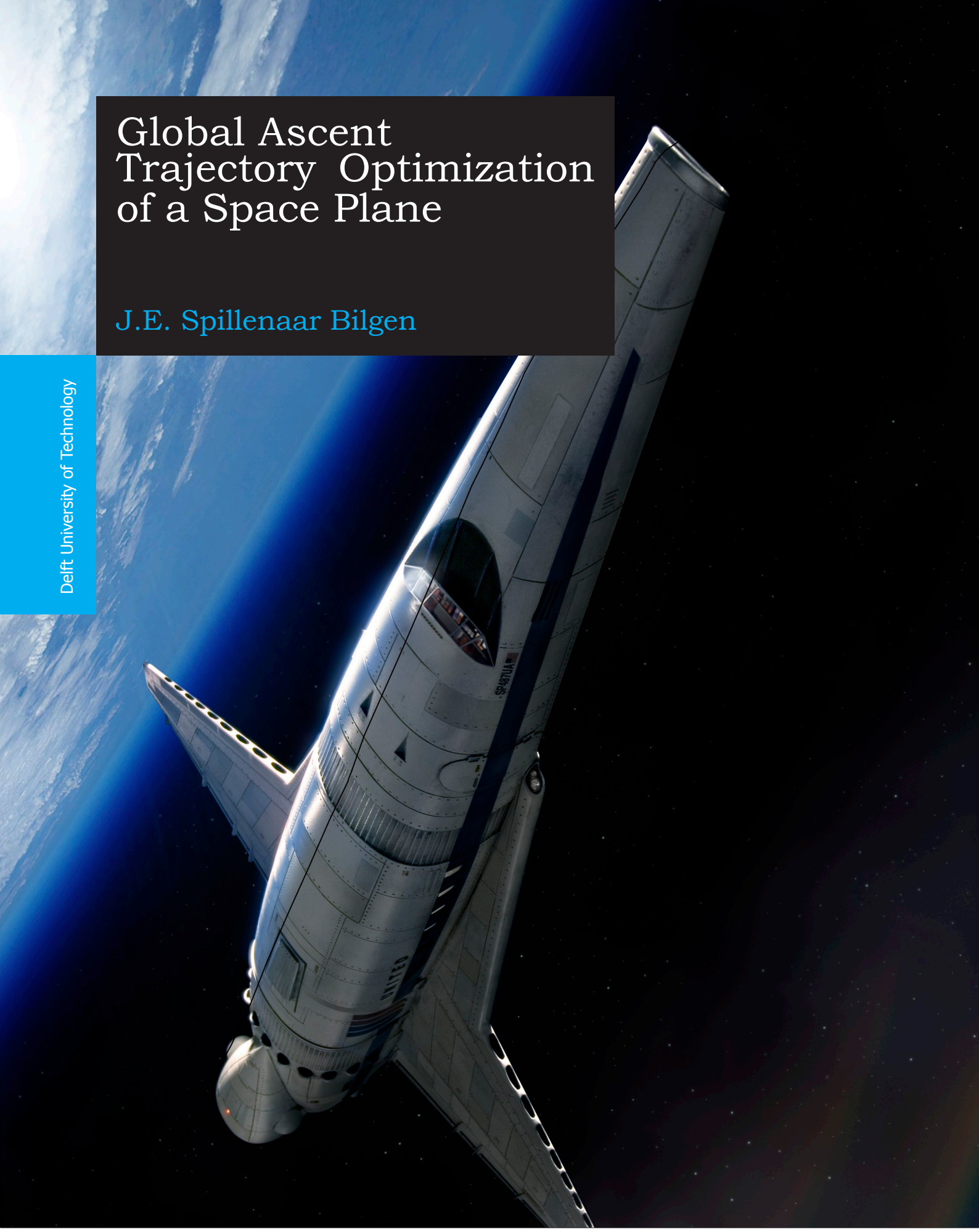
Global Ascent Trajectory Optimization of a Space Plane

J.E. Spillenaar Bilgen

Delft University of Technology



TU Delft Delft University of Technology

Challenge the future

Global Ascent Trajectory Optimization of a Space Plane

by

J.E. Spillenaar Bilgen

in partial fulfillment of the requirements for the degree of

Master of Science
in Applied Physics

at Delft University of Technology,
to be defended publicly on Friday October 27, 2017 at 09:30 AM.

