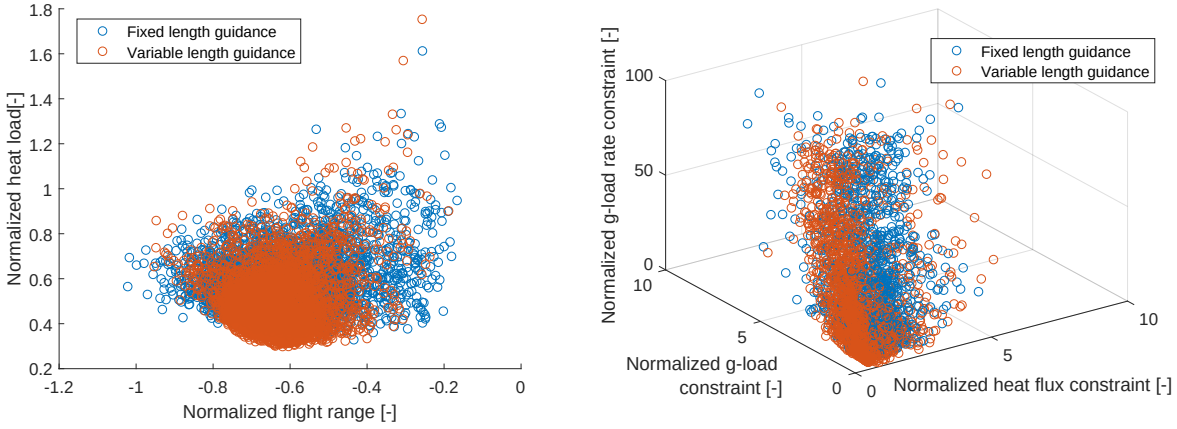


Figure 9.2: Final generations of the numerical and fixed entry state problems from the MOEA/D algorithm.



(a) Objectives (b) Constraints

Figure 9.3: Results of a Monte Carlo analysis with a size of 4,000.

9.3.1. SCEA Results

First, the results of the MOEA/D and SCEA algorithms were generated, due to their high comparability. They are analyzed for their Pareto front and convergence.

Pareto Fronts

The results for four numbers of generations are presented in Figure 9.4. Note that the lines present the space in between the non-dominated individuals in that specific generation.

The results show that the SCEA algorithm is capable of finding similar Pareto fronts compared to MOEA/D. Figure 9.4(a) shows that after 12 generations, the Pareto optimal fronts that are found by SCEA outperform the fronts found from MOEA/D. Later generations show that MOEA/D starts finding better fronts compared to SCEA as the number of generations increases, although the region of the fronts for low heat load trajectories are still showing better results from the SCEA algorithm.

Both population sizes for MOEA/D and the SCEA problem with a population size of 64 are found to obtain values in a similar range of flight range and heat load values. When applying 32 individuals and using the SCEA algorithm, a less diverse Pareto front is obtained, which is maintained throughout the generations. To further investigate the cause of this phenomenon, the convergence rate of the algorithms is analyzed next.

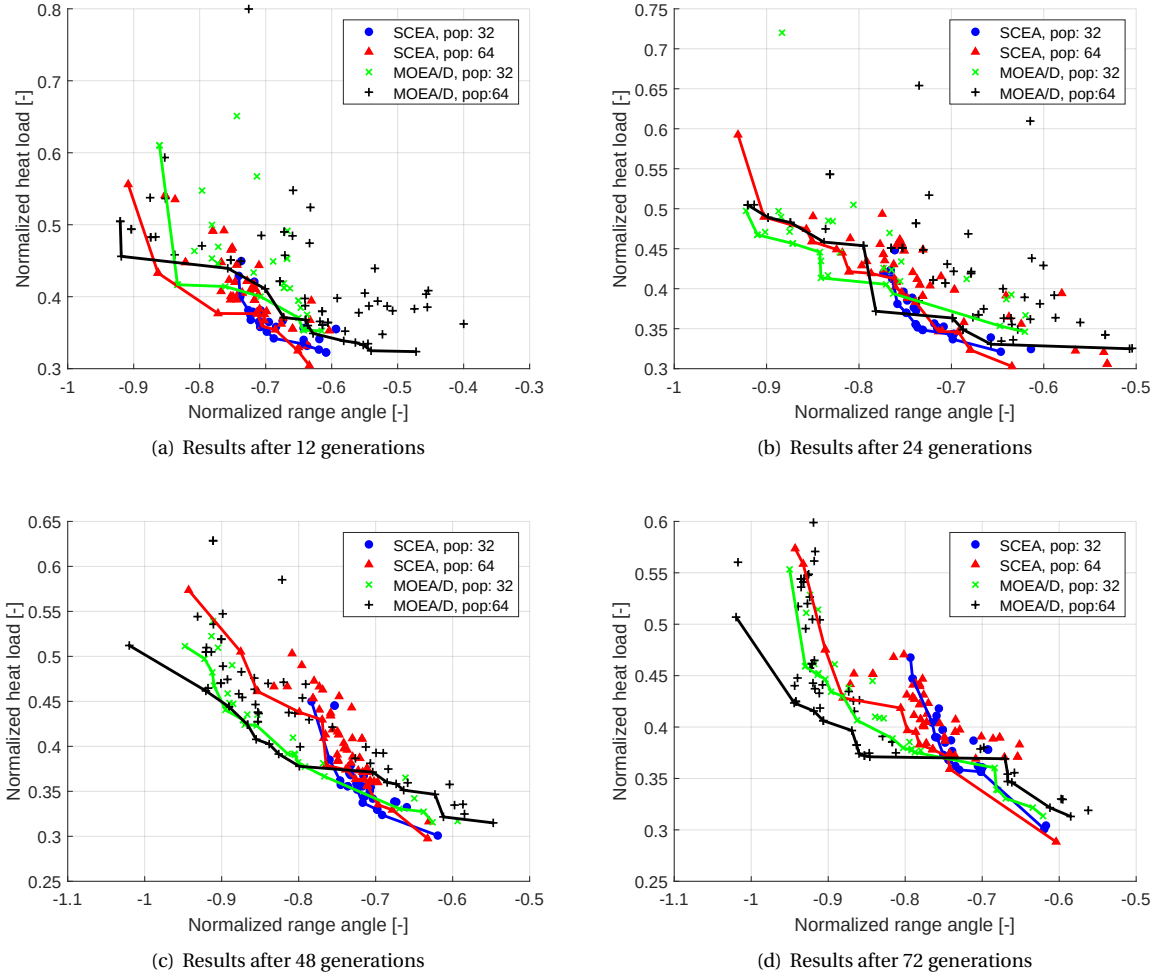
Convergence Rate

Although the optimal values and shape of the Pareto front provide important information about the performance of the algorithms, the convergence rate cannot be overlooked. For the convergence of the algorithms, two elements are important. Firstly, the convergence towards a feasible solution space and secondly the convergence towards the Pareto optimum.

The convergence rate towards a feasible solution space for the algorithms is presented in Figure 9.5. Two main conclusions can be drawn from the figures. First of all, the population size does not significantly influence the convergence rate of the algorithms as only a small difference in the number of generations before as feasible solution space is reached can be observed.

Secondly, the SCEA algorithm shows a much faster convergence towards a feasible solution space. Within five generations, the SCEA algorithm has cleared all infeasible candidates from its population, while the MOEA/D algorithm requires 19 generations in the best scenario. It now also becomes clear why the SCEA algorithm performs well in terms of Pareto front in early generations, as solutions are no longer compared for their constraint violation when both the parent and child are feasible. The SCEA algorithm thus starts comparing based on the decomposition value for all individuals after five generations, compared to after 19 generations for MOEA/D.

A note should be made about the g-load rate constraint shown in Figure 9.5(d). For the SCEA evolution, the values of the constraint start at zero as no individuals in the population violate the constraint. At first, this



(a) Results after 12 generations

(b) Results after 24 generations

(c) Results after 48 generations

(d) Results after 72 generations

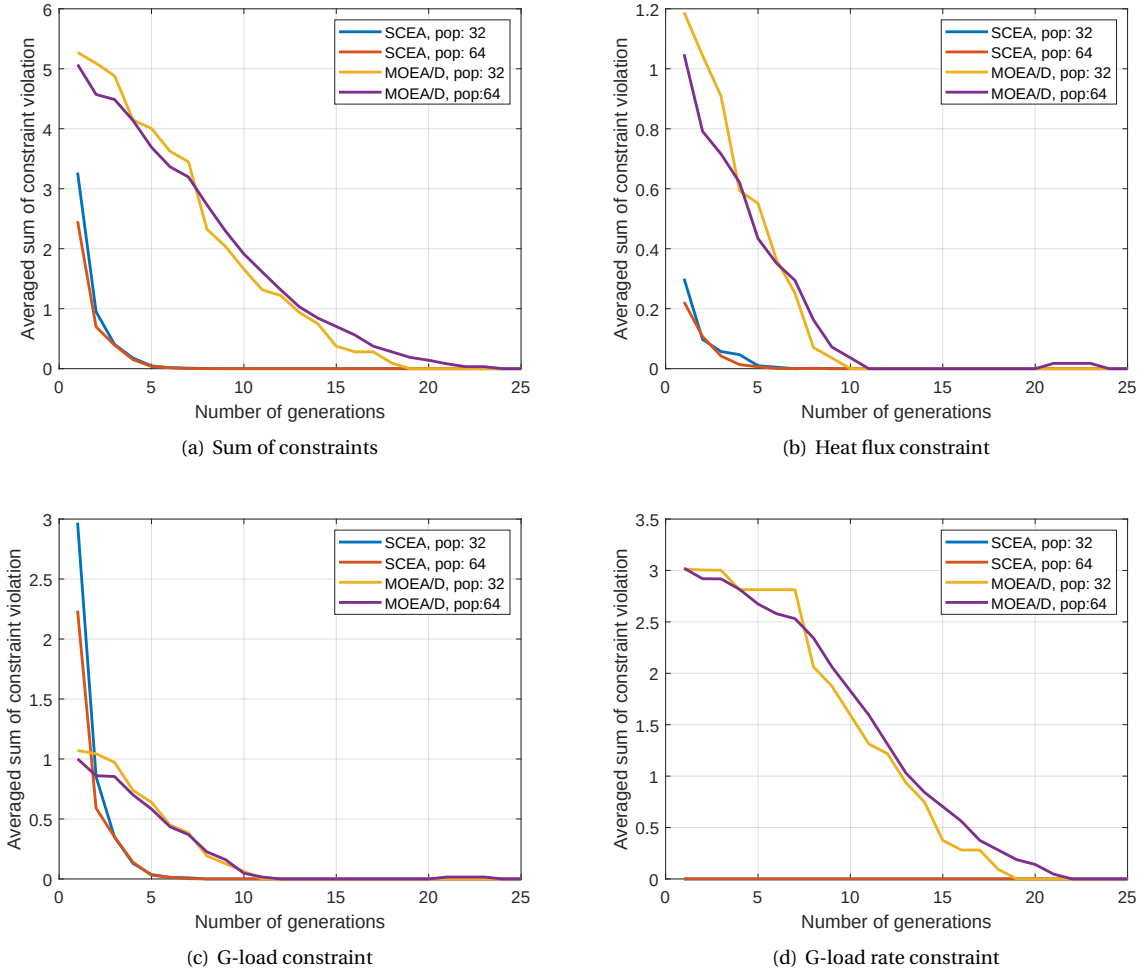
Figure 9.4: Results of the SCEA algorithm compared to MOEA/D after selected numbers of generations.

seems illogical, although the simulator that was used for both the fixed-length and variable length problem was equal, and thus this phenomenon will have to be accredited to the influence of the current random seed.

