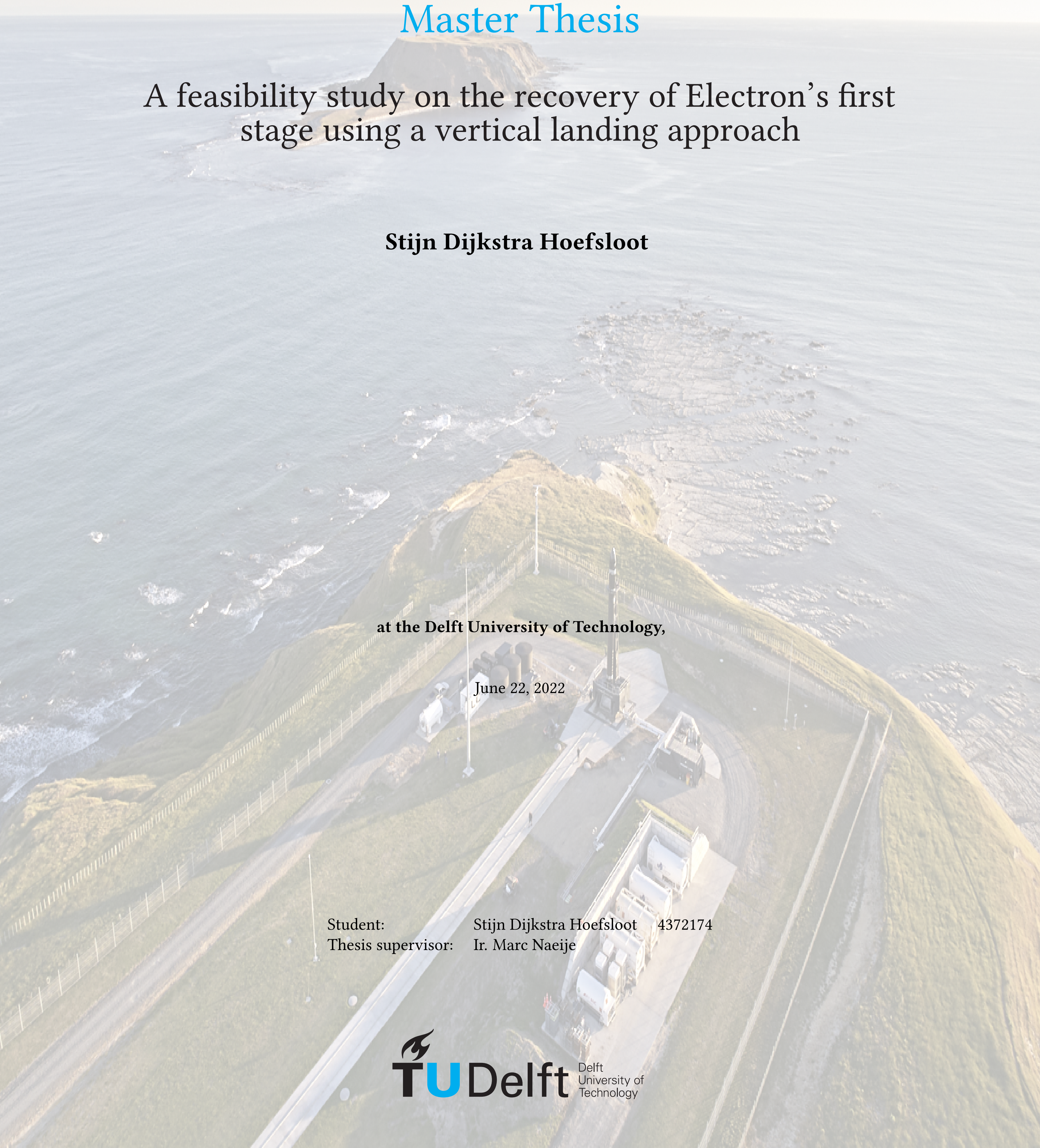


Master Thesis

A feasibility study on the recovery of Electron's first stage using a vertical landing approach

Stijn Dijkstra Hoefsloot

at the Delft University of Technology,

June 22, 2022

Student:	Stijn Dijkstra Hoefsloot	4372174
Thesis supervisor:	Ir. Marc Naeije	

Preface

As an Aerospace Engineering student on the brink of graduating the question ‘what is next?’ becomes more commonly asked by the day. In response to that question I often express my passion for the UN Sustainable Development Goals, especially regarding climate change, preservation of nature and affordable clean water supplies everywhere in the world. It’s not always easy to link the above mentioned to the often abstract and non-environmental friendly Space Industry. Fortunately for me (and the Earth) there is a growing demand for Earth Observation applications. With this study I am happy to contribute to making space more accessible for those kind of applications.

I would like to dedicate a a word to the people who are directly or indirectly involved in reaching this milestone. My thesis supervisor, Marc, thanks for the countless virtual tea meetings we shared throughout the year. Your positivity, understanding and feedback helped a lot getting through the difficult phases of the process. I would also like to thank my parents, Ellis en Michiel, for their unconditional love, patience and support throughout my studies. Many thanks to my roommates and friends who have had to listen to the many ’thesis life is much fun’ complaints longer than anyone had wished for. Thanks Juul for just being there.

Stijn Dijkstra Hoefsloot,
June 2022

Abstract

The small satellite market is rapidly growing and becomes more competitive by the day. In order to reduce launch costs, the concept of reusable launch vehicles wins popularity. Only few companies have managed to successfully recover and re-use their launch vehicle, of which SpaceX is the only one using a powered descent technique. This study focuses on applying that same technique on a small satellite launcher: Rocket Labs Electron. A 3-DOF trajectory optimization is performed to find the fuel optimal ascent and descent trajectories of the Electron. The original design of the Electron is adhered to as much as possible. The optimization is performed for a variety of Single- and Multi objective optimizers and different settings. Although this is only a feasibility study in which many assumptions are made, the results look promising for further research to be done.

List of Figures

List of Tables

Nomenclature

Abbreviations

CAGR	Compound Annual Growth Rate
CFD	Computational Fluid Dynamics
DE	Differential Evolution
DOF	Degree Of Freedom
DRL	Down Range Landing
ECEF	Earth Centered Earth Fixed
ECI	Earth Centered Inertial
EOM	Equation Of Motion
FFA	Federal Aviation Administration
GA	Genetic Algorithm
GE	Genetic Evolution
GSO	Geo-Synchronous Orbit
HBVP	Hamiltonian Boundary-Value Problem
ICAO	International Civial Aviation Organisation
LEO	Low Earth Orbit
LoS	Line of Sight
MAR	Mid-Air Retrieval
MEO	Medium Earth Orbit
NLP	Non-Linear Programming
PSO	Particle Swarm Optimization
RTLS	Return To Launch Site
SmallSat	Small Satellite
SQP	Sequential Quadratic Programming
SSO	Sun-Synchronous Orbit
TVC	Thrust Vector Control

US76	United States Standard Atmosphere Model 1976	
WP	Work Package	

Symbols

α	Angle of Attack	deg
β	Sideslip angle	deg
δ	Deflection angle	deg
δ	latitude	deg
δ_{max}	Maximum gimbal angle	deg
$\dot{\delta}_{max}$	Maximum gimbal rate	$\mathrm{deg} \cdot \mathrm{s}^{-1}$
$\dot{m}$	mass flow	$\mathrm{kg} \cdot \mathrm{s}^{-1}$
γ	flight-path angle	deg
κ	heading angle	deg
λ	Line of Sight angle	deg
$\mathbf{a}$	Euler axis	deg
Φ	Euler angle	deg
ϕ	Roll angle	deg
ψ	Yaw angle	deg
ρ	density	$\mathrm{kg} \cdot \mathrm{m}^{-3}$
σ	Bank angle	deg
τ	longitude	deg
θ	Pitch angle	deg
C_D	Drag force coefficient	-
C_L	Lift force coefficient	-
D	Drag force	kN
g	Gravity constant	$\mathrm{m} \cdot \mathrm{s}^{-2}$
h	Geometric altitude	km
H_s	Scale height	km
I_{sp}	Specific Impulse	s
L	Lift force	kN
L_{z_i}	Thermal lapse rate	$\mathrm{K} \cdot \mathrm{km}^{-1}$
M	Mach number	-
p	Roll rate	$\mathrm{deg} \cdot \mathrm{s}^{-1}$
q	Pitch rate	$\mathrm{deg} \cdot \mathrm{s}^{-1}$

q_c	Convective heat flux	$\mathrm{W \cdot m^{-2}}$
q_{max}	Maximum heat flux	$\mathrm{Wm \cdot ^{-2}}$
r	Yaw rate	$\mathrm{deg \cdot s^{-1}}$
R_N	Nose radius	m
S_{ref}	Reference Area	$\mathrm{m^2}$
T	Thrust force	kN
T_{max}	Maximum thrust force	kN
T_{min}	Minimum thrust force	kN
U	Gravity potential	$\mathrm{J \cdot kg^{-1}}$
V_c	Circular velocity	$\mathrm{m \cdot s^{-1}}$
V_g	Groundspeed	$\mathrm{m \cdot s^{-1}}$
V_{inf}	Free stream velocity	$\mathrm{m \cdot s^{-1}}$
z	Geopotential altitude	km

Constants

μ_E	Gravitational parameter of the Earth	$3.986004418\mathrm{x}10^{-14}\ \mathrm{m^3 s^{-2}}$
ω_E	Angular velocity of the Earth in Inertial space	$7.2921150\mathrm{x}10^{-5}$ rad/s
ρ_0	Air density at sea level	$1.225\ \mathrm{kg \cdot m^{-3}}$
g_0	Gravity constant at sea level	$9.80665\ \mathrm{m \cdot s^{-2}}$
R^*	Universal gas constant	$8.31446261815324\ \mathrm{J \cdot K^{-1} \cdot mol^{-1}}$
R_0	Radius of the Earth (Equatorial)	6356.766 km

Contents

1 | Introduction

The current global SmallSat market has an estimated value of 2.8 billion USD, according to a research conducted by MarketsAndMarkets [16]. The same research estimates an average CAGR of 20.5% for the coming 5 years, forecasting a global market value of 7.1 billion USD by 2025. Furthermore, in 2020 more than twice the number of SmallSats have been sent into space with respect to 2019 [17], despite the setbacks of the pandemic. NASA classifies satellites with a mass below 500kg as 'small'. SmallSats are relatively cheap and are widely adapted for commercial, communication and research purposes. Currently, the gross amount of SmallSats is delivered to space as piggyback payload (rideshare on a launch with bigger payload and higher priority). If a company wants to sent a SmallSat to space it is thus fully dependent on the launch schedule of the larger prioritized payload. With the increasing popularity of the market more and more launch companies offer launches for SmallSats only. An example of such company is Rocket Lab, which has successfully delivered its first payload to space in 2018. One way of reducing the cost of such launches is to re-use the launch vehicles. Rocket Lab currently focuses on Mid Air Retrieval (MAR) to retrieve the first stage of their launch vehicle, Electron. This study focuses on the retrieval of Electrons first stage by means of a powered descent, which finds its relevance in the rapidly growing market. This study focuses on the technical feasibility of Return To Launch Site (RTLS) mission for a SmallSat launcher, Rocket Labs Electron in specific. The configuration of the Electron is kept as close to the original as possible. Most of Rocket Labs missions have a 500x500km SSO orbit as their destination. This orbit will serve as the nominal target orbit that has to be reached by the ascent trajectory. The following research question and sub-questions were established of which the answers will be discussed in Chapter 10:

Research question: Can an implementation of vertical powered descent techniques safely bring the Electron back to Earth - while flying a fuel optimal trajectory?

Sub-questions:

- To what extent should the Electron be modified in order to return back to Earth?
- What guidance algorithm suits the application best and how does the model perform without (or heavily simplified) guidance in place?
- What is the optimal amount of used fuel for the return of Electrons' first stage w.r.t the landing accuracy, and what are the trade-offs?
- How does the choice of integrator and optimizer algorithm affect the accuracy of the model?

In Chapter 2 the concept of different recovery methods is briefly discussed, followed by a breakdown of the Electron vehicle and its launch sites in Chapter 3. In Chapter 4 the Equation of

Motions and aerodynamics of the model are discussed. In addition an appropriate integrator and propagator are selected. In Chapter 5 several atmosphere and gravity models are discussed. In Chapter 6 the ascent and descent trajectory models are presented and validated. In Chapter 7 the optimization of the trajectory model is explained, followed by the optimization results presented in Chapter 8. This report concludes with a sensitivity analysis and conclusion and recommendation in Chapter 9 and Chapter 10, respectively.

2 | Recovery

The recovery of a first stage, or any other part of the vehicle, can be accomplished in several ways. Roughly spoken the recovery can be divided into two phases. One phase comprising the return trajectory, the *retrieval* phase. The second phase comprises the refurbishment of the retrieved parts, the *refurbishment* phase. In this study, and many others, the *recovery* is referred to the *retrieval* phase. Although refurbishment is a not to be underestimated field of study that plays a large role in calculating the cost efficiency of re-usable rocket(s) parts, it is not included in the scope of this project. Examples of vehicle recoveries are the Apollo capsules, Space shuttle and Falcon 9's first stage/booster, each making use of different landing techniques.

2.1 Feasibility

No matter the technological possibilities, most problems diverge to the question whether solving the problem at hand is profitable. Especially for commercial applications. The Space shuttle e.g. was praised for its ability to return to Earth such that it could be used again for future missions. However, the main reason why the Space Shuttle program was eventually shut down, was because of the expensive refurbishment costs. SpaceX claims to reduce the overall launch costs of the Falcon 9 by 30% [18], which gives them a strong competitive advantage in the launcher market.

2.2 First Stage Recovery

2.2.1 Retrieval configurations

Mid-air retrieval

Mid-air recovery aims to intercept a descending object using a helicopter. In the beginning of the re-entry phase the object typically uses a heat shield to decelerate and dissipate heat from the vulnerable components. After a while the vehicle deploys a parachute. A helicopter approaches the vehicle and captures the parachute, with a cable attached underneath the helicopter and returns it to land or a nearby drone ship. In the work of Grevlee it is concluded that MAR is an especially good way to retrieve the most costly components of a launch vehicle and not so much the entire first stage [19]. Bachelor students of the TU Delft aerospace faculty conducted a study aimed to recover the key components of a heavy launch vehicle, reducing the cost per launch by an estimated 15% for the Ariane 6 launch vehicle [20]. In the work of Merle Snijders a cost effectiveness analysis of the first stage recovery of the European SMILE project is conducted by optimizing the 3-DOF descent trajectory using MAR techniques [21]. In Snijders study a MAR approach was favoured over a vertical descent recovery because of the aerospike engine featured in the SMILE.

Rocket Lab is currently investigating the possibilities to recover the Electron's first stage using a MAR method. This study has the same objective but uses an alternative method.

Powered descent

SpaceX has gained worldwide recognition for the successful powered descent of their Falcon 9. Powered descent trajectories are typically divided in two groups, Return to Launch Site (RTLS) and Down Range Landing (DRL) trajectories. The RTLS and DRL shematic flight profiles are illustrated in Figure 6.2 and Figure 2.2 respectively. Contrary to DRL, RTLS requires an extra boost-back burn to change its direction back to the launch site (see Figure 2.1). Contant estimated an extra 17% and 12% of the first stage fuel mass for a RTLS and DRL trajectory respectively [1].

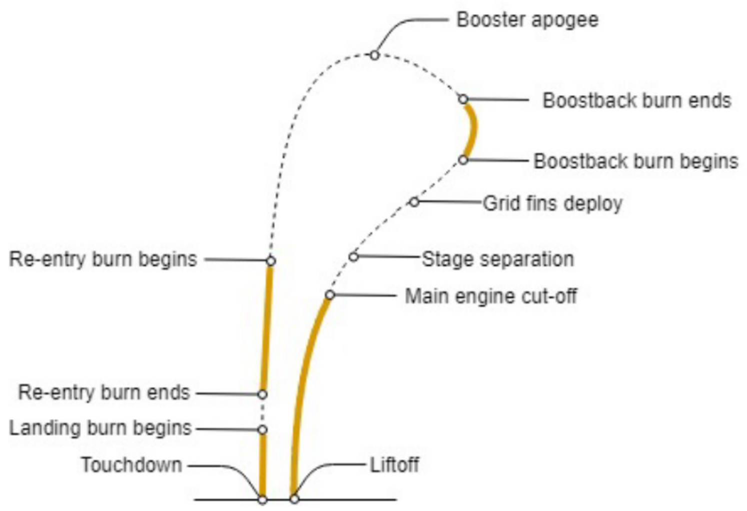


Figure 2.1: Return To Launch Site flight profile [1]

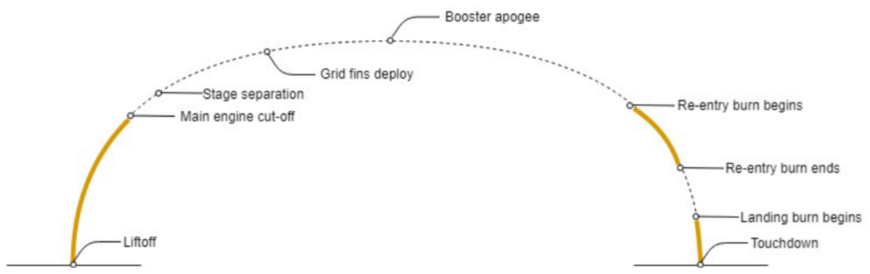


Figure 2.2: Down Range Landing flight profile [1]

2.3 second stage recovery

On the day of writing SpaceX is the only company that has successfully performed a complete first stage recovery. Hitherto it seems far grasped to consider a second stage recovery as a noteworthy solution to reduce overall launch costs. Nevertheless, Pepermans conducted a thorough research on the cost feasibility of a second stage recovery[22]. The main findings conclude a 6.3% overall cost reduction for LEO missions and infeasible findings for MEO and GSO. As Pepermans mentions, a second stage recovery should only be considered once the first stage recovery methods are fully successful.

3 | Vehicle: Electron

All vehicle characteristics, including geometry, propulsion systems, vehicle constraints etc. is limited to the amount of information Rocket Lab has made publicly. Electrons Payload User's Guide 2020 [2] provides a clear picture of the Electron. The Electron is a two-stage orbital class rocket specially designed by Rocket Lab to meet the increasing demand of launching small satellites into space. An optional kick-stage can be added to the vehicle and allows for unique orbit payload injections. The Electron can carry a nominal payload of 200kg to a nominal sun-synchronous (SSO) orbit. Table 3.1 gives an overview of the Electrons mass and geometry characteristics. Electrons first stage typically separates from the second stage at an altitude of 78km [2].

Table 3.1: Electron specifications [2][12]

	Overall	**first stage**	**second stage**	**Kickstage/ payload**
Length [m]	18	12.1	2.4	∼3.5
Diameter [m]	1.2	1.2	1.2	∼1.2
Mass at liftoff [kg]	13,000	10,200	2,300	500
Dry mas [kg]	1,200	950	250	-
Fuel mass [kg]	11,300	9,250	2,050	-
Engine	-	9x Rutherford	1x Rutherford vacuum	1x Curie
Thrust [kN] (single engine)	-	24	22	-
Specific Impulse [s]	-	311	341	-
Fuel/Oxidizer	-	RP-1/LO$_x$	RP-1/LO$_x$	-

3.1 Geometry and mass

Rocket Lab is currently trying to recover the Electrons first stage by means of a mid-air recovery. My study however, focuses on the first stage recovery using powered descent techniques. In the thesis work of Contant an estimated 10% and 17% of fuel should be kept reserved for a DRL and RTLS trajectory respectively[1]. Contrary to a mid-air capture, a vehicle performing a powered descent trajectory requires landing gear. The Falcon 9 landing gear is estimated by Contant to be 10% of the dry mass of the first stage whereas the extra mass of the grid fins is considered negligible[1]. This is similar to the estimated landing gear mass of a Mars Rover (9.41%) in the work of Price [23]. The mass of the MAR system (parachute etc.) is estimated to be around 12.5% of the vehicles first stage[24]. The mass initially needed for the MAR system can for this application be redistributed to the landing legs and grid fins.

The landing gear is assumed to have little effect on the aerodynamic properties of the vehicle as they are only unfolded at the last phase of the descent trajectory. Figure 3.1 schematically depicts how the landing gear is folded against the rocket. A similar system is assumed for the Electron.

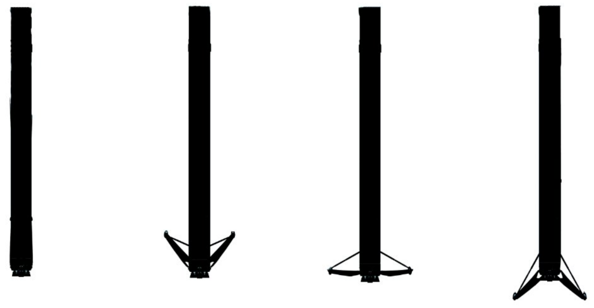


Figure 3.1: Schematic representation of the Falcon 9's landing gear folding capabilities [1]

3.2 Engine and Propulsion

Accurate trajectory simulation requires for an accurate and complete thrust model. The thrust model is highly dependent on the engine and its design parameters. Electrons first stage makes use of 9 identical Rutherford sea level engines (Figure 3.2) whilst the second stage makes use of a single Rutherford vacuum engine.

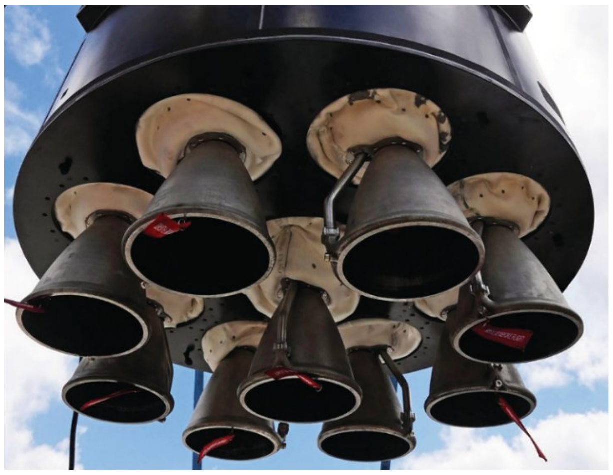


Figure 3.2: Electrons first stage engine configuration featuring 9 Rutherford engines [2]

The Rutherford engine is specifically designed for the Electron rocket and can deliver a thrust of 24kN. The engine features a fully electric propulsion cycle, making use of brushless DC electric motors. The Electron is the first orbital class rocket using an electric pumped (RP-1/LO$_x$) engine. The propellant pumps are powered by a set of high-performance lithium polymer batteries. Contrary to traditional gas cycle engines, electric engines are relatively easy to build. The Rutherford engines reach an efficiency of 90%. Furthermore, Rutherford is the first RP-1/oxygen engine that uses 3D-printed parts for all of its primary components. The Rutherford engine is fully designed and manufactured by Rocket Lab itself, at Long Beach headquarters in California, USA.

3.3 Launch sites

Rocket Lab currently operates two launch facilities, located in New Zealand and the USA. Both sites together offer more than 130 launch opportunities per year. The results obtained in chapter 8 would be applicable to any launch site with only minor modifications. This research only focuses on launches from Launch Compex 1, Rocket Labs main launch site.

Launch Complex 1, Mahia, New Zealand

Launch Complex 1 is the world's only private orbital launch range [2]. The base is licensed by the FFA and supports up to 120 launches per year. The base is located in the Hawke's Bay at (39.262°S, 177.865°E). Complex 1 supports trajectories with inclinations ranging from 39° to 120°. Other inclinations are available upon request.

(a) Launch Complex 1, Maghia, New Zealand [2]

(b) Launch Complex 2, Virginia, USA [2]

Figure 3.3: Rocket Labs launch complexes

Launch Complex 2, Virginia, USA

Launch Complex 2 'only' supports up to 12 missions annually. Complex 2 is located Wallops Island at (37. 834°N, 75.488°W) allows inclinations between 30° and 60° [2].

4 | Flight Mechanics

4.1 reference frames

Reference frames are essential in studying the motion of an object in space. Without any, its impossible to define the state of an object. To put it technically, a reference frame is specified by a set of three mutually orthogonal direction vectors. The origin of the frame is located at the point where the three vectors coincide, point (0,0,0). In celestial mechanics the c.o.m. of the central body often serves as the origin of a frame. A coordinate system specifies how the state variables are defined within a reference frame, Cartesian or spherical elements for example. Reference frames exist in all sorts of forms with endless variations and can be chosen arbitrarily. Although there exist no 'wrong' reference frames, the formulation of a bodies motion can become unnecessarily complex if a reference frame in combination with a coordinate system is not selected cautiously.

Two types of reference frames can be distinguished; *inertial* and *non-inertial* reference frames. According to Newton's Law of Motions, an *inertial* frame is per definition at rest or moves in a linear fashion with constant velocity. For some (Earthly) applications it has practical value to express the motion of an object in a rotating reference frame. For re-entry applications for example. As the Earth rotates around its own axis, the designated landing spot of a re-entry vehicle on Earth's surface moves w.r.t the *inertial* frame, at the same angular velocity as the Earth (ω_E). Introducing a *non-inertial* rotating frame with a angular velocity identical to the central body eliminates this problem.

Translational motion
Many variations of (inertial) reference frames can be distinguished. In re-entry studies two reference frames are commonly used to describe the translational motion of a vehicle, the *inertial planetocentric* and *rotating planetocentric* frame[3]. Both illustrated in Figure 4.1 and discussed below.

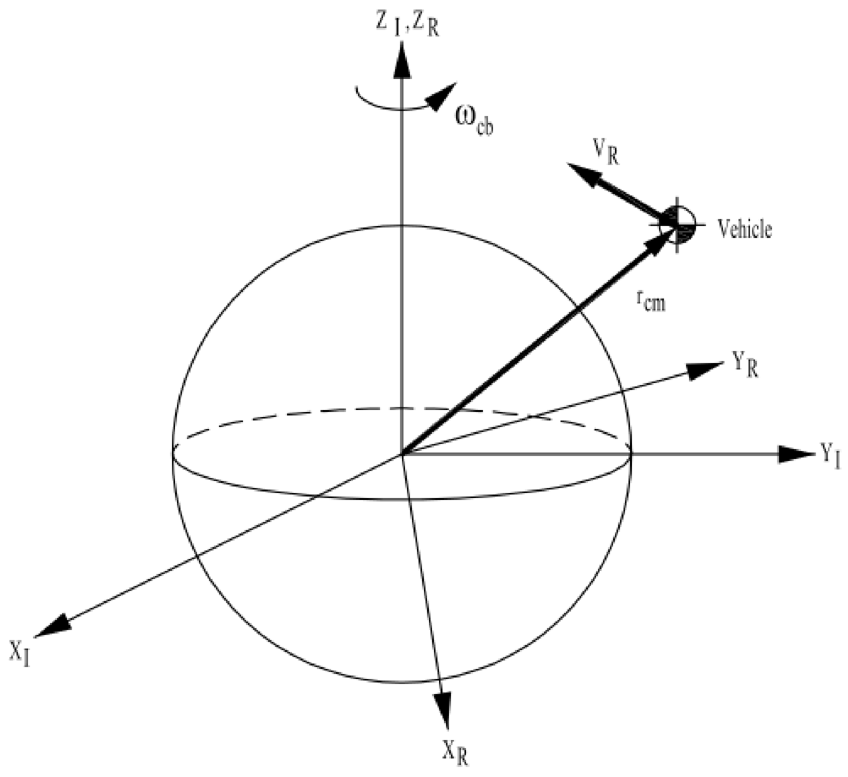


Figure 4.1: A vehicle w.r.t. two similar reference frames, I and R represent the *inertial planetocentric* and *rotating planetocentric* frame respectively.

Inertial planetocentric reference frame, Index I

One of the characteristics of a planetocentric frame is that the OX_IY_I plane coincides with the equatorial plane of the central body. The Z_I-axis points north and is thus coincident with the rotational axis of the central body. The direction of the X_I-axis is defined by by prime meridian at zero time. The Y_I-axis fulfills the right-handed system [3].

For Earthly applications the above mentioned definitions are generally not used. In a Earth Centered Inertial frame (ECI) the X_I is pointed towards a fixed point in inertial space. In the commonly used $J2000$ frame this corresponds to the Vernal Equinox[3].

Equation 4.1 describes the translational motion of an arbitrary vehicle with variable mass in the inertial frame. In this study we will restrict ourselves to rigid body dynamics, meaning that the mass distribution of the vehicle is constant over time. Consequently the last two terms on the right hand side of Equation 4.1 become zero, simplifying the equation to Equation 4.2.

$$\mathbf{F_I} = m\frac{d^2\mathbf{r_{cm}}}{dt^2} + 2\boldsymbol{\omega} \times \int_m \frac{\delta\tilde{\mathbf{r}}}{\delta t}dm + \int_m \frac{\delta^2\tilde{\mathbf{r}}}{\delta t^2}dm \tag{4.1}$$

$$\mathbf{F_I} = m\frac{d^2\mathbf{r}_{\text{cm}}}{dt^2} \tag{4.2}$$

The above described frame is considered inertial. However, technically such frame is pseudo-inertial because of the motion of the central body itself. For an Earth orbiting object the most convenient reference frame is the ECI frame. However, the Earth itself rotates around the sun. The effects of this rotation are marginal and negligible in most cases. Such frames can thus considered to be inertial [3].

Rotating planetocentric reference frame, Index R

This frame is very similar to the inertial planetocentric frame. In this case the frame is fixed to the (rotating) central body. It coincides with the inertial planetocentric frame at zero time and after every full rotation of the central body.

Rotational motion
To define the rotational motion of a vehicle, reference frames are used that have their origin at the c.o.m. of the vehicle. Again, many different combinations of reference frames and coordinate systems are available. The most commonly used frames in re-entry studies are discussed below. It is assumed that the vehicle has at least one plane of symmetry. Right handed coordinate systems are commonly used in this field. For a more complete overview of the available reference frames the reader is referred to [3]

Body reference frame, index B

This body frame is fixed to the vehicle. The X_B- and Z_B-axis both lie in the plane of symmetry and are defined positive in forward and downward direction respectively. The Y_B-axis conforms to the right-handed system. This is the general definition used for aircraft. The signs of the direction vectors can be chosen differently if it suits the application better. This could be desirable for a re-entry vehicle that enters the Earth's atmosphere backwards, like the Apollo capsule[3].

Aerodynamic reference frame (airspeed based), index AA

The X_{AA}-axis is defined along the velocity vector of the vehicle relative to the atmosphere. The Z_{AA}-axis is collinear with the aerodynamic lift force (based on airspeed), but opposite in direction. The Y_{AA}-axis again completes the right-handed system. A similar definition exists for a ground-speed based reference frame. In this case the X_{AA}-axis would be defined relative to the ground-speed (the rotating planetocentric frame).

4.2 state variables

The state of a vehicle is defined by its state variables. The state describes the vehicles position and velocity and the vehicles attitude and angular rates in translational and rotational dynamics, respectively. The state variables can be expressed in different ways, dependent on the coordinate system that suits the application best. It has to be emphasized that any arbitrary set of state variables can be converted to any other set. As aforementioned with the reference frames, carefully picking the state variables is key in reducing the complexity of the problem. Another important reason to carefully chose the state variables is the occurrence of possible singularities. A good example is the Gimbal-lock phenomena, further elaborated on in Section 4.2.2.2. Other singularities regarding the Kepler elements are well explained in the Propagation and Optimization lecture notes [25]. One has to be aware of the possible singularities and chose the state variables such that they are avoided.

4.2.1 Position and velocity

The position and velocity can be described in several state representations. TUDAT (see Section 5.4 for more information) supports the following 5 representations[26] of which the first 3 will be discussed below.

- Keplerian elements
- Cartesian elements
- Spherical elements
- Modified Equinoctial elements
- Unified State Model elements.

4.2.1.1 Keplerian elements

In many applications related to astrodynamics, such as vehicles orbiting celestial bodies or interplanetary trajectories. In case of orbital elements, the position and velocity of an object in an elliptical orbit can be defined by six parameters, w.r.t. to an inertial frame. The state vector then equals $\mathbf{x} = [e, a, i, \omega, \Omega, M]$. The orbital elements are visualized in Figure 4.2 and Figure 4.3.