Supervisor:	dr. ir. E. Mooij	
Thesis committee:	dr. D.M. Stam	TU Delft
	dr. ir. M. Snellen	TU Delft
	dr. ir. D. Dirkx	TU Delft

Cover image credit: http://hubpages.com/education/What-Ever-Happened-to-the-Space-Plane

TU Delft Delft University of Technology

Preface

When I started my Masters degree in Aerospace Engineering I did not even know what a space-plane was. It was only after I started studying the literature that I realized how much research has actually been done on this topic. Studying space-planes has shown me how many questions are still unanswered and it has opened my eyes to a world of opportunities.

I would like to take this opportunity to thank my supervisor, Erwin Mooij, for his guidance throughout this process. His lectures on space-planes inspired me to start my research on this topic, and his advice and guidance kept me going to the end.

A special thanks goes to Dominic Dirkx who has helped me countless times improve and solve problems with my software. Finally I would like to thank my parents, my housemates and my girlfriend who have helped me to stay focused and were always there when I needed someone to talk to.

J.E. Spillenaar Bilgen
Delft, October 2017

Abstract

With the increasing number of space missions every year, reusable launch vehicles (RLV) are thought to be more cost effective in the long run than the traditional expendable launchers. Over the last decades many projects were proposed and cancelled again. The enormous number of projects that have been cancelled before completion, shows that designing a reusable launcher is not an easy task.

The development costs of fully reusable systems are high, since not all required knowledge and technology is available yet. The high initial investment will pay off if the vehicle has a short turn-around time and can be launched frequently. The upcoming of space tourism in the last years has opened a new segment on the launch market. Furthermore, the possibility of hypersonic passenger transport is also being studied. Especially for these purposes, reusable launchers with a fast turn-around time are essential for success. The addition of these newcomers on the launch market will make the development of a true reusable launch vehicle more feasible in the near future, which makes RLVs an interesting topic for further research.

As with any launch system, the primary goal of an RLV is to bring as much payload as possible to its target destination. The enormous amount of energy that is required to bring a vehicle into orbit, means that only a small fraction of the total mass of the system is available for payload. As a result, the cost of bringing a kilogram of payload into orbit is very high as well. We therefore strive to gain every kilogram possible. This has led to the following research question:

> *How can an ascent trajectory of an HTHL SSTO launch vehicle to ISS be found that maximizes the payload capacity for a given set of flight-path and control constraints?*

where the Horizontal Take-off Horizontal Landing (HTHL) Single Stage To Orbit (SSTO) Winged Cone Configuration (WCC) is selected for this research based on its promising turn around time, fuel efficiency and safety, and the availability of aerodynamic and propulsive models. The International Space Station (ISS) is selected as a suitable destination as it is currently the most frequently visited place in space.

In this research the focus will be on optimizing the payload capacity by flying a minimum fuel and heat load trajectory. Every kilogram that can be saved on the fuel consumption and the Thermal Protection System (TPS) can be directly used for payload. Multi-objective (MO) global optimizers are identified as a promising method to find an optimal solution to the ascent trajectory optimization problem. Many global optimization methods exist and both the ascent trajectory problem and the optimization can be defined in numerous ways. The objective of this research is therefore to find a general good method and approach to optimize the ascent trajectory.

To achieve this objective and find an answer to the research question a simulation model is set up. This simulation model propagates the ascent trajectory based on a guided angle of attack and throttle setting as a function of the normalized energy state of the vehicle. The guidance law is defined at a number of control nodes and is stored in a decision vector. This vector is randomly initialized and subsequently optimized with the different optimization algorithms. The performance of different optimization methods and problem settings is assessed based on the convergence of the optimized populations with respect to a set of evaluation objectives.

Four optimization methods are used: MOEA/D, NSGA-II, NSGA-II-tabu and NSGA-II-tabu-reintro, with each using default settings. Of the optimization methods compared in this research, MOEA/D performs best in terms of consistency of the results and convergence speed. Some initial tests are also performed with a parallel implementation of MOEA/D and NSGA-II-tabu, which show promising results.