Secondly, the convergence rate towards the optimal solutions is investigated. An overview of the average and best values of the objectives is given in Figure 9.6. As was already observed from the obtained Pareto fronts, the SCEA algorithm also shows in the convergence that it prefers solution with lower heat load and lower flight range. Especially the best found solutions presented in Figure 9.6(b) and Figure 9.6(d) for the flight range and heat load, respectively, indicate this clearly as the SCEA algorithm finds better heat load trajectories and the MOEA/D algorithm finds the better flight range trajectories.

Also the course of the convergence should be discussed here. For MOEA/D a relatively smooth improvement for the trajectories can be found, while SCEA has a strongly fluctuating development over the generations. A relation between this fluctuation and the average number of guidance nodes per generation as seen in Figure 9.6(e) can be observed. As the SCEA algorithm also takes the number of guidance nodes as an optimization objective, the algorithm will trade solutions not only based on the flight range and heat load, but also the number of nodes and thus accept worse values in the flight range/heat load Pareto front if they show an improvement in the number of guidance nodes that is required to obtain it.

Now a cause for the distribution of the Pareto front of the SCEA algorithm can also be found. Figure 9.6(e) shows that the algorithm tends to maintain an average number of nodes between 4.5 and 5.5 over its population. If the improvements for the flight range are thus only found when a larger number of nodes is used, it will still not be accepted by the SCEA algorithm.



(a) Sum of constraints

(b) Heat flux constraint

(c) G-load constraint

(d) G-load rate constraint

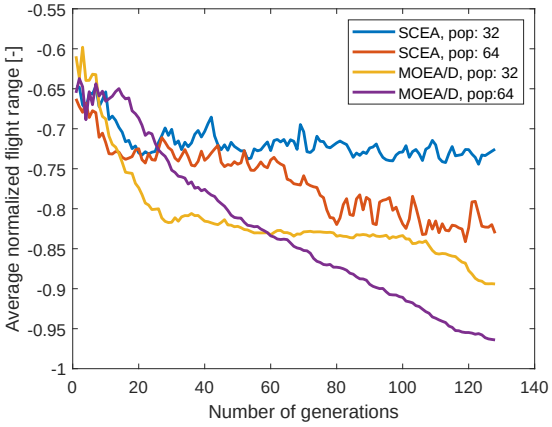
Figure 9.5: Convergence rate towards a feasible solution space.

Function Evaluations

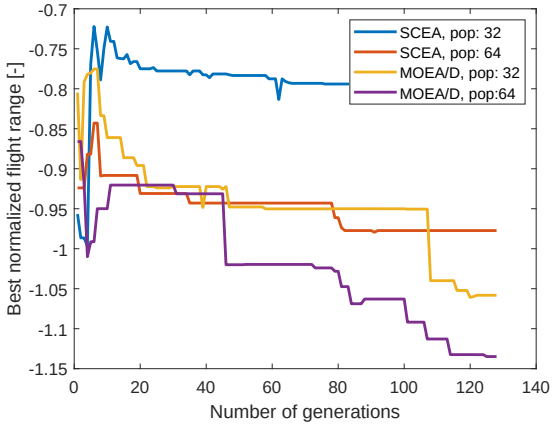
To conclude the study on the convergence rate of the SCEA algorithm, the number of function evaluations is shown in Figure 9.6(f). First, note that the large difference between the problems with a population size of 32 and the population size of 64 is caused by the fact that less individuals are tested for each generation. Still, a clear difference can be observed between the SCEA and MOEA/D problems. For both population sizes, the SCEA algorithm requires a larger number of function evaluations per generation. For the population size of 32, this is maintained over all generations and a slight increase is present. For the population of 64 individuals, a different profile is found, where initially a larger number of function evaluations is required, However, the algorithm becomes more efficient over the generations and at 128 generations has performed slightly less function evaluations than MOEA/D.

A cause for the difference between the two algorithms can be found in the implementation of SCEA. As SCEA changes the number of nodes for an individual, it now randomly generates the added nodes. These are thus unlikely to be optimal as they use no historical knowledge of the population. MOEA/D on the other hand uses fixed nodes and can always use information from the parents when generating offspring, thus maintaining a higher chance of maintaining near optimal nodes in the offspring.

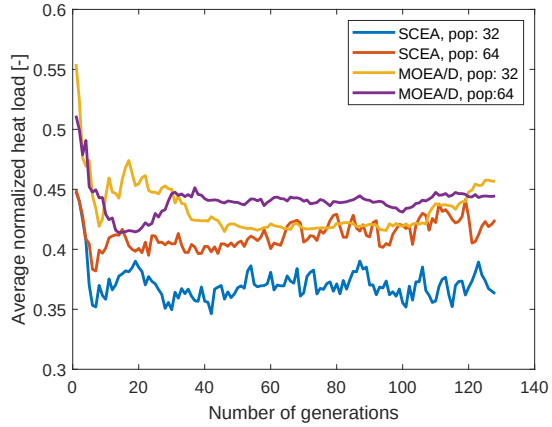
Although being more likely to find optimal offspring, this does not directly affect the number of function evaluations. However, when considering that the offspring in MOEA/D is also more likely to meet the constraints, which are largely based on rates of variables, the explanation becomes clearer. For high rates of change of a variable, a variable step size integrator is likely to decrease the step size as larger differences between the intermediate integration steps are obtained.



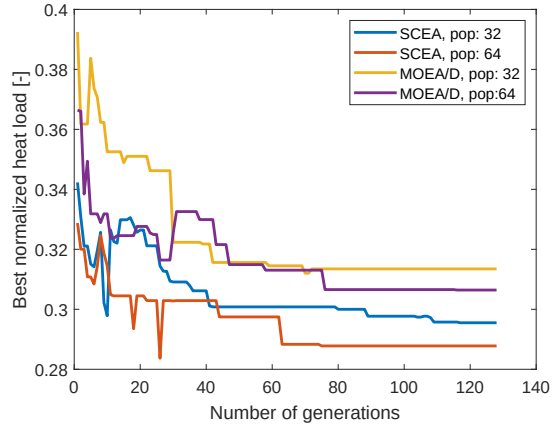
(a) Average normalized flight range



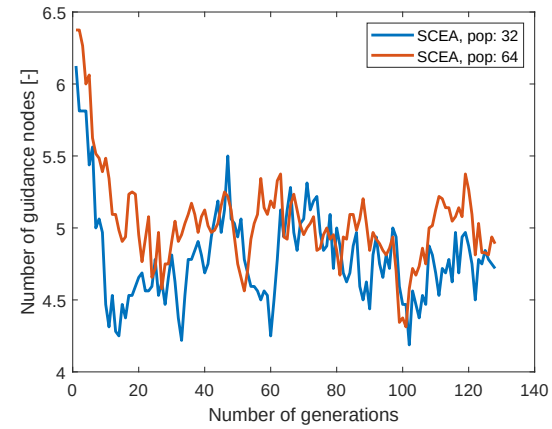
(b) Best normalized flight range



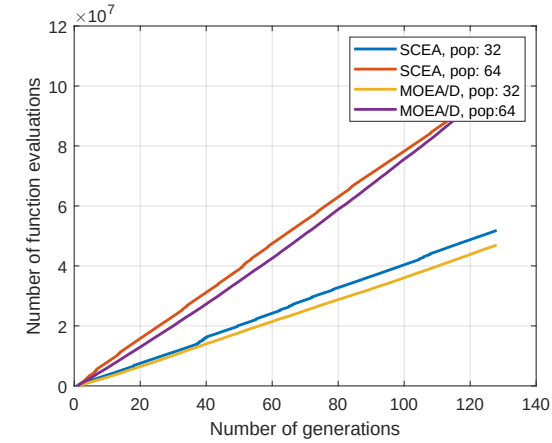
(c) Average normalized heat load



(d) Best normalized heat load



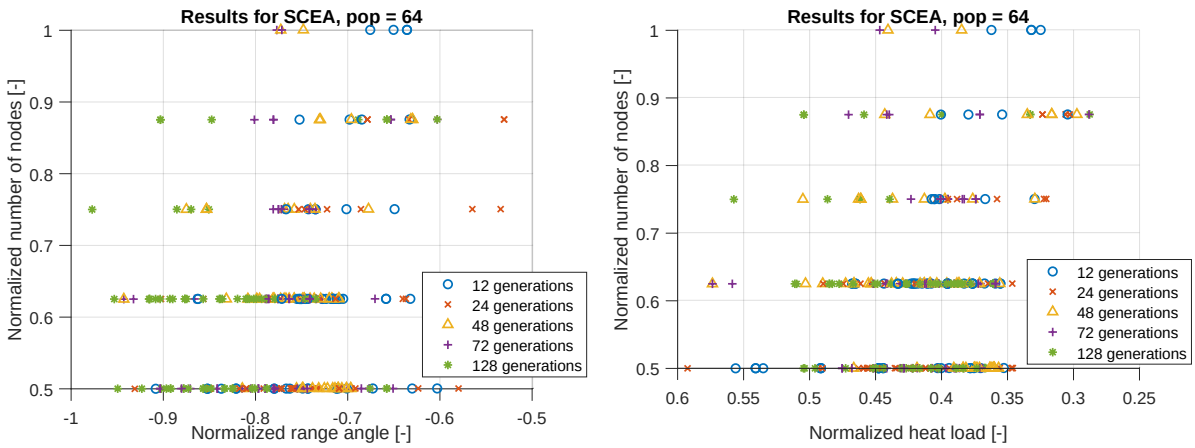
(e) Average number of guidance nodes



(f) Function evaluations per generation

Figure 9.6: Convergence rate towards the Pareto optimal solutions.