Orbital element	**description**
e	eccentricity ($0 \leq e < 1$)
a	semi-major axis ($a > R_e$)
i	inclination ($0^\circ \leq i \leq 180^\circ$)
ω	argument of pericentre ($0^\circ \leq \omega \leq 360^\circ$)
Ω	longitude of the ascending node ($0^\circ \leq \Omega \leq 360^\circ$)
M	true anomaly ($0^\circ \leq \Omega \leq 360^\circ$)

Table 4.1: Orbital elements for an elliptical orbit [3]

The shape of the ellipse is defined by the eccentricity. The eccentricity of the ellipse is defined as $1 - a/b$, where a and b represent the height and width of the ellipse respectively. When the eccentricity is 0, this means that the orbit is a perfect circle. For $e = 1$, the orbit becomes parabolic and for $e > 1$ the orbit becomes hyperbolic. This means that the orbit is no longer closed and the object escapes the gravitational pull from the central body. To determine the position of an object on the orbit at a given time, an extra parameter is required, the time of pericentre passage. It is deemed irrelevant to dive deeper into the orbital elements and the math behind it. If interested, the reader is referred to [6], covering all the ins and outs of celestial- and astrodynamics.

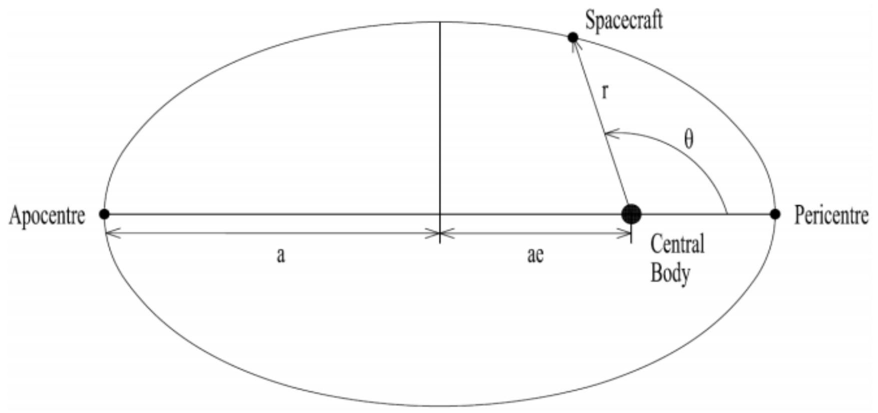


Figure 4.2: Definition of semi-major axis a and the eccentricity e. The spacecraft is moving at a distance r; the true anomaly is indicated by θ

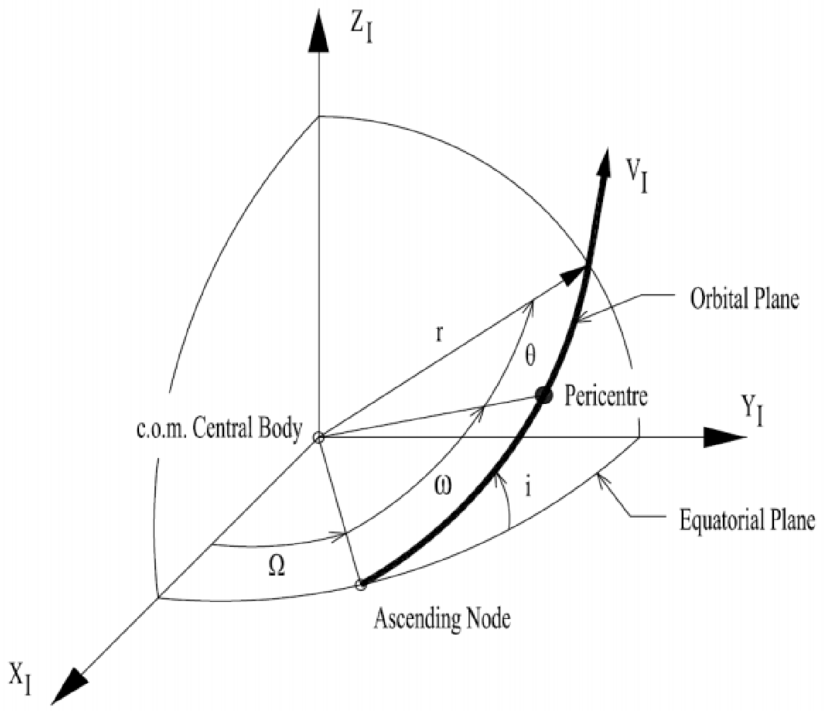


Figure 4.3: Definition of the three orbital parameters Ω, ω and i. The spacecraft is moving at a distance r with a velocity $\mathbf{V_I}$ w.r.t. to the inertial planetocentric frame (index I). The true anomaly is indicated by θ [3]

4.2.1.2 Cartesian elements

Cartesian coordinate systems are very straightforward and without doubt the most widely used. Cartesian elements have revolutionized the field of mathematics by providing the first methodical link between algebra and Euclidean geometry. Position and velocity can either be described w.r.t the I- or R-frame. The same variables are used but with a different subscript (I or R). The position and velocity are defined as (x, y, z) and $(\dot{x}, \dot{y}, \dot{z})$ respectively. In the R-frame, the velocity components are often expressed as (u, v, w) instead of $(\dot{x}_R, \dot{y}_R, \dot{z}_R)$, illustrated in Figure 4.4.

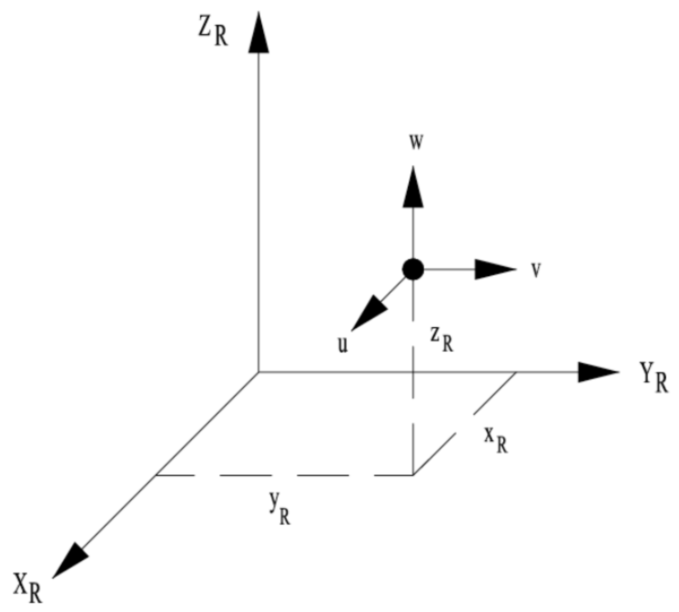


Figure 4.4: Cartesian components illustrated w.r.t the R-frame [3]

4.2.1.3 Spherical elements

Spherical coordinates are very popular. In the field of avionics and re-entry studies, spherical coordinates are more intuitive than for example Cartesian coordinates. The position and velocity components are expressed as follows and illustrated in Figure 4.5:

Position: distance R, longitude τ, latitude δ
Velocity: groundspeed V_g, flight-path angle γ_g, heading angle χ_g

The longitude is measured positively to the east ($0^\circ \leq \tau < 360^\circ$). The latitude is measured from the Equator, positive in northern direction and negative toward the south. R represents the distance of the origin of the reference (c.o.m of central body) toward the c.o.m of the vehicle. This distance is the equivalent of the modulus of the Cartesian position coordinates (x, y, z). V_g resembles the relative velocity of the vehicle w.r.t the rotating planetocentric frame. In contrast to the Cartesian elements, where the velocity is made up of three velocity vectors, the velocity is defined by a magnitude, (V_g), and two direction angles, γ_g and χ_g. γ_g defines the angle between the velocity vector and the local horizontal plane $(-90^\circ \leq \gamma_g \leq 90^\circ)$, positive for a velocity vector below the local horizon. χ_g defines the direction of the velocity vector in the local horizontal plane w.r.t the local northern direction $(-180^\circ \leq \chi_g \leq 180^\circ)$. $\chi_g = 0^\circ$ the vehicle moves in northern direction. For $\chi_g = 90^\circ$ and $\chi_g = -90^\circ$ the vehicle moves parallel to the central bodies equator in Eastern and Western direction respectively.

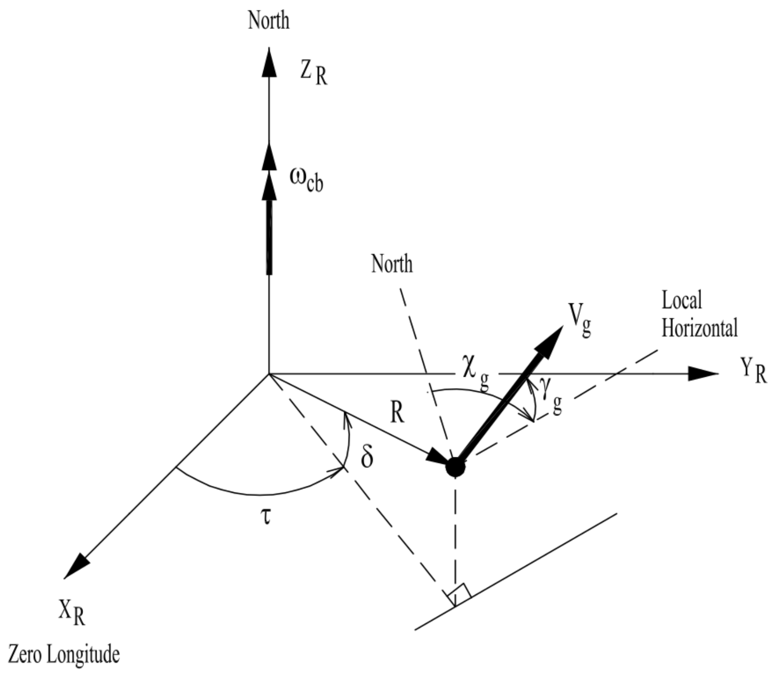


Figure 4.5: Spherical components illustrated w.r.t the R-frame [3]

4.2.2 Attitude and angular rates

Whereas the position and velocity describe the translational motion of a vehicle, the attitude and angular rates describe the rotational motion of the vehicle. The *attitude* of a vehicle is defined as the orientation of a body-fixed reference frame to another one and can be expressed in several ways. Three commonly used definitions in re-entry studies are discussed here.

4.2.2.1 Euler angles

Euler angles define a body-fixed frame w.r.t an inertial frame. The *classical attitude angles* are the roll angle Φ, pitch angle θ and yaw angle ψ. Usually these angles are defined w.r.t inertial space (inertial planetocentric frame e.g.) but they can also be defined w.r.t the local horizontal plane. The latter definition is used in the attitude indicator instruments of aircraft for example. The order in which a transformation (or rotation) is performed is important. In aerospace applications the 3-2-1 sequence is most commonly used[3]. That means three consecutive rotations around the Z- (yaw rotation), Y- (pith rotation) and X-axis (roll rotation) respectively.

Aerodynamic angles also form a set of Euler angles, defined as a sequence with order 2-3-1. Instead of roll, pitch and yaw the angle of attack α, bank angle σ and angle of sideslip β are used. The *aerodynamic angles* describe the orientation of the aerodynamic frame w.r.t the body reference frame (α and β) and w.r.t the trajectory frame (σ). Recall the earlier mentioned reference frames. The angles are illustrated in Figure 4.6.

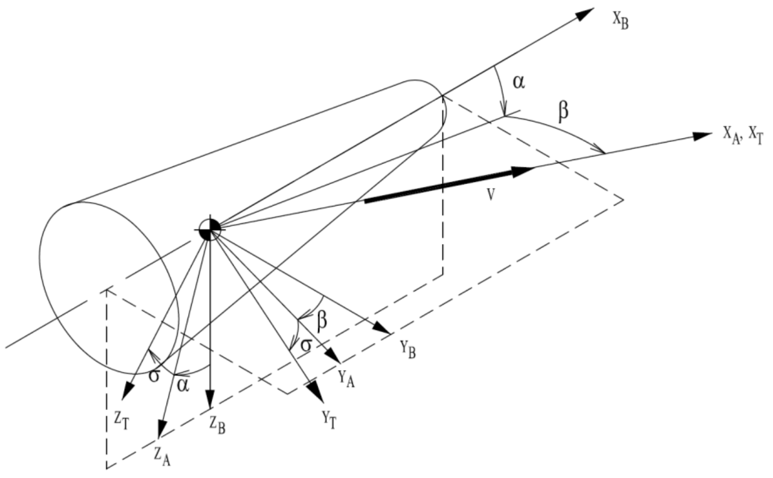


Figure 4.6: Aerodynamic angles α, β and σ. The depicted reference frames are the body frame (B), trajectory frame (T) and aerodynamic frame (A). α, β and σ are defined positive here.

4.2.2.2 quaternions

Quaternions are a very elegant way to describe the attitude of a vehicle. A quaternion is a 4-dimensional hyper-complex number consisting of 3 imaginary and 1 real number. The imaginary numbers satisfy the following constraint:

$$i^2 = j^2 = k^2 = ijk = -1$$

A rotation is specified by 4 quaternions, a vector and scalar part, given below. **a** and **Φ** denote the Euler axis and Euler angle respectively.

$$\mathbf{q} = (q_1, q_2, q_3)^T = \mathbf{a} \sin\Phi/2 \quad \text{and} \quad q_4 = \cos\Phi/2$$

The rotation can be expressed as $(\mathbf{q}, q_4)$. The 4 quaternions satisfy the following constraints and are therefore not mutually independent.

$$\mathbf{q}^T\mathbf{q} + q_4^2 = q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1$$

Its difficult to interpret the physical meaning of a set of quaternions, other than for example a pitch angle. The math behind the quaternions is not discussed here. It is a rather complicated topic with its own algebra rules and specific properties. Quaternions have proven themselves to be very efficient, computationally wise. Another advantage of a quaternion representation is that its made up of 4 parameters, eliminating all possible singularities, such as a *gimbal lock*. In TUDAT all attitude calculations and transformations happen through quaternions[26]. To output a desired attitude representation, the quaternions are simply multiplied by the corresponding transformation matrix.

4.2.2.3 angular rates

The angular rate of a vehicle is defined as the rotational velocity of the body-fixed frame w.r.t the inertial frame. This leads to a rotation vector $\omega = (p, q, r)$, where p, q and r define the roll, pitch and yaw rate respectively, illustrated in Figure 4.7

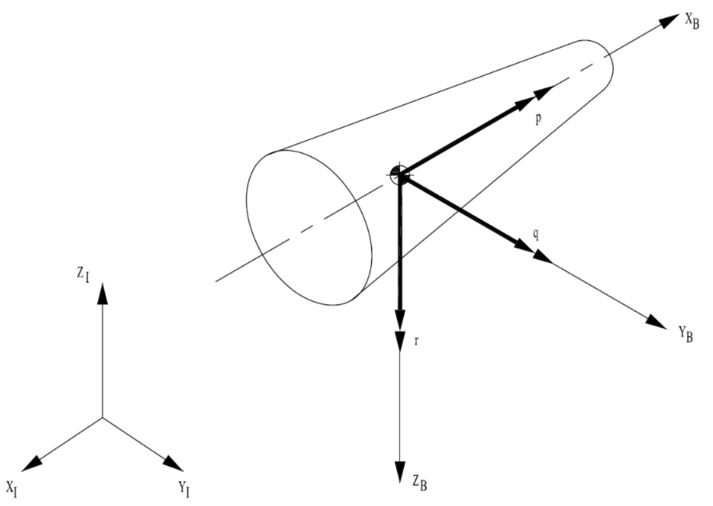


Figure 4.7: Definition of $\omega = (p,q,r)^T$

4.3 Frame transformations

A frame transformation is nothing less than going from one frame to another. A complete transformation consists of a translation and a rotation. A translational transformation is easily achieved by adding a translational vector to the center of origin of the current frame. A rotation between two arbitrary frames can always be decomposed into a sequence of unit axis rotations. The transformation matrices to rotate over the X-, Y- and Z-axis respectively are as follows, for any arbitrary angle α:

$$\mathbf{C}_1(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix}$$

$$\mathbf{C}_2(\alpha) = \begin{bmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 0 & \cos\alpha \end{bmatrix}$$

$$\mathbf{C}_3(\alpha) = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The re-entry systems lecture notes [3] provide a neat overview of several standard transformation matrices. Many of these are incorporated and tested in TUDAT and ready to use.

4.4 Equations of Motion

4.4.1 Mass depletion

In most aerospace applications that have to do with a thrust force, the mass of the vehicle can not considered to be constant. The used fuel reduces the current mass of a the vehicle significantly over time. The thrust force of a rocket engine is generally computed using the Rocket Thrust Equation (Equation 4.3).

$$T = \dot{m} I_{sp} g_0 \tag{4.3}$$

With $\dot{m}$ the mass flow and I_{sp} the specific impulse of the engine. The thrust forces that Electrons engines are able to deliver are publicly known, as is the specific impulse. Rewriting Equation 4.3 leads to an expression for the mass flow (Equation 4.4). The mass equation is added to the set of state differential equations. TUDAT provides build-in models that account for the mass depletion, only having to specify the thrust force, specific impulse, initial time and the number of engines.

$$\dot{m} = \frac{T}{I_{sp} g_0} \tag{4.4}$$

4.4.2 Translational dynamics

Translational dynamics describe the translational motion of an object subjected to a force. The dynamics can be expressed in many different ways and reference frames where each discipline favours its own representation. The general expression for the translational motion of a vehicle in a rotating planetocentric frame is given by Equation 4.5. The equation holds for any arbitrary rigid body with variable mass. It is assumed that the angular rate ($\boldsymbol{\omega}_\mathbf{R}$) of the central body is constant.

$$\mathbf{F}_\mathbf{R} = m\frac{d^2\mathbf{r}_{\text{cm}}}{dt^2} + 2m\boldsymbol{\omega}_\mathbf{R} \times \frac{d\mathbf{r}_{\text{cm}}}{dt} + m\boldsymbol{\omega}_\mathbf{R} \times (\boldsymbol{\omega}_\mathbf{R} \times \mathbf{r}_{\text{cm}}) \tag{4.5}$$

$\mathbf{F}_\mathbf{R}$	=	sum of all external forces in the rotating frame
$\frac{d^2\mathbf{r}_{\text{cm}}}{dt^2}$	=	apparent acceleration of the vehicle in the rotating frame
$2\boldsymbol{\omega}_\mathbf{R} \times \frac{d\mathrm{r}_{\text{cm}}}{dt}$	=	Coriolis acceleration due to the motion of the vehicle in the rotating frame
$\boldsymbol{\omega}_\mathbf{R} \times (\boldsymbol{\omega}_\mathbf{R})$	=	Centrifugal acceleration due to the rotational velocity of the Earth.

In re-entry studies the EOM's are commonly expressed in the previously discussed spherical flight parameters and can be written as follows. For the derivation of the equations the reader is referred to [3] as they become rather long and tedious.

$$\dot{V} = -\frac{D}{m} + g\sin\gamma + \omega_{cb}^2 R\cos\delta(\sin\gamma\cos\delta - \cos\gamma\sin\delta\cos\chi) \tag{4.6}$$

$$\begin{aligned} V\dot{\gamma} = \frac{L\cos\sigma}{m} - g\cos\gamma + 2\omega_{cb}V\cos\delta\sin\chi + \frac{V^2}{R}\cos\gamma + \\ + \omega_{cb}^2 R\cos\delta(\cos\delta\cos\gamma + \sin\gamma\sin\delta\cos\chi) \end{aligned} \tag{4.7}$$

$$V\cos\gamma\dot{\chi} = \frac{L\sin\sigma}{m} + 2\omega_{cb}V(\sin\delta\cos\gamma - \cos\delta\sin\gamma\cos\chi) + \\ + \frac{V^2}{R}\cos^2\gamma\tan\delta\sin\chi + \omega_{cb}^2 R\cos\delta\sin\delta\sin\chi \tag{4.8}$$

Whereas from the dynamic EOM's the velocity components of the vehicle are obtained, the position parameters are obtained from the kinematic equations, listed below, in the flight-path reference frame.

$$\dot{R} = \dot{h} = V\sin\gamma \tag{4.9}$$

$$\dot{\tau} = \frac{V\sin\chi\cos\gamma}{R\cos\delta} \tag{4.10}$$

$$\dot{\delta} = \frac{V\cos\chi\cos\gamma}{R} \tag{4.11}$$

This representation will be used throughout my model.

4.5 Choice of integrator and propagator

To calculate the actual flight path of the trajectories the equations of motion established in Chapter 4 are numerically integrated. Many different numerical integration techniques are available where each method has its own pro's and cons. The propagator of a system defines the type of state variables in which the system is propagated. In the field of astrodynamics propagating the system in Cartesian coordinates can becomes cumbersome and counter intuitive. Instead the system could be propagated in using Unified State Model (USM) parameters or Gauss Orbital elements. The available propagators and integrators available within the TUDAT environment are listed Table 4.2 and Table 4.3 respectively. Due to the 3-DOF nature of the problem at hand, the propagation of the rotational, and therefore rotational propagators, are not considered.

Propagators

Table 4.2: List of available propagators in the TUDAT environment[1]

Propagator	State Variables
Cowell	Cartesian
Encke	Cartesian
Gauss Keplerian[27]	Keplerian elements
Gauss Modified Equinoctial[28]	Modified equinoctial elements
USM quaternions[29]	quaternions
USM Rodrigues[29]	modified Rodrigues parameters
USM exponential[29]	exponential map

The Cowell propagator has proven itself very well for ascent and descent trajectories with relatively short flight times and is used amongst others in the work of van Kesteren[30] and Rozemeijer[31]. An in-depth propagator analysis is therefore not conducted and the propagator choice for this study will be Cowell.

[1]`https://py.api.tudat.space/en/latest/propagator.html`

Integrators

Two important types of integrators can be distinguished: fixed step-size and variable step-size integrators. Fixed step-size integrators make use of the same step-size for each integration step while variable step-size integrators can vary the step-size during the propagation dependent on the forces acting on the system. Variable step-size integrators are especially useful interplanetary spaceflight and propagation of orbital trajectories. In case the propagated object reaches the pericentre of an orbit or performs a fly-by around a celestial body, a small step-size is desired because of the higher gravitational accelerations around that point. For other parts of the trajectory such small step-sizes might not be required and unnecessarily increases the number of function evaluations (and therefore computational effort). The Runge-Kutta Fehlberg (RKF) variable step-size method was first published in the work of Fehlberg[32], to which the reader is referred for a thorough explanation of how the method works exactly. The first number after RKF denotes the order of the integrator and the second number denotes the order of the error estimator that is used to control the step-size. A higher order generally means higher accuracy's at the cost more integration steps. In order to pick an integrator a trade-off has to be made between the accuracy of the integrator and the number of required function evaluations.

Table 4.3: Integrator settings available within the TUDAT environment[2]

Integrator	Step	Step size type
Euler	Single	Fixed
RK4	Single	Fixed
RKF4(5)	Single	Variable
RKF5(6)	Single	Variable
RKF7(8)	Single	Variable
RKF7(8)DP	Single	Variable
Adam Bashfourt Moulton	Multi	Fixed
Bulirsch Stoer	Extrapolation	Variable

In order to identify the best integrator an integrator analysis is conducted for both the ascent and descent trajectory. In Figure 4.9 the performance of several integrators with different settings is plotted. All integrators with corresponding settings below the dashed line meet the accuracy requirements. From that selection the integrator with the lowest number of function evaluation is selected: $RKF7(8)$ with tolerances of $10E-9$.

A similar analysis is conducted for the ascent trajectory of the second stage to orbit. It has to be noted the propagation of the second stage already starts with an initial error (final error of the Launch Vehicle propagation). The results are plotted in Figure 4.10. Following the same reasoning as aforementioned the $RKF5(6)$ integrator with tolerances $10E-9$ is selected.

At last an integrator analysis for the descent trajectory is conducted. For the descent trajectory both the position error and the velocity error are examined, plotted in Figure 4.11 and Figure 4.12 respectively. It immediately stands out that none of the integrators, not even the higher order ones, are capable of achieving high accuracy's. At first the position accuracy requirement for the descent trajectory was set at 100m instead of 1000m. However, due to the integrator performance the requirement was raised to 1000m. Although the Bulirsh Stoer integrator with tolerances of

[2]`https://py.api.tudat.space/en/latest/integrator.html`

$10E-13$ lies just below the 100m mark, the amount of function evaluations is not practical for the optimization process and therefore not considered as a potential integrator. The maximum velocity accuracy error is set to 1 m/s. It can be noticed that there is no integrator that satisfies both the position and the velocity requirement. The integrator of choice for the descent trajectory is $RKF5(6)$ with tolerances of $10E-9$, achieving an accuracy of 686 m and 0.1 m/s for the position an velocity respectively.

The 'poor' performance of the descent trajectory integrator is best explained with the help of Figure 4.8, where the accumulated position error is plotted over time. The system is propagated for a number of (very small) step-sizes, using an $RKF7(8)DP$ integrator. It can be noticed that the position error skyrockets for all step-sizes at $t = 150s - 160s$ and can be linked to the start of the boostback burn. The sudden acceleration change in opposite direction of the velocity vector apparently causes a large offset in the accuracy of the integrator.

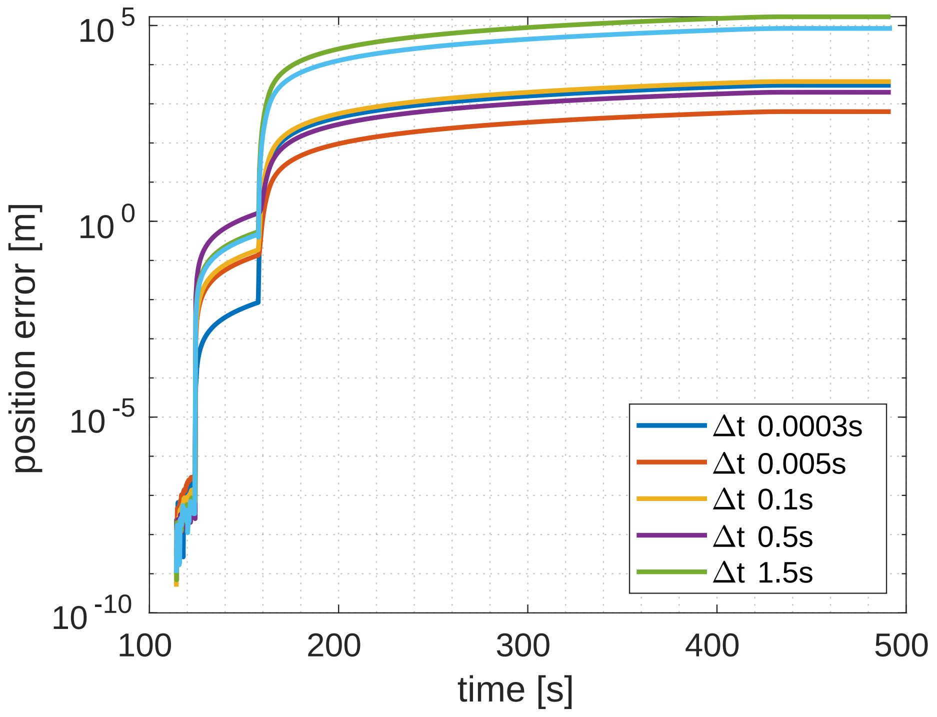


Figure 4.8

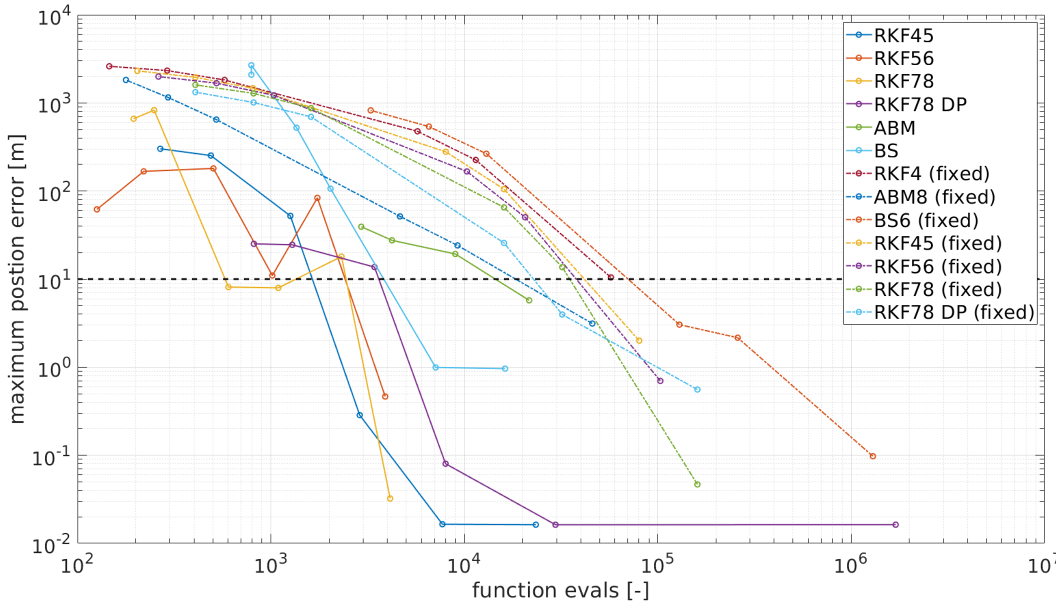


Figure 4.9: Integrator analysis for multiple variable and fixed step-size integrators, using the Cowell propagator. Variable step-size tolerances: *10E-13, 10E-11, 10E-9, 10E-7, 10E-5, 10E-3*; Fixed step-size stepsizes: *0.01s, 0.1s, 0.5s, 1s, 2s, 4s*

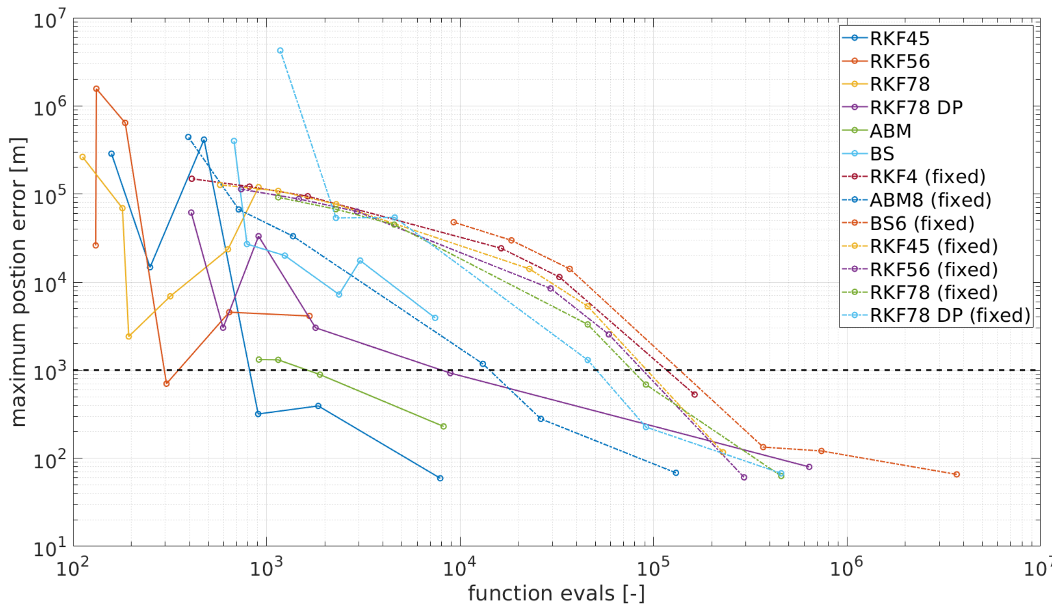


Figure 4.10: Integrator analysis for multiple variable and fixed step-size integrators, using the Cowell propagator. Variable step-size tolerances: *10E-13, 10E-11, 10E-9, 10E-7, 10E-5, 10E-3*; Fixed step-size stepsizes: *0.01s, 0.1s, 0.5s, 1s, 2s, 4s*