Different control parameters are included in the decision vector. The thrust is guided by either setting the equivalence ratio or the throttle factor as guidance parameters. Thrust was also computed by a throttle law effectively excluding it from the guidance and thus the set of optimization parameters. Finally thrust vector control (TVC) was included, allowing the direction of thrust to be deflected up and down.

Settings are tested for different numbers of control nodes. The size of the problem dimension is found to be important for the optimization performance. The throttle law is effective in that sense as it reduces the dimension by eliminating thrust guidance from the decision vector. The reduced problem dimension allows for inclusion of TVC, which was found to have a positive effect on the optimization performance.

The ascent trajectory is optimized with respect to different sets of objectives. The different objectives are conflicting for the ascent trajectory leading to a number of complications. Premature convergence to non-flying solutions is found to be a problem specific to the ascent trajectory. A no-flight penalty function is included to the constraint objective function and helps initial convergence of the population.

A number of optimization approaches is then tested to find if different objectives can be optimized in separate stage. Given the mutual conflict of interest between objectives, it is found that the final stage of the optimization should include all objectives that need to be optimized. Defining a separate function for each objective is most robust as the performance of objective functions that combine multiple objectives are very dependent on the weights that are applied to each objective.

The best found problem settings, optimization method and approach are finally combined to optimize the ascent trajectory from take-off to an altitude of ISS at orbital velocity. Optimizing the complete trajectory as single problem is found to be impossible with the current implementation. The complete trajectory is therefore split in three phases: take-off, atmospheric acceleration and pull-up. Separating the different flight phases allows for more confined domains of the optimization parameter for each phase. This resulted in a considerably reduced search space of each of the flight phases.

The separate phases can be optimized with the current implementation. Both the take-off and pull-up trajectories comply to all constraints. For the atmospheric acceleration phase the optimizer struggles to find a trajectory without constraint violations. The best results are found when the throttle law is used and 50% of the trim was managed by TVC. For this setup only the axial acceleration constraint was violated by just 1% of its objective value. Combining these separate phases results in a complete ascent trajectory capable of bringing 401 kg of payload to an altitude of the ISS at orbital velocity.

In conclusion, the ascent trajectory of an HTHL SSTO launch vehicle to ISS can be optimized with the MO global optimization method MOEA/D. However, the resulting pseudo optimal solution requires further local optimization to find the real optimum. Furthermore, the current implementation requires the trajectory to be evaluated in phases. Improvements on the implementation of the guidance, the objective functions and the optimization settings are required to be able to effectively optimize the trajectory as a whole.

Contents

List of Acronyms

ACO	Ant Colony Optimization
APAS	Aerodynamic Preliminary Analysis System
ASA	Adaptive Simulated Annealing
ASTOS	AeroSpace Trajectory Optimization Software
c.g.	center of gravity
c.o.m.	center of mass
c.o.t.	center of thrust
DE	Differential Evolution
DGMOPSO	Dual-Grid Multi-Objective Particle Swarm Optimization
DOF	Degrees Of Freedom
EA	Evolutionary Algorithms
EOM	Equations of Motion
EP	Evolutionary Programming
ESA	European Space Agency
FESTIP	Future European Space Transportation Investigations Program
FLOP	Future Launcher Optimization Program
FLPP	Future Launchers Preparatory Program
GA	Genetic Algorithm
GP	Genetic Programming
GESOP	Graphical Environment for Simulation and Optimization
GRAM	Global Reference Atmosphere Model
HTHL	Horizontal Take-off Horizontal Landing
HTO	Horizontal Take-off
ISS	International Space Station
LEO	Low Earth Orbit
MO	Multi-Objective
MOEA/D	Multi-Objective Evolutionary Algorithm based on Decomposition
MPSO	Multiple Particle Swarm Optimization
m.r.c.	moment reference center
MSIS	Mass Spectrometer and Incoherent Scatter
NASA	National Aeronautics and Space Administration
NASP	National Aero-Space Plane
NRLMSISE	Naval Research Laboratory Mass Spectrometer and Incoherent Scatter radar Extended from the ground to the upper atmosphere
NSGA-II	Non-dominated Sorting Genetic Algorithm 2
PaGMO	Parallel Global Multiobjective Optimizer
POST	Program to Optimize Simulated Trajectories
PSO	Particle Swarm Optimization
RK	Runge-Kutta
RKF	Runge-Kutta-Fehlberg
RLV	Reusable Launch Vehicle
SA	Simulated Annealing
SBX	Simulated Binary Crossover
SO	Single Objective
SSTO	Single Stage To Orbit
SSTSO	Single Stage To Sub-Orbit
TPS	Thermal Protection System
TRL	Technology Readiness Level
TSTO	Two Stage To Orbit
Tudat	TU Delft Astrodynamics Toolbox