This reasoning poses a potential improvement in the SCEA algorithm, where a method should be implemented that uses historical data of the past generations to generate new offspring in the case of an increasing number of nodes.



(a) Number of nodes compared against flight range

(b) Number of nodes compared against heat load

Figure 9.7: Convergence rate of the SCEA algorithm for two sets of nodes.

Relation between Nodes and Objectives

An overview of the nodes that were obtained compared against the flight range and heat load values is shown in Figure 9.7. The results show that the algorithm has a preference for the lower numbers of nodes, as can be expected due to their implementation as an objective. Over the generations, the results are shown to slightly converge towards the lower node values.

Interestingly, the results do not show that the extremities of the Pareto front can only be obtained by the guidance vector with a larger number of nodes. The extremities with a small heat load and small flight range are however shown to be found mostly using 6 or 7 nodes, whereas the trajectories with higher heat load and flight range are found for 4 nodes in almost all cases. The results also show that diversity of the number of nodes is preserved throughout the generations.

Effect of the Maximum Guidance Nodes

The maximum number of guidance nodes was expected to influence the optimization results for two reasons. Firstly, because the original SCEA settings defined the maximum number of nodes as 8, which was the same number as used for the fixed length guidance. Thus, the SCEA algorithm would see the solutions that used the same number of nodes as the fixed length problem as the worst possible outcome in terms of nodes. Secondly, due to the normalization that was used against the maximum allowable guidance nodes, only allowing a difference of four would result in very large improvements in the objective for the nodes when the number of nodes was even decreased by one.

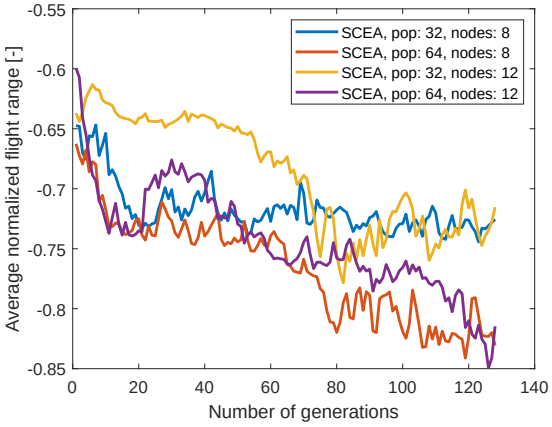
Thus, the effect of the maximum number of guidance nodes was studied by performing the optimization problem with a maximum number of guidance nodes that was set to 12. The results are presented in Figure 9.8 for both the optimization with a maximum of 8 nodes and 12 nodes. Again, the population size is shown to be the main contributor to the convergence of the solutions.

The average values of the objectives show similar results, regardless of the number of nodes. The exception here is the average heat load, which attains significantly higher values for the 12 node problem compared to the 8-node problem.

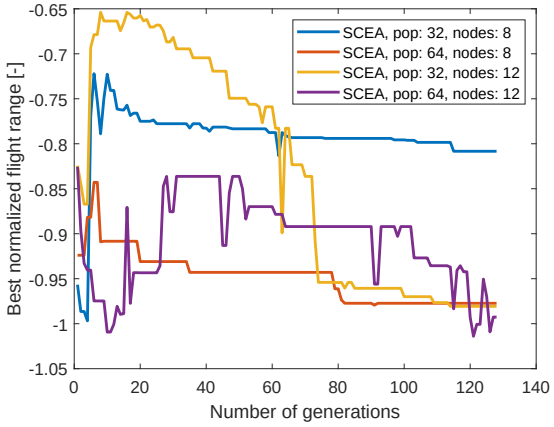
The best values for both objectives show a different profile. It is shown that the best obtained values for both objectives are lower in the case of 12 nodes. For the 8 node problem with a population of 64, the results reach a similar optimal value, but for a population of 32, the performance of 8 nodes is significantly worse, almost 10% for the optimal heat load and approximately 20% for the flight range.

The average number of nodes in the population is shown in Figure 9.8(e). Clearly, the number of nodes is higher when the total number of nodes that is allowed increases to 12. A larger discrepancy between the population sizes is also observed, where the average number of nodes becomes 7 for a population size of 32 and slightly below 6 for a population size of 64.

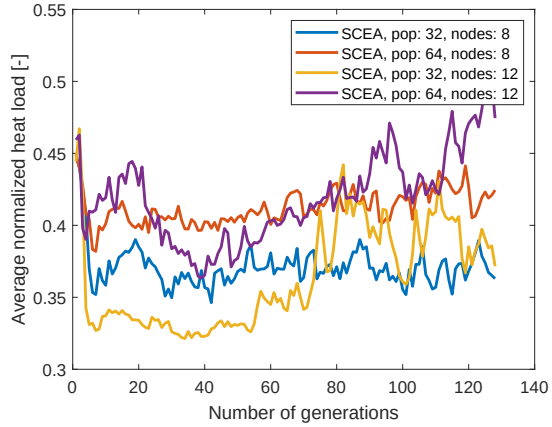
Lastly, a note can be made about the convergence towards a feasible solution space. Regardless of the maximum number of nodes, SCEA still manages to converge to a feasible solution space within 10 genera-



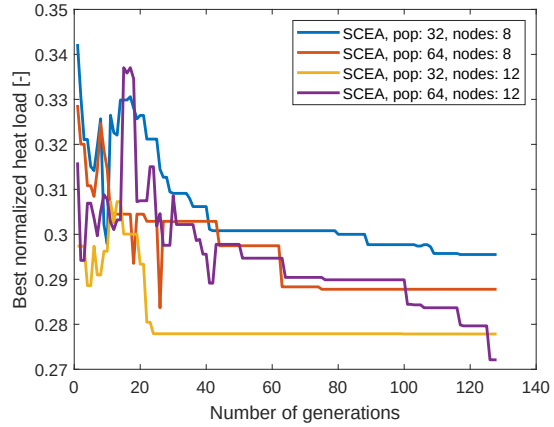
(a) Average normalized flight range



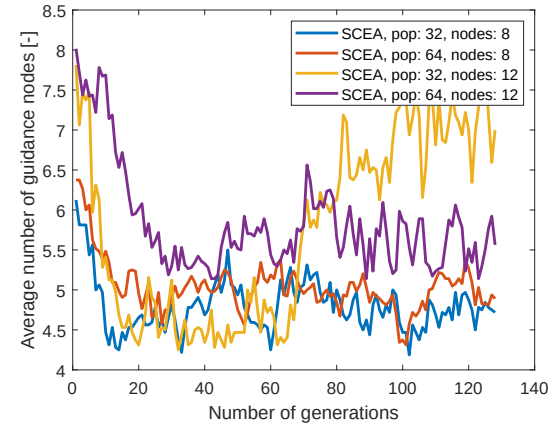
(b) Best normalized flight range



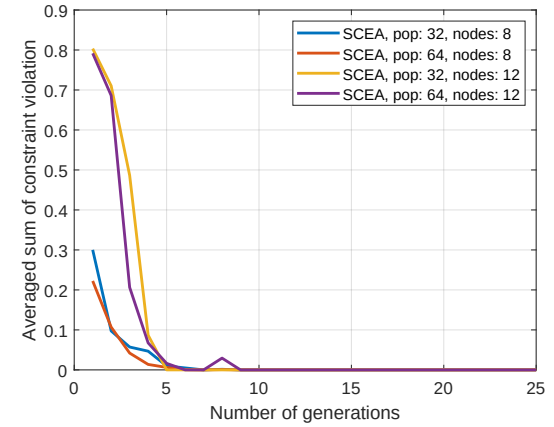
(c) Average normalized heat load



(d) Best normalized heat load



(e) Average number of guidance nodes



(f) Convergence towards feasible solution space

Figure 9.8: Convergence rate of the SCEA algorithm for two sets of nodes.

tions.

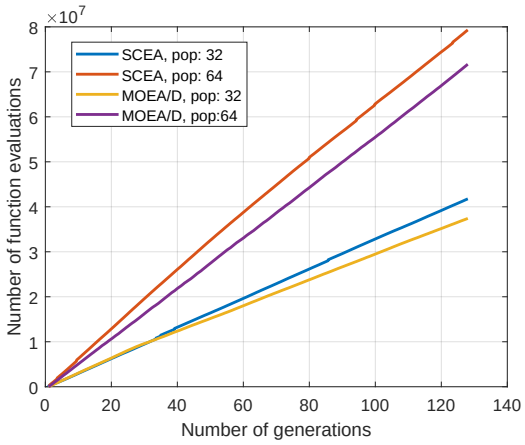


Figure 9.9: Function evaluations of the MOEA/D and SCEA algorithms for population sizes of 32 and 64 with a guidance sampling time of 4.0 seconds.

Effect of the Zero-Order Hold

Given that the SCEA algorithm finds optimal trajectories using fewer guidance nodes, the initial expectation was that the number of function evaluations that was required to obtain convergence would be lower. However, the results proved that the SCEA algorithm requires slightly more function evaluations. It was then reasoned that the guidance sampling time that the implementation of the zero-order hold could influence this phenomenon.

To verify this assumption, the guidance sampling time was increased to 4.0 seconds to ensure that the integration time step would be less dominated by the guidance sampling time and more by the dynamics of the problem. The number of function evaluations for these settings are presented in Figure 9.9. The difference between the two algorithms now becomes clearer. For both population sizes, the MOEA/D algorithm outperforms the SCEA algorithm and thus the assumption that SCEA would perform better for a larger guidance sampling time does not hold.

The explanation of the number of function evaluations thus has to be accredited to the method in which the offspring is created, which is more efficient for the MOEA/D algorithm, due to the better use of historical data and lower dependency on newly generated genes. This hypothesis was already suggested based on the fact that MOEA/D finds a better Pareto front after it has converged to a feasible solution space and is again supported by these results.