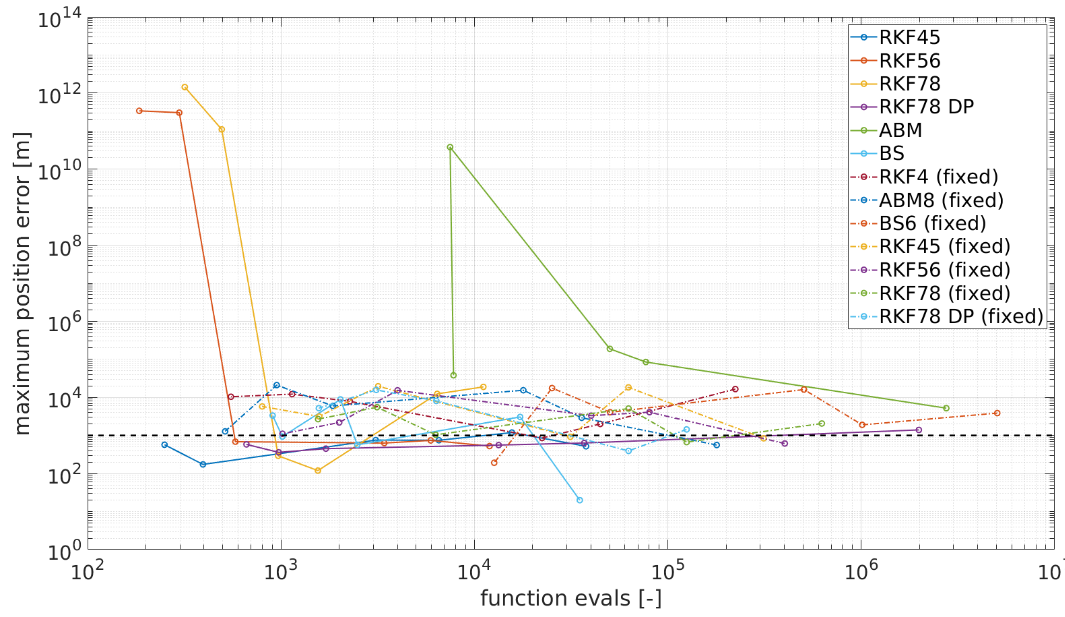


Figure 4.11: Integrator analysis for multiple variable and fixed step-size integrators, using the Cowell propagator. Variable step-size tolerances: *10E-13, 10E-11, 10E-9, 10E-7, 10E-5, 10E-3*; Fixed step-size stepsizes: *0.01s, 0.1s, 0.5s, 1s, 2s, 4s*

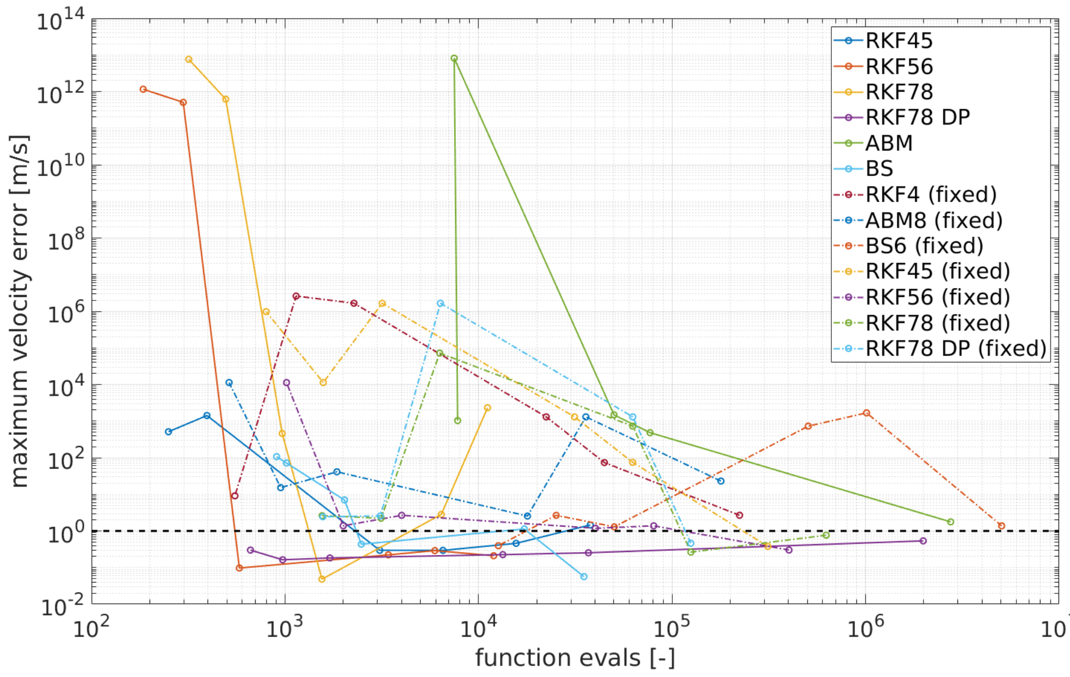


Figure 4.12: Integrator analysis for multiple variable and fixed step-size integrators, using the Cowell propagator. Variable step-size tolerances: *10E-13, 10E-11, 10E-9, 10E-7, 10E-5, 10E-3*; Fixed step-size stepsizes: *0.01s, 0.1s, 0.5s, 1s, 2s, 4s*

4.6 Aerodynamics

For many space applications the translational and rotational dynamics of a body are studied separately as they (barely) influence each other. For atmospheric flights however, the translational and rotational dynamics are coupled with one another through the aerodynamic forces acting on the vehicle, clearly seen in Equations 4.6, 4.7 and 4.8.

The aerodynamic lift- (L) and drag-force (D) can be computed using Equation 4.12 and 4.13. The tricky part is finding the right values for the aerodynamic coefficients (C_D, C_S, C_L or C_X, C_Y, C_Z, in aerodynamic and Cartesian components respectively). C_S represents the aerodynamic side-force, primarily caused by wind. For the sake of simplicity the side-slip angle is set to 0 deg throughout this study. C_S and the resulting side force are therefore also 0. The coefficients are comprised of the complex dependencies of several factors (Mach number, angle of attack, vehicle shape, flow conditions, etc.) and no analytical solution exists to compute these coefficients. As a result the coefficients are estimated making use of Missile DATCOM software.

$$L = 0.5 * C_L * \rho_{air} * V_\infty^2 * S_{ref} \tag{4.12}$$

$$D = 0.5 * C_D * \rho_{air} * V_\infty^2 * S_{ref} \tag{4.13}$$

4.6.1 Missile DATCOM

Missile DATCOM[33] is a powerful tool used in many preliminary rocket/missile design problems to estimate the required aerodynamic coefficients. Within the TUDAT environment this works as follows. First the user has to define the flight conditions and the vehicle shape. The flight conditions consist of the range of Mach numbers and angle of attack for which the coefficients are created, Table 4.4. Missile DATCOM supports angle of attacks up to around 20-30 degrees due to the the non-linear behaviour in computing the coefficients above that range.

Table 4.4: Missile DATCOM flight conditions

M [-]	0.3, 0.6, 0.8, 0.9, 0.95, 1.0, 1.05, 1.1, 1.2, 1.3, 1.4, 1.6, 1.8, 2.0, 2.5, 3.0, 3.5, 4.0
α [deg]	-20, -15, -10, -7, -4, -2, -1, 0, 1, 2, 4, 7, 10, 15, 20

For a large range of conventional rocket shapes there is a coefficient database available, only having to specify the length and diameter of the rocket. The database consists of rockets with uniform diameter and a (blunted) nose. Contant used this database to efficiently simulate many different vehicle configurations[1]. Knowing the length and diameter of the Electron, the coefficients that correspond best to the shape of Electron can be retrieved from the database. However, the exact shape of the Electron is made public by Rocket Lab - including the shape of the fairing - and depicted in Figure 4.14. Its well worth to define Electron's exact vehicle characteristics in Missile DATCOM and compare the estimated coefficients to the coefficients available in the database. In Figure 4.13 a significant difference in C_D between both methods is observed. The difference is significant enough to use the coefficients specifically generated for the Electron throughout this study.

Once the look-up tables of coefficients is created it can be imported to the TUDAT environment. C_D and C_L can now be used for each integration step as a function of M and α, using bi-linear interpolation.

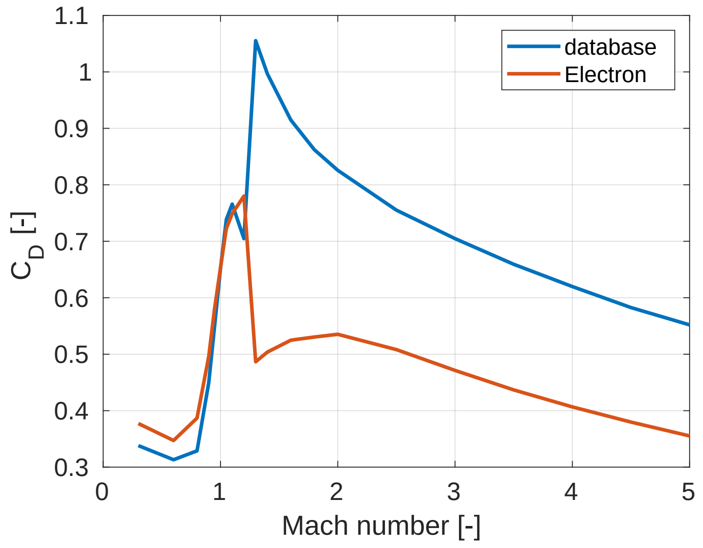


Figure 4.13: Aerodynamic drag force coefficient (C_D) as a function of Mach number (M) for angle of attack ($\alpha = 0$).

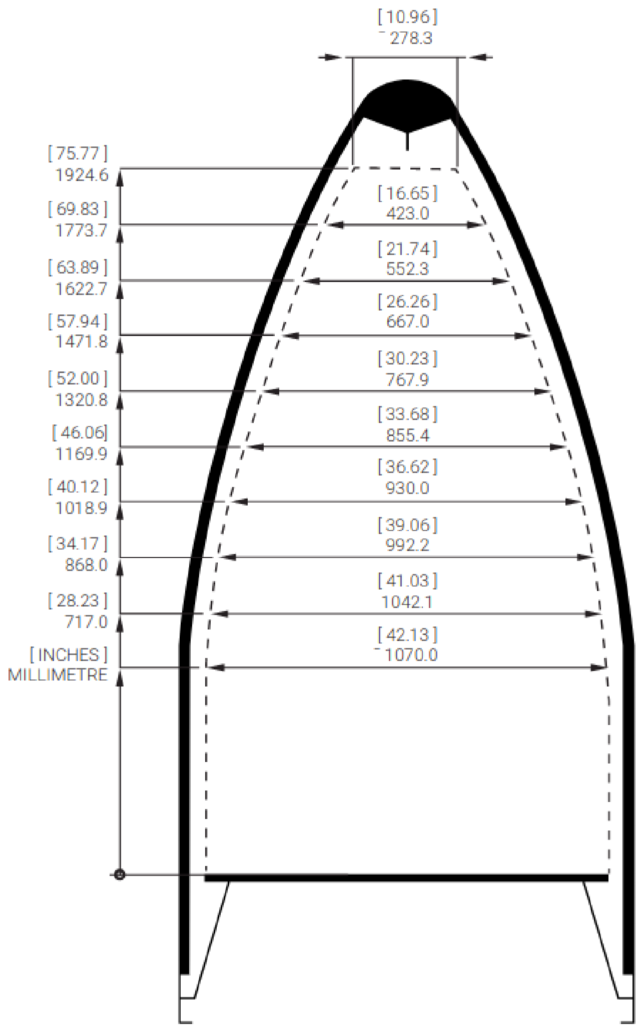


Figure 4.14: Shape characteristics for Electrons fairing[2]

Grid fins

Grid fins are a rather unconventional type of control surfaces used for aerodynamic guidance. A grid fin is characterised as an outer frame attached to the rocket having an internal grid framework. Grid fins are most famous for their application in Space X' Falcon 9 where the fins are used to both decelerate the vehicle and control its attitude [34]. The main advantage of a grid fin is its low hinge moment due to a low chord length, allowing the actuator to be small. Other advantages are efficient packaging, good lift capabilities in the subsonic velocity regime and high strength to weight ratio [35]. Due to its advantages, grid fins have been applied to a wide variety of missiles. Examples include; the OTR-21 Tochka tactical ballistic missile of the Former Soviet Union, R-77 air-to-air missile of Russia, GBU-43/B Massive Ordnance Air Blast (MOAB) of the USA, Falcon 9 rocket, Falcon heavy rocket, etc. [36]

Grid fins rotate about their axes of rotation (Figure 4.15), referred to as hinge lines. The rotational motion of the respective grid fins, determined by a guidance command, are capable of controlling the vehicles attitude. Defining the orientation of each individual fin relative to the vehicles body requires two angles; the azimuthal and deflection angle. The azimuthal position is defined in the Y-Z plane and positive counterclockwise from the Y axis. The fin deflection angle is defined such that a positive deflection results in a counterclockwise roll of the vehicle, referenced about the respective hinge line. The azimuthal and deflection angle and their sign convention are illustrated in Figure 4.16 on the left and right respectively. Any other sign convention or coordinate system can be used to define the angles as long as the same convention is used consistently throughout the process.

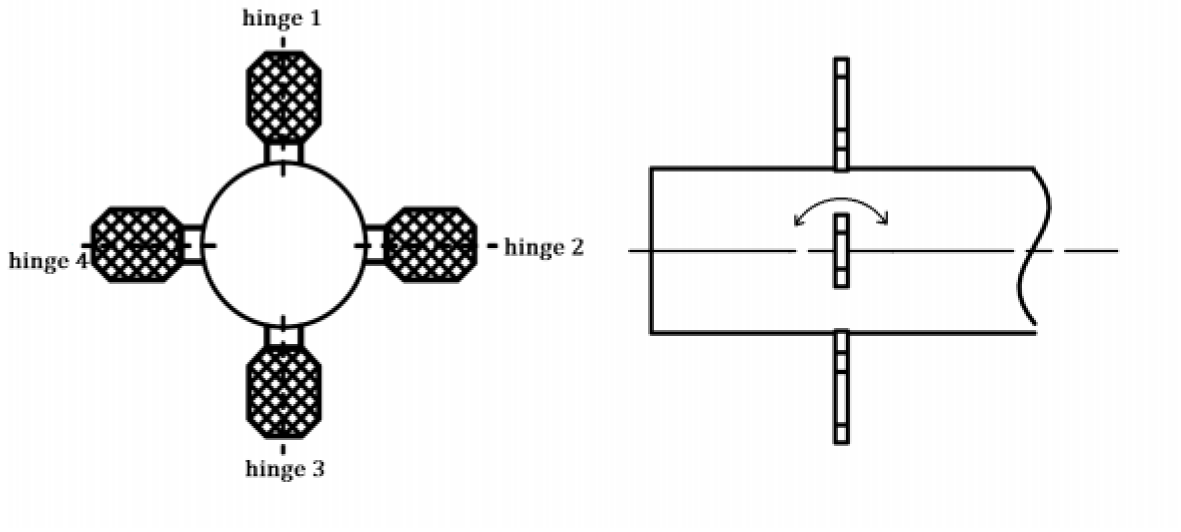


Figure 4.15: Schematic bottom and side view of 4 grid fins attached to a rocket

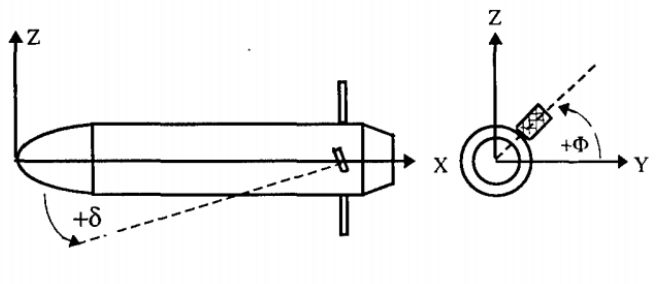


Figure 4.16: Grid fin sign convention [4]

Due to the complexity of the grid fin geometry, numerous geometric parameters have to be defined to characterise the grid fin accurately. Figure 4.17 provides an illustration of a grid fin attached to its vehicle, including its geometric parameters. The geometry of the grid structure is left out of the illustration, but has an influence on how the air will flow through the grid fin. Optimizing the grid structure and its effect on a vehicles state is a field of research on its own, leaning more towards aerodynamic and CFD analysis.

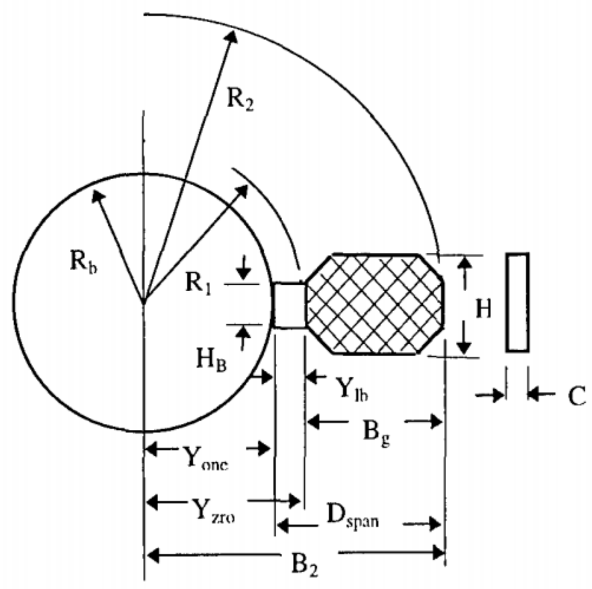


Figure 4.17: Grid fin geometry parameters [4]

To use the grid fins as control surfaces, the forces and moments acting on the grid fins have to be calculated. Doing so is non-trivial. The flow going through the grid fin will behave differently for the different velocity regimes, see Table 4.5. Consequently, different models have to be used to describe the flow throughout the trajectory. In the subsonic flow a vortex lattice proves itself well[37][35], whilst in the supersonic regime a modified version of the Evvards theory is used in [4][38][39]. Grid fin control has proven itself to be most efficient in the sub- and supersonic flow regimes. In the transonic regime the grid fin primarily suffers from a high drag force. In transonic flight the flow through the fin will choke, causing a detached bow shock (Figure 4.18 in front of the grid fin which makes it harder to control[34]. For a grid fin with a non-zero angle of attack the upwash effect is defined by [4][39]. Due to the lack of available data on aerodynamic grid fin behaviour and the complex modeling of such aerodynamic forces, grid fins are not included in this study and requires a 6-DOF approach.

Mach number	**Velocity regime**	**Assumed Flow Structure**
M <0.8	Subsonic	Local Flow structure completely subsonic. Compressible subsonic formulation is required.
0.8 <M <1.0	Transonic	Flow through Grid Structure is Typically Choked Requiring Application of Coefficient Correction Factors
1.0 <M <1.4	Transonic	Bow shock in front of grid fin. Flow is subsonic behind the bow shock and a compressible subsonic solution is required.
1.4 <M <1.9	Supersonic	Grid fin swallows shock. Each element acts as a thin wing with an attached leading shock, producing a reflected shcok dominated flow region.
1.9 <M 3.5	Supersonic	Leading edge shock on each fin element does not impinge on adjacent elements and internal flow is primarily supersonic with minimal reflected shock effects.
M >3.5	Supersonic	Strong leading edge shock transitioning to hypersonic flow.

Table 4.5: Velocity regimes and their assumed flow structure [4]

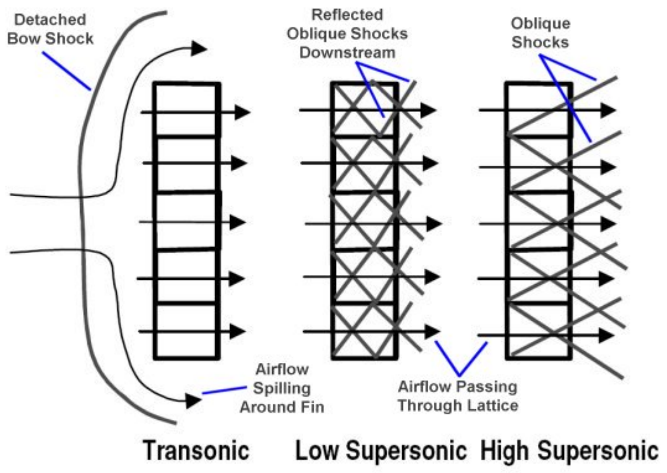

Detached Bow Shock
Reflected Oblique Shocks Downstream
Oblique Shocks
Airflow Spilling Around Fin
Airflow Passing Through Lattice
Transonic
Low Supersonic
High Supersonic

Figure 4.18: Tran- and supersonic flow through a grid fin[5]

5 | Environment

An environment model defines a set of physical properties aiming to simulate the real environment a vehicle is subjected to. The atmospheric and gravity model are of particular importance for this study. A full list of supported physical properties that can be modeled is provided on Tudats website [40]. Finding a balance between simulating the environment as accurate as possible on the one side and reducing its complexity on the other side, is a delicate process. Its common practice to run a simulation multiple times featuring different environment models, ranging in complexity. If a 'simple' model suffices the requirements (in terms of a predefined error margin), there is no reason to use a more accurate method as it will only complexify the model.

5.1 Atmospheric Model

Accurately modeling atmospheric models has proven to be a difficult task. Partly because atmospheres can not considered to be static mediums, meaning the atmospheric conditions differ over time and per location, primarily caused by solar radiation [3]. To compute the aerodynamic loads on a vehicle, the atmospheric density, temperature and pressure have to be, among others, computed as a function of altitude. Doing so is non-trivial as no analytical expression exists between these variables, not without making (simplifying) assumptions. This section provides a selection of commonly used atmospheric models. A distinction is made between *standard* and *reference* models. Disregarding and respecting the aforementioned dynamical (time-dependent) effects on the atmosphere respectively. For performance comparison of different simulation set-ups, *standard* models are often used as its characteristics do not differ over time and every simulation is subjected to the same atmospheric conditions.

5.1.1 Exponential Model

The exponential model is a simplified atmosphere model used for analytical problem approaches. The simplifications include the assumption of the ideal gas law (Equation 5.1), constant temperature and molecular mass and hydrostatic equilibrium in the atmosphere (Equation 5.2). A derivation of the equations, presented in Mooij[3], leads to Equation 5.3. This set of equations is used to compute the air- density, pressure and temperature (constant),

$$p = \rho R T = \rho \frac{R^*}{M} T \tag{5.1}$$

$$\mathrm{d}p = -\rho g \mathrm{d}h \tag{5.2}$$

$$\frac{\rho}{\rho_0} = e^{-\beta h} = e^{-\frac{h}{H_s}} \tag{5.3}$$

with R^*/M the gas constant of air, h the altitude and H_s the scale height. Commonly used values for Earthly applications are $\rho_0 = 1225 \text{kg/m}^3$ and $H_s = 7050 - 7200m$, resulting in a constant temperature of $T_c = 240K$[3].

5.1.2 The United States Standard Atmosphere 1976

The US standard Atmosphere 1976 (US76) is a tabulated atmosphere model, independent of time, build from experimental data. It is a revised version of the earlier developed US67 model. Below 32km the model is identical to the International Civil Aviation Organization [3]. US76 provides data for altitudes up to 1000km at a latitude of 45° North [41]. The main difference between US67 and US76 is that the latter uses geopotential instead of geometric altitude. The relation between the two is presented as Equation 5.4, with h the geometric altitude, z the geopotential altitude, g_0 and g the gravity constant at sea level and h respectively and $R_0 = 6356.766$ km the (Polar) radius of the Earth [13].

$$g_0 \mathrm{d}z = g \mathrm{d}h \Rightarrow z = \int_0^h \frac{g}{g_0} \mathrm{d}h \approx \frac{R_0 h}{R_0 + h} \tag{5.4}$$

The US76 model separates the atmosphere in several layers where each layer is subjected to a specific set of equations. For heights up to 85km the temperature can be computed using Equation 5.5 and Equation 5.6.

$$T_M = T_{M_i} + L_{z_i}(z - z_i) \tag{5.5}$$

$$T = T_M \frac{M}{M_0} \tag{5.6}$$

T_M is the molecular scale temperature and can computed with the data provided in Table 5.1, depending on the current layer. T_{M_0} equals T_0 and has a value of 288.15K [13]. Up to altitudes of 86km the molecular mass is assumed to be constant, $M_0 = M = 28.964$kg/kmol and thus $T = T_M$.

subscript i	Geopotential heigth z_i [km]	Molecular scale temperature gradient L_{z_i} [K/km]	Form of function relating T to z
0	0	-6.5	Linear
1	11	0	Linear
2	20	+1	Linear
3	32	+2.8	Linear
4	47	0	Linear
5	51	-2.8	Linear
6	71	-2.0	Linear
7	84.852	-6.5	Linear

Table 5.1: Defined reference levels and gradients [13]

For altitudes between $86km \leq z < 120km$, the temperature profile can be expressed in terms of 3 successive functions, presented by Mooij[3]. For $86km \leq z < 91km$ an isothermal layer is defined with $T = 186.8673K = constant$. For $91km \leq z < 110km$ the temperature is computed using Equation 5.7.

$$T = T_c + A\sqrt{1 - \left(\frac{z - z_8}{b}\right)^2} \tag{5.7}$$

with constants $T_c = 263.1905K$, $A = -76.3232km$ and $b = 19.9429km$. $z_8 = 91km$.
Equation 5.8 represents the temperature at altitudes $110km \leq z \leq 120km$, with $T_9 = 240K$, $L_9 = 12K/km$ and $L_9 = 110km$.

$$T = T_9 + L_9(z - z_9) \tag{5.8}$$

Below altitudes of 86km two expressions exist to compute the respective pressure. Equation 5.9 for $L_{z_i} = 0$ and Equation 5.10 for $L_{z_i} \neq 0$ Density ρ is computed using the ideal gas law (Equation 5.1).

$$p = p_i \exp\left[\frac{-g_0 M_0 (z - z_i)}{R^* T_{M_i}}\right] \tag{5.9}$$

$$p = p_i \left[\frac{T_{M_i}}{T_{M_i} + L_{z_1}(z - z_i)}\right]^{K_i} \tag{5.10}$$

With $K_i = \frac{g_0 M_0}{R^* L_{z_i}}$ and R^* the universal gas constant. At sea-level, index $i = 0$, the pressure is defined as $p_0 = 101325N/m^2$. Figure 5.1 depicts the typical temperature profile as a function of altitude for the exponential and US76 atmosphere.

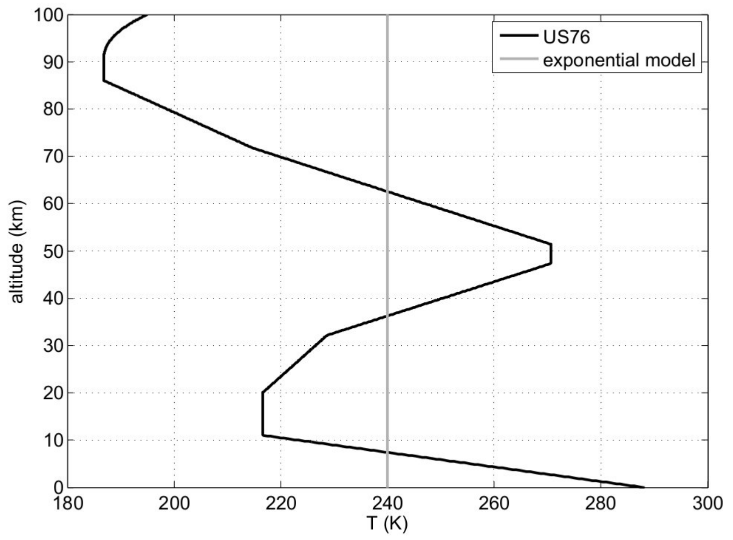


Figure 5.1: Temperature as a function of altitude for the exponential ($H_s = 7050m, \rho_0 = 1.225kg/m^3$) and US76 atmosphere model[3]

The first stage of the Elektron separates itself from the second stage at an altitude of approximately 78km[2]. For a deeper and more thorough understanding of the US76 atmosphere model the reader is referred to the US76 documentation[13]. Previous studies in the field of launcher optimization, including the work of Vandamme[41], Contant[1], van Kesteren[30] and Castellini[42] have implemented the US76 atmosphere model. For this study it is convenient to continue on a similar note and implement the US76 model. For the second stage however, reaching altitudes up to 500km, the US76 atmosphere model does not suffice.

5.1.3 NRLMSISE-00

For simulations at higher altitudes the US76 model becomes less accurate as it only provides average data. For applications that require high precision at high altitudes (decay of satellites, or space-debris predictions e.g.) the NRLMSISE-00 model presents itself as a good candidate. Contrary to the exponential and US76 model, NRLMSISE-00 is a *reference* atmosphere model which allows for the input of time-dependent parameters such as solar radiation, longitude, latitude etc. A full list of inputs and outputs of the NRLMISE-00 model is thoroughly discussed in the work of Picone et. al. [43]. For the simulation of the second stage the NRLMISE-00 model is implemented, which is readily available in TUDAT. It is expected that the choice of an atmosphere model for the second stage is of little influence because of the high altitudes.

5.2 Gravity Model

The four primarily forces acting on a re-entry vehicle are aerodynamic forces (lift, drag), thrust force and the gravitational force. For Earthly re-entry studies the only gravitational force to be considered is the one from Earth. Two ways of modeling the Earths gravitational force are discussed in this section: Central gravity model and Spherical Harmonics.

5.2.1 Central gravity

Computing the central gravity force on a vehicle is very straightforward and follows Equation 5.11, Newtons gravitational law. With $\mu_E = 3.986004418\text{x}10^{-14}\ \text{m}^3\text{s}^{-2}$ the standard gravitational parameter of the Earth and $\vec{r}$ the position vector of the vehicle w.r.t the center of the Earth. This method assumes the Earth to be a point mass.

$$\mathbf{F_G} = \frac{\mu_E}{r^3}\mathbf{r} \tag{5.11}$$

5.2.2 Spherical Harmonics

In reality Earth can not considered to be a perfect homogeneous sphere. Consequently, the gravitational field of the Earth is not uniform. To approximate the deviations from the homogeneous sphere, a gravity potential is introduced in Equation 5.12. U may be expressed as a summation of the central gravity field (point mass) and a set of spherical harmonics terms representing the non-symmetric mass distribution of the Earth[3]. U can be approximated by Equation 5.13.

$$\mathbf{F_G} = -m\nabla U \tag{5.12}$$

$$U(R,\delta,\tau) = \frac{\mu}{r} + \frac{\mu}{r}\sum_{l=0}^{\infty}\sum_{m=0}^{l}\left(\frac{R_e}{r}\right)^l P_{lm}(\cos\delta)\left(C_{lm}\cos(m\tau) + S_{lm}\sin(m\tau)\right) \tag{5.13}$$

with δ and τ the respective latitude and longitude, P_{lm} the Legendre polynomials and C_{lm} and S_{lm} spherical harmonic coefficients. If the density function of a body is known, C_{lm} and S_{lm} can be found by integrating the mass integrals over the bodies volume. However, C_{lm} and S_{lm} are usually determined empirically (because the body's density function is usually not known) using data provided by orbiting satellites. Coefficients that are independent of longitude ($m = 0$) are called zonal coefficients. Note that for $m = 0$, all S_{l0} terms become 0. The remaining zonal terms are commonly denoted as $J_l = -C_{l0}$ [44]. $J_2 = 1082.645\text{x}10^{-6}$ is by far the most dominant term, almost a tenfold of J_4 [45]. For ascent and descent trajectories the zonal terms are most important as the vehicle tends to move over, or parallel to the equator.

Both in case of a descent and ascent trajectory, the covered surface over the Earth will be relatively small, contrary to an orbiter for example. For ascent and descent trajectories it would make sense to only model the Spherical Harmonics that are relevant for the vehicles trajectory, saving computation time. However, in Tudat its currently only possible to create a Spherical Harmonics gravity model for the *whole* body.