TVC	Thrust Vector Control
US76	United States Standard Atmosphere 1976
VTHL	Vertical Take-off Horizontal Landing
VTO	Vertical Take-off
VTVL	Vertical Take-off Vertical Landing
WCC	Winged-Cone-Configuration

List of Symbols

Latin symbols

$\boldsymbol{A}$	State coefficient matrix [-]
b	Span of the wings [m]
$\boldsymbol{B}$	Control coefficient matrix [-]
c	Mean aerodynamic chord [m]
c	Child individuals [-]
c_E	Energy state cost function [-]
c_{fm}	Fuel mass cost function [-]
c_Q	Heat load cost function [-]
c_{eff}	Effective exhaust velocity [m s^{-1}]
cr	Crossover probability [-]
$\boldsymbol{C}$	Rotation matrix [-]
C_D	Drag coefficient [-]
$C_{D,f}$	Basic vehicle fuselage drag increment coefficient [-]
C_L	Lift coefficient [-]
$C_{L,f}$	Basic vehicle fuselage lift increment coefficient [-]
C_m	Pitching moment coefficient [-]
$C_{m,f}$	Basic vehicle fuselage pitch moment increment coefficient [-]
$dC_{m,c}$	Canard pitch moment increment coefficient [-]
$dC_{m,el}$	Elevon pitch moment increment coefficient [-]
$dC_{m,r}$	Rudder pitch moment increment coefficient [-]
C_T	Thrust coefficient [-]
$dC_{D,c}$	Canard drag increment coefficient [-]
$dC_{D,el}$	Elevon drag increment coefficient [-]
$dC_{D,r}$	Rudder drag increment coefficient [-]
$dC_{L,c}$	Canard lift increment coefficient [-]
$dC_{L,el}$	Elevon lift increment coefficient [-]
$dC_{m,c}$	Canard pitch moment increment coefficient [-]
$dC_{m,el}$	Elevon pitch moment increment coefficient [-]
$dC_{m,r}$	Rudder pitch moment increment coefficient [-]
D	Drag force [N]
D	Problem dimension [-]
e	Error [-]
e	Ellipticity of Earth [-]
$\boldsymbol{f}$	Objective function vector [-]
E	Energy state [J kg^{-1}]
$\hat{E}$	Normalized energy state [-]
EP	External population [-]
F	Differential evolution factor [-]
$\boldsymbol{F}$	External force vector [N]
F_T	Thrust scaling factor [-]
F_{TVC}	TVC trim fraction [-]
g	Gravitational acceleration [m s^{-2}]
$\boldsymbol{g}$	Constraint vector [-]
G	Number of generations [-]
h	Altitude above the surface of the Earth [m]
h	Integration step size [s]
H_s	Scale height [m]
i_{dist}	Crowding distance [-]
i_{rank}	Non-dominated rank [-]