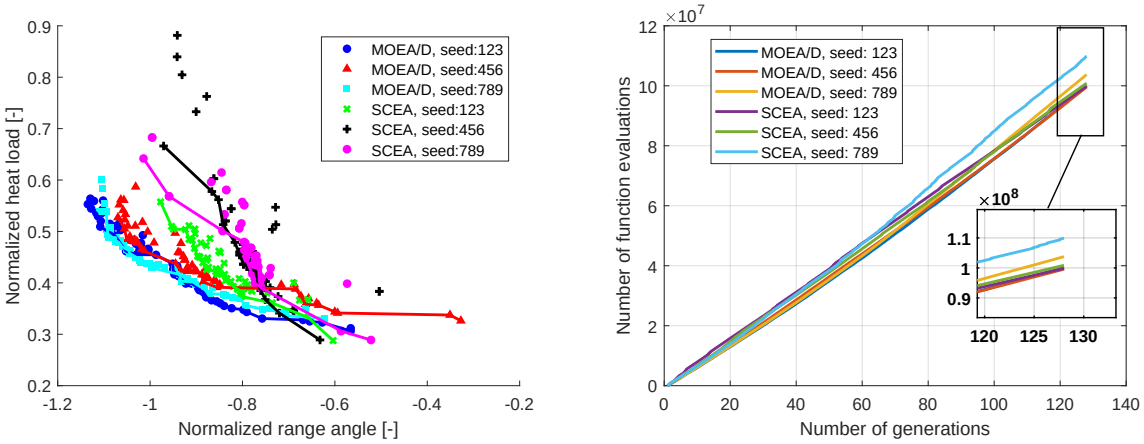
Random Seed Influence

Finally, to verify that the optimization algorithms can robustly find the optimal solutions, it is important to test them for different random seeds. The comparison was made for the algorithms after 128 generations and for a population size of 64, as these had previously shown the best results. The results of both MOEA/D and SCEA for a random seed of 123, 456 and 789, respectively, are shown in Figure 9.10(a). The results show that MOEA/D consistently finds very similar results, regardless of the random seed. The SCEA algorithm shows larger differences between the three selected random seeds. As was found before, the results are still competitive for the lowest heat load trajectories, where the algorithm finds similar results regardless of the random seed. When the heat load and flight range increase, larger discrepancies are observed.

As was suggested before, the higher dependency on random generation of the SCEA algorithm when the chromosome length changes is the likely cause of these differences.

The required number of function evaluations is presented in Figure 9.10(b). It can be observed that SCEA requires slightly more function evaluation after 128 generations for the first two random seeds and significantly more for the third option. MOEA/D also shows this increase for the random seed of 789 and the cause of the increase is thus partially explained by the random seed. As the number of variables that is generated for both algorithms is not the exact same since SCEA takes an integer value for the number of nodes as well. The larger difference is thus mainly found as the random seed has an even larger effect on the function evaluations for SCEA.

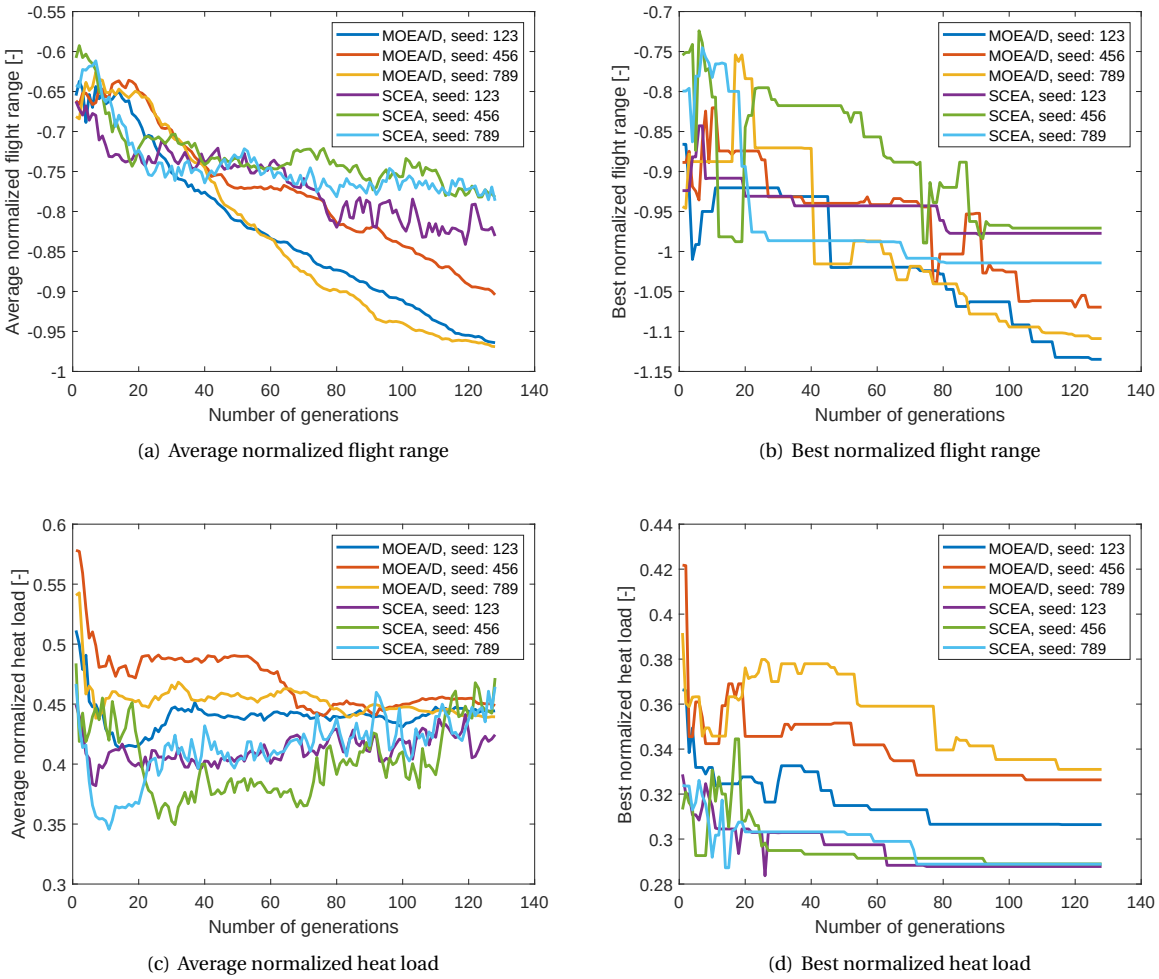
Next, the differences for the convergence of the objectives for the algorithms for the different random seeds is presented in Figure 9.11. Again, MOEA/D is shown to have a smoother change in the average ob-



(a) Pareto fronts after 128 generations

(b) Function evaluations

Figure 9.10: MOEA/D and SCEA results for different random seeds.



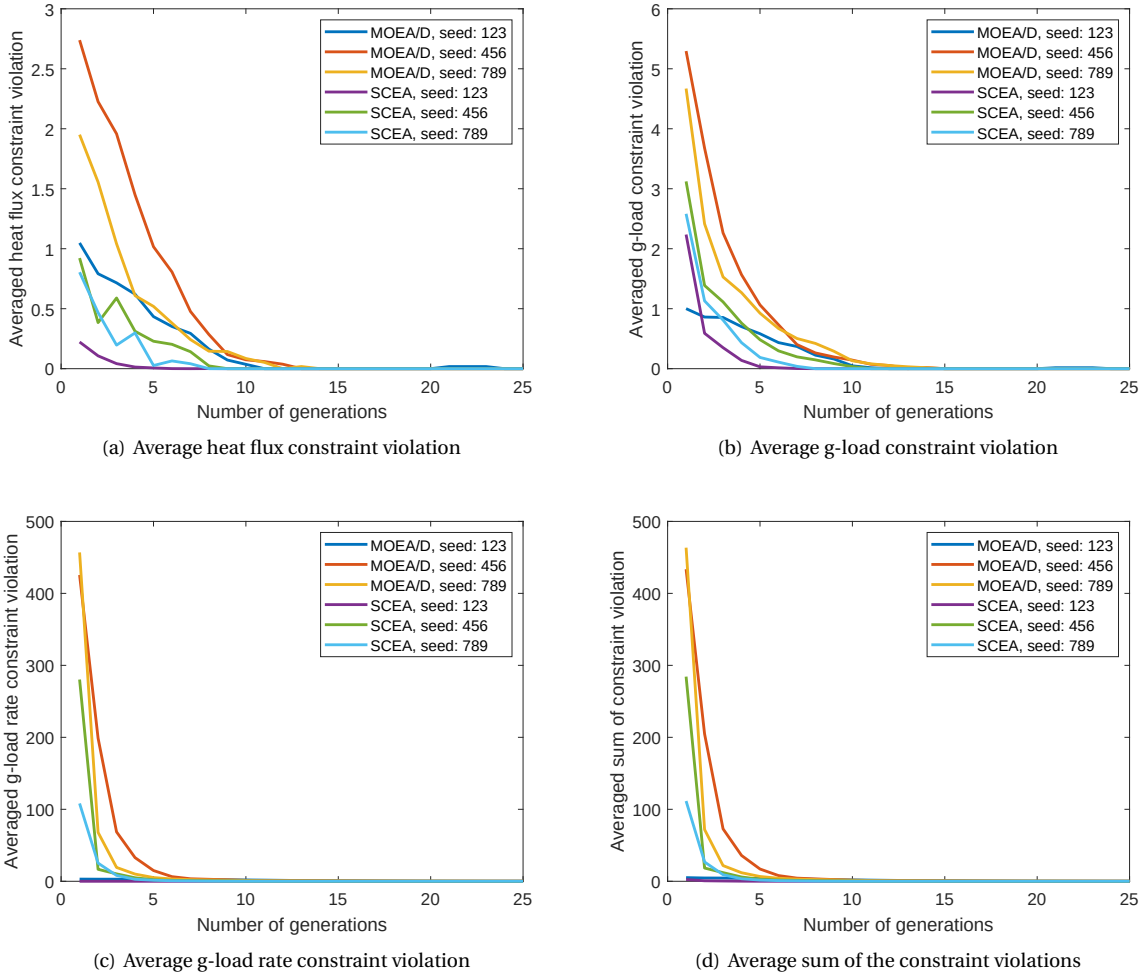
(a) Average normalized flight range

(b) Best normalized flight range

(c) Average normalized heat load

(d) Best normalized heat load

Figure 9.11: Convergence rate of the SCEA and MOEA/D algorithms for the three selected random seeds.



(a) Average heat flux constraint violation

(b) Average g-load constraint violation

(c) Average g-load rate constraint violation

(d) Average sum of the constraint violations

Figure 9.12: Convergence rate of the constraint for the SCEA and MOEA/D algorithms for the three selected random seeds.

jective values as shown in Figure 9.11(a) and Figure 9.11(c) for the flight range and heat load, respectively. SCEA still shows the rougher nature of the convergence. For the flight range, the results for a random seed of 456 and 789 show a very similar progress over the generations, while for a random seed of 123 better average flight range results are observed before 30 and after 70 generations. It is reasonable to assume that this effect depends on the random seed, as the average value of the range can improve strongly if one of the randomly generated offspring finds a strong improvement. These results also show their relation with the Pareto fronts that were obtained in Figure 9.10(a), where the fronts for a seed of 456 and 789 show a clustering around the middle of the front and are shown to perform especially worse compared to the seed of 123 in this region.