The difference between a central gravity model and a model that includes the J_2 term is estimated around $5\mu g$ [46]. Both van Kesteren and Contant adopted (and validated) a central gravity model following this reasoning [30][1]. Throughout the course of this study the central gravity model will be used.

5.3 Simulation Software

In this section the available simulation/optimization software is briefly discussed.

5.4 Software packages

Trajectory simulation/optimization software is widely available on the internet. Choosing software that suits the application and the user best is beneficial for the simulation set-up. ASTOS, widely used by ESA, is a very advanced software package with extensive toolboxes and many possible applications [47]. GPOPS-II is another software package that serves a similar purpose [48]. The problem with most (advanced) software is the underlying commercial interests. Neither of the above mentioned software packages is licensed by the TU Delft. This makes it less attractive to use commercial software. The TU Delft Aerospace Engineering Faculty has however developed its own trajectory simulation/optimization software (TUDAT), only to be used for educational purposes. Being already familiar with this software it adds up to analyse the problem in TUDAT.

5.4.1 TUDAT

"The TU Delft Astrodynamics Toolbox (TUDAT) is a powerful set of C++ libraries that support astrodynamics and space research. These libraries are publicly available on Github[1] for anyone to use and contribute to.

TUDAT includes a wide variety of libraries, all the way from gravity models to numerical integrators and other mathematical tools. One of the key strengths within TUDAT is its ability to combine such libraries in a powerful simulator framework. Such framework can be used for a wide variety of purposes, ranging from the study of reentry dynamics, interplanetary missions, etc." [49]

TUDAT is under active development and widely used in the TU Delft Aerospace Engineering faculty by researchers, master students and is taught in some courses. This means knowledge about the software is easy accessible through students/researches who use or develop TUDAT.

5.4.2 PaGMO

Parallel Global Multiobjective framework for Optimzation, abbreviated PaGMO is an extensive C++ scientific library used to solve optimization problems. It allows a broad range of problems to be optimized, including single- and multi-objective problems. One of the main advantages of the PaGMO toolbox is the wide variety and easy interchangeable optimization algorithms it offers. Optimization algorithms can therefore be easily compared. PaGMO is developed within ESA by Biscani and Izzo [11] and is very well supported by the TUDAT environment. As a result the implementation of PaGMO in TUDAT is relatively easy.

[1] `https://github.com/Tudat`

6 | Trajectory Model

The trajectory model is divided into two phases, the ascent- and descent phase. The goal of simulating the ascent trajectory is to reach a desired target orbit without violating the mission constraints. In this study the semi-major axis, eccentricity and inclination are used to define the target orbit, as mentioned in section 4.2. At what point in time and where exactly on the orbit the payload is delivered is deemed irrelevant for this study. The argument of perigee and longitude of the ascending node are therefore left unconstrained. In this chapter the ascent- and descent trajectory models are explained in further detail as well as validated using flight data from the Falcon 9.

6.1 Design parameters

In this study two types of design parameters are distinguished, trajectory- and vehicle configuration parameters. The parameters form the user defined input of the model. In the process of choosing the design parameters a trade-off has to be made between the number of parameters and the freedom the user is given to define the model. More parameters generally means more design freedom at the cost of increased computational effort and vice versa.

The chosen design parameters for both the ascent and descent trajectory are given in Table 6.1. The equispaced altitude is defined as the altitude range between two consecutive pitch nodes, further elaborated on in subsection 6.4.1. There is no need to define h_{eqs} for the launch vehicle as its directly dependent on the MECO altitude through $h_{LV,eqs} = \text{h}_{MECO}/\text{nr. of nodes}$.

Table 6.1: Design parameters for ascent and descent trajectory

design parameters	
Ascent	**Descent**
MECO Altitude (h_{MECO}) [m]	Boostback time ($t_{boostback}$)
Equispaced altitude Second Stage ($h_{SS,eqs}$) [m]	Re-entry burn start altitude ($h_{reentry_{start}}$) [m]
Pitch angle at node i (θ_i) [deg]	Re-entry burn end altitude ($h_{reentry_{end}}$) [m]
	Re-entry burn start altitude ($h_{landing_{start}}$ [m]
	Angle of side-slip (β) [deg]
	Pitch-over angle (θ_c) [deg]
	Thrust throttle factor landing (ε_1) [-]
	Thrust throttle factor landing (ε_2) [-]

6.2 Ascent trajectory model

In Figure 6.1 two types of basic launch vehicle ascent trajectories are distinguished: *Direct Ascent* (DA) and *Hohmann Transfer Ascent* (HTA). Hohmann ascent trajectories are generally more fuel efficient than Direct Ascent trajectories. This is primarily caused by the steeper ascent of the DA, resulting in higher gravity losses. In case of flying a HTA, the launch vehicle is first launched to a circular *parking* orbit, typically with altitudes ranging from 180km to 220km, just outside the densest part of the atmosphere. The final target orbit is reached by means of two orbital manoeuvres. First changing from the parking orbit to an elliptical transfer orbit and then changing from the elliptical transfer orbit to the final target orbit.

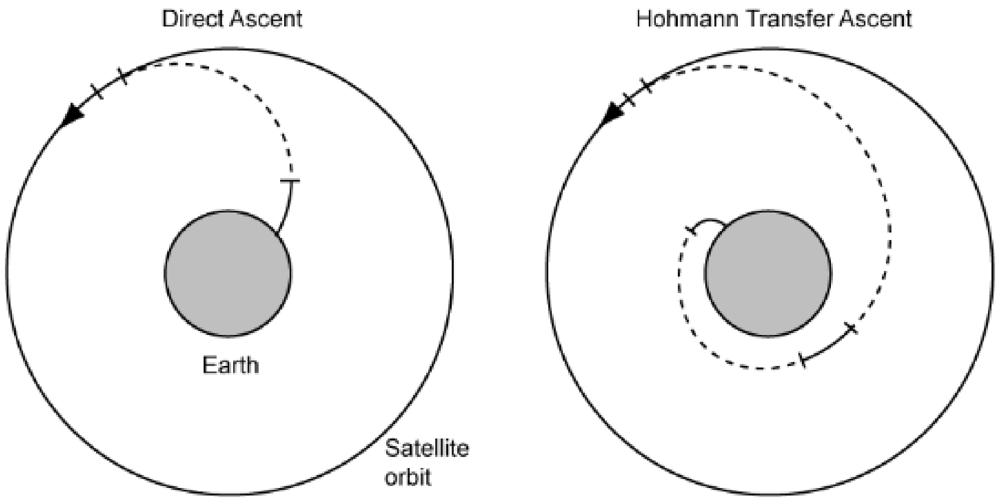


Figure 6.1: Typical Direct Ascent and Hohmann Transfer Ascent trajectory[6]

Hohmann transfers are usually used to reach higher orbits while for LEO orbits a DA trajectory is often favored. Several factors can play a role in deciding whether to use DA or HTA, other than fuel consumption. In terms of recovery of the first stage or a booster, the steeper ascent trajectory for DA results in a shorter horizontal distance traveled at MECO than HTA. This is the main reason a DA trajectory is used in this study, alongside the fact that the inherited pitch control law discussed in Section 6.4.1, is designed for the simulation of DA trajectories.

6.3 Descent trajectory model

When it comes to powered descent trajectories typically two methods can be distinguished: *Return To Launch Site* (RTLS) and *Down Range Landing* (DRL), displayed in Figure 6.2 and Figure 6.3 respectively. The refurbishment costs of DRL trajectories are higher than RTLS trajectories mainly because of a drone ship that has to be operated in order for the booster to land on. Contant has shown that for small satellite launchers the refurbishment costs of a DRL trajectory do not weigh up to the financial benefits of retrieving the booster[1]. For that reason only an RTLS method is investigated and analysed in this study.

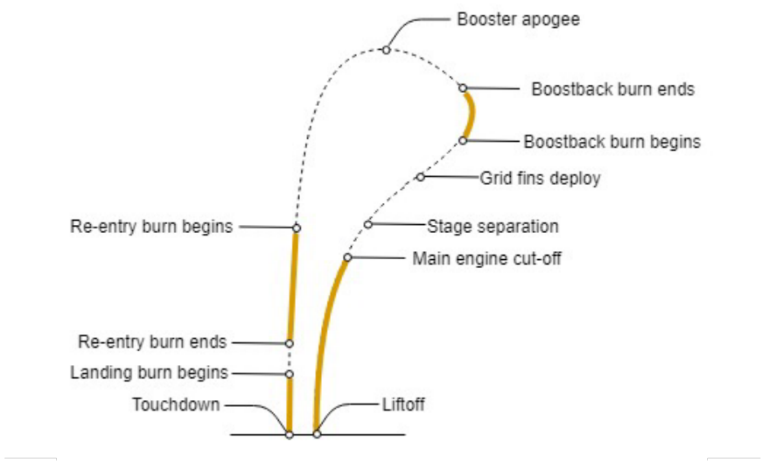


Figure 6.2: Return To Launch Site flight profile [1]

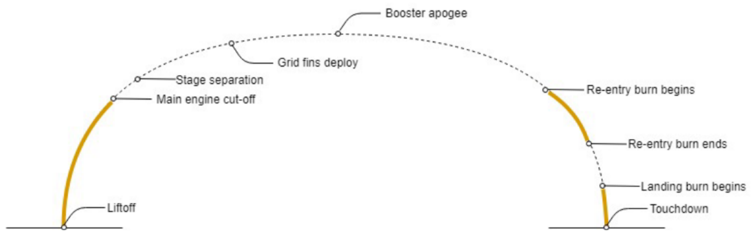


Figure 6.3: Down Range Landing flight profile [1]

RTLS trajectories can be divided in three phases: *boost-back burn*, *Re-entry burn* and a *landing burn*. In case of the boost-back burn a pitch-over manoeuvre is executed by gimbaling the thrust vector, such that the booster is directed back to the launch site. The re-entry burn is primarily performed in order to slow down the vehicle and mitigate for large dynamic pressure loads.

6.4 Guidance

The basic guidance problem can be defined as finding the trajectory that transfers a vehicle from its initial to its final state without violating any constraints. For atmospheric ascent or descent trajectories, aerodynamic and thrust guidance are often regarded.

In practical flight, guidance works as follows. The attitude (angles and angular velocities) of a vehicle can be controlled by applying a moment $\mathbf{u} = [M_x, M_y, M_z]$ around the c.o.m of the vehicle. This can be done using thrust vector control (TVC) or deflecting apparent control surfaces. Due to the 3-DOF nature of the problem at hand, attitude control (6-DOF) is left outside the scope of this study. However, a guidance algorithm to control the aerodynamic angles is still required as they largely influence the translational state of the vehicle through the lift and drag forces, as explained in Chapter 4.

Instead of computing the aerodynamic angles through the control vector **u**, the aerodynamic angles are simply imposed on the vehicle. Using this method it has to be assumed that the imposed aerodynamic angles can be achieved by the actuators of the vehicle. This assumption is backed by setting a maximum pitch rate $\dot{\theta}$ of 8 deg/s, derived from the maximum engine gimbal ranges found in literature[50][9][51][52]. As mentioned in Section 4.6, the angle of side-slip β is set to zero. The angle of attack α is computed using a (simple) pitch guidance law, illustrated in Figure 6.4 and Equation 6.1.

$$\alpha = \theta - \gamma \tag{6.1}$$

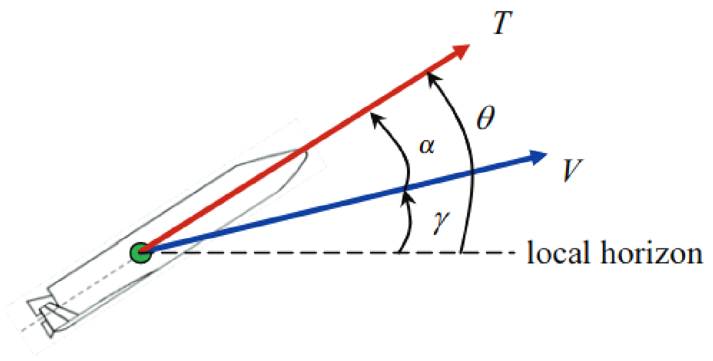


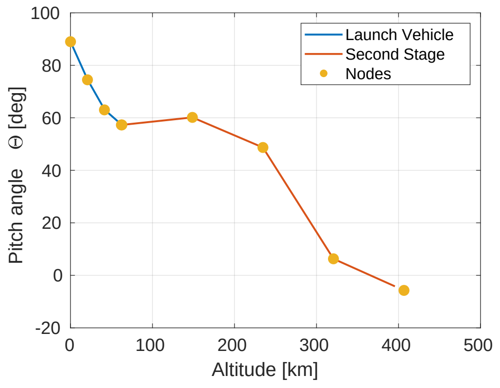
Figure 6.4: Thrust and velocity vectors of a rocket in the vertical plane[7]

6.4.1 Ascent

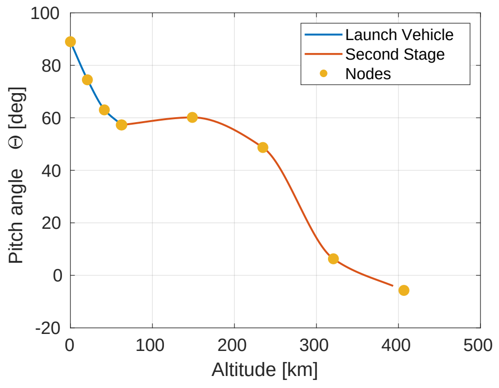
In order to guide the launch vehicle and the corresponding second stage to orbit, a uniform independent node control law is used to control the pitch angle throughout the trajectory. The pitch control law has been widely used in previous studies and is validated in the work of Van Kesteren[30]. The nodes were initially spaced linearly in time. However, for re-entry and ascent trajectories, when the flight time is left unconstrained, time is often not the best choice as an independent variable. Due to the unconstrained flight-time, spacing the nodes in time could lead to large parts of the trajectory being left virtually uncontrollable. In order to mitigate this problem, the nodes are spaced in altitude. The MECO altitude and the altitude of the target orbit are user defined, leading to a more accurate pitch controlled flight profile. The altitude with which the nodes are spaced differ for the launch vehicle and second stage. Both trajectories are assigned their own pitch profiles, leaving the second stage with less frequent spacing.

$$\theta(h) = \begin{cases} \theta_{LV}(h), & 0 \leq h < h_{MECO} \\ \theta_{SS}(h), & h_{MECO} \leq h < h_{final} \end{cases} \tag{6.2}$$

Within the Tudat environment several interpolation techniques are available, from which linear interpolation is the most straightforward. However, to ensure a continuous and more realistic pitch profile, Hermite Spline interpolation is implemented. Figure 6.5 shows two interpolated pitch profiles for a Falcon 9 launch trajectory. For a more detailed explanation of several interpolation techniques, including Hermite Spline interpolation, the reader is referred to the work of Moler[53].



(a) Linear



(b) Hermite Spline

Figure 6.5: Linear and Hermite Spline interpolation for a simulated Falcon 9 ascent trajectory.

6.4.2 Descent

The aerodynamic and thrust guidance are characterised by Equation 6.3 and Equation 6.4 respectively. For the boost-back burn and the re-entry burn the Falcon 9 fires 3 of its 9 Merlin 1D engines, contrary to the landing burn where one engine suffices. The thrust throttle factor ε is added to the design parameters for more design freedom and to dose the thrust force better. It is unknown whether the Rutherford engines from Rocket Lab have throttling capabilities. However, based on the similarities the Rutherford engine shows with the Merlin 1D engine, it is assumed that they can.

$$\theta(t), \theta(h) = \begin{cases} \theta_{LV}(h), & 0 \leq h < h_{MECO} \\ \theta_{MECO+coast} + \dot{\theta} \cdot t, & t_{MECO+coast} \leq t \quad \& \quad \theta(t) < \theta_c \\ \theta_c, & t \leq t_{boostback} \\ \theta_c - \dot{\theta} \cdot t, & \frac{1}{2}\pi \leq \theta(t) < \theta_c \\ \frac{1}{2}\pi, & h \geq 0 \end{cases} \tag{6.3}$$

$$T(t), T(h) = \begin{cases} 3 \cdot T_{1,eng}, & t_{MECO+coast} \leq t < t_{boostback} \\ 3 \cdot T_{1,eng} \cdot \varepsilon_1, & h_{reentry_{end}} \leq h < h_{reentry_{start}} \\ 0, & h_{landing_{start}} \leq h < h_{reentry_{end}} \\ T_{1,eng} \cdot \varepsilon_2, & 0 \leq h < h_{landing_{start}} \end{cases} \tag{6.4}$$

In-Plane Pitch Over manoeuvre

In this study an *In-Plane Pitch Over* manoeuvre is performed by commanding a pitch-over angle (θ_c) on the vehicle just after stage separation. Figure 6.4 displays the first stage right before the pitch-over manoeuvre. Initiating the manoeuvre, the pitch angle (θ) is quickly increased to reach θ_c, typically ranged between 160 - 190 deg. The manoeuvre is performed primarily by the Cold Gas Nitrogen thruster mounted on top of the first stage. This is an efficient way to 'flip' the first stage because of the high leverage w.r.t the c.o.m. Figure 6.6 shows the influence of θ_c on the downrange distance.

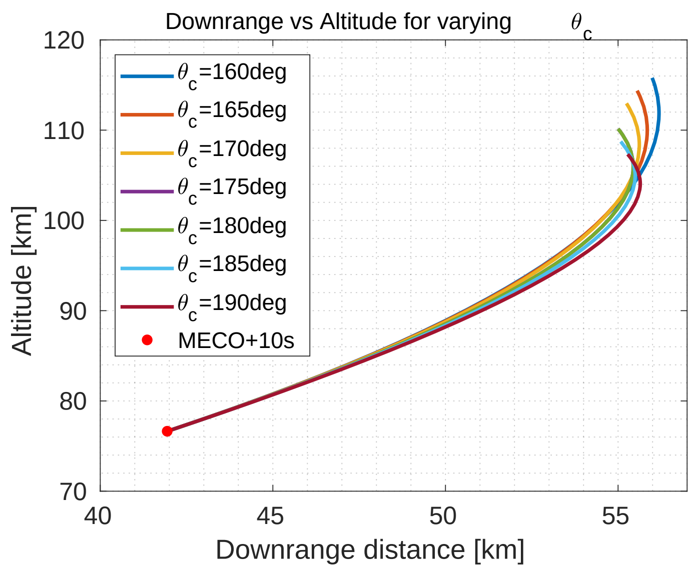


Figure 6.6: Downrange distance for several values of θ_c [1]

It is not possible for the pitch angle θ to reach θ_c instantaneously. It is found that for assuming an instant pitch-over manoeuvre the burn-back time is decreased by around 5 seconds. Multiplying by the the mass-flow ($\dot{m}$) of three Electrons Rutherford engine, an estimated 100kg of fewer fuel is needed for the RTLS trajectory. That is 10% of the initial fuel mass of the first stage at MECO. To ensure a more realistic flight path, in this study a constant pitch rate ($\dot{\theta}$) is used to gradually reach the value of θ_c.

Higher pitch rates are generally desirable. However, the pitch rate is restricted by the the maximum gimbal angle of the engine and the actuator responsiveness. Additionally, propellant sloshing and potential engine starvation issues can also play a role but falls outside the scope of this study. Typical values used for $\dot{\theta}$ in similar studies range from 5-20 deg/s[54][55].

After the boost-back burn the booster has to reorient itself for the re-entry phase (going from θ_c to $\theta = 90$ deg). Based on the aforementioned studies a constant pitch rate of 10 deg/s is used for Electrons in-plane pitch manoeuvre. Higher pitch rates for reorientation of the booster are not necessarily desired as lower pitch rates allow the booster to glide back longer. The constant pitch rate for reorienting the booster is set to 10 deg/s, a similar approach is used in the work of Simplico[56]. In Figure 6.7 θ, γ and α are plotted as a function of time according to the pitch law described in Equation 6.3

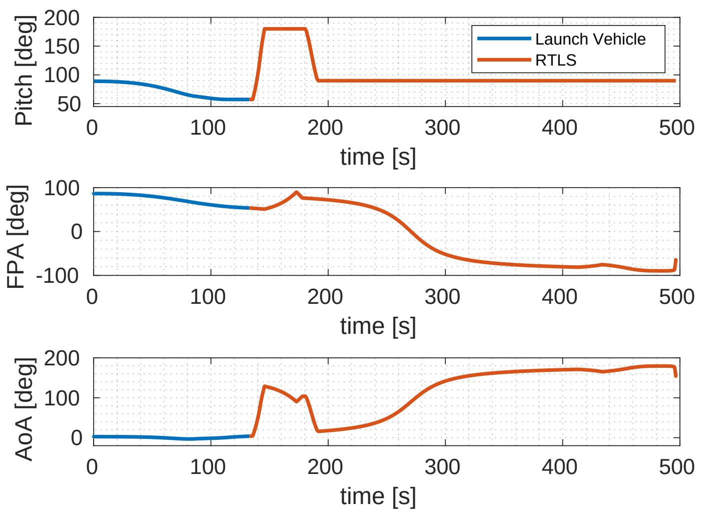


Figure 6.7: θ, α, γ guidance profile for the Launch Vehicle and RTLS trajectory.

Re-entry phase
After the first stage has reached a certain altitude a re-entry burn is initiated to slow the vehicle down and mitigate for high dynamic pressures. The re-entry burn is assisted by a set of grid-fins, set in place to slow the vehicle down and to effectively control the vehicles attitude. Modeling of the naturally complex grid-fin aerodynamics is outside the scope of this study. Implementing such model however, is expected to positively contribute to the model output as it helps slowing the vehicle down.

Landing phase
The landing burn is typically initiated at an altitude of 10-20 km and lasts until touch-down. At this stage the attitude of the vehicle is primarily controlled by the use of cold gas thrusters mounted on top of the rocket. Due to the relative low velocities during the landing phase, the use of grid fins is not as efficient anymore.

6.5 Validation

Most Flight data from previous SpaceX missions is classified and laying hands on such manufacturer verified data is almost impossible. For the validation of both the ascent and descent trajectory the available flight data from the CRS-10 Dragon Resupply Mission (2017) is used. This mission launched a payload from Kennedy Space Center to ISS, and successfully recovered the first stage flying a RTLS trajectory. The CRS-10 'real data' as displayed in Table 6.3 is retrieved from a mission overview published by SpaceX[15] and a live webcast of the launch[14], also made available by SpaceX.

The initial launch conditions for launches from Cape Canaveral and Launch Complex 1 are displayed in Table 6.2. The initial altitude is set to 100m instead of 0m to mitigate for potential singularities that might occur within the Tudat environment as a result of R_{Earth} being equal to

h_0. The initial heading angle χ_0 of the launch vehicle is simply computed using Equation 6.5, with i the inclination of the target orbit and δ_0 the initial latitude.

$$\cos i = \cos\delta_0 \sin\chi_0 \quad ; \quad 0 \leq i < \pi \tag{6.5}$$

Table 6.2: Initial launch conditions in spherical coordinates for Cape Canaveral (Falcon 9) and Launch Complex 1 (Electron)

	Cape Canaveral	**Launch Complex 1**
Latitude δ [deg]	28.608389	-39.261967
Longitude τ [deg]	-80.60433	177.86500
Altitude h [m]	100.0	100
Velocity V [m/s]	0.1	0.1
Flight path angle γ [deg]	89	89
Heading angle χ [deg]	50	-7

6.5.1 Ascent

It has to be stretched that the exact pitch profile of the CRS-10 mission is unknown. The model input parameters found in Table 6.3 have been manually tuned to validate the trajectory. The maximum dynamic pressure (Q_{max}) that the Falcon 9 endures is also unknown. However, Q_{max} generally lies between 30-40 kPa for launch trajectories. The output value for the eccentricity suggests that the final orbit is not as circular as the target orbit. This being the case its well worth to evaluate the pericentre- and apocentre altitude (h_p and h_a) of the final orbit. With the values for h_p and h_a as presented in Table 6.3, a minor circularization manoeuvre would suffice to enter the final 400 km circular orbit. Figure 6.8 shows the altitude and velocity profile of the validated ascent and descent trajectory of the CRS-10 mission. It is noted that V_0 and V_{final} of the first stage are not equal to zero, contrary to the data displayed in Table 6.3. This is due to the velocity being measured as the relative airspeed velocity instead of ground speed velocity. V_0 and V_{final} are therefore equal to the rotational velocity of the Earth, 406 m/s measured at Cape Canaveral.

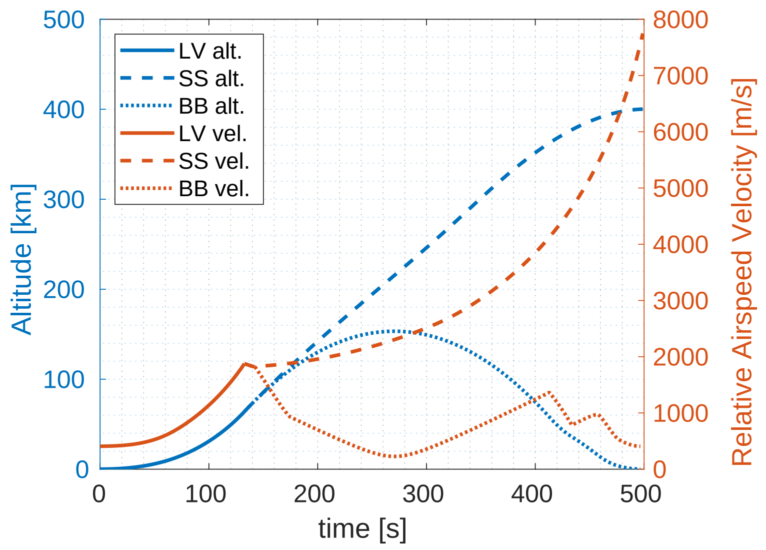


Figure 6.8: Altitude and Velocity profile for the ascent and RTLS trajectory of the Falcon 9 CRS-10 mission

Table 6.3: Ascent trajectory validation using flight data from the CRS-10 ISS resupply mission in 2017[14][15]

Variables	model output	real data
Launch Vehicle		
MECO time [s]	132.7	141.0
MECO velocity [m/s]	1876	1667
MECO altitude [m]	64097	65100
MECO Downrange distance [km]	32.46	-
Q_{max} [kPa]	32.25	-
Second stage		
final altitude [km]	400.1	-
Apocentre altitude [km]	677.1	409.9
pericentre altitude [km]	397.5	400.1
velocity [m/s]	7748	7520
semi-major axis [km]	6908	6783
eccentricity [-]	0.02024	0.000715
inclination [deg]	50.81	51.64
RTLS trajectory		
Landing altitude [m]	100	0
landing velocity [m/s]	2.81	0
Down range distance [m]	184	0
Boostback burn time [s]	41.9	39.0
Re-eentry burn time [s]	20.6	17.0
Landing burn time [s]	38.2	29.0

Model input parameters			
Ascent		**Descent**	
h_{MECO} [m]	64000	$t_{boostback}$	41.9
$h_{SS,eqs}$ [m]	89000	$h_{reentry_{start}}$ [m]	58000
$\theta_{1,\ldots 9}$ [deg]	89	$h_{reentry_{end}}$ [m]	36000
	63.03	$h_{landing_{start}}$ [m]	15100
	57.30	β [deg]	0
	57.30	θ_c [deg]	180
	51.56	ε_1 [-]	0.755
	51.56	ε_2 [-]	1
	5.73		
	-11.46		

6.5.2 Descent

Data on the RTLS trajectory of the Falcon 9 is even more difficult to find than the ascent data. Luckily the final state of the vehicle is known as it is supposed to land back at the launch site with a velocity equal to zero. From the live webcast the burn times of the different phases can be roughly extracted and are displayed in Table 6.3. The differences between the burn times can be explained by the fact that the exact altitudes at which the burns are initiated, alongside the throttle profiles of the burns, are unknown. As previously mentioned the simulation is terminated at an altitude of 100 to avoid singularities. The downrange distance of 184m is deemed accurate enough to validate the RTLS trajectory, taking into account that attitude control and precision landing techniques such as on-board guidance and navigation systems are left outside the scope of this study.

Figure 6.9) shows the downrange distance as a function of altitude. The shape of the flight profile closely resembles to flight profiles for RTLS trajectories found in literature[7][56]. Based on Figure 6.9 and the data displayed in Table 6.3 the RTLS model is sufficiently validated. The model can now also be used to model and optimize the Electron ascent and descent trajectory.

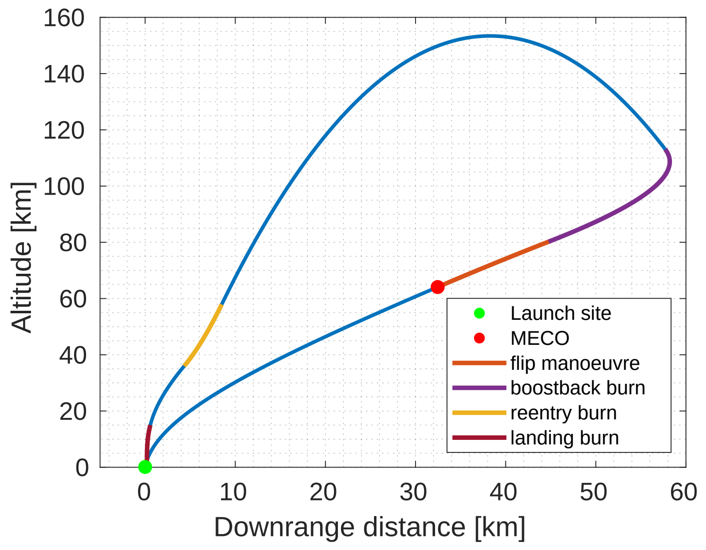


Figure 6.9: Downrange distance - altitude profile for the RTLS trajectory of the CRS-10 mission

6.6 constraints

Several types of trajectory constraints can be identified. *Path* constraints, which have to be satisfied at every time-point during the propagation. *Boundary conditions* define the constraints at t_0 and t_{end}. *Control* constraints define the limitations of the control variables. *State-triggered* constraints can be 'toggled on and off' if specific (state) conditions are met. The different types of constrained are explained in the following sections. The chosen values and implementation of the constraints are elaborated on in Chapter 7

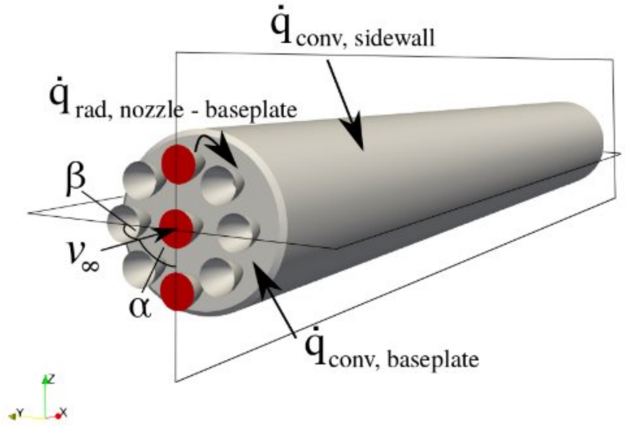
6.6.1 Path constraints

Heat flux

Heat flux is usually composed of two components, radiative and convective heat flux. For Earthly re-entry trajectories the radiative heat flux is very small compared to the convective heat flux[41]. Therefore, the convective heat flux is usually used to determine the maximum heat flux. The maximum heat flux occurs at the vehicles' stagnation point. An approximation for the heat flux in this point is given by the Chapman Equation (Equation 6.6[3]):

$$q_c = c^* \frac{1}{R_N^n} \left(\frac{\rho}{\rho_0}\right)^{1-n} \left(\frac{V}{V_c}\right)^m \tag{6.6}$$

with $c^*(\sqrt{m})$ a constant, $\rho_0 = 1.225\,\mathrm{kg/m^3}$ and ρ the air density at sea level and time t respectively, V and V_c the free stream and circular velocity respectively and R_N the nose radios of the vehicle. $m = 3$ and $n = 0.5$ are commonly used constants[3]. Equation 6.6 is is especially suited for vehicles with a blunt nose, the Space Shuttle or Apollo re-entry capsule e.g. The base-plate geometry (see Figure 6.10) of a descending rocket is more complex than a blunt nose and can therefore be less accurately described by Equation 6.6.