I_{sp}	Specific impulse [s]
I_{xx}, I_{yy}, I_{zz}	Moment of inertia around the body x-,y- and z-axis [kg m^2]
J_2	Gravity zonal harmonic coefficient [-]
K	Gain [-]
lb	Lower bound [-]
L	Lift force [N]
L_{ref}	Vehicle reference length [m]
m	Mutation probability [-]
m	Number of objective functions [-]
m	Number of optimization parameters [-]
m	Mass of the vehicle [kg]
m_0	Mass of the vehicle at take-off [kg]
m_{circ}	Fuel mass required for circularization [kg]
m_{con}	Fuel mass consumed by the air-breathing engine during ascent [kg]
m_{dry}	Dry vehicle mass [kg]
m_f	Final vehicle mass before circularization [kg]
m_{fuel}	Total fuel mass on board at take-off [kg]
$m_{penalty}$	fuel mass penalty [kg]
m_{pl}	Payload mass [kg]
m_{TPS}	Mass of the thermal protection system [kg]
$\dot{m}$	Fuel mass rate [kg s^{-1}]
M	Mach number [-]
$\bar{M}$	Pitching moment about c.g. [Nm]
$\bar{M}_{mrc}$	Pitching moment about m.r.c. [Nm]
$\bar{M}_{TVC}$	Pitching moment caused by TVC [Nm]
n	Population size [-]
n	Number of control nodes [-]
n	Number of constraint violations [-]
n	Number of guidance evaluations [-]
n_a	Axial load [m s^{-2}]
$n_{a,c}$	Axial load constraint [m s^{-2}]
n_r	Neighborhood replacement constant [-]
N	Total density number [m^{-3}]
N_A	Avogadro's constant [mol^{-1}]
$\boldsymbol{O}$	Objective function vector [-]
$\boldsymbol{O_f}$	Feasibility objective function vector [-]
$\boldsymbol{O_o}$	Optimality objective function vector [-]
p	Atmospheric pressure [N m^{-2}]
p	Selection constant [-]
p	Parent individual [-]
$p_{E_{mon}}$	Penalty function for monotonic energy state [-]
p_{fm}	Penalty function for fuel mass [-]
p_{n_a}	Penalty function for axial load [-]
p_{nf}	Penalty function for non-flying trajectories [-]
$p_{\bar{q}}$	Penalty function for dynamic pressure [-]
$p_{\dot{Q}}$	Penalty function for heat rate [-]
p_γ	Penalty function for oscillating flight-path angle [-]
P	Penalty function [-]
P	Period [s]
$\bar{q}$	Dynamic pressure [N m^{-2}]
$\bar{q}_c$	Dynamic pressure constraint [N m^{-2}]
Q	Integrated heat load [J m^{-2}]
$\dot{Q}$	Heat rate [W m^{-2}]
$\dot{Q}_c$	Heat rate constraint [W m^{-2}]
r	Distance from the center of the Earth to the c.o.m. of the vehicle [m]
$\boldsymbol{r}$	Position vector [m]

R_E	Mean equatorial radius of Earth [m]
R_S	Radius of Earth at the surface [m]
R_P	Mean polar radius of Earth [m]
R_n	Nose or leading edge radius [m]
R^*	Universal gas constant [J mol^{-1} K^{-1}]
S_{ref}	Reference area [m^2]
sc_T	Thrust scaling factor [-]
t	Time [s]
T	Temperature [K]
T	Thrust [N]
T	Size of the neighborhood [-]
T_M	Molecular scale temperature [K]
$\boldsymbol{u}$	Control vector [-]
$\boldsymbol{u}$	Trial vector [-]
ub	Upper bound [-]
$\boldsymbol{v}$	Mutant vector [-]
V	Velocity [m s^{-1}]
$\boldsymbol{V}$	Velocity vector [m s^{-1}]
$\dot{\boldsymbol{V}}$	Acceleration vector [m s^{-2}]
ΔV	Velocity increment for circularization of the trajectory [m s^{-1}]
$\boldsymbol{x}$	State vector [-]
$\boldsymbol{x}$	Optimization parameter vector [-]
x_{com}	Distance between the m.r.c. and the c.o.m of the vehicle [m]
x_{cot}	Distance between c.o.t. and the m.r.c. [m]
x_v	Decision vector [-]
x, y, z	Cartesian components of position [m]
X, Y, Z	Axes [-]
$\dot{x}, \dot{y}, \dot{z}$	Cartesian components of velocity [m s^{-1}]
Y	Side force [N]
z	Modulus [-]
$\boldsymbol{z}$	Reference objective vector [-]