The best obtained values for the flight range and heat load are shown in Figure 9.11(b) and Figure 9.11(d), respectively. The MOEA/D results show that for all seeds it finds better range angle results, although the SCEA algorithm consistently finds better results for the heat load. Importantly, the best obtained heat load is consistent for the three runs for SCEA. Comparison with Figure 9.10(a) shows that these do not correspond to the same range angle. This inconsistency mainly occurs at the extremes of the trajectory, where a more dominating solution will only be found if an improvement is found for the range angle.

The effect of the random seed on the constraint violation is presented in Figure 9.12. All three constraints shows that SCEA outperforms MOEA/D in its convergence to the feasible solution space. However, it is also shown that for the random seed of 456, SCEA converges to feasible solutions much slower and more in line with the MOEA/D results. Still, it does not perform worse and it is thus safe to say that the convergence of SCEA towards feasible solutions is better than the convergence of MOEA/D.

9.3.2. MHACO Results

The MHACO algorithm has previously been proven to provide good results for multi-objective optimization problems. For this thesis, it was tested for the problem of re-entry mission design. The results are found below.

Problem Adaptation

When the first tests were run for the current re-entry problem, it was shown that the problem settings caused the MHACO algorithm to find only a very limited number of feasible solutions, if any at all. The cause was identified to lie in the values of the constraints, which typically took on values that are much higher than the objectives and thus resulting in an unfair comparison in the hypervolume that was used to determine the Pareto front.

The best results were obtained when ensuring that the solution space for both the objectives and constraints was similar. For this reason, the flight range normalization was altered. Originally, this was defined as:

$$f_r = -\frac{\theta}{\pi} \tag{9.1}$$

Thus normalizing it against half the circumference of the Earth and setting it to a negative values to ensure its maximization in PaGMO. The new flight range objective is set to:

$$f_{r,MHACO} = 2.0 - \frac{\theta}{\pi} \tag{9.2}$$

thus ensuring that the objective value is between 1 and 2 for almost all cases, with the exception of the small number of solutions that find a flight range over half the circumference of the Earth.

The heat load objective was also adapted to find a more suitable range of solution. The new definition of the heat load is found as:

$$f_{Q,MHACO} = 1.0 + \frac{\theta}{\pi} \tag{9.3}$$

Again, this ensure that the theoretical optimum of the heat load objective is found at a value of 1.0. By applying this modifications to the objectives, the range is almost equal to those that are found for the constraints, which are adapted as follows.

As MHACO treats the constraints as objectives, it was no longer valid to treat these as zero for feasible solutions. Thus, also the feasible results were normalized against the constraint value. However, as large constraint violations negatively impacted the convergence, all normalized violations over 2.0 were set equal to 2.0. This does significantly limit the knowledge of the constraints when optimizing the problem. However, the information about the constraints for slight violations is still available and it would thus still be possible to identify solutions that are very close to the feasible solution space and might require further analysis to find if a higher fidelity simulator provides more accurate results.

Lastly, it was found that without further alteration, MHACO would still accept almost only infeasible solutions, as the improvement for the objectives was significant. To circumvent this, a penalty was added to both objectives for each constraint that was violated. The value of this penalty is 0.5, thus leading to a maximum objective penalty of 1.5 when all constraints was violated. This method showed to cause the MHACO algorithm to strongly prefer feasible solutions as will be described in the rest of this section.

Algorithm Parameters

Next, the performance of the MHACO algorithm depends strongly on the parameters that are given to it. To find the proper tuning of the parameters, several cases were tested. The parameters that were tested are:

- **Number of generations**: 24, 48 and 72.
- **Solution archive**: 32 individuals, with a population size of 32 and 64.
- **threshold**: 5 generations before the final generation
- **n_gen_mark**: 3, 7 and 10.

Table 9.1: Number of feasible individuals for the selected test cases with an archive size of 32.

	gen = 24	gen = 48	gen = 72
n_gen_mark = 3	pop 32: 4 pop 64: 3	pop 32: 4 pop 64: 6	pop 32: 5 pop 64: 13
n_gen_mark = 7	pop 32: 3 pop 64: 7	pop 32: 2 pop 64: 6	pop 32: 4 pop 64: 9
n_gen_mark = 10	pop 32: 1 pop 64: 3	pop 32: 5 pop 64: 8	pop 32: 3 pop 64: 17

All other parameters were left to their standard definition in PaGMO.

First, the number of feasible individuals in the population is identified for the selected settings, as too little diversity in feasible individuals would result in insufficiently filled Pareto fronts and thus an unrealistic representation of the actual front. For the test cases that have been described before, the results are presented in Table 9.1. The clearest relation with the number of feasible individuals is found for a large number of generations. The larger population size also shows to provide more feasible solutions for all but the top left test case. Given that the archive size remains fixed at 32 for all test cases, increasing the ratio between the population size and the archive thus proves to improve the result.

Finally, the *n_gen_mark* value of 10 provides the best results for both 48 and 72 generations, while the worst performance is found for 24 generations. As most feasible solutions are only found for the larger number of generations, the value of 10 for *n_gen_mark* thus proves most useful for use in an extended simulation for MHACO and will be used for the further investigation of the MHACO optimizer.

To ensure that sufficient feasible individuals were obtained and to allow for comparison with the MOEA/D results, the maximum number of generations will be set to 128 for the optimization of the re-entry mission. To summarize, the following settings are used for the following optimization:

- **Random seed**: 123
- ***n_gen_mark***: 10
- ***q***: 1.0
- ***threshold***: 5 generations before the final generation.
- ***evalstop***: 10000
- ***focus***: 0.0

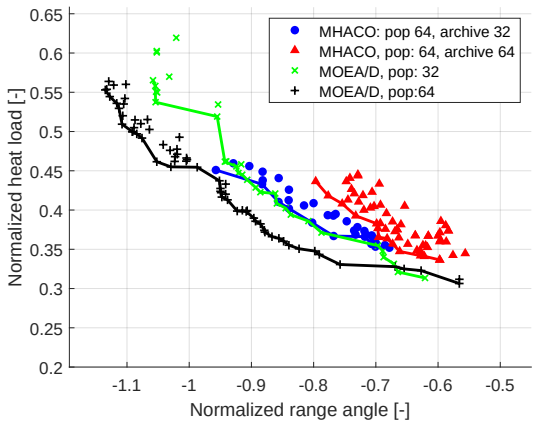


Figure 9.13: Pareto front of MHACO compared to MOEA/D after 128 generations.

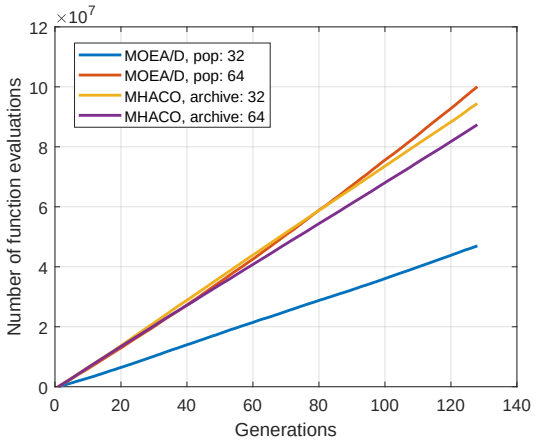


Figure 9.14: Function evaluations of MHACO compared to MOEA/D over 128 generations.

Pareto Fronts

The obtained Pareto fronts for both MHACO and MOEA/D after 128 generations is presented in Figure 9.13. An important note that has to be made about the solutions is that all obtained solutions for MHACO satisfy the constraints of the optimization problem and are thus useful for mission design.

The results show that MHACO is capable of obtaining the same fronts that were found by MOEA/D, even though no direct constraint handling method was implemented. A difference is observed for the different solution archive sizes of MHACO, for an archive of 64, the results are less optimal than for an archive size of 32. As was observed in Table 9.1, the results are better when the population size is larger than the solution archive.

Similar to what was observed for SCEA, the MHACO algorithm has trouble finding the solutions for large range angle and heat load values.

Convergence

The number of function evaluations for MHACO and MOEA/D are presented in Figure 9.14. The obtained results confirm the excellent behavior MHACO regarding the computational requirements. Initially, MHACO requires slightly more function evaluations than MOEA/D, although after 80 generations MHACO becomes more efficient. At the final generation, which is 128 for this case, the MHACO algorithm has been more efficient. For the solution archive size of 32, which provided the best Pareto results MHACO has required only 95% of the function evaluations compared to MOEA/D with a population size of 64. With a solution archive size of 64, MHACO becomes even more efficient and requires 87% of the function evaluations, although the quality of the Pareto front has become worse.

Still, MHACO did not outperform a population of 32 for the MOEA/D algorithm in its Pareto results, which still shows that MOEA/D is more efficient when trying to find optimal solutions.

Influence of the Population Size

The population and archive size of MHACO were found to influence the results as shown in Figure 9.15. The results were obtained for population sizes of 32 and 64, both with the solution archive set to either the entire population size or half the population size. A similar diversity in the Pareto front is obtained, although for the smaller population and archive sizes, more individuals are found to lie in the Pareto front itself.

The best performance is clearly shown for a population size of 64 and an archive size of 32, while the other results show very similar Pareto fronts. Given that the population size of 32 requires only half the function evaluations, these settings thus outperform the results of a population size of 64 with an equal archive size.

Contrary to the results for a population of 64, it is shown that no clear difference is present when the population size is set to 32 between the two archive settings. Also, when an archive size of 16 is selected all values are found to lie in the Pareto front. For such settings, it is thus very likely that valuable information on the problem is lost.

From these results, it can be concluded that from among the tested settings, the optimum is found when applying a population of 64 and a solution archive with a size of 32.

Effect of the Number of Generations

It was shown that MHACO provides competitive results with MOEA/D after 128 generations. However, MOEA/D was previously already shown to also provide strong results for earlier generations. The results for different generations for MHACO are presented in Figure 9.16. Note that infeasible solutions have been left out of the figure.

The infeasible solutions are still abundant for the results of 48 generations, where for both archive sizes around half of the solutions remain infeasible. Still, the convergence towards a feasible solution space thus remains poor for MHACO in comparison to MOEA/D and SCEA.

Otherwise, the results show that the obtained Pareto front for MHACO are excellent between the generations. The best solutions are still find by having a larger population size compared to the solution archive, regardless of the number of generations. Initially, little improvement is seen between the results of 48 and 72 generations, but it should be noted that all infeasible individuals have disappeared when 72 generations are reached. After 128 generations a clear improvement is observed, while the solutions still remain within the feasible solution space for all individuals.