$\dot{q}_{conv, sidewall}$
$\dot{q}_{rad, nozzle - baseplate}$
β
V_∞
α
$\dot{q}_{conv, baseplate}$
Z
Y
X

Figure 6.10: Schematic base-plate geometry of a rockets first stage [8]

In case of a retro-burn during re-entry, a bow-shock will form around the exhaust plume of the engine, gaining similar characteristics as a blunt nose[8][57]. [8] evidently states that a supersonic retro-propulsion maneuver redistributes the heat loads from the base-plate to the entire vehicle surface (sidewall, Figure 6.10), therefore reducing local thermal loads. Nevertheless, accurately computing the thermal loads for a powered descent trajectory can become very complex. The 'nose radius' generated by the plume e.g. will not be constant along the trajectory because of a change in density, thrust magnitude/direction etc. The heat flux constraint for the the RTLS trajectory is left unconstrained throughout this study because of the added complexity to the model.

Dynamic pressure

The dynamic pressure experienced by a vehicle is related to the velocity of the flow around the vehicle, V, and the density of the fluid the vehicle is moving through (ρ). The dynamic pressure, q, is simply computed by Equation 6.7.

$$q = \frac{1}{2}\rho V^2 \tag{6.7}$$

The maximum dynamic pressure, q_{max}, varies over a wide range, dependent on the vehicle at hand and differs for ascent and descent trajectories. For descent trajectories few data dynamic pressure profiles is available. However, with the data that is available from previous SpaceX descent trajectories, the work of Sippel and Wilken surmise a q_{max} of 200 kPa[58][59]. This value is also used throughout this study. The Falcon 9 stays well below this boundary with values around 120 kPa. Electrons first stage however, encounters higher dynamic pressures, around the

200 kPa.

Acceleration (g-force)

The maximum allowable g-force for ascent trajectories is often determined by the fragility of the payload rather than the rocket itself. The axial and lateral acceleration for a payload during ascent typically lies between -2 and 8 and -2 and 2 respectively [60][2]. For a descent trajectory (without payload) however, there is no payload that constrains the maximum allowable acceleration. In the work of Snijders a maximum axial acceleration of 20g is assumed for the recovery of a small satellite launcher [21]. This value will also be used in this this study. It is found that the value of 20g is never reached during the simulations.

6.6.2 Boundary conditions

Boundary conditions are comprised of all the constraints that have to be satisfied only in the beginning or end of a trajectory. For precision landing this could be a set of coordinates that have to be reached. For vertical powered descents, the boundary conditions are comprised of the final position, velocity and attitude of the vehicle. The final position should be within range of the landing site, the final velocity should be 0m/s and the vehicle has to land vertically.

6.6.3 State triggered constraints

In optimal control the most commonly used constraints are *temporally-scheduled constraints*, meaning they are valid for a certain time-interval of the trajectory. *state-triggered constraints* are similar to the aforementioned, however, in this case the constraints are activated by an if-statement conditioned on the vehicles state (or other variable). *State triggered constraints* can for example be used to impose a *Line-of-Sight* constraint when the vehicle reaches a certain distance from the landing site, illustrated in Figure 6.11. Application of this constraint could be the autonomous landing of a powered descent vehicle as its sensors have to remain a clear overview of the landing site.

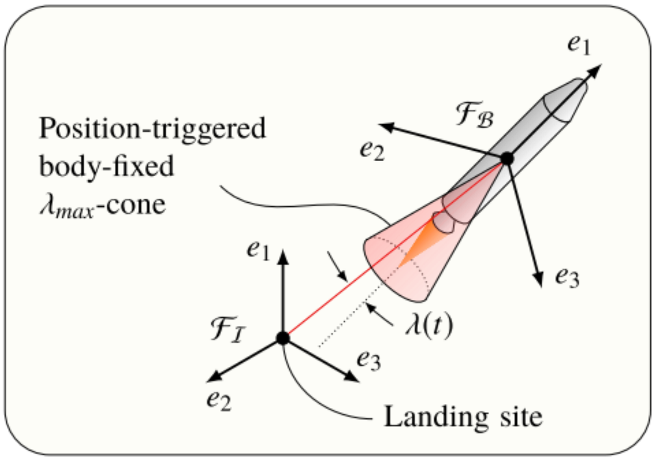

e_1
$\mathcal{F}_\mathcal{B}$
Position-triggered body-fixed λ_{max}-cone
e_2
e_1
e_3
$\mathcal{F}_I$
$\lambda(t)$
e_3
e_2
Landing site

Figure 6.11: Line of Sight constraint limiting the LoS angle λ when vehicles closes in on the landing site [9]

For a sophisticated mathematical formulation of the *state triggered constraints* the reader is referred to the work of Szmuk and Reynolds [9][51].

7 | Trajectory Optimization

In this chapter an overview of the general optimization problem is given and is applied to the trajectory optimization problem at hand. Objective formulation, different optimization techniques and constraint handling will be discussed.

7.1 General optimization problem

The general optimization problem can be defined as finding a set of parameters **x** that maximizes or minimizes an arbitrary objective function dependent on those parameters, $f(\mathbf{x})$. The objective function is often subjected to a set of constraints, limiting the space in which the optimal solution can be found. The general optimization problem can be stated as follows:

$$\min f(\mathbf{x}) \tag{7.1}$$

subjected to a set of constraints divided in equality (Equation 7.2) and inequality constraints (Equation 7.3) :

$$g_i(\mathbf{x}) = 0 \quad \text{for } i = 1,2,\dots,j \tag{7.2}$$

$$g_i(\mathbf{x}) \geq 0 \quad \text{for } i = j+1,\dots,m \tag{7.3}$$

where j and n are the number of equality and inequality constraints respectively. On a fundamental level maximization and minimization problems are the same. One can easily be written and converted to the other using Equation 7.4.

$$\max f(\mathbf{x}) = -\min[-f(\mathbf{x})] \tag{7.4}$$

7.2 Optimal control theory

Optimal control theory describes the process of finding the optimal control variables needed to minimize the objective function(s) in a system of ordinary differential equations. The state equation of such problem can be described as:

$$\mathbf{x} = \mathbf{f}[\mathbf{x}(t),\mathbf{u}(t),t] \tag{7.5}$$

where $\mathbf{x}(t)$ and $\mathbf{u}(t)$ are defined as the state- and control variables respectively. Both $\mathbf{x}(t)$ and $\mathbf{u}(t)$ might both be subjected to a set of path constraints along the trajectory (see Section 6.6.1). The initial and final conditions of the system are defined as a set of boundary conditions (see Section

6.6.2).

The optimal control problem can be mathematically expressed by Equation 7.6, where the cost function J is to be minimized by finding the optimal control $\mathbf{u}(t)$ and state variables $\mathbf{x}(t)$. Φ and $\mathscr{L}$ are both scalar functions.

$$J = \Phi\left[\mathbf{x}\left(t_0\right), t_0, \mathbf{x}\left(t_f\right), t_f\right] + \int_{t_0}^{t_f} \mathscr{L}[\mathbf{x}(t), \mathbf{u}(t), t] \mathrm{d}t \tag{7.6}$$

7.3 Multi-Objective design optimization

A difference can be made between single-objective (SO) and multi-objective (MO) optimization problems. In the latter case, multiple (coupled) objective functions are optimized simultaneously. A typical MO problem can be described as given in Equation 7.7, with N the number of objectives that are to be minimized.

$$\min \mathbf{f}(x) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_N(\mathbf{x})] \tag{7.7}$$

Within the field of trajectory optimization, optimizing for multiple objective functions is a popular approach. Think of a problem where both the payload mass and the travel time of the object have to be optimized for (manned mission to Mars e.g.).

The final solution of a multi-objective optimization problem will yield a Pareto front consisting of solutions that are all Pareto-optimal. That means that no objective function can be improved without having a negative impact on the other objective function(s). Dependent on how the objectives are weighed, a single solution can be drawn from the results. The work of Pagano and Pepermans are one of the many applications using this MO approach [61][22].

7.4 Objectives and constraint handling

Ascent

In Chapter 6 it is mentioned that the accuracy of the final orbit is measured by the semi-major axis, eccentricity and inclination of the final orbit. For the evaluation of the ascent trajectory fitness of the Electron a slightly different approach is used. The majority of missions launched from Launch Complex 1 are headed to a circular 500x500km (near) polar orbit Sun Synchronous Orbit (SSO). In Chapter 6 it was mentioned that for altitudes higher than 200-300km a direct ascent becomes less fuel efficient than a Hohmann transfer.

A more efficient way to reach the final orbit is to make use of an orbital coast period, schematically depicted in Figure 7.1. In this case the second stage is launched to the pericentre of a an elliptical *transfer orbit*. The pericentre altitude of the *transfer orbit* should be high enough for aerodynamic forces to be negligible and the apocentre altitude should be equal to the radius of the final target orbit. When the pericentre altitude of the *transfer orbit* is reached the engines of the second stage are cut off. Then follows the orbital coast period which lasts until the vehicle arrives at the apocentre of the *transfer orbit* (true anomaly +180 deg). At the apocentre the engine of the second stage is re-ignited to deliver the required ΔV for the circularization manoeuvre.

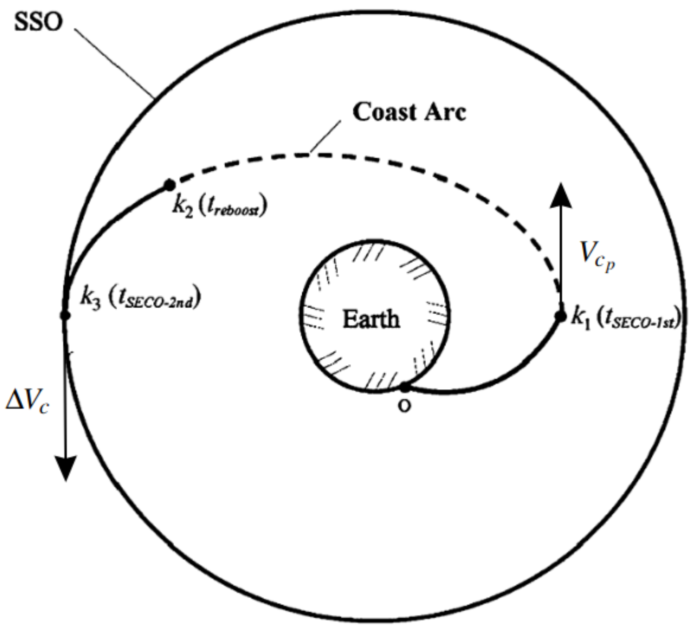


Figure 7.1: Ascent trajectory to SSO orbit using an orbital coast period[10]

The required ΔV_c for a circularization manoeuvre as depicted in Figure 7.1 is given by Equation 7.8[6]. With V_{c_p} and e the circular velocity at the pericentre of the *transfer orbit.* In addition to circularizing the final orbit, an extra ΔV manoeuvre might be needed to change orbital planes and reach the inclination of the target orbit. ΔV_i can be computed using Equation 7.9[6], where V_a and Δi represent the final velocity of the vehicle at the apocentre of the *transfer orbit* and the inclination difference between the orbital planes of the target orbit and final orbit respectively. ΔV_c and ΔV_i are lumped together into one objective function (Equation 7.10); minimizing the total ΔV.

$$\Delta V_c = V_{c_p} - V_{c_p}\sqrt{1-e} \quad ; \quad e = \frac{r_a - r_p}{r_a + r_p} \tag{7.8}$$

$$\Delta V_i = V_a 2 \sin\frac{1}{2}\Delta i \tag{7.9}$$

$$\Delta V_{tot} = \Delta V_c + \Delta V_i \tag{7.10}$$

Based on the Falcon 9 RTLS trajectory, a minimal of 10% of the launch vehicles initial fuel mass is required in order for the first stage to return to the launch site. Throughout this study the geometry, engine performance and mass distributions are tried to keep as close to the original vehicle as possible. Without altering any of the vehicles characteristics it makes sense to optimize for the launch vehicles leftover propellant at MECO. At the same time the second stage still has to be able to reach its target orbit. Hence the objective function can be written as Equation 7.11, with $m_{f,used_{LV}}$ the used fuel by the launch vehicle. For the second stage ΔV_{max} is computed at the pericentre of the transfer orbit (SECO). To account for the ΔV_{tot} manoeuvre and potential unforeseen other manoeuvres, 25kg (1.17% of total second stage fuel mass) of propellant is saved such that $m_0/m_{dry} > 0$ always holds and sub-consequently $\Delta V_{max} > 0$.

$$\min \mathbf{f}(x) = [\Delta V_{tot}(\mathbf{x}), m_{f,used_{LV}}(\mathbf{x})] \tag{7.11}$$

For the sake of clarity and a more intuitive understanding of the outcome of the optimization process, the objective functions are normalized according to the equations given below. The theoretical maximum ΔV that a vehicle is able to deliver can be computed using the Tsiolkovsky rocket equation, Equation 7.12. m_0 and m_{dry} represent the wet mass and dry mass of the vehicle respectively.

$$\Delta V_{\max} = I_{sp} g_0 \ln\left(\frac{m_0}{m_{dry}}\right) \tag{7.12}$$

$$\Delta \bar{V} = \Delta V / \Delta V_{\max} \tag{7.13}$$

$$\bar{m}_{f,\text{ used}} = m_{f,\text{ used}} / m_{f,\max} \tag{7.14}$$

The following constraints, previously discussed in Chapter 6, will be implemented in the ascent trajectory optimization model. The violation of the constraints is measured in a penalty function. The penalty function is then added to the objective function to worsen its result.

- Pitch angle rate: $\dot{\theta} \leq 10 deg/s \quad \forall \quad \dot{\theta}$
- The maximum acceleration acting on the vehicle: $a_{max} \leq 20g \quad \forall \quad a_{max}$
- Final altitude h_{final} and pericentre altitude h_p at SECO: $h_{final}, h_p \geq 150km$
- Apocentre altitude h_a of the final orbit: $480km \leq h_a \leq 600km$
- Maximum dynamic pressure: $Q_{max} \leq 200kpa$

The penalized objective function can be written as Equation 7.15, with $\mathscr{L}$ the penalized objective function, f the non-penalized objective function, λ_j the penalty value on the objective and g_i the respective constraint.

The total (accumulated) violation over the whole trajectory is calculated and added to the ΔV objective function. The process of finding a fitness function, or deciding what penalty functions to use is non-trivial. The best method does not exist and defining such functions often requires trial and error work. One could for example decide to square the penalty function in order to penalize higher violations more quickly. However, in this study the penalty is linearly computed according to Equation 7.15.

$$\mathscr{L} = f + \Sigma \lambda_j g_j \tag{7.15}$$

The violation of the pitch angle rate constraint is integrated over time, capturing both the severity and the duration of the violation. In this way a small but continuous violation could yield the same integrated violation value as a one-off large violation. The final apocentre altitude should ideally be exactly 500km. However, this condition is impossible for the optimizer to reach exactly. h_a is therefore allowed to a maximum upper and lower bound, Equation 7.16. The closer it lies to the target apocentre, the smaller the penalty.

$$Penalty = \begin{cases} \frac{|h_a - h_{a,target}|}{h_{a,target}} \cdot f, & 480km \leq h_a \leq 600km \\ 10^{10}, & 480km > h_a > 600km \end{cases} \tag{7.16}$$

Some constraints are not allowed to be violated under any circumstances, so called 'hard constraints'. In the case that such constraint is violated a *death penalty* is assigned to the objective function (Equation 7.17), instantly marking the solution as very bad. The remaining constraints (h_{final}, h_p and Q_{max}) are all assigned a *death penalty* when violated.

$$Penalty = 10^{10} \tag{7.17}$$

Descent

The objective functions and constraint handling for the descent optimization problem are of a similar nature. Since the goal of this research is to find the fuel optimal RTLS trajectory, the main objective function is comprised of minimizing the used fuel. The accuracy of the RTLS trajectory is measured w.r.t the final boundary conditions, Equation 7.18. x_i and $x_{TARGET,i}$ denote the final value and target value of the respective variables. tol_i equals the allowed tolerance between x_i and $x_{TARGET,i}$. Hence, $E_i \leq 1 \quad \forall \quad E_i$ is within the user defined set of tolerances. An extra variable ($tol_{pen,i}$) is introduced to define the cases where $E_i > 1$, described in Equation 7.18. In this way, the solution is not immediately discarded when the tolerance value is violated. This gives the user more freedom of interpretation, as a slightly violated solution might still be of use. Furthermore, in case the tolerance is set too tight, this can lead to too many individuals of a population being penalized and therefore not being able to generate a good enough off-spring. As a result, the solution might not converge.

$$E_i = \begin{cases} \frac{|x_i - x_{target,i}|}{\text{tol}_i}, & (x_i - x_{target,i}) \leq \text{tol}_{pen,i} \\ 10^{10}, & (x_i - x_{target,i}) > \text{tol}_{pen,i} \end{cases} \tag{7.18}$$

The path constraints of the trajectory are handled in accordance to Equation 7.19, with y_j the respective path constraint and $C_{pen,j}$ its boundary value. The fitness of a solution is computed by taking the Root Sum Square (RSS) for all values of E_i, Equation 7.20. n denotes the number of boundary constraints taken into account. In case of the RTLS trajectory $n = 3$, representing the downrange distance (D_{final}), final velocity (v_{final}) and the final altitude (h_{final}). The path constraints are simply included by adding them to the RSS.

$$Penalty_{path,j} = \begin{cases} 0, & y_j < C_{pen,j} \\ 10^{10}, & y_i > C_{pen,j} \end{cases} \tag{7.19}$$

$$RSS = \sqrt{\sum_{i=1}^{n} E_i^2 + \sum_{j=1}^{m} Penalty_{path,j}} \tag{7.20}$$

Dependent on the value of n, something can be said about the RSS values. At first sight, one could argue that for all $RSS \leq \sqrt{n}$ all boundary conditions are met since $E = 1$ lies exactly on the tolerance boundary. However, $\sum E_i = n$ does not necessarily mean that the individual values of E_i are all equal to 1, which is only the case when all E_i preform equally well. The following can be stated and should be taken into account interpreting the results:

- For $RSS > \sqrt{n}$, at least one of the boundary conditions is not met.
- For $1 < RSS \leq \sqrt{n}$, at least one boundary condition is met, and potentially all.
- For $RSS \leq 1$, all boundary boundary conditions are met for sure.

The RTLS trajectory constraints are given below. As aforementioned, the final velocity, downrange distance and altitude constraints are lumped together into an objective function that is to be minimized alongside the used fuel of the first stage. $5m/s$ is thought of as a reasonable limit for the final velocity. During a real flight the velocity can be more accurately controlled by an onboard control system and brought to zero. The same holds for the downrange landing distance of the first stage. It is assumed that distances of $\approx 2kmandlowercaneasilybeadjustedforbyanonboardnavigationsystem.Toac$
2000.

- The maximum acceleration acting on the vehicle: $a_{max} \leq 20g \quad \forall \quad a_{max}$
- Maximum dynamic pressure: $Q_{max} \leq 200kpa$
- Final groundspeed velocity: $V_{g,final} \leq 5m/s; \qquad tol_{pen,vel} = 15m/s$
- Final altitude: $h_{final} \leq 100m; \qquad tol_{pen,h} = 200m$
- Final distance to the landing site: $D_{final} \leq 1km; \qquad tol_{pen,D} = 5km$
- Final pitch angle: $85deg \leq \theta_{final} \leq 95deg$

7.5 Design Space Exploration

The solution obtained from an optimization process is highly dependent on the search space the optimizer is initiated with. At the one hand the search space has to be large enough for the optimizer to find the global optimum instead of a local optimum. On the other hand a search space that is too large can lead to longer computation times and non-convergence of the solution. The search space can be narrowed by conducting a Design Space Exploration (DSE) prior to the optimization process.

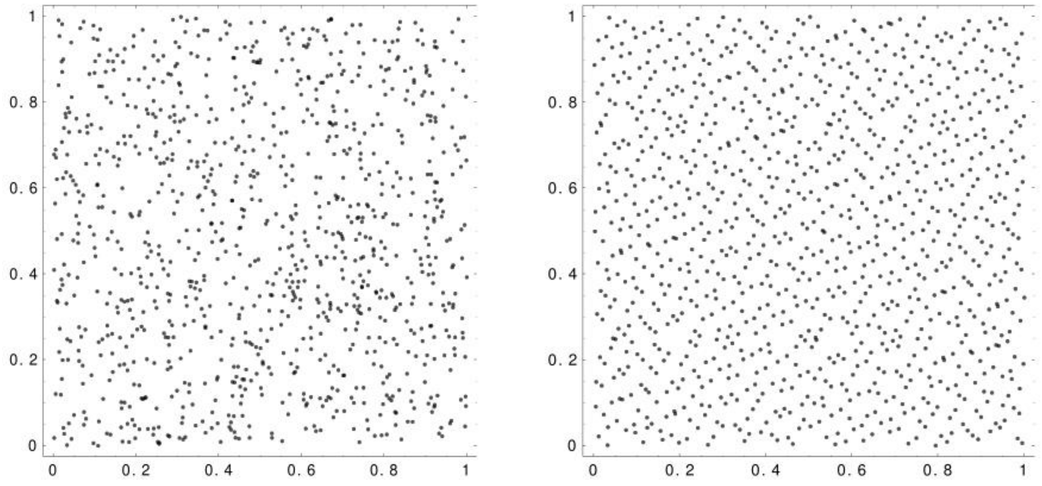


Figure 7.2: *Pseudorandom* Monte Carlo (left) and *Quasirandom* sobol (right) simulation for 1024 points in a 2D plane.

First the initial search space has to defined. Then a random number generator is used to compute a large set of *pseudorandom* or *quasirandom* numbers representing the design variables used in the simulation. The difference between a set of *quasirandom* and *pseudorandom* is best explained by figure Figure 7.2. *Quasirandom* numbers are more uniformly distributed as the algorithm generates each successive number as far away as possible from previously generated numbers in the set. These kind of test are often too uniform to pass randomness tests. However, in case of the DSE, the degree of randomness is irrelevant and a more uniformly distributed set of design variables is desired to give a better coverage of the search space.

In this study a *quasirandom* sobol sequence is used for the DSE of both the ascent and descent optimization problem. For the generation of the numbers the settings in Table 7.1 are used. The reader is referred to the Mathworks website[1] for more information on the sobol method and how it is implemented.

Table 7.1: Sobol sequence settings for reproducibility

	Samples	Dimensions	skip	leap	Scramble method	Type
Ascent	10000	nr. of design variables	800	0	0x0 structure	sobol
Descent	300000	nr. of design variables	800	0	0x0 structure	sobol

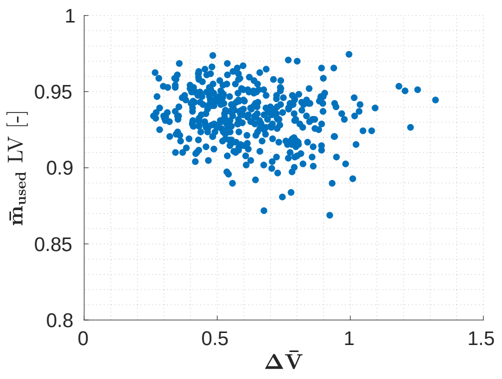
In Table 7.2 the lower and upper values of the design variables used for the DSE are displayed. In Figure 7.3 only the feasible solutions are plotted, in accordance to the constraints and objectives formulated in section 7.4. For the ascent problem 10 000 sample points resulted in a good coverage (Figure 7.3a)of the design space while for the descent problem only minor coverage is achieved for 300 000 sample points (Figure 7.3b). Two possible reasons can be given to explain this behaviour. The initial search space of the descent problem could be too large and therefore not enough feasible design variables are generated. However, for 300 000 sample points this is deemed unlikely. The

[1]https://www.mathworks.com/help/stats/sobolset.html

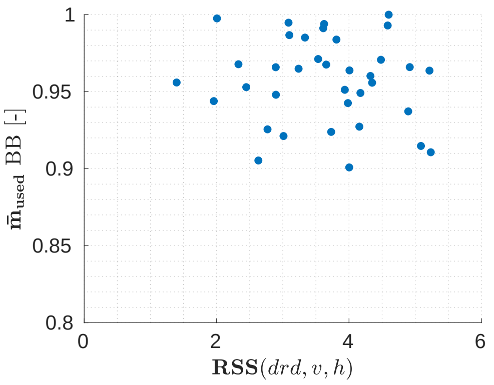
second, and more likely reason, is the high sensitivity of one or more design variables on the output. A sensitivity analysis on the design variables will be further discussed in Chapter 9

Table 7.2: Initial search space used for the sobol Design Space Exploration

Ascent			Descent		
	Lower bound	**Upper bound**		**Lower bound**	**Upper bound**
MECO alt. [m]	60000	85000	$t_{boostback}$ [s]	45	51
Equispaced altitude [m]	20000	50000	$h_{reentry_{start}}$ [m]	50000	65000
θ_1 [rad]	$\frac{1}{2}\pi$	$\frac{1}{2}\pi$	$h_{reentry_{end}}$ [m]	30000	48000
θ_2 [rad]	0.8	1.3	$h_{landing_{start}}$ [m]	10000	20000
θ_3 [rad]	0.5	1.2	yaw angle [rad]	-0.1	0.1
θ_4 [rad]	0.2	0.8	θ_c [deg]	160	190
θ_5 [rad]	0	0.3	ε_1 [-]	0	1
θ_6 [rad]	-0.2	0.1	ε_2 [-]	0	1



(a) Ascent DSE for 10 000 sample points



(b) Descent DSE for 300 000 sample points

Figure 7.3: Design Space Exploration for ascent and descent problems

A logical next step is to plot the DSE results as a function of the design variables to evaluate their contribution to the objective functions. From Figure 7.3 it can be observed that for higher MECO altitudes a larger amount of fuel is used and vice versa. This is simply because the launch vehicle has to thrust for a longer period of time for higher MECO altitudes. It can also be noticed that lower values for θ_2 are favored. θ_2 represents the second pitch node and consequently the range of altitude where the highest aerodynamic and gravity losses occur. The lower bound of θ_2 is restricted to ensure a vertical lift-off. At last, for the final pitch node (θ_6) negative pitch angles are not desired. The pitch angle is defined as the angle between the longitudinal axis of the body and the local horizon. A pitch angle equal to 0 would therefore correspond to a circular orbit (assuming that the flight path angle is also 0). However, because a coast-arc is used to transfer the second stage to its final circular orbit, the final pitch node should be higher than 0 as the transfer orbit is elliptical of shape ($0 < e < 1$).

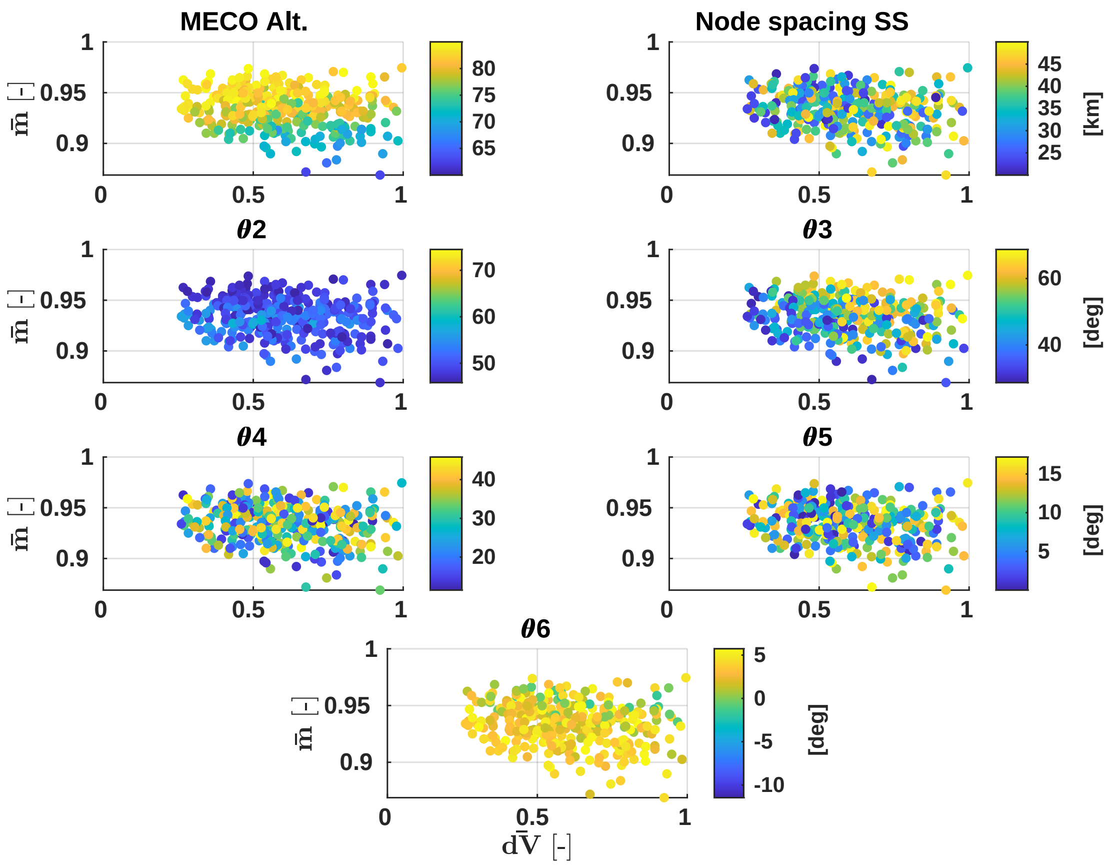


Figure 7.4: DSE for the ascent optimization problem as a function of the design variables.

A similar analysis for the descent DSE is performed. In Figure 7.5 the DSE of the descent optimization problem is plotted as a function of the design variables. It can be observed that all feasible solutions feature a boostback time of around 43s. The range for $t_{boostback}$ can therefore be narrowed down drastically. The same reasoning can be followed for the throttling factors. The insights obtained from Figure 7.4 and Figure 7.5 will be used as a starting point for the actual optimizations discussed in Chapter 8.

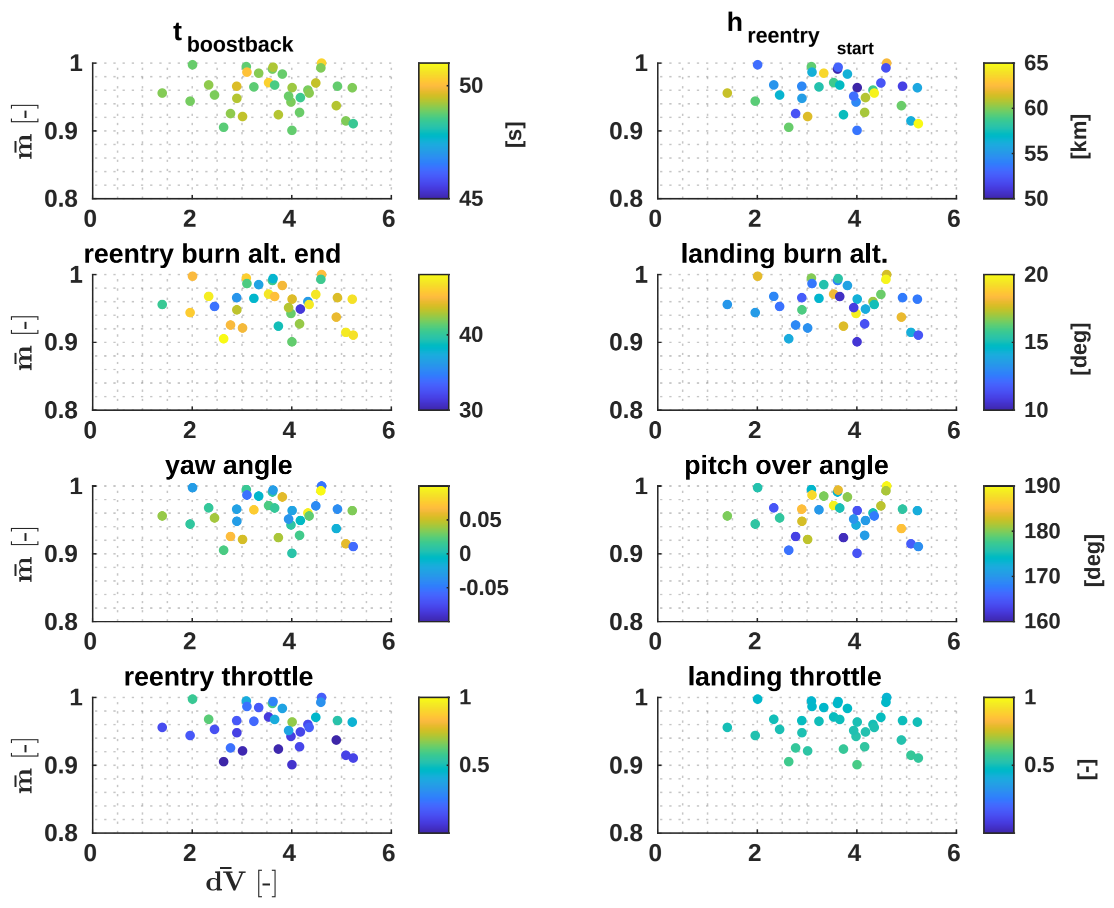


Figure 7.5: DSE for the descent optimization problem as a function of the design variables.

7.6 Global Optimization Methods

Genetic Algorithm

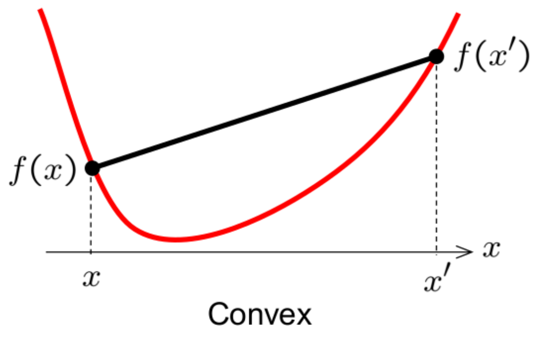
Genetic Algorithm (GA) is a global optimization method based on biological inspired processes, part of the larger Evolutionary Algorithm class. Examples include Particle Swarm Optimization (PSO) and Differential Evolution (DE) techniques. PaGMO (Parallel Global Multi-objective framework for Optimization, see Section 5.4 for more information) is a software package developed by ESA and features a wide range of optimization algorithms. A full list of PaGMO supported algorithms is provided on the PaGMO website[2]

Similar to other global optimization techniques, the main advantage of Genetic Algorithms is the large search space. This makes the algorithm insensitive to the initial guess of the solution and therefore very well suited for complex global optimization purposes. The Genetic Algorithm can be divided in 5 sequential phases;

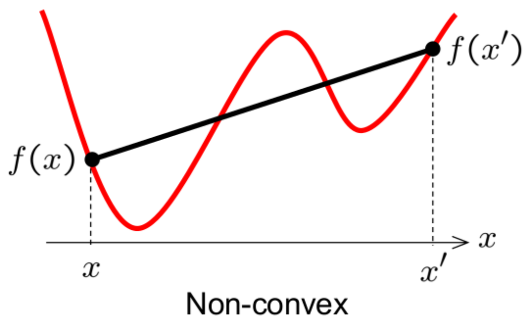
[2]https://esa.github.io/pagmo2/docs/cpp/cpp_docs.html#implemented-algorithms

- *initial population*: The initial population exists of set of individuals, each representing a different solution to the optimization problem.
- *fitness function*: Next a fitness function is defined, this function determines how 'good' a solution is.
- *selection*: The best individuals are selected based on the fitness evaluation.
- *crossover*: Individuals randomly exchange parts of their solution creating a set of 'offspring' solutions.
- *mutation*: Within the new offspring, parts of the solution change order. Mutation often happens with a low probability and is applied to prevent premature convergence of the solution.

The algorithm finishes when a termination condition set by the user is met. This is ideally when a set of solutions meets the defined requirements, but could also be a limit on the maximum number of generations or a when the solution stops converging. The disadvantages of this method include relatively large computation times and more importantly, the GA can only approach the optimum but is incapable of finding the exact optimum. To find the exact optimum, or check the convergence of the solution, a local (gradient based) optimizer could be used to find the exact optimum. The difference between global and local optimization is best explained by Figure 7.6. For convex functions it can be said that the local optimum is equal tot the global optimum. For non-convex functions this is not the case. For the non-convex function on the right in Figure 7.6 a local optimizer might find the local minimum on the right while the global optimum lies further left. Local optimizers are very sensitive to the initial guess (search space) of the solution. The final solution of the global optimization problem can serve as the initial guess for the local optimizer. As such, both techniques are often used together. Many different local optimization methods exist, *direct shooting*, *gradient based*, *collocation* etc. In this study a local optimization will be performed using a simple Monte Carlo simulation.



(a)



(b)

Figure 7.6: Example of a convex and non-convex function

7.7 Choice of optimizer

The majority of literature focused on the ascent trajectory optimization problem makes use of one of the following optimizers[62][63][64]. Particle Swarm Optimizer (PSO), non-linear programming

(NLP) solvers and Genetic Algorithms (GA). For the sake of convenience the optimizer choice is restricted to the optimizers available in PaGMO. NLP solvers are therefore not taken into account.

For descent and reentry optimization problems a similar set of often used optimizers is found. The work of Rahimi successfully uses PSO to optimize a spacecraft re-entry trajectory [65] and a study conducted by De Ridder uses a GA to optimize the flight range of a re-entry vehicle[66]. In the work of Ghosh, PSO is used to compute near-optimal solutions to multiple nontrivial trajectory optimization problems. These include, amongst others, minimum fuel use trajectories [67].

To back the choice of optimizer, the performance of four multi-objective optimizers (Table 7.3) is examined for the ascent and RTLS trajectories. The Non-dominated Sorting Particle Swarm Optimization (NSPSO) and Improved Harmony Search (IHS) were quickly discarded because of their poor performance. The results for MOEAD and NSGA-II are discussed later in this chapter.

NSPSO		**MOEA/D**		**IHS**		**NSGA-II**	
parameter	value	parameter	value	parameter	value	cr	0.95
w_{min}	0.95	weight generation	grid	phmcr	0.85	eta_c	10
w_{max}	10	decomposition	"tchebycheff"	$ppar_{min}$	0.35	mr	0.01
c_1	0.01	neighbours	20u	$ppar_{max}$	0.99	eta_m	50
c_2	0.5	CR	1.0	bw_{min}	1E-5	seed	123
χ	0.5	F	0.5	bw_{max}	1		
v_{coeff}	0.5	eta_m	20	seed	123		
diversity mechanism	"crowding distance"	realb	0.9				
leader selection range	2u	limit	2u				
seed	123	preserve diversity	true				
		seed	123				

Table 7.3: Default settings of multi-objective optimizers. Detailed explanation of the parameters can be found on the ESA website[3]

Throughout this study the default optimization parameters as given in Table 7.3 are used. For a more in depth optimizer analysis the parameters can be varied to customize and improve the optimizer performance. In this section only the population size is varied and carried out for 300 generations. Tuning the optimizer parameters, and in this case the population size, is non-trivial and highly dependent on the problem at hand. Although larger population sizes can generate more off-spring, this does not necessarily lead to a better optimizer performance. In addition, unnecessarily large population sizes can lead to increased computational effort. To put this in perspective, for the ascent optimization problem initiated with a population size of 100 and being evolved for 200 generations, the optimization process may take over 24h to finish, on a single CPU thread.

[3]`https://esa.github.io/pagmo2/docs/cpp/cpp_docs.html#implemented-algorithms`

Ascent

In Figure 7.7 and Figure 7.8 optimization runs for both NSGA-II and MOEAD are executed for 300 generations and different population sizes. For MOEAD (Figure 7.8) it can be concluded that for population sizes larger than 64 the Pareto front already converges after 100 generations and even more so for 200 generations. The same can not be said for the NSGA-II optimizer, where a population size of 200 outperforms the other populations for all generations. Furthermore, for for a population size of 32 individuals, both NSGA-II and MOEAD are unable to converge to a single solution.

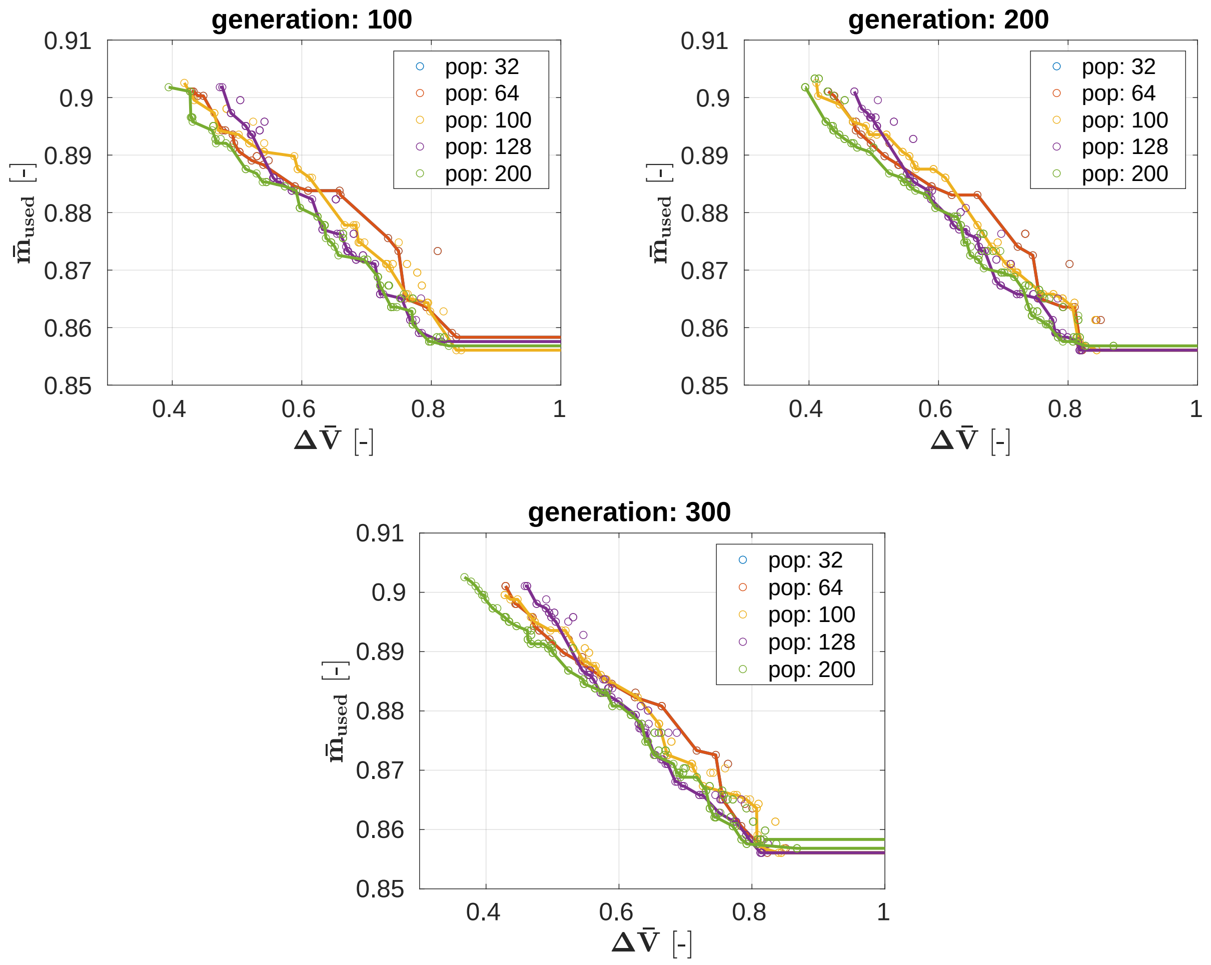
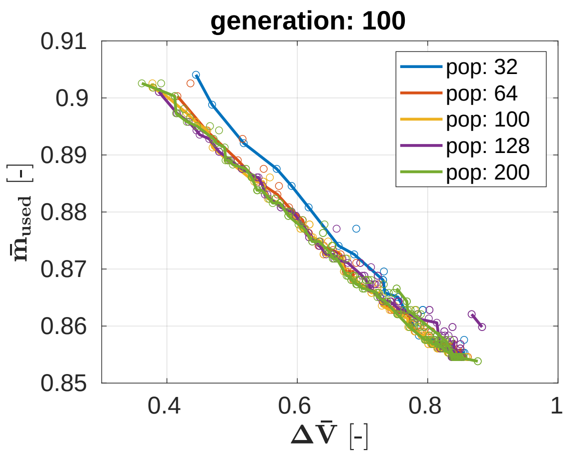
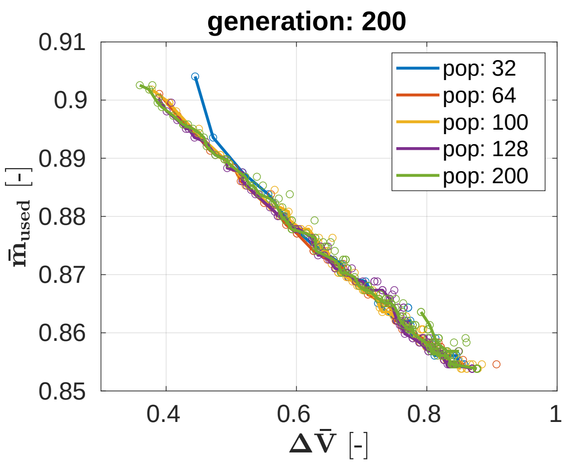


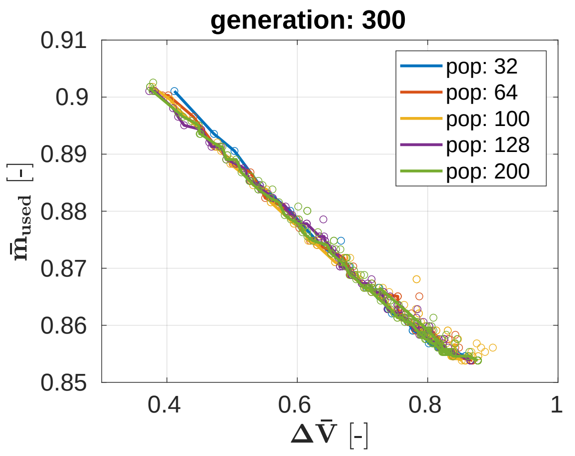
Figure 7.7: Ascent optimization for NSGA-II different generations and populations. $seed = 123$



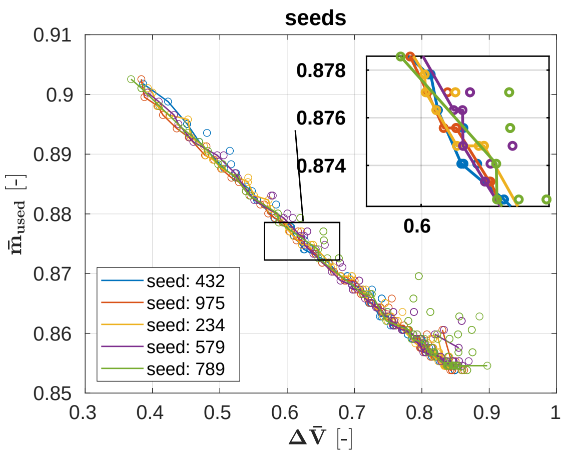
(a) Multiple populations for generations: 200; seed: 123



(b) Multiple populations for generations: 200; seed: 123



(c) Multiple populations for generations: 200; seed: 123



(d) Different seeds for population: 100 and generations: 200

Figure 7.8: Ascent optimization for MOEAD for different populations, generations and seeds

From Figure 7.9 it becomes clear that MOEAD outperforms the NSGA2 algorithm for all population sizes. Although NSGA-II with a population size of 200 performs equally well for some parts of the Pareto front, this comes at the cost of an increased computational effort (200 versus 100 population size). Based on the aforementioned, MOEAD serves as the choice of optimizer for the ascent optimization process.

For each individual in a population a new set of design variables is generated by a random number generator. The initialization of the random number generator is dependent on the *seed* value. The comparison of optimization results can become difficult when different seeds are used because the difference in performance could both be attributed to the change of a system parameter or a fluctuation that stems form the internal sampling randomness. The seed value is best described as a key to generate the same sequence of random numbers over and over again. For the comparison of different optimizer settings (such as population size and evolved generation), the seed is kept constant to rule out the contribution of the aforementioned fluctuations. The robustness of the optimization results (e.g. error margins) can be found by running the same setting for multiple seeds. In Figure 7.8d the ascent optimization results are displayed using the settings from Table 7.4, for 5 different seeds. Based on the observed results it can be said with a certain certainty that the

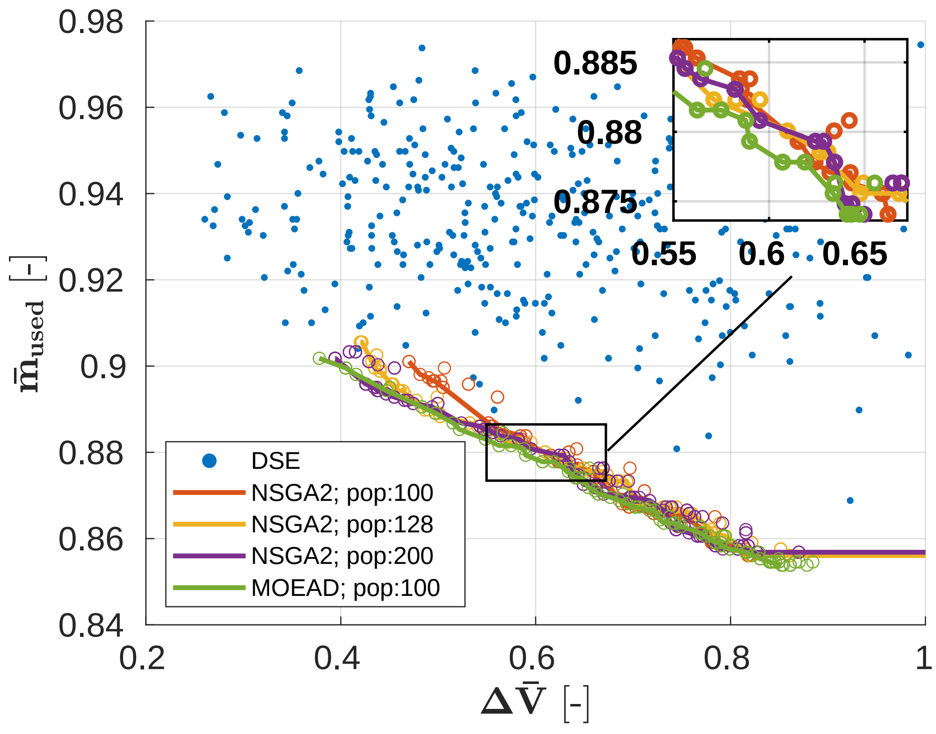


Figure 7.9: NSGA-II and MOEAD for same settings. *gen* = 200; *seed* = 123

optimizer performance is only little influenced by the sampling randomness and therefore deemed *robust*.

Note: the horizontal behaviour of some of the Pareto fronts is caused by the fact that some individuals of the population are heavily penalized but still treated as part of the Pareto front.

Descent

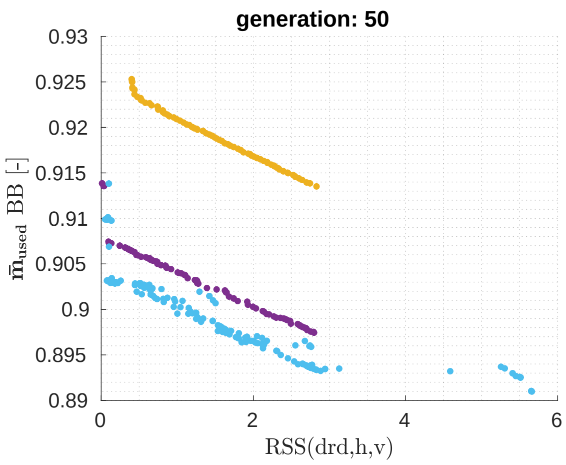
The same method as described in the previous section is followed for the descent optimizer analysis, using the settings from Table 7.3. From Figure 7.10 it becomes clear that for population sizes of 32 and 64 individuals no feasible solutions exist, irregardless of the evolved generations. After 150 generations the population size of 200 has converged and after 250 generations the populations of 100 and 200 individuals perform equally well. Either a population size of 200 evolved for 150 generations or a population size of 100 evolved for 250 generations can be chosen from.

The MOEAD results shown in Figure 7.11 show less promising results and erratic behaviour. None of the populations have converged after 350 generations. The population size of 200 evolved for either 250 or 350 generations show the best best results, but nowhere close to the NSGA-II performance of Figure 7.10. The performance of the NSGA-II and MOEAD optimizer are plotted on the DSE for several population sizes and generations in Figure 7.12. It becomes clear that NSGA-II outperforms MOEAD in every aspect for the descent optimization problem.

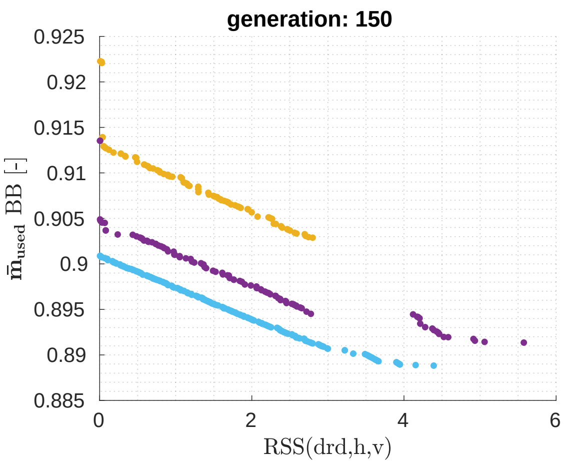
It is hard to say why MOEAD behaves the way it does for the descent problem. Analyzing Figure 7.11 it seems that most solutions tend to cluster around one solution, giving more priority to the RSS objective than the mass objective. This might be a result of the decomposition technique MOEAD uses, but remains speculation. Based on the conducted optimizer analysis, NSGA-II is the chosen optimizer, with 100 individuals and evolved for 250 generations. In Figure 7.10e the descent optimization is plotted for 5 different seeds. All 5 runs show similar behaviour and little variance w.r.t each other.

Table 7.4: Optimizer settings ascent and descent

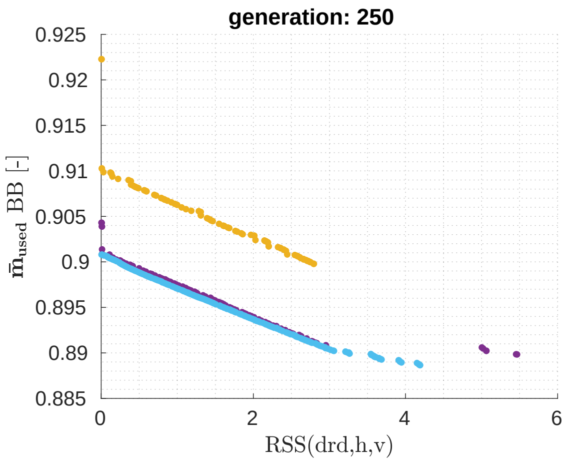
	optimizer	population	#generations
ascent	MOEAD	100	200
descent	NSGA-II	100	250



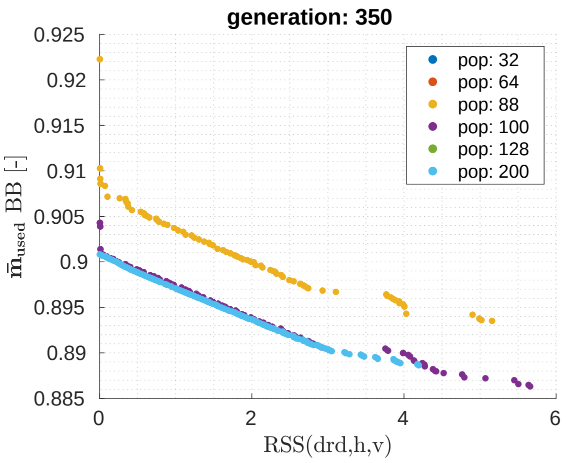
(a) Multiple populations for generations: 250; seed: 123



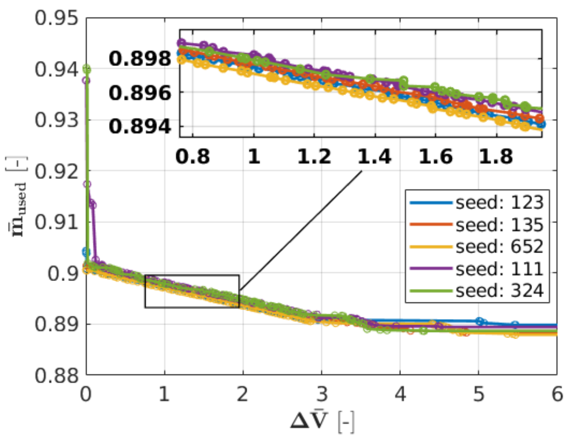
(b) Multiple populations for generations: 250; seed: 123



(c) Multiple populations for generations: 250; seed: 123



(d) Multiple populations for generations: 250; seed: 123



(e) Different seeds for population: 100; generations: 250

Figure 7.10: Optimizer performance NSGA-II for different generations and population sizes for the RTLS trajectory

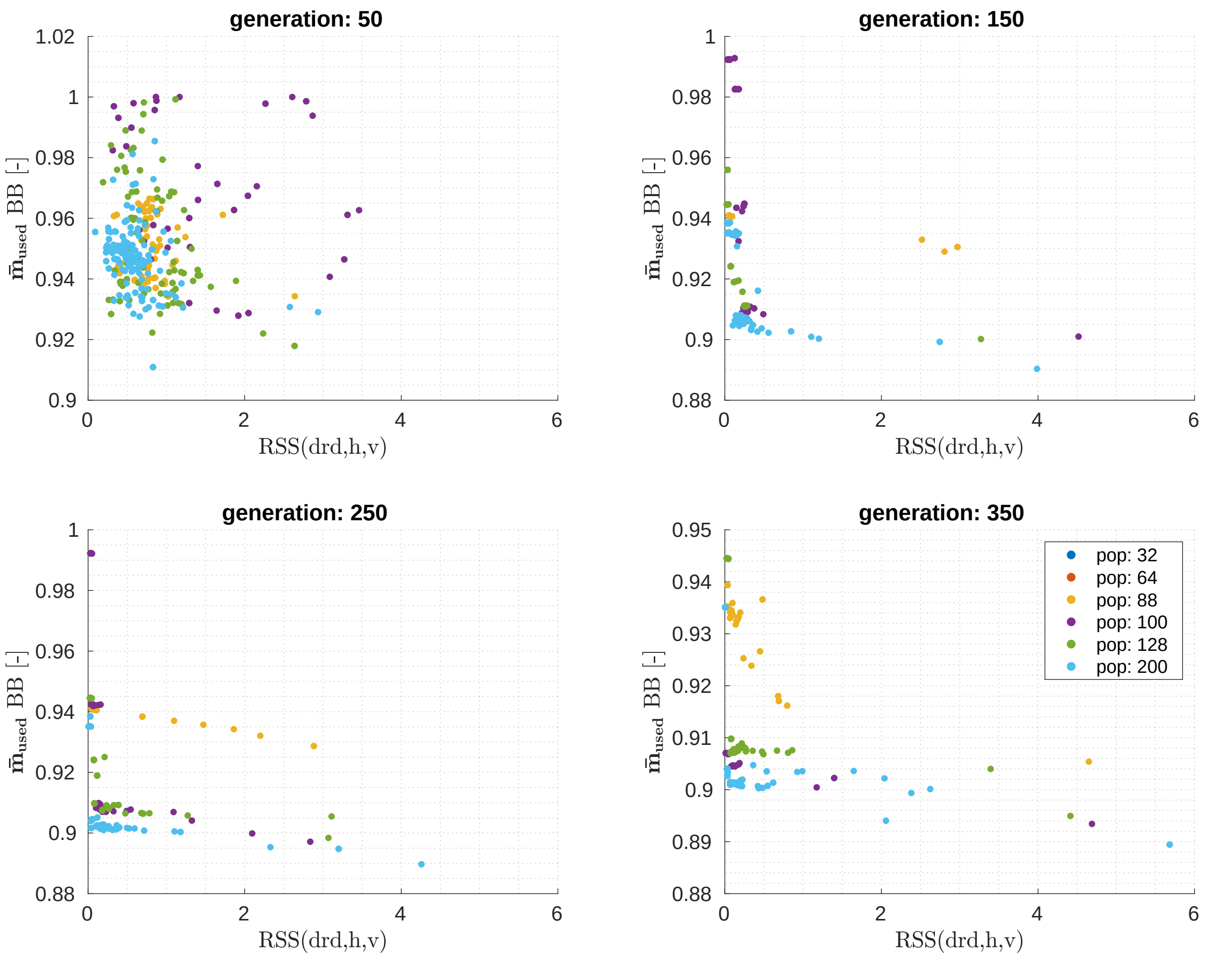


Figure 7.11: Optimizer performance MOEAD for different generations and population sizes for the RTLS trajectory

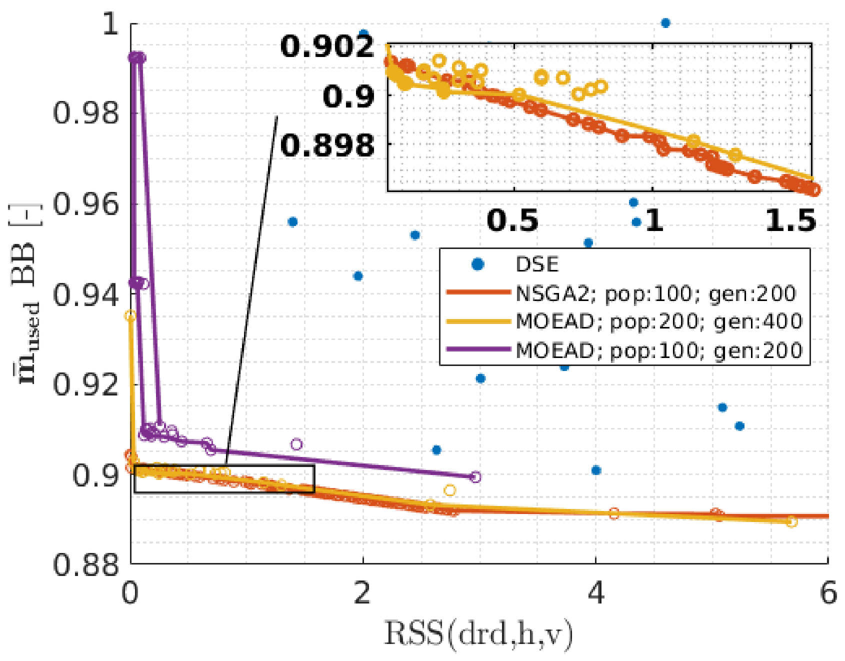


Figure 7.12: NSGA-II and MOEAD performance for several settings $seed = 123$

NSGA-II and MOEA/D

The main objective of NSGA-II is to find multiple Pareto-optimal solutions in one single simulation run. This algorithm has been demonstrated as one of the most efficient algorithms for multi-objective optimization on a number of benchmark problems [68]. It uses non-dominated sorting, crowding distance and elitism techniques to find a solution as close to the Pareto-optimal solution as possible, provide diversity and preserve the best solution of a current population in the next generation respectively [69]. Just like NSGA-II, MOEA/D is a multi-objective optimization optimizer that uses the same techniques as the NSGA-II method. The most important difference is that MOEA/D uses a decomposition technique (hence the D) that decomposes the multi-objective problem into multiple single-objective sub-problems and solves these simultaneously [70]. This makes MOEA/D very well suited for highly complex multi-objective optimization problems and it tends to outperform NSGA-II for more than 2-3 objectives [71]. The optimization problem addresed in this study is rather simple and only consists of two objectives. For a similar two objective optimization problem NSGA-II and MOEA/D perform in a similar fashion [72].