Random Seed Influence

To test the robustness of MHACO for re-entry mission optimization, it is also run for three random seed settings, the original seed of 123 and with a seed of 456 and 789. The results are presented in Figure 9.17(a). It is shown that between the different random seed settings only small differences are found in the Pareto front after 128 generations. The newly generated data even shows that MHACO can provide results that are closer to the previously found results of MOEA/D, which becomes especially clear for the random seed of 456. For high range angle trajectories, MOEA/D finds better solutions. The reason for this phenomenon is likely

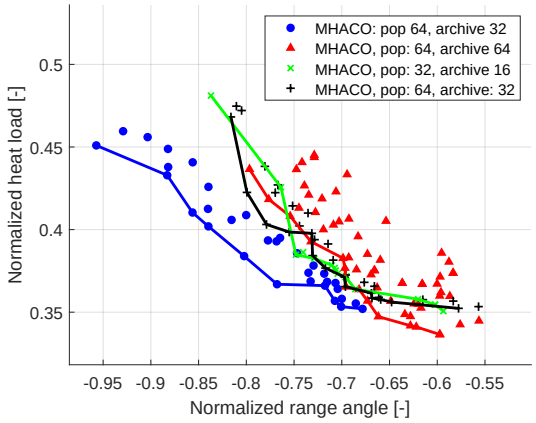


Figure 9.15: Results for different population and archive sizes of MHACO.

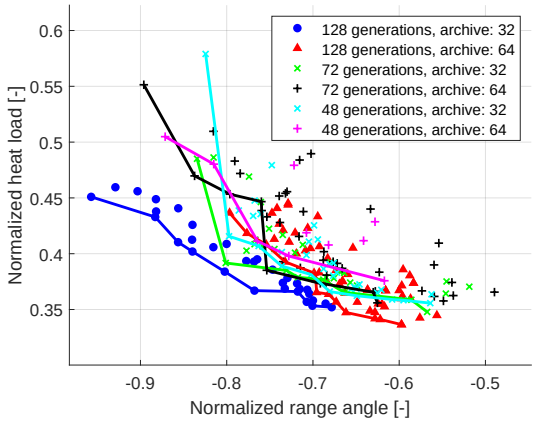
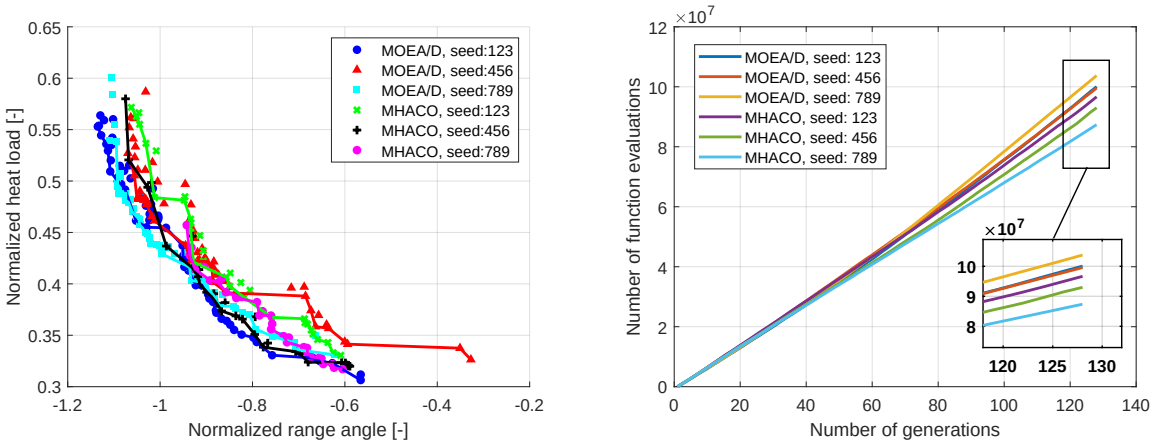


Figure 9.16: MHACO results for different generations.



(a) Pareto fronts after 128 generations

(b) Function evaluations

Figure 9.17: MHACO results for different random seeds.

related to the random seed, as the difference between the maximum obtained flight range value between MHACO and MOEA/D is only small.

Also note that the MHACO Pareto set for a random seed of 789 finds a more limited Pareto front, where the most optimal flight range value is -0.94. As the other two random seeds settings still yield the higher range angle trajectories, this can also be accredited to the random seed effect.

Secondly, the number of functions evaluations that is required for each random seed can be analyzed from Figure 9.17(b). It is shown that for each of the selected random seeds MHACO requires fewer function evaluations, thus saving computation time with respect to MOEA/D. The effect only becomes clear after 60 generations and a likely cause for this is that MHACO initially has more difficulties to find feasible solutions.

Given the results of the random effect, it becomes clear that MHACO can provide competitive Pareto fronts when compared to MOEA/D.

9.3.3. Optimal Trajectories

Below, the optimal trajectories for the different algorithms are presented. Their altitude and velocity profiles will be discussed as well as the resulting profiles of the angle of attack and the bank angle and the constraints. The results are presented in Figure 9.18. Firstly, a clear difference in the flight time for the maximum range and minimum heat load trajectories can be observed, as was expected.

When observing the altitude profiles in Figure 9.18(a) it becomes clear that all trajectories follow a skipping or lofting flight. Typically this is unwanted behavior as more load cycles are introduced. Importantly, these also differ significantly from the HORUS-2b reference trajectory, which follows a gliding profile. It should be noted that no direct constraint or objective is implemented in the algorithm that limits the number of peaks in the trajectory. For example, the work done by Papp (2014) did include a constraint on the number of heat flux peaks and there gliding trajectories are consistently obtained. It should thus be noted that this is still a significant drawback of the simulator that was developed for this work as especially a large number of heating cycles can cause significant stresses in the vehicle.

Observation of the constraints in Figure 9.18(e) and Figure 9.18(f) does show that the constraints that were applied to the current simulator are not exceeded for the trajectories. As expected, the heat flux constraint is nearly reached for all trajectories as it proves to limit the trajectory profile especially in the early stages when the velocity is high. Comparison with the bank angle profile in Figure 9.18(d) shows that the bank angle remains small at the start of the trajectory and especially when the vehicle starts skipping at approximately 300 s. For a gliding trajectory it would be expected that the value of the bank angles rises more strongly at the start of the trajectory to values over 60°, which is also the case for the HORUS reference trajectory. As this does not occur for the currently obtained guidance profiles, this is a clear cause of the skipping flight as the vertical lift remains too large to ensure gliding flight.

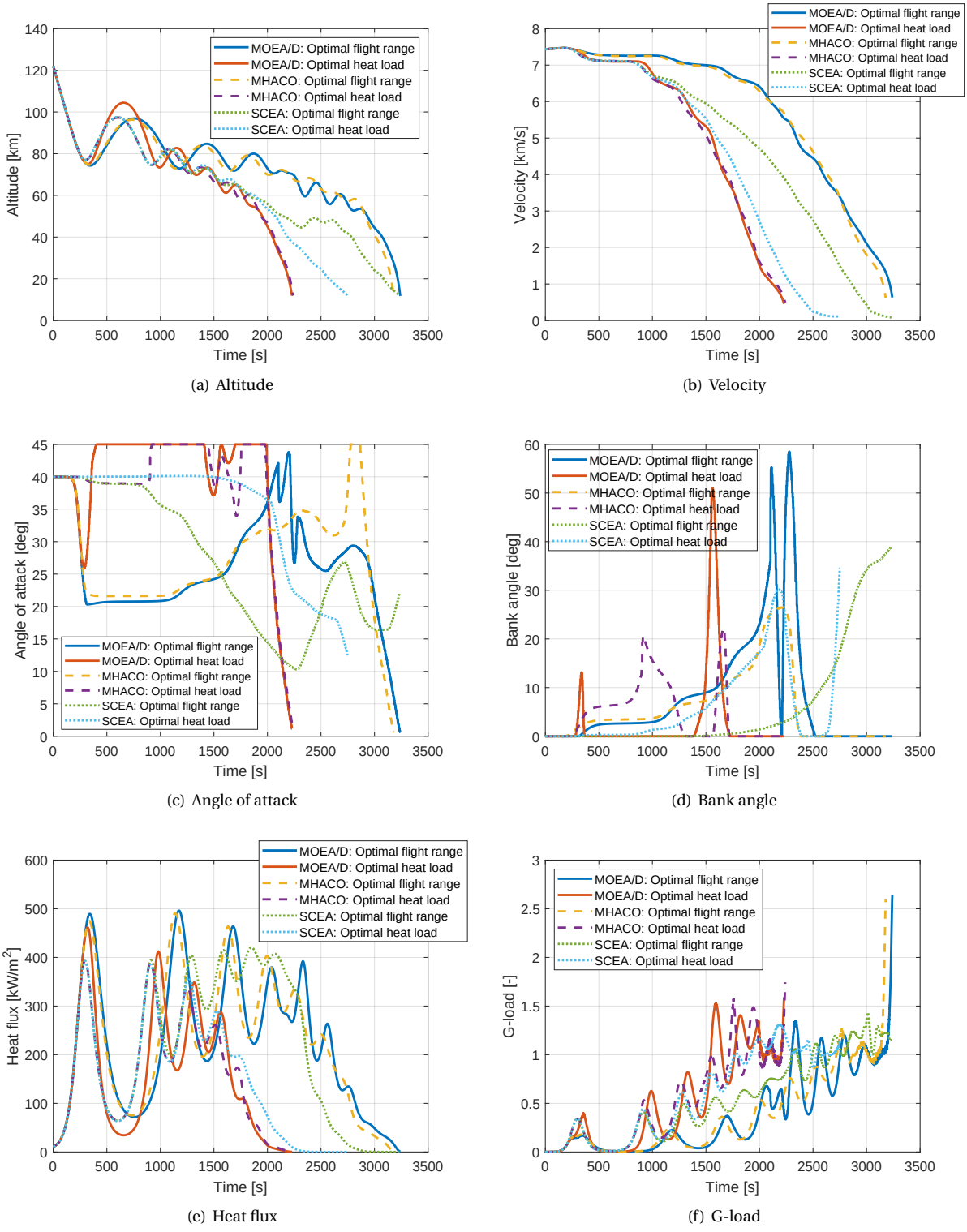
Going into the guidance profiles more specifically, again profiles are observed that are atypical for a gliding entry. As was stated before, larger bank angle values are expected at the beginning of the trajectory, while

they are observed at the end for the obtained simulation results. The angle of attack does show to be large for most of the optimal heat load trajectories, which is in line with expectations and similar to the reference trajectory that was used for verification. The strong decrease in angle of attack that is observed for the optimal flight range is not common for re-entry trajectories. The reason for this dip might be the reaction of the algorithm to the skip that occurs at the same and thus the algorithm incorrectly selects the angle of attack to lower the vertical lift, instead of the bank angle.