PSO and de-1220

In addition to the Multi-Objective NSGA-II and MOEAD algorithms, the performance of two single-objective optimizers is examined, a Differential Evolution and (*de-1220*) and a Particle Swarm Optimization (*PSO*) algorithm. The obtained results for both algorithms is discussed in Chapter 8.

Differential Evolution

The idea behind the differential evolution is best described by Figure 7.13. A population is evolved for an arbitrary number of generations. For each generation the population generates an off-spring.

In case the off-spring performs better w.r.t the objective function it replaces the parent, otherwise the parent remains in the population. The *DE* was first introduced in the work of Storn and Price[73]. The reader is referred to this work for more technical explanation of the *DE* algorithm. The *de*1220 algorithm is variation of the conventional *de* developed by ESA and has proven itself well in the field of trajectory optimization.

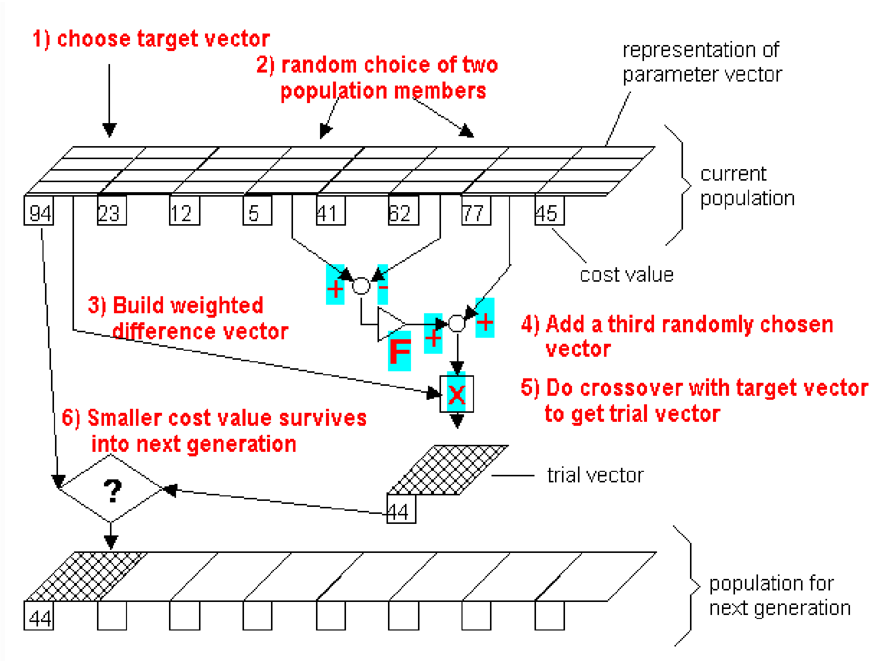

1) choose target vector
2) random choice of two population members
representation of parameter vector
current population
94
23
12
5
41
62
77
45
cost value
+
-
3) Build weighted difference vector
F
+
+
4) Add a third randomly chosen vector
X
5) Do crossover with target vector to get trial vector
6) Smaller cost value survives into next generation
?
trial vector
44
44
population for next generation

Figure 7.13: Schematic representation of the DE algorithm[11]

Particle Swarm Optimization

The *PSO* algorithm is best described by Figure 7.14. In PSO a number of individuals from a population size is placed within the search space of an optimization problem. For every generation each individual evaluates the fitness of the objective function at its current location. For the next generation the individuals move to a new position and once again evaluate their fitness. The direction and distance traveled for this move is determined using the individuals fitness history of previous locations, a random perturbation, but also the fitness history information of all other individuals located in the search space. The individuals thus communicate with each other on where the best solution is located. Similar to a flock of birds foraging for food, the individuals tend to cluster around the optimal solution. A mathematical representation of the *PSO* algorithm, alongside many of its applications, can be found in the work of Poli and Kennedy[74].

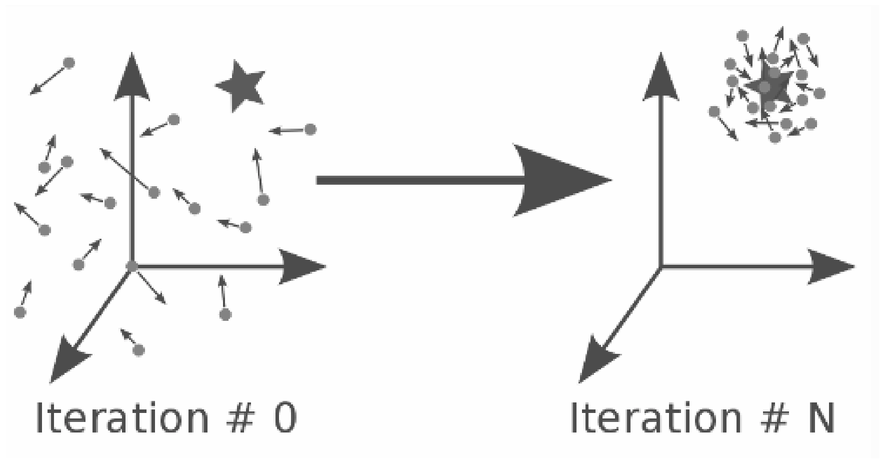


Figure 7.14: Schematic representation of the PSO algorithm[11]

Local optimization

Heuristic global optimization methods, like MOEAD and NSGA-II, are only able to *approach* the optimal solution of a problem and never find the *exact* optimum. In this section a Monte Carlos analysis is performed on the results obtained with the settings from Table 7.4. For every set of design variables that corresponds to a Pareto optimal solution, 500 new sets of design variables are created around that point using a Monte Carlo analysis, uniformly distributed with a variance of 10% for the ascent trajectory and 5% for the RTLS trajectory. This method requires a very accurate *initial guess* of the solution and can therefore not be efficiently used without the global optimization results. The results of the Monte Carlo analysis are displayed in Figure 7.15 and Figure 7.16 for the ascent and RTLS trajectory respectively. For both cases it can be concluded that no improvements on the existing Pareto front were found. For a more in depth analysis other local optimization techniques could be used. However, these techniques surpass a Monte Carlo analysis in complexity and are therefore not touched upon in this study.

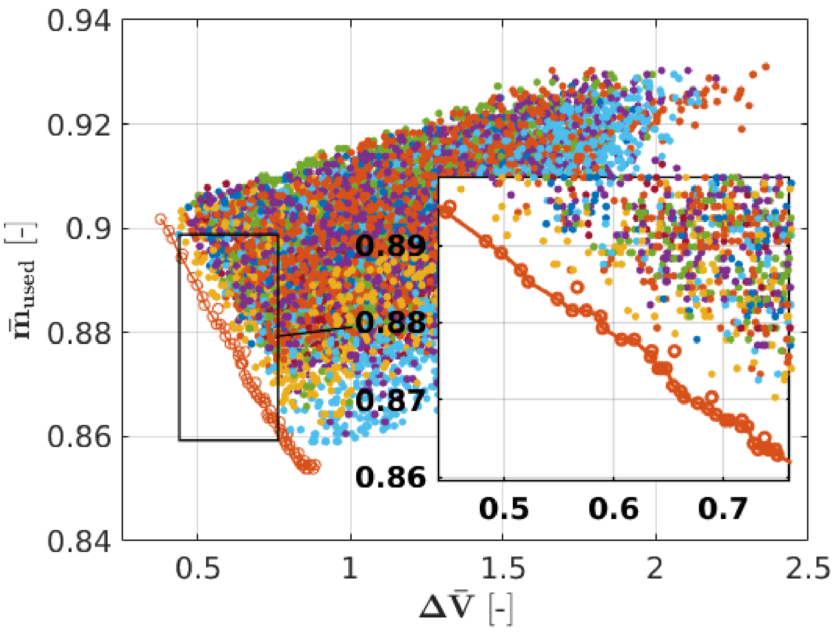


Figure 7.15: Local refinement for the ascent optimization around the Pareto front for 500 Monte Carlo samples, uniformly distributed with a variance of 10%. *seed=123*

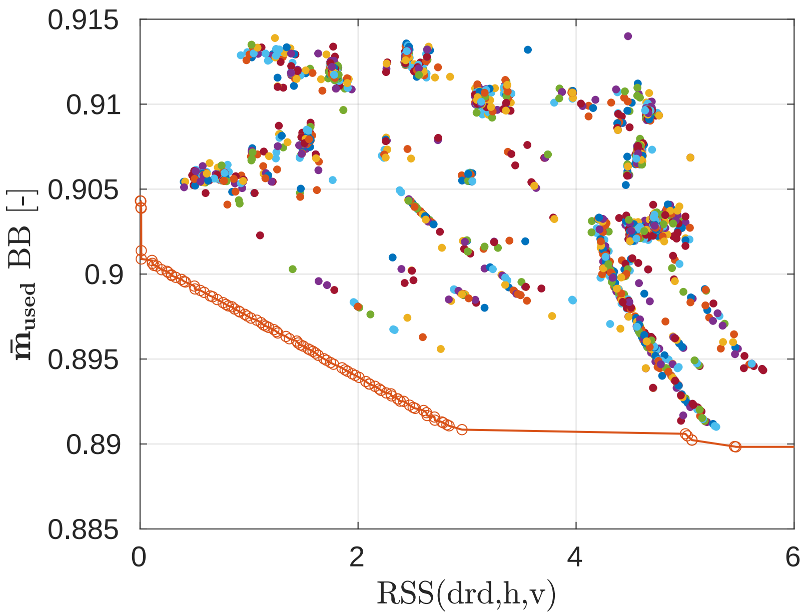


Figure 7.16: Local refinement for the descent optimization around the Pareto front for 500 Monte Carlo samples, uniformly distributed with a variance of 5%. *seed=123*

8 | Results

To reduce the complexity of the optimisation model, the ascent and descent optimization processes are separated. First the ascent optimization model is executed. After interpreting the results the user can manually pick a Pareto optimal solution from the solution set. The model output of the ascent simulation corresponding to that particular solution will serve as input for the descent optimization model.

Ascent

In this section the ascent optimization results are discussed. Ultimately a Pareto optimal solution is picked from the ascent results and used as a starting point for the descent optimization process. Following up on the DSE analysis from Section 7.5, the new search space for both the ascent and descent problem is given in Table 8.1.

Table 8.1: Initiated search space for the optimization runs, based on the DSE analysis.

Ascent			Descent		
	Lower bound	**Upper bound**		**Lower bound**	**Upper bound**
MECO alt. [m]	60000	75000	$t_{boostback}$ [s]	46.5	49.5
Equispaced altitude [m]	20000	50000	$h_{reentry_{start}}$ [m]	50000	65000
θ_1 [rad]	$\frac{1}{2}\pi$	$\frac{1}{2}\pi$	$h_{reentry_{end}}$ [m]	30000	48000
θ_2 [rad]	0.78	1.05	$h_{landing_{start}}$ [m]	10000	20000
θ_3 [rad]	0.52	0.88	yaw angle [rad]	-0.01	0.01
θ_4 [rad]	0.2	0.8	θ_c [deg]	160	190
θ_5 [rad]	0.0	0.3	ε_1 [-]	0.2	0.8
θ_6 [rad]	0.0	0.1	ε_2 [-]	0.4	1.0

The nominal payload that the Electron can bring to a SSO is 175kg[2]. Although it would be interesting to add the payload mass to the objective function, in this study the payload mass is varied but kept constant throughout the optimization process. The optimization is executed for the following payloads: [150kg 175kg 200kg 225kg 250kg] and plotted in Figure 8.1. It becomes clear that for payloads larger than 200kg the optimizer is unable to find a feasible trajectory.

For the Falcon 9 a minimum of 14% of the initial first stage fuel mass is required to perform a successful RTLS trajectory. The same value is used to limit the range of feasible ascent trajectories of the Electron. Following this reasoning, ascent trajectories with a 200kg payload are not suited for RTLS trajectories. For Electrons RTLS mission any trajectory below the dotted line can be used as a starting point. The most fuel efficient trajectory from the 175kg payload configuration is used as the starting point for the RTLS trajectory (green dot in Figure 8.1).

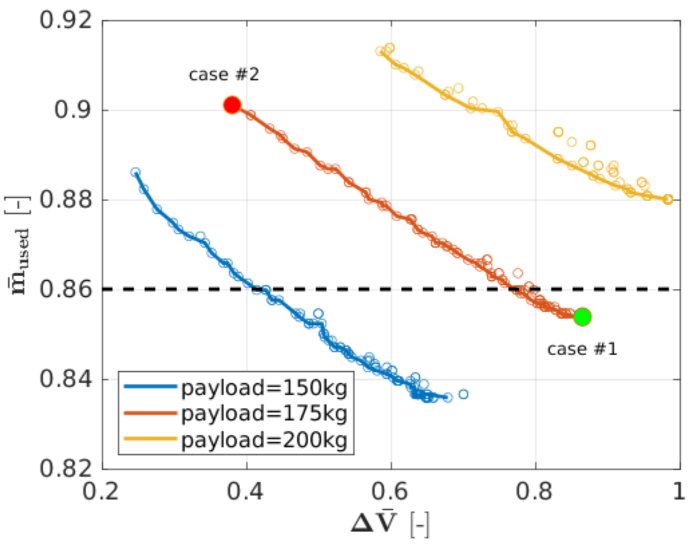
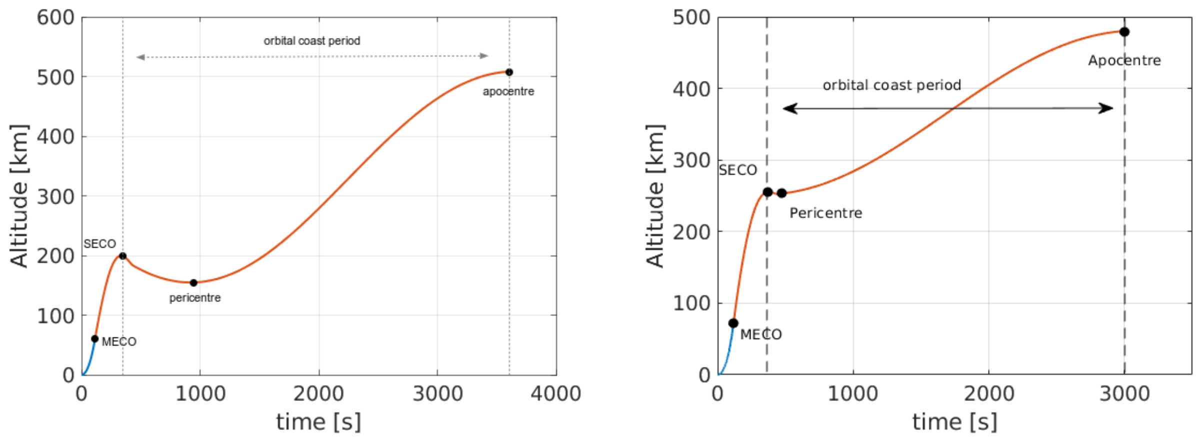


Figure 8.1: Ascent optimization performance for different payloads. *optimizer: MOEAD; seed=123; population=100; generations=200*

In Section 7.4 it was mentioned that the final orbit is reached by means of an orbital coast period and a corrective circularization manoeuvre at the apocentre of the transfer orbit. It was also mentioned that the second stage was launched to the pericentre and start the coast period from there. However, in Figure 8.2a it can be observed that the second stage arrives at the transfer orbit prior to the pericentre and is therefore not as efficient.



(a) case #1

(b) case #2

Figure 8.2: Ascent trajectory flight profile for the selected solution from Figure 8.1

The exact location of where a vehicle arrives or intersects with a specified orbit is directly related to the Argument of periapsis and therefore the time of launch. The argument of periapsis could be added to the model as a design variable to further minimize the fuel consumption or ΔV. As aforementioned in Chapter 6, this is deemed outside the scope of this study and will be left as a recommendation. From Figure 8.2 and Table 8.2 it becomes clear that for case #2 a higher pericentre and a closer distance to the pericentre at SECO is obtained and. As a result the required ΔV is also lower. However, the amount of fuel saved due to the more efficient ΔV manoeuvre is only little in comparison to the extra used fuel of the launch vehicle. It is up to the end-user make a balanced decision based on the mission requirements. The final trajectory parameters corresponding to the two optimal trajectories are displayed in Table 8.3.

Table 8.2: Trajectory parameters of 2 different Electron mission to SSO. Parameter values at MECO are used as the initial conditions of the RTLS trajectory.

parameter	case #1	case #2
MECO		
time [s]	114.2	120.2
altitude [m]	60055	74927
velocity [m/s]	1821	1967
longitude [deg]	-38.9057	-38.952
latitude [deg]	177.789	177.791
flight path angle [deg]	39.88	48.54
heading angle	-9.66	-10.8
Downrange distance [km]	40.1	44.2
Q_{max} [kPa]	38.9	38.8
fuel mass left [kg]	1246	925.2
fuel mass left [%]	14.1	10
Second stage		
final altitude [km]	182.4	252.6
Apocentre altitude [km]	508.7	480.7
pericentre altitude [km]	155	251.7
semi-major axis [km]	6703	6737
eccentricity [-]	0.0264	0.0170
inclination [deg]	97.53	98.12
final fuel mass [kg]	25	25
ΔV_{total} [m/s]	202.5	88.34
ΔV_{total} required fuel [kg]	21.84	9.72

Descent

The descent optimization results obtained with the settings from Table 7.4 are plotted in Figure 8.3. Looking at the y-axis, it becomes clear that the margin of $\bar{\mathbf{m}}_{\mathbf{used}}$ between the two extreme Pareto optimal solutions is very small, around 1 to 2% of the initial fuel mass of the first stage prior to the RTLS trajectory (1246kg, see Table 8.2). In Section 7.4 it was mentioned that for all RSS < 1 the final distance to the landing site, velocity and altitude are all within the set tolerances. For all RSS > 1 there exists a possibility that one or more tolerances are not met.

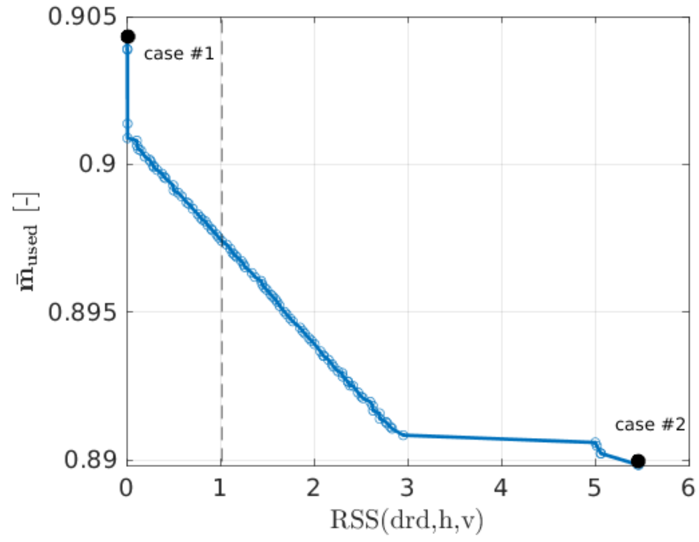


Figure 8.3: Descent optimization performance for different payloads. *optimizer: MOEAD; seed=123; population=100; generations=200*

The results show that the optimizer is capable of finding feasible RTLS trajectories, but no real correlation between the RSS and $\bar{\mathbf{m}}_{\mathbf{used}}$ values is found. Contrary to the optimal design variables found for the ascent trajectory, the RTLS trajectory parameters are almost the same for each Pareto optimal solution (e.g. comparing case #1 to case #2).

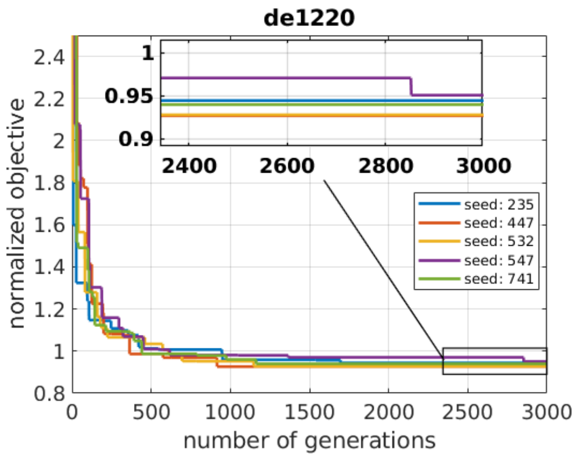
Table 8.3: Trajectory design variables corresponding to two of the Pareto optimal solutions for the ascent and RTLS trajectory and design variables obtained with the Single-Objective optimization.

Ascent			**Descent**				
	MOEAD			**NSGA-II**		**de1220**	**PSO**
	case #1	**case #2**		**case #1**	**case #2**		
MECO alt. [m]	60028	74997	$t_{boostback}$	48.8	48.5	48.7	48.7
Equispaced altitude [m]	60028	74997	$h_{reentry_{begin}}$	50016	50017	50764	50000
θ_1 [rad]	$\frac{1}{2}\pi$	$\frac{1}{2}\pi$	$h_{reentry_{end}}$	47988	47999	43717	40479
θ_2 [rad]	0.822	0.974	$h_{landing_{begin}}$	10645	10748	10513	10000
θ_3 [rad]	0.582	0.579	yaw angle	0.0090	0.0089	0.0091	0.0041
θ_4 [rad]	0.579	0.474	θ_c	167.8	167.8	170.4	171.7
θ_5 [rad]	0.196	0.267	ε_1	0.202	0.202	0.218	0.2
θ_6 [rad]	0.0369	0.0675	ε_2	0.594	0.6	0.568	0.561

The small margins of the mass objective make the Pareto front less interesting because the trade-off between the Pareto optimal solutions is barely noticeable. The added value of a Multi-Objective optimization process w.r.t a Single-Objective optimization can therefore be questioned. In addition a Single-Objective optimization is conducted for two different optimizers, *de1220* and *PSO*. For the objective function the $\bar{m}_{used}$ and RSS objective of the Multi-Objective optimization are lumped together into one objective function (Equation 8.1). In Figure 8.4 the results of both optimizers are plotted for multiple seeds. It stands out that *de1220* outperforms *PSO* in terms of better performance and quicker convergence.

$$objective = \bar{m}_{used} + RSS \tag{8.1}$$

Although the results of case #1 of the Multi-Objective optimizer show a slightly better performance than *de1220*, Table 8.4, the differences are small. In Table 8.3 it can also be noticed that the design variables for the SO tend to converge to the same values as for case #1 and case #2 of the MO optimization. The sensitivity of the design parameters on the outcome of the optimization process will be discussed in chapter 9.



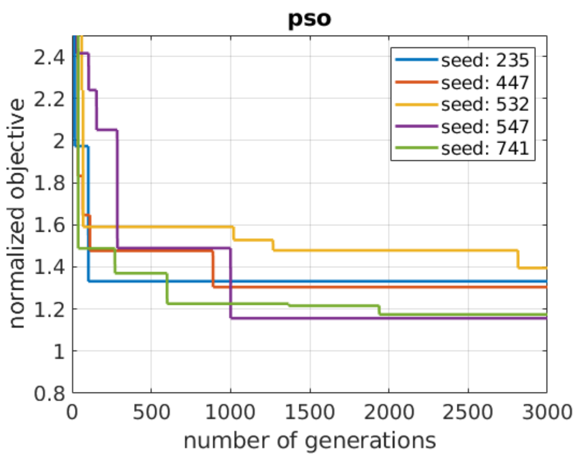
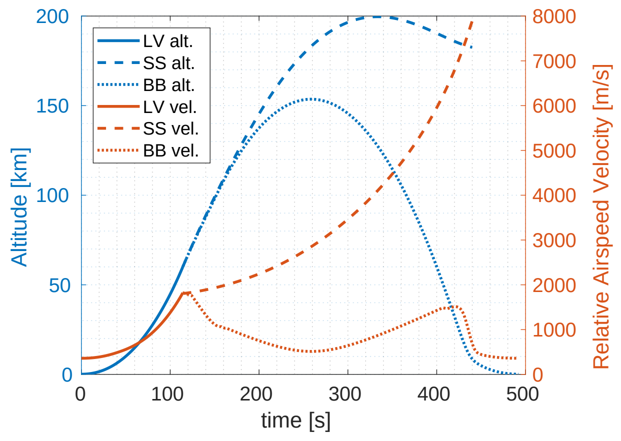


Figure 8.4: Single Objective optimzation for PSO and de1220 algorithms for multiple seeds and population: 64

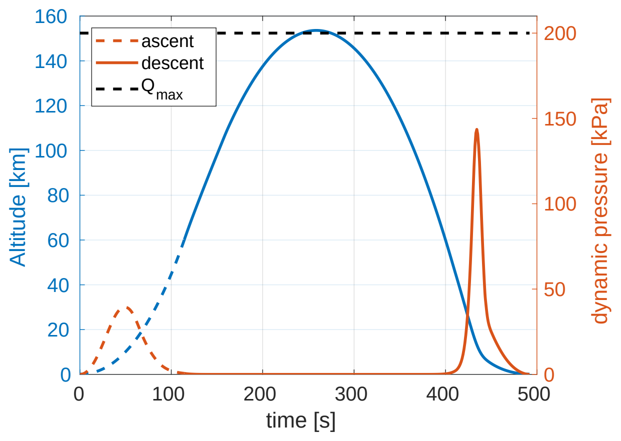
Table 8.4: Optimization results for all optimizers

	NSGA-II		de1220	PSO
	case #1	case #2		
Initial fuel mass ($m_{fuel,BB}$)	1345	1345	1345	1345
final fuel mass [kg]	128.9	148.3	111	106
D_{final} [m]	6.9	4735	9.34	201
V_{final} [m/s]	0.54	14.7	1.0	0.2
h_{final} [m]	101	100	100	104
Q_{max} [kPa]	162	164	143	136
$(m_{fuel,BB}/m_{fuel,LV})_{available}$ %	14.4	14.4	14.4	14.4
$(m_{fuel,BB}/m_{fuel,LV})_{used}$ %	13.01	12.8	13.2	13.3

In Figure 8.5 the flight profile of the optimal descent trajectory is illustrated. A similar trajectory is obtained as the validated Falcon 9 trajectory from chapter 6, which is to be expected. Many potential solutions have been discarded during the optimization process because the maximum dynamic pressure was exceeded. However, the obtained optimal solution stays well below the constraint of 200kPa. A noticable difference between the Falcon 9 and Electron descent trajectory is the reentry burn time, which is longer for the Falcon 9. In practical flight the reentry burn time might have to be increased to slow the vehicle down at an earlier stage and allow for better attitude control. Since attitude control lies outside the scope of this study the (reentry) burn times are left unconstrained.



(a) Altitude velocity profile



(b) Dynamic pressure profile

Figure 8.5: Altitude, velocity and dynamic pressure profile for optimal ascent and descent trajectory

In Figure 8.6 the descent trajectory of Electrons first stage is illustrated. Because only little is known about the actual Electron and its engines, some assumptions were made to arrive at the results. In order to fly a RTLS trajectory it has to be assumed that the engines are re-ignitable and throttable. These are potential alterations that need be made on the existing Electron configuration. No alterations were made on the initial fuel mass of the Electron as the lower MECO altitude proved to be efficient in saving fuel for the descent trajectory. The nominal payload of 200kg however, had to be reduced to 175kg to find feasible initial conditions for the descent trajectory.

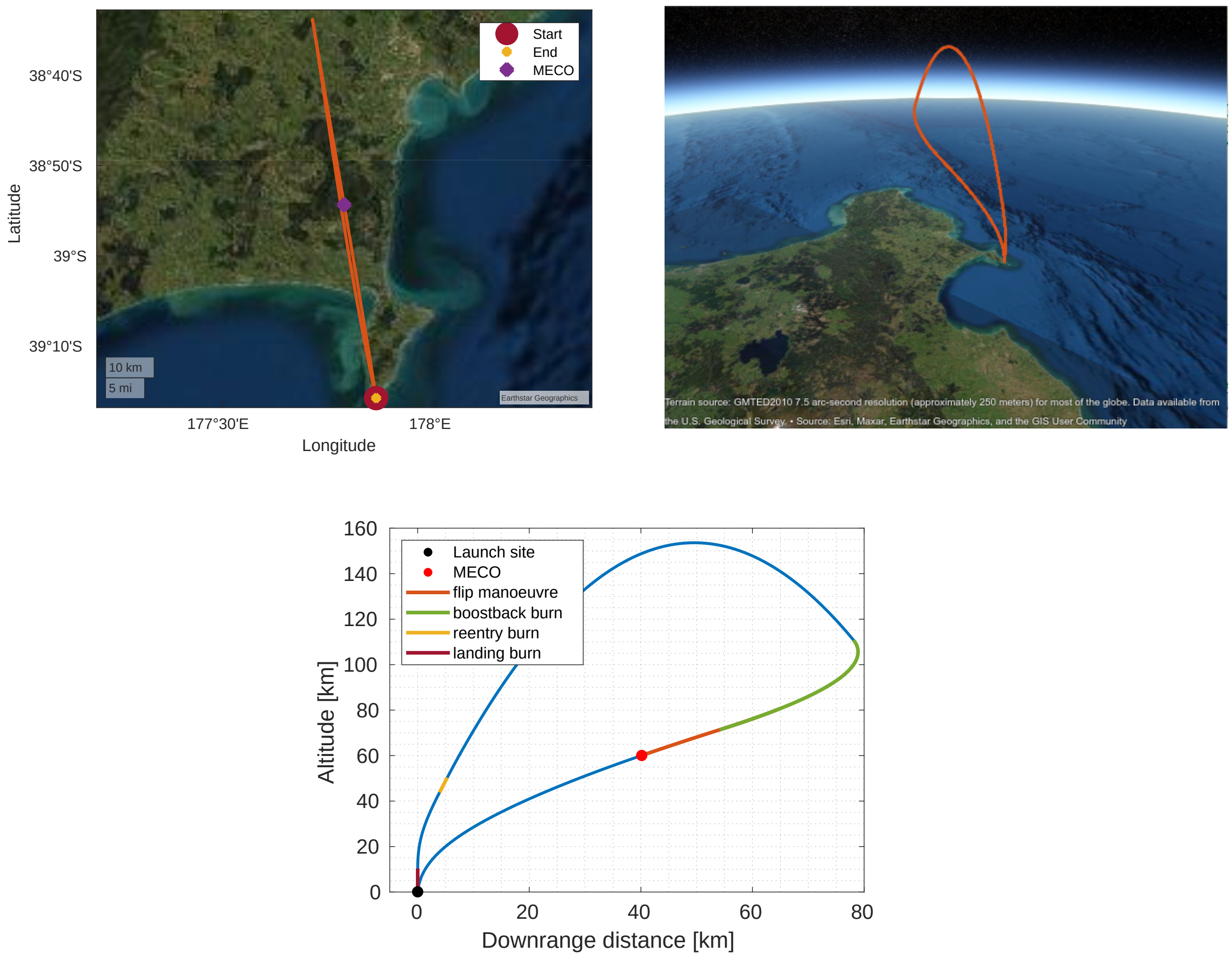


Figure 8.6: Illustrative descent trajectory plots for the case #1 optimal descent trajectory

9 | Sensitivity Analysis

The results of an arbitrary simulation/optimization problem heavily rely on the set of (design) variables that is used to describe the problem. In reality, variables are always subjected to uncertainties. For example, the I_{sp} of an engine might deviate from the nominal value (maximum deviation in this case is usually provided by the manufacturer). A sensitivity analysis is nothing more than subjecting an arbitrary variable to a plausible (random) deviation and then investigate the effect on the output. One way to execute a sensitivity analysis is by deviating one parameter at a time (*one-at-a-time approach*). However, any correlation between the individual parameters is then neglected. To bypass this issue, all parameters can be subjected to an uncertainty at the same time. However, doing so it becomes difficult to analyze how much the variance of a specific variable contributes to the variance of the outcome. To solve this problem a *factorial design* method can be used. The full factorial design method examines all possible combinations of levels for all factors (design variables). The number of experiments that have to be conducted for a full factorial design increases with the power of the number of design variables and can be become cumbersome very soon[75].

In this study an extended Fourier amplitude test (EFAST) is used to perform a sensitivity analysis on the results. The method was first published in the work of Saltelli[76] and amongst others used in the work of Van Damme[41]. With this method the effect of each input variable on the variance of the output is computed and measured. A Latin Hypercube Sampling (LHS) method is used to vary the input parameters around their optimal value. A Latin Hypercube is represented by a n-by-m matrix where n equals the number of samples taken and m the number of input parameters. For each column of the Latin Hypercube matrix the n values are randomly distributed with one from each interval, and randomly permuted:

$$(0,1/n),(1/n,2/n),...,(1-1/n,1)$$

Due to the more evenly distributed sample space of the LHS method it requires much smaller sample sizes than e.g. Monte Carlo sampling and still achieves a reasonably accurate random distribution. The method works especially well for larger number of design variables. In the work of Seaholm, a *factorial design* required 14 times the sample size of a LHS method for the sensitivity analysis on an epidemic model, obtaining the same results[77]. The total effect of each individual input parameter on the output variance is computed using so called Sobol sensitivity indices [75]:

$$S_i = \frac{V_i}{V(\Upsilon)} \qquad\qquad S_{Ti} = 1 - \frac{V_{-i}}{V(\Upsilon)} \tag{9.1}$$

$V(\Upsilon)$ is the total output variance of the model and V_i is the fraction of the output variance that is caused by varying only input parameter x_i. S_i is sometimes called the first order sensitivity indices. V_{-i} on the other hand represents the variance of the output caused by all input parameters except x_i. S_{Ti} is therefore also called the total-effect sensitivity indices. For the computation of the sensitivity indices, $m+1$ hypercubes have to be generated, one for each input parameter and one extra to compute $V(\Upsilon)$, were all input parameters are varied at the same time.

This study leans more towards finding the optimal descent trajectory rather than the optimal ascent trajectory. It is therefore decided only to perform a sensitivity analysis on the descent optimization results. The sensitivity analysis can be performed for every Pareto optimal solution. However, due to the large computational effort that would require, an arbitrary optimal solution is selected for the sensitivity analysis. Besides, it is reasonable to assume that all optimal solutions show the same behaviour in the sensitivity analysis (because the solutions are very similar to each other). The Latin Hypercubes are generated with Matlab and then imported to the Tudat environment. A uniformly distributed sample-size of $N = 1000$ and 10% variance on the input parameters is used. Two sensitivity analyses are conducted, one for varying the design variables and another for varying the initial state parameters of the RTLS trajectory. In this study only the total-effect indices will be discussed.

The output variance is measured by the previously mentioned RSS value and the components it consists of. From Figure 9.1 it can be concluded that the boostback time has the biggest effect on the model output. This also strokes with the results from the DSE obtained in section 7.5, where only a very small range of the boostback time resulted in feasible solutions. The boostback time and the pitch-over are the only two parameters that can effectively change the direction of the flight path and therefore the final distance to the landing site (DRL). This is also concluded from Figure 9.1. The final velocity of the vehicle is mainly dependent on the throttle factors and the altitudes at which the burns start. In case the throttle factor is too low, or the burn altitude too low, the vehicle is not able to slow down quick enough before impact. On the other hand, if the throttle factor is too high, this can result in reaching 0m/s at higher altitudes or fuel burning up too quick, terminating the simulation at a high altitude. This explains the relatively high contribution of the final altitude w.r.t the RSS value. In the previous chapter it was found that the optimal solutions for the RTLS trajectory all feature short burn-times for the reentry burn phase and therefore does not contribute as much to the final solution as e.g. the landing burn phase. This can also be concluded from Figure 9.1 where the reentry burn altitudes and reentry throttle factor barely contribute to the total variance of the output.

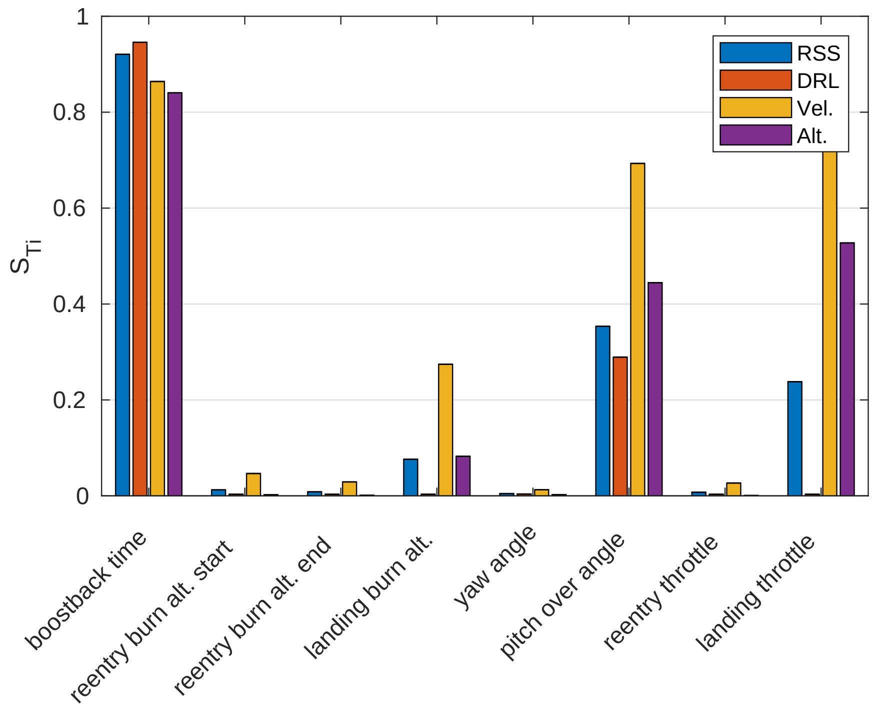


Figure 9.1: Mean of the total effect sensitivity indices on the output variance for given input parameters.

A similar analysis can be conducted for varying the initial state parameters of the RTLS trajectory, displayed in Figure 9.2. In practical flight the uncertainties in the initial state parameters could e.g. be caused by inaccuracies in the (GPS) measurement equipment on board of the vehicle. In it stands out that an offset of the radius (or altitude) has a high impact on the final velocity and altitude of the vehicle. This can be explained by the fact that the radius is defined as the distance from the center of the Earth to the vehicle ($R_{earth} + h$). A variance of 10% on the radius leads to an absolute offset in the order of 10^2km. These numbers are not realistic and the results therefore not reliable. To mitigate for this to happen a variance on altitude could have been used instead. The longitude and latitude represent the position of the vehicle projected on the service of the Earth. An offset on the initial longitude and latitude is therefore expected to influence the downrange landing distance to a high level. Although this is the case for the latitude, the same does not hold for the longitude. This is remarkable taking into account that both parameters play an equal role in computing the downrange distance.

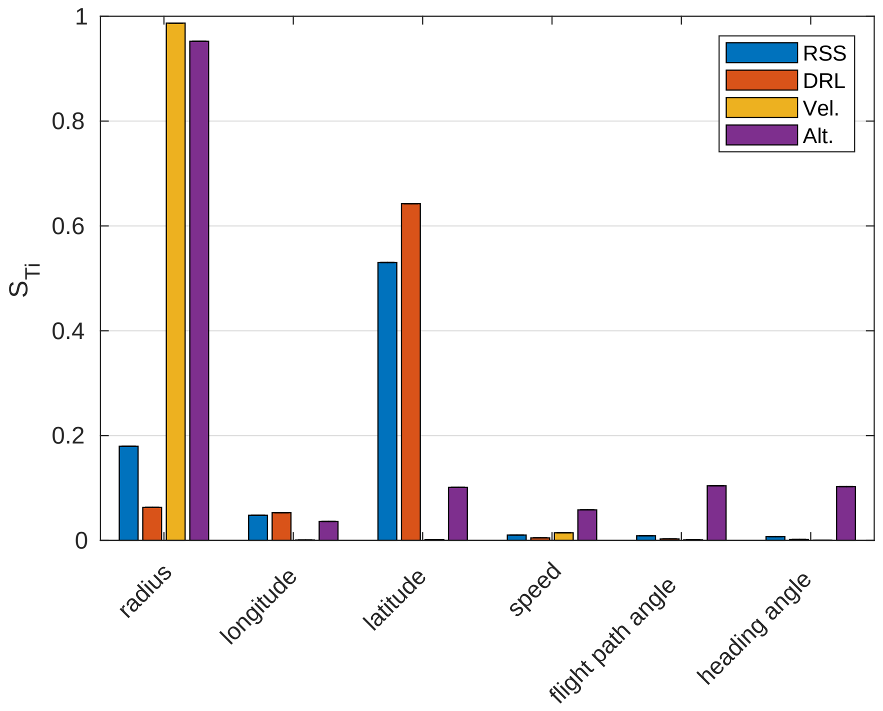
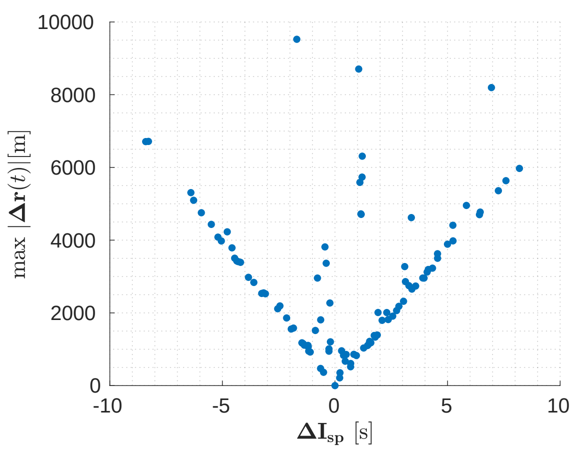
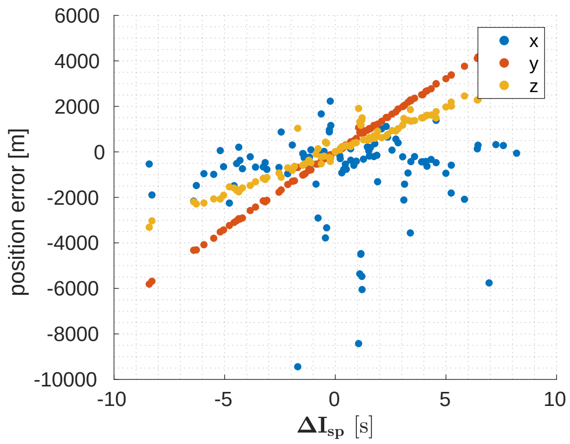


Figure 9.2: Mean of the total effect sensitivity indices on the output variance for given input parameters.

Last but not least, the previously mentioned specific impulse (I_{sp}) is subjected to an uncertainty. The I_{sp} directly influences the thrust magnitude an engine is capable of delivering and therefore interesting to investigate. Engine performance is always subjected to a certain degree of uncertainty. The specific impulse uncertainties are sometimes provided by the manufacturer, however, such data is not found for Electrons Rutherford engine. In a paper written by Davidian and Dieck the I_{sp} of rocket engines is studied. The study concludes with an 1.3% uncertainty on the I_{sp}. The largest contributions to the uncertainty were caused by calibration errors from the measurement devices[78]. For this analysis the I_{sp} will also be varied with a uncertainty of 1.3%. From Figure 9.3 it can be concluded that the I_{sp} uncertainty seriously effects the final position error of the vehicle, more than the 1km tolerance that was introduced in section 7.4. Due to the predictive behaviour of the uncertainty the offset in the final position vector can be compensated for. This could e.g. be done with a guidance system that monitors offset in position and acts accordingly by decreasing or increasing the thrust magnitude or direction.



(a) Norm of the position error



(b) Contributions of the x,y and z coordinate on the position error

Figure 9.3: Uncertainty analysis of the specific impulse I_{sp} on the position error of the model output

10 | Conclusion and Recommendations

In this chapter the findings of the study are concluded and the previously established research questions are tried to be answered. Furthermore a list of of recommendations is presented to point out and discuss the shortcomings of this study and potential improvements on the model.

The focus of this study was to investigate the feasibility of an RTLS mission using vertical powered descent techniques for an already existing small satellite launcher: Electron. The lesser changes made on the vehicle configuration the more interesting the results become for e.g. Rocket Lab. Any positive findings that would involve large changes of the vehicle are not desired because the Electron already exists in its current form and Rocket Lab is not expected to drastically change the design of the rocket anytime soon. In terms of mass distribution no big changes had to be made because the estimated 12.5%[24] of the vehicles dry mass originally used for the MAR system can now be used for the landing gear and grid fins.

It was also found that by assuming an instantaneous pitch-over manoeuvre for the RTLS trajectory (e.g. used in the work of Contant[1]), significantly less fuel is required than for a more realistic flight profile that embeds a gradual pitch-over manoeuvre. The difference between the two methods was estimated to be around 10%.

For the computation of the aerodynamic forces a Missile DATCOM database was provided. The database contains the aerodynamic force coefficients for a broad range of simple vehicle configurations, taking the length and diameter as input. The database then finds the set of coefficients that comes closest to the actual vehicle. For the Electron, with a length of 18.0m and a diameter of 1.2m, the database provides the coefficients corresponding to a vehicle with length 20m and diameter 1.6m. The database might serve well for the preliminary design of launch vehicles, it is deemed too inaccurate for this study. For that reason the Missile DATCOM software was used to generate the aerodynamic coefficients corresponding to the exact configuration of the Electron, leading to a more accurate aerodynamic flight profile.

Previous real life missions of the Electron to SSO orbit involved MECO altitudes of around 75km. It is not known what type of ascent trajectories Rocket Lab currently uses for its missions, but using an orbital coast period presented in this study, the MECO altitude was reduced to 60km (similar to Falcon 9's MECO altitude) while still reaching SSO orbit. Reducing the MECO altitude directly affects the fuel use of the Launch Vehicle. This is an important aspect of this study because the extra saved fuel during ascent can now be used for the RTLS trajectory.

An optimization framework for the ascent and descent trajectory was presented in this study. For the ascent trajectory it can be concluded that the MOEAD optimizer outperforms the NSGA-II optimizer. After optimization the reader is left with a Pareto front of optimal solutions of which the 'best' solution can be picked manually. For the optimization of the descent trajectory the Multi-Objective optimizers were not able to give a better insight in the problem than the proposed Single-Objective optimizers. This is mainly caused by the fact that the Multi-Objective descent optimization converges for one solution only, with a useless Pareto front as a result. It can therefore be concluded that the required fuel mass is not strongly correlated to the final accuracy of the landing site. For the SO objective optimization the $de1220$ algorithm outperformed the PSO algorithm both in terms of convergence and performance.

As presented in Table 8.4, 128kg of fuel is still left after the first stage has successfully reached the landing site. Because this amount of fuel is not used, it could e.g. be used to increase the payload mass if the optimization process were to be iterated. From the same table it is observed that the first stage approach the landing site with a accuracy of the order 10m. This is an extremely good result were it not that the maximum achievable accuracy of the integrator is 686m.

From the sensitivity analysis it was concluded that the boostback time and pitch-over angle are the most determining factors in computing the model output. In practical flight those variables are not necessarily fixed and could be adjusted during flight as required by the on-board guidance system. High variance of the output caused by those variables can than be compensated for during flight and as such will not accumulate to a high variance of the output. Another interesting outcome is the maximum achievable accuracy by the integrator. Ideally the choice of integrator should not be leading in determining the accuracy requirements of the model. A more in-depth analysis on the integrator performance is therefore strongly recommended.

To conclude, no major modifications to the existing Electron rocket had to be made to safely return it to the launch site. The payload was reduced from the nominal 200kg to 175kg to make sure the orbit was still reached. The most optimal descent run was able to land with a remaining 148kg of fuel. As mentioned before, the choice of integrator seriously effects the final accuracy of the model. In the end a simple aerodynamic and thrust guidance algorithm were implemented.

10.1 Recommendations

Simulations or models of real time events are always subjected to simplifications and assumptions. Which assumptions and until what degree can be made is highly dependent on the goal of the study. A few recommendations are discussed, which if implemented, are believed to improve the overall performance of the designed (optimisation) software tool.

- The addition of an extra objective function in the form of the payload mass for the ascent trajectory. Adding such objective gives more design freedom to the user because the desired payload mass can is not fixed but computed as a function of the other objective function.
- To further improve the performance of the ascent trajectory the Argument of periapsis could be added as a design variable. This allows the optimizer to find the optimum launch window the rocket should be launched in.

- In this study a simple guidance law is used to impose an attitude on the vehicle during flight. The slenderness of the rocket results in a naturally unstable behaviour. If no control would be added (or attitude be imposed), the attitude of the vehicle would show very erratic behaviour. Keeping the rocket up right requires active attitude control by means of grid-fins, engine gimballing and cold gas nitrogen thrusters. It would be interesting to analyze whether the attitude of the vehicle can remain stable during one of the optimal trajectories and if yes, how much extra fuel this would require.
- The addition of grid-fins. Grid fins are used to both decelerate the vehicle and to control its attitude.
- It may well be worth to further investigate the performance of the integrator and see if any improvements on the achievable accuracy can be made.
- This study shows that its technically possible for the first stage of the Electron to fly back to the landing site. It would be interesting to compare the results of this study with the results obtained for a MAR method. Rocket Lab has recently executed a successful MAR recovery and in addition a student at the TU Delft is currently working on this topic.

Bibliography

[1] Contant, S., "Design and Optimization of a Small Reusable Launch Vehicle Using Vertical Landing Techniques," 2019.

[2] Rocket Lab, "Launch: payload USER'S GUIDE," , No. August, 2020.

[3] Mooij, E., *AE4870B Re-entry systems*, Faculty of Aerospace Engineering, Delft University of Technology, 2019.

[4] R.W. Kretzschmar, J. B., "Aerodynamic Prediction Methodology for Grid Fins," *RTO AVT Symposium on "Missile Aerodynamics", held in Sorrento, Italy*, 1998.

[5] Aerospaceweb.org, "Missile Grid Fins," `http://www.aerospaceweb.org/question/weapons/q0261.shtml`, 2006.

[6] Wakker, K., *Fundamentals of Astrodynamics*, 2015.

[7] Filho, W. and Mallaco, L., "Strategy for pitch-over maneuver control," 08 1999.

[8] Ecker, T., Zilker, F., Dumont, E., Karl, S., and Hannemann, K., "Aerothermal Analysis of Reusable Launcher Systems During Retro-Propulsion Reentry and Landing," *Space Propulsion 2018*, , No. May, 2018, pp. 1–12.

[9] Szmuk, M., Reynolds, T. P., Açıkmeşe, B., Mesbahi, M., and Carson, J. M., "Successive convexification for 6-dof powered descent guidance with compound state-triggered constraints," *AIAA Scitech 2019 Forum*, 2019.

[10] Zheng, Y., Fu, X., Xu, M., Li, Q., and Lin, M., "Ascent trajectory design of small-lift launch vehicle using hierarchical optimization," *Aerospace Science and Technology*, Vol. 107, 2020, pp. 106285.

[11] Biscani, F. and Izzo, D., "A parallel global multiobjective framework for optimization: pagmo," *Journal of Open Source Software*, Vol. 5, No. 53, 2020, pp. 2338.

[12] SpaceLaunchReport, "Rocket Lab Electron Data Sheet," `https://www.spacelaunchreport.com/electron.html#:~:text=Electron$%$20weighed$%$2012.55$%$20tonnes$%$20at,$%$2Dkilometer$%$20sun$%$2Dsynchronous$%$20orbit.`

[13] US Government, "U.S. Standrad Atmosphere, 1976," *National Oceanic and Atmospheric Administration National Aeronautics and Space Administrat'on United States Air Force*, 1976, pp. 241.

[14] "CRS-10 Technical Webcast," `https://www.youtube.com/watch?v=rUDLxFUMC9c`, Accessed: 2022-05-2.

[15] SpaceX, "CRS-10 Dragon Resupply Mission," 2017, `https://www.nasa.gov/sites/default/files/atoms/files/crs10presskitfinal.pdf`.

[16] Markets and Markets, "Small Satellite Market Global Forecast to 2025," `www.marketsandmarkets.com`.

[17] Catapult Satellite Aplications, "Small Satellite Market Intelligence Report, Q3 2020," `https://s3.eu-west-1.amazonaws.com/media.newsa.catapult/wp-content/uploads/2020/11/18155227/20110985-Small-SAT-Market-Q3-2020-Final.pdf`, visited on November 2020.

[18] Davis, L. A., "First Stage Recovery," *Engineering*, Vol. 2, No. 2, 2016, pp. 152–153.

[19] Gravlee, M., Kutter, B., Zegler, F., Mosley, B., and Haggard, R. A., "Partial rocket reuse using mid-air recovery," *Space 2008 Conference*, , No. September, 2008, pp. 1–11.

[20] Dek, C., Overkamp, J. L., Toeter, A., Hoppenbrouwer, T., Slimmens, J., van Zijl, J., Areso Rossi, P., Machado, R., Hereijgers, S., Kilic, V., and Naeije, M., "A recovery system for the key components of the first stage of a heavy launch vehicle," *Aerospace Science and Technology*, Vol. 100, 2020, pp. 105778.

[21] Snijders, M., "Cost effectiveness of the first stage recovery of a small satellite launcher," 2017.

[22] Pepermans, L., "Development of a Multidisciplinary Design Optimisation Tool to Determine the Feasibility of Upper Stage Reusability," 2019.

[23] Price, H., Manning, R., Sklyanskiy, E., and Braun, R., "A high-heritage blunt-body entry, descent, and landing concept for human Mars exploration," *54th AIAA Aerospace Sciences Meeting*, Vol. 0, No. January, 2016, pp. 1–16.

[24] Antonenko, S. V. and Belavskiy, S. A., "Mid-Air Retrieval technology for returning of reusable launch vehicles' boosters," 2009.

[25] Dirkx, D. and Mooij, E., *AE4866 – Propagation and Optimization Lecture notes*, Faculty of Aerospace Engineering, Delft University of Technology, 2019.

[26] Tudat, "5.1 Frame/State Transformations," `https://tudat.tudelft.nl/tutorials/tudatFeatures/astroTools/framestateTransformations.html`.

[27] Vallado, D. A., "Fundamentals of Astrodynamics and Applications," 1997.

[28] Hintz, G. R., "Survey of Orbit Element Sets," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 3, 2008, pp. 785–790.

[29] Vittaldev, V., Mooij, E., and Naeije, M., "Unified State Model theory and application in Astrodynamics," *Celestial Mechanics and Dynamical Astronomy*, Vol. 112, 2012, pp. 253–282.

[30] Van Kesteren, M., "Air Launch versus Ground Launch: A Multidisciplinary Design Optimization Study of Expendable Launch Vehicles on Cost and Performance," 2013.

[31] Rozemeijer, M., "High Altitude Platform assisted launching," .

[32] Fehlberg, E., "Classical Fifth-, Sixth-, Seventh-, and Eighth-Order Runge-Kutta Formulas with Stepsize Control," *National Aeronautics and Space Administration*, 10 1968.

[33] Rosema, C., Doyle, J., Auman, L., Underwood, M., and Blake, W. B., "AFRL-RB-WP-TR-2011-3071 Missile DATCOM User's Manual - 2011 Revision." Vol. 3071, 2011.

[34] DIKBAS, E., "Design of a Grid Fin Aerodynamic Control Device for Transonic Flight Regime," , No. June, 2015, pp. 104.

[35] Theerthamalai, P., "Aerodynamic characterization of grid fins at subsonic speeds," *Journal of Aircraft*, Vol. 44, No. 2, 2007, pp. 694–698.

[36] Peng, K., Hu, F., Wang, D., Okolo, P. N., Xiang, M., Bennett, G. J., and Zhang, W., "Grid fins shape design of a launch vehicle based on sequential approximation optimization," *Advances in Space Research*, Vol. 62, No. 7, 2018, pp. 1863–1878.

[37] Burkhalter, J. E. and Frank, H. M., "Grid fin aerodynamics for missile applications in subsonic flow," *Journal of Spacecraft and Rockets*, Vol. 33, No. 1, 1996, pp. 38–44.

[38] Burkhalter, J. E., "Grid fins for missile applications in supersonic flow," *34th Aerospace Sciences Meeting and Exhibit*, , No. January, 1996.

[39] Theerthamalai, P. and Nagarathinam, M., "Aerodynamic analysis of grid-fin configurations at supersonic speeds," *Journal of Spacecraft and Rockets*, Vol. 43, No. 4, 2006, pp. 750–756.

[40] Tudat, "1 Environment set-up," `https://tudat.tudelft.nl/tutorials/tudatFeatures/environmentSetup/index.html`.

[41] Vandamme, J., "Assisted-Launch Performance Analysis Using Trajectory and Vehicle Optimization," 2012, pp. 150.

[42] Castellini, F., "Multidisciplinary Design Optimization For Expendable Launch Vehicles," 2012, pp. 179.

[43] Picone, J., Hedin, A. E., Drob, D., and Aikin, A., "NRLMSISE-00 empirical model of the atmosphere: Statistical comparison and scientific issues," *Journal of Geophysical Research*, Vol. 107, 2002.

[44] Boere, M., "Optimization of Descent Trajectories," 2010.

[45] Kozai, Y., "New Determination of Zonal Harmonics Coefficients of the Earth's Gravitational Potential," , Vol. 16, Jan. 1964, pp. 263.

[46] Hsu, D. Y., "Comparison of four gravity models," *Proceedings of Position, Location and Navigation Symposium - PLANS '96*, 1996, pp. 631–635.

[47] ASTOS, "ASTOS solutions," `https://www.astos.de/resources`, visited on November 2020.

[48] Patterson, M. A. and Rao, A. V., "GPOPS-II: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using Hp-Adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming," *ACM Trans. Math. Softw.*, Vol. 41, No. 1, 2014.

[49] Tudat, "Start page," `https://tudat.tudelft.nl/`, visited in November 2020.

[50] Ma, L., Wang, K., Chen, Y., Shao, Z., Song, Z., and Biegler, L. T., "Six-degree-of-freedom trajectory optimization for powered landing of reusable rockets," , No. August, 2019.

[51] Reynolds, T. P., Szmuk, M., Malyuta, D., Mesbahi, M., Acikmese, B., and Carson, J. M., "Dual Quaternion Based Powered Descent Guidance with State-Triggered Constraints," , No. April, 2019.

[52] Gunnell, E., Mansfield, E., and Rodriguez, D., "Powered Descent and Landing of an Orbital-Class Rocket," 2019.

[53] Moler, C., *Numerical Computing with MATLAB*, 2022, `hhttps://www.mathworks.com/matlabcentral/fileexchange/37976-numerical-computing-with-matlab`, MATLAB Central File Exchange. Retrieved May 1, 2022.

[54] Bradford, J. and Germain, B. S., "Rocket Back Trajectory Sensitivity Analyses for a Reusable Booster System," *AIAA SPACE 2010 Conference \&amp; Exposition*.

[55] Mallacof, L. M. R., "STRATEGY FOR PITCH-OVER MANEUVER CONTROL," 1999.

[56] Simplicio, P., Marcos, A., and Bennani, S., "A Reusable Launcher Benchmark with Advanced Recovery Guidance," April 2019, 5th CEAS Conference on Guidance, Navigation amp; Control, EuroGNC19 ; Conference date: 03-04-2019 Through 05-04-2019.

[57] Ecker, T., Karl, S., Dumont, E., Stappert, S., and Krause, D., "A numerical study on the thermal loads during a supersonic rocket retro-propulsion maneuver," *53rd AIAA/SAE/ASEE Joint Propulsion Conference, 2017*, , No. September, 2017.

[58] Wilken, J., Stappert, S., Bussler, L., Sippel, M., and Dumont, E., "FUTURE EUROPEAN REUSABLE BOOSTER STAGES: EVALUATION OF VTHL AND VTVL RETURN METHODS," *69th International Astronautical Congress (IAC)*, September 2018.

[59] Sippel, M., Stappert, S., Bussler, L., and Dumont, E., "Systematic assessment of reusable first-stage return options," *Proceedings of the International Astronautical Congress, IAC*, Vol. 15, 2017, pp. 9919–9930.

[60] Space Exploration Technologies Corp, "Falcon User' s Guide," , No. April, 2020, pp. 72.

[61] Pagano, A. and Mooij, E., "Global launcher trajectory optimization for lunar base settlement," *AIAA/AAS Astrodynamics Specialist Conference 2010*, 2010.

[62] Pontani, M. and Cecchetti, G., "Ascent Trajectories of Multistage Launch Vehicles: Numerical Optimization with Second-Order Conditions Verification," *ISRN Operations Research*, Vol. 2013, Dec. 2013, pp. 498765.

[63] Roshanian, J., Bataleblu, A. A., and Ebrahimi, M., "Robust ascent trajectory design and optimization of a typical launch vehicle," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, Vol. 232, No. 24, 2018, pp. 4601–4614.

[64] Zhou, H., Wang, X., and Cui, N., "Ascent trajectory optimization for air-breathing vehicles in consideration of launch window," *Optimal Control Applications and Methods*, Vol. 41, No. 2, 2020, pp. 349–368.

[65] Rahimi, A., Kumar, K. D., and Alighanbari, H., "Engineering notes :Particle swarm optimization applied to spacecraft reentry trajectory," *Journal of Guidance, Control, and Dynamics*, Vol. 36, No. 1, 2013, pp. 307–310.

[66] De Ridder, S. and Mooij, E., "Optimal longitudinal trajectories for reusable space vehicles in the terminal area," *Journal of Spacecraft and Rockets*, Vol. 48, No. 4, 2011, pp. 642–653.

[67] Ghosh, P. and Conway, B. A., "Numerical trajectory optimization with swarm intelligence and dynamic assignment of solution structure," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 4, 2012, pp. 1178–1191.

[68] SUBASHINI, G. and BHUVANESWARI, M., "Comparison of multi-objective evolutionary approaches for task scheduling in distributed computing systems," *Sadhana*, Vol. 37, 12 2013.

[69] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 2, 2002, pp. 182–197.

[70] Zhang, Q. and Li, H., "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition," *IEEE Transactions on Evolutionary Computation*, Vol. 11, No. 6, Dec 2007, pp. 712–731.

[71] Tan, Yanyan, L. X. L. Y. W. Q. Z. H., "Decomposition-Based Multiobjective Optimization with Invasive Weed Colonies," *Hindawi*, Vol. 2019, No. 4, 2019.

[72] Vinh Ho-Huu, Sander Hartjes, H. G. V. R. C., "An Efficient Application of the MOEA/D Algorithm for Designing Noise Abatement Departure Trajectories," *MDPI*, 2017.

[73] Storn, R. and Price, K., "Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces," *Journal of Global Optimization*, Vol. 11, No. 4, 1997, pp. 341–359.

[74] Poli, R., Kennedy, J., and Blackwell, T., "Particle swarm optimization," *Swarm Intelligence*, Vol. 1, No. 1, 2007, pp. 33–57.

[75] Ridolfi, G., Mooij, E., and Corpino, S., "Complex-Systems Design Methodology for Systems-Engineering Collaborative Environment," *Systems Engineering*, edited by B. Cogan, chap. 2, IntechOpen, Rijeka, 2012.

[76] Saltelli, A., Tarantola, S., and Chan, K. P., "A quantitative model-independent method for global sensitivity analysis of model output," *Technometrics*, Vol. 41, No. 1, 1999, pp. 39–56.

[77] Seaholm, S. K., Ackerman, E., and Wu, S. C., "Latin hypercube sampling and the sensitivity analysis of a Monte Carlo epidemic model," *International Journal of Bio-Medical Computing*, Vol. 23, No. 1-2, 1988, pp. 97–112.

[78] Davidian, Kenneth J., "A detailed description of the uncertainty analysis for High Area Ratio Rocket Nozzle tests at the NASA Lewis Research Center," 1987.