Moving to the final part of the trajectory, it is shown that the MOEA/D and MHACO trajectories move towards a value of zero for the angle of attack and bank angle, as is commanded by the guidance at the end of the trajectory. SCEA shows a different behavior, even though the final conditions are prescribed in the same manner. It should however be noted that the terminal conditions are applied for sea-level, standstill conditions and thus the exact final conditions have not yet been reached. The results from the SCEA trajectories thus show that this is not sufficiently accurate and should be altered for if the simulator is developed further. One clear suggestion would be to impose the final conditions at a higher altitude and include a proper terminal area interface.

The g-load behavior of MOEA/D and MHACO for the optimal flight range trajectories also still show unwanted behavior in the sharp increase of the g-load. Currently, the value still remains below the constraint of 3.0, although it is not unlikely that the constraint would be violated if the trajectory were to be propagated further. Again, the trajectory meets the imposed constraints and thus is accepted by the optimization algorithms, but it would be unlikely that such a trajectory would be selected for a flight planner as the uncertainty about the continuation of the g-load is large.

Finally, it should be noted that especially the trajectories that are obtained by MOEA/D and MHACO show strong similarities, as was also already observed in their Pareto fronts. This does confirm that for the currently implemented simulator, guidance and optimization constraints, the optimization are finding similar solutions. The SCEA trajectories show larger discrepancies as they have a longer flight time and thus the integrated heat load values will be larger.



(a) Altitude

(b) Velocity

(c) Angle of attack

(d) Bank angle

(e) Heat flux

(f) G-load

Figure 9.18: Maximum range and minimum heat load trajectories for the tested algorithms.

IV

Part IV: Conclusions and Recommendations

10

Conclusions

The following chapter presents the conclusions and recommendations of this thesis in Section 10.1 and Section 10.2, respectively.

10.1. Conclusion

The goal of this research was to test the performance of analytical trajectory methods for optimization of re-entry trajectories. The research questions and sub-questions that were formulated to support this research are repeated here:

To what extent can optimal two-dimensional re-entry trajectories be developed using a second-order analytical method with the objectives of maximizing the flight range and minimizing the heat load, while adhering to operational constraints?

1. *To what extent can the second order analytical solutions be used to generate an optimal guidance profile for re-entry?*
2. *How do results of the second-order methods for skipping and gliding flight compare for use in an optimization problem?*
3. *What differences are present in the Pareto optimal set of numerical and second order solutions?*
4. *How does Ant Colony Optimization compare to the previously used MOEA/D algorithm for re-entry trajectory optimization?*
5. *How does a variable chromosome length algorithm compare to the traditional fixed chromosome length optimization algorithms?*
6. *How does the increased complexity of a variable chromosome length weigh against the circumvention of node selection?*

To answer these questions, a re-entry simulator was developed and verified against data from literature. The simulator is capable of using either the HORUS-2B reference vehicle or a sample return mission capsule. The former has the benefit of being a winged vehicle, which allows for better controllability of the trajectory.

The environment in the simulator can be easily controlled, although for the most accurate simulations the following settings were used. The Earth was presented as a rotating sphere. The selected atmosphere model was the US1976 standard atmosphere model and the gravity terms were included up to J_2.

Through verification of the elements and the entire system, the simulator was shown capable of accurately representing the trajectory of HORUS-2B.

The developed guidance algorithm was based on angle of attack and bank angle commands that were located in the trajectory based on the normalized specific energy of the vehicle. The algorithm was extended to be able to handle a variable length input vector, based on a commanded number of guidance nodes. Through these guidance commands, the vehicle attitude is determined throughout the trajectory based on an interpolation scheme. It should be noted that a sampling time for the guidance is commanded to create a more realistic algorithm.

In parallel to the numerical simulator, an analytical simulator was set up based on the second order analytical equations developed by Vinh et al. (1993). These simulators were set up separately for the gliding and skipping flight.

Finally, these simulators were used in an optimization environment, which had the objectives of maximum the flight range and minimizing the heat load of the trajectory, while staying withing the operational constraints of the vehicle. The optimization environment made use of the MOEA/D, SCEA and MHACO algorithms to optimize the problem for the conflicting objectives.

The SCEA algorithm was developed specifically for this thesis based on previous literature. Instead of working with a fixed decision vector, it took additional integer elements, which prescribed the number of decision vector nodes that were actually used by the guidance. Except this more elaborate evolutionary operator, the method was otherwise based on the MOEA/D algorithm for fair comparison.

The first subquestion has been discussed for both skipping and gliding flight and it is now possible to conclude to what extent the second-order relations can be used to generate optimal guidance profiles. It was shown that both methods are capable of providing accurate results compared to an analytical simulator, although a number of significant drawbacks were identified.

For the gliding flight, this was the strong limitation in its flexibility. First, it was shown that no flexibility in the initial state was available as this was only dependent on the input entry altitude. Due to the equilibrium flight assumption present in the derivation of the method, the flight path angle remained below 1° for acceptable entry conditions. When the flight path angle showed larger values, the initial velocity had decreased to a point that was unrealistic for any re-entry mission. For the larger HORUS-2B vehicle, this caused a problem where skipping flight resulted from numerical simulations, whereas the analytical method would only generate trajectory with a monotonically decreasing altitude. The SRM capsule did provide accurate results for a range of values, although for values above 110 km, also here skipping flight was observed. Values below this altitude provide accurate results with errors below 3% when compared to numerical simulations.

The skipping flight also showed accurate results compared to numerical simulations. However, also here a drawback was found. When the trajectory would cross below the estimated deepest penetration depth, only imaginary values were returned by the method, thus losing valuable information of the most severe part of the skipping entry. This was mainly observed for low entry velocities, whereas also low values of C_{D_0} caused part of the trajectory to return imaginary values.

Now conclusions can be drawn with regard to the applicability of the second-order methods for use in the optimization problem. Although the flexibility of the method for gliding flight was low, it did provide accurate results and had a large benefit of not losing important information of the trajectory for all cases. The choice to move on to the optimization was made based on this conclusion and supported by the fact that presently the gliding flight is the main mode of entry.

The Pareto optimal solutions that were obtained for the analytical method and numerical method showed a close match and their Pareto optimal sets find values that are close to one another. However, due to the small flight path angle, the trajectory would show a strong bias towards optimizing the range by flying high in the atmosphere for as long as possible. The obtained Pareto optimal values would thus converge to a single point, instead of a front that is typically expected. The optimization algorithms would converge to this point for both numerical and analytical simulations, but the conclusion had to be that due to the limitations of the analytical method, the second order methods do not present a useful replacement for the numerical simulations.

The applicability of the analytical second-order methods for re-entry mission optimization can now be summarized as follows:

- Results for the gliding flight proved to be accurate when compared to numerical simulations.
- The methods for gliding flight carry the significant drawback that they impose the initial velocity and flight path angle based on the input altitude to ensure an equilibrium glide trajectory.
- Due to the imposed initial conditions it was shown that the obtained solution sets include a single point that is dominant over all other solutions.
- The methods for skipping flight were shown to be accurate with respect to numerical methods although due to the formulation of the method, complex numbers were found in the solution, thus rendering part of the trajectory useless.

Next, it was investigated how the variable chromosome length algorithm, SCEA, performed ti the traditional fixed-length MOEA/D algorithm. Both algorithms showed good convergence towards a feasible solution space and the Pareto optimum. It was found that SCEA outperformed the MOEA/D algorithm in its convergence to a feasible solution space and due to the selected constraint handling method, it would also find better Pareto fronts in the early generations. For later generations, MOEA/D was shown to outperform the SCEA algorithm in finding the best optima. This can be accredited to the implementation of the SCEA algorithm, where random values are generated when the number of nodes changes during crossover or mutation in the algorithm. The limitation in the number of nodes also resulted in a bias towards low heat load trajectories for a population size of 32, whereas a population of 64 would still provide a more diverse Pareto front.

Additionally, the number of function evaluations was shown to be larger for the SCEA algorithm, again caused by the less efficient evolutionary operators. Still, the number of function evaluations remained close to MOEA/D. Tests were performed to find an improvement by decreasing the sample rate of the guidance and increasing the search space of the number of nodes, but both proved to decrease the performance of SCEA.

However, when one notes that SCEA is capable of selecting its own nodes, which would typically be a design problem, a case can still be made for the algorithm. The required number of function evaluations is close to that of MOEA/D, although no need for designing the optimal number of nodes is required.

A final note on the performance of SCEA can be made with regard to the influence of the random effect. After testing both MOEA/D and SCEA for three different random seeds, it was found that SCEA was more susceptible to the random effect as larger discrepancies where found between the obtained Pareto fronts for SCEA and also the number of functions evaluations differed more strongly. Overall, this behavior corresponds with earlier conclusions where the random effect played a significant role when the number of nodes in the guidance vector was altered. Again, when the algorithm is extended to also include historical data when changing the guidance vector size for a new individual, it is likely that these discrepancies will decrease.

The conclusions for the performance of SCEA are as follows:

- SCEA was shown to be capable of finding Pareto sets similar to MOEA/D, although slightly worse and for a slight increase in the number of function evaluations.
- The random effect was shown to be more important for SCEA due to its method of generating an individual with a changed guidance vector length.
- It was shown that SCEA is capable of selecting its own required number of nodes, thus saving effort in the design of the guidance vector length.

Finally, the performance of MHACO with respect to MOEA/D was investigated to answer the final subquestion. First, it was shown that MHACO has stricter requirements with respect to the normalization of the problem, as it handles constraints as problem objectives. The obtained results showed a similar performance to MOEA/D and for an equal population size, MHACO was more efficient in terms of function evaluations. The Pareto fronts that were found by both algorithms where also close to each other, although MHACO results had difficulty finding values for a large range angle and heat load, similar to SCEA. Additionally, the influence of the random effect was also studied for MHACO and it was shown that MHACO was actually able to find a Pareto set that was close to MOEA/D for one of the selected random seeds. Also, for all seeds, MHACO required fewer function evaluations to obtain the Pareto front after 128 generations. This was partially caused by the shorter trajectories that were found by MHACO, but even for the more diverse Pareto fronts, which were similar to MOEA/D the required number of evaluations was found to be smaller.

The performance of MHACO is thus summarized as follows:

- MHACO was shown to provide Pareto sets that were competitive with MOEA/D.
- Due to the hypervolume method that was used to find the optimal solutions, MHACO is more sensitive to the normalization of the objectives and constraints and alterations had to be made.
- MHACO consistently required fewer function evaluations to find the optimal solutions.

Overall it can thus be concluded that MHACO and SCEA provide new algorithms that perform well for re-entry trajectory optimization, although MOEA/D still outperforms both in their current state.

Based on the previous results, the main research question of this thesis can be answered and it is now possible to conclude to what extent the analytical second-order methods can be used to develop optimal two-dimensional re-entry trajectories with regards to flight range and heat load, while adhering to the operational constraints. The selected methods do provide accurate results compared to the numerical models, but carry significant drawbacks when tested for a wider range of entry conditions. Mainly this inflexibility of the gliding method proved that it is not suited for re-entry mission design in its current form. As the goal of the thesis was to select one of these methods, the skipping flight was not further investigated in the optimization problem, as this had also previously been done in literature for a single skip. The results do prove promising, but a method should be developed that robustly omits the imaginary values that were now obtained.

10.2. Recommendations

Finally, the recommendations for future work in this field are listed below. They are categorized based on the analytical methods and the optimization methods.

Analytical Methods

- **Flexibility of the gliding phase**: Although no improvement to the analytical method was found in this thesis, it is clear that the equilibrium glide assumption limits the second order analytical method in its flexibility. Further research into these assumptions might yield better results and a more flexible entry state.

- **Application of skipping method**: Applying the skipping method to an optimization problem might provide more promising results than the gliding flight, when the current drawbacks are solved. Especially, the combination with the variable chromosome length, which was already developed for a variable number of legs, might show good results. Especially when considering that the introduction of legs prevents that the nodes have to be spread among the atmospheric phases in the skipping flight.

Optimization Methods

- **Historical knowledge of SCEA**: The developed SCEA algorithm could be extended and likely improved by improving the method by which genes are generated when the number of guidance node changes. For example, the data could be taken from individuals in the population that use these nodes or data from past generations could be stored.

- **Flexibility of SCEA**: The current SCEA was set up to deal with a fixed number of variables per node in the decision vector, which was 3 for this problem ($\hat{E}$, α and σ). However, the algorithm should be capable of dealing with a variable number of variables per node as well. This would become especially interesting when considering the addition of thrust to the trajectory, which would only occur at a very limited number of nodes. The current algorithm is capable of including such phases, although it would be in an inefficient way where a binary variable would have to be added to indicate whether the thrust is performed.

- **Constraint handling for MHACO**: MHACO proved to have difficulties in finding feasible solutions. A proposed method for added constraint handling would be to separate the feasible and infeasible solutions into separate archives and sort them in a manner similar to what was now applied for MOEA/D. Thus, feasible solutions are passed to the solution archive first and only afterwards infeasible solutions are added. However, the influence of the hypervolumes on the applicability of this method should still be studied.

- **Variable length MHACO**: Currently, only tests on a variable chromosome length where applied for SCEA, based on MOEA/D as the original literature on the topic was strongly based on genetic algorithms. Given the drawbacks of not using historical data, MHACO might provide a valuable algorithm for variable length individuals as it already stores a solution archive and thus has historical data readily available. More effort would especially have to be put into the method in which the variable length ants are generated by MHACO, as this is currently not easily supported.

Bibliography

Acciarini, G. (2020). "Optimizing a Solar Sailing Polar Mission to the Sun: Development and Application of a New Ant Colony Optimizer". Master's thesis, Delft University of Technology.

Acciarini, G., Izzo, D., and Mooij, E. (2020). "MHACO: A Multi-Objective Hypervolume-Based Ant Colony Optimizer for Space Trajectory Optimization". In: "2020 IEEE Congress on Evolutionary Computation (CEC)", pp. 1–8.

Anderson Jr., J. (1999). *Aircraft Performance and Design*. WCB/McGraw-Hill.

Bate, R., Mueller, D., and White, J. (1971). *Fundamentals of Astrodynamics*. Dover Publications, Inc. New York.

Boyd, I., Trumble, K., and Wrigth, M. (2010). "Modeling of Stardust Entry at High Altitude, Part 1: Flowfield Analysis". *Journal of Spacecraft and Rockets*, volume 47, no. 5, pp. 708–717.

Burden, R. and Faires, J. (2001). *Numerical Analysis*. Books/Cole, 7th edition.

Chapman, D. (1960). "An Analysis of the Corridor and Guidance Requirements for Supercircular Entry into Planetary Atmospheres". *Technical Report TR-R-55*, NASA, Washington D.C.

Coello Coello, C. (1999). "A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques". *International Journal of Knowledge and Information Systems*, volume 1.

Colorni, A., Dorigo, M., and Maniezzo, V. (1991). "Distributed Optimization by Ant Colonies". In: "European Conference on Artificial Life", pp. 134–142.

Dasgupta, D. and McGregor, D. (1992a). "Engineering Optimizations using the Structured Genetic Algorithm". In: "European Conference on Artificial Intelligence", .

— (1992b). "sGA: A Structured Genetic Algorithm". *Technical report*, University of Strathclyde.

Davies, C. and Arcadi, M. (2006). "Planetary Mission Entry Vehicles - Quick Reference Guide Version 3.0". *Technical report*, NASA. Technical report SP-2006-3401.

Dijkstra, M., Mooij, E., and Sudmeijer, K. (2013). "Trajectory Optimization to Support the Study of Hypersonic Aerothermodynamic Phenomena". In: "AIAA Atmospheric Flight Mechanics Conference", .

Eggers, A., Allen, H., and Neice, S. (1957). "A comparative study of the Performance of Long-Range Hypervelocity Vehicles". *Naca tn 4046*, National Advisory Committee for Aeronautics.

Engelsma, J. (2019). "Aerocapture Mission Analysis - Investigation and Optimisations for Earth, Mars, and Venus Trajectories". Master's thesis, Delft University of Technology.

Graves, C. and Harpold, J. (1970). "Reentry Targeting Philosophy and Flight Results from Apollo 10 and 11". In: "AIAA 8th Aerospace Sciences Meeting", AIAA Paper No. 70-28.

— (1979). "Shuttle Entry Guidance". *Journal of Aeronautical Sciences*, volume 27, pp. 239–268.

Lampinen, J. (2002). "A Constraint Handling Approach for the Differential Evolution Algorithm". In: "2002 Congress on Evolutionary Computation", pp. 1468–1473.

Li, H. and Deb, K. (2017). "Challenges for Evolutionary Multiobjective Optimization Algorithms in Solving Variable-Length Problems". In: "2017 IEEE Congress on Evolutionary Computation (CEC)", pp. 2217–2224.

Lu, P., Brunner, C., Stachowiak, S., Mendeck, G., Tigges, M., and Cerimele, C. (2017). "Verification of a Fully Numerical Entry Guidance Algorithm". *Journal of Guidance, Control, and Dynamics*, volume 40.

Lucassen, K. (2019). "Robust Non-linear Control for Winged Re-Entry Vehicles". Master's thesis, Delft University of Technology.

Mooij, E. (1995). "The Horus-2B Reference Vehicle". Delft University of Technology, Faculty of Aerospace Engineering. Memorandum M-682.

— (1998). "Aerospace-Plane Flight Dynamics. Analysis of guidance and control concepts." Ph.D. thesis, Delft University of Technology.

— (2016). *AE4870B - Re-entry Systems - Draft Lecture Notes*. Delft University of Technology.

Mooij, E. and Hänninen, P. (2009). "Distributed Global Trajectory Optimization of a Moderate Lift-to-Drag Re-entry Vehicle". In: "AIAA Guidance, Navigation, and Control Conference", doi:10.2514/6.2009-5770.

National Oceanic and Atmospheric Administration (1976). "U.S. Standard Atmosphere, 1976". *Technical report*, NASA. NOAA-S/T 76-1562.

Nyew, H., Abdelkhalik, O., and Onder, N. (2012). "Autonomous Interplanetary Trajectory Planning Using Structured-Chromosome Evolutionary Algorithms". In: "AIAA/AAS Astrodynamics Specialist Conference", pp. 1–29.

Olson, J., Craig, D., Maliga, K., Mullins, C., Hay, J., Graham, R., Graham, R., Smith, P., , Johnson, S., Simmons, A., Williams-Byrd, J., Reves, J., Herrmann, N., and Troutman, P. (2012). "Voyages - Charting a Course for Sustainable Human Space Exploration". Retrieved on November 3, 2020 from `https://www.nasa.gov/exploration/whyweexplore/voyages-report.html`.

Papp, Z. (2014). "Mission Planner for Heating-Optimal Re-Entry Trajectories with Extended Range Capability". Master's thesis, Delft University of Technology.

Rijnsdorp, J. (2017). "Performance of a Flush Airdata Sensor in a Particle Filter-Based Re-entry Navigation System". Master's thesis, Delft University of Technology.

Robinson, J., Wurster, K., and Mills, J. (2009). "Entry Trajectory and Aeroheating Environment Definition for Capsule-Shaped Vehicles". *Journal of Spacecraft and Rockets*, volume 46, no. 1, pp. 74–86.

Smith, M., Craig, D., Herrmann, N., Mahoney, E., Krezel, J., McIntyre, N., and Goodliff, K. (2020). "The Artemis Program: An Overview of NASA's Activities to Return Humans to the Moon". In: "2020 IEEE Aerospace Conference", pp. 1–10.

Vinh, N., Kim, E.K., and Greenwood, D. (1993). "Second-Order Analytic Solutions for Re-Entry Trajectories". In: "Flight Simulation and Technologies", doi:10.2514/6.1993-3679.

Watanabe, S. (2010). "Type definition mt19937". `https://www.boost.org/doc/libs/1_64_0/doc/html/boost/random/mt19937.html`.

Wertz, J. (2001). *Orbit and Constellation Design and Management*. Microcosm Press and Wiley.

While, L., Bradstreet, L., and Barone, L. (2011). "A fast way of calculating exact hypervolumes". *IEEE Transactions on Evolutionary Computation*, volume 16, no. 1, pp. 86–95.

Zhang, Q. and Li, H. (2007). "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition". *IEEE Transactions on Evolutionary Computation*, volume 11, no. 6, pp. 712–731.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C., and Grunert da Fonseca, V. (2003). "Performance Assessment of Multiobjective Optimizers: An Analysis and Review". *IEEE Transactions on Evolutionary Computation*, volume 7, no. 2, pp. 117–132.

Zuiani, F. and Vasile, M. (2013). "Multi Agent Collaborative Search based on Tchebycheff decomposition". *Computational Optimization and Applications*, volume 56, pp. 189–208.