Greek symbols

α	Angle of attack [rad]
β	Sideslip angle [rad]
γ	Flight-path angle [rad]
δ	Geocentric latitude [rad]
δ	Deflection angle [rad]
ϵ	Local truncation error [-]
ϵ_{tol}	Integration tolerance [-]
ϵ_T	Thrust-elevation angle [rad]
ζ	Damping ratio [-]
η_c	Crossover distribution constant [-]
η_m	Mutation distribution constant [-]
θ	Phase shift [rad]
$\boldsymbol{\lambda}$	Weight vector [-]
$\boldsymbol{\lambda}$	Vector with eigenvalues [-]
μ	Gravitational parameter of Earth [m^3 s^{-2}]
$\boldsymbol{\mu}$	Eigenvector [-]
ρ	Atmospheric density [kg m^{-3}]
σ	Bank angle [rad]
ϕ	Fuel-to-air equivalence ratio [-]
τ	Geocentric Longitude [rad]
χ	Heading angle [rad]
ψ_T	Thrust-azimuth angle [rad]
ω_{cb}	Rotational rate of Earth [rad s^{-1}]

ω_n	Natural frequency [rad s^{-1}]
$\boldsymbol{\omega}$	Rotation vector [rad s^{-1}]

Indices

0	Initial condition
0	Sea level
a	Basic vehicle
A	Aerodynamic
c	Canards
c	Constraint
c	Commanded
c	Crossover
cb	Central body
cg	Center of gravity
$circ$	Circularization
con	Consumed fuel
cot	Center of thrust
C	Coriolis
d	Derivative
D	Drag
e	Outgoing
el	Elevons
E	Equatorial radius
f	Final
f	Feasible
$fuel$	Fuel
g	Guidance
G	Gravity
i	Integral
in	Incoming
int	Integration
L	Lift
m	Mutation
m	Pitch
max	Maximum
mrc	Moment reference center
o	Optimal
p	Proportional
P	Polar radius
pl	Payload
r	Rudder
ref	Reference
rel	Relative
S	Surface
tol	Tolerance
T	Thrust
TVC	Thrust vector control
α	Angle of attack
γ	Flight-path angle

Reference Frames

A	Aerodynamic frame
B	Body frame
I	Inertial frame
P	Propulsion frame
R	Rotating frame

T	Trajectory frame
V	Vertical frame

1

Introduction

Reusable launch vehicles (RLV) have been studied for decades. Even before the first mission to space was a fact, plans were made to build hypersonic planes that could travel into space (Sänger and Bredt, 1944). Initially most of these projects were driven by military purposes. Nowadays, most missions have a scientific or commercial goal.

With the increasing number of space missions every year, reusable launchers were thought to be more cost effective in the long run than the traditional expendable launchers. Many projects were proposed and cancelled again. This led eventually to the development of the semi-reusable Space Shuttle that transported astronauts to and from the International Space Station (ISS) for more than 30 years (Darling, 2003).

The large number of projects that have been cancelled before completion, shows that designing a reusable launcher is not an easy task. Even the Space Shuttle is seen by many people as a failure, because it did not meet the requirements of easy, cheap and frequent access to space[1]. Moreover, the system was not fully reusable.

The development costs of fully reusable systems are high, since not all required knowledge and technology is available yet. The high initial investment will pay off if the vehicle has a short turn-around time and can be launched frequently, but with the constant launch market over the last years these investments have not been feasible yet (Koelle and Janovsky, 2007).

However, the upcoming of space tourism[2,3] in the last years has opened a new segment on the launch market. Furthermore, the possibility of hypersonic passenger transport is also being studied (Sippel, 2015). Especially for these purposes, reusable launchers with a fast turn-around time are essential for success. The addition of these newcomers on the launch market will make the development of a true reusable launch vehicle more feasible in the near future, which makes RLVs an interesting topic for further research.

As with any launch system, the primary goal of an RLV is to bring as much payload as possible to its target destination. The enormous amount of energy that is required to bring a vehicle into orbit, means that only a small fraction of the total mass of the system is available for payload. As a result, the cost of bringing a kilogram of payload into orbit is very high as well. We therefore strive to gain every kilogram possible. This has led to the following research question:

> *How can an ascent trajectory of an HTHL SSTO launch vehicle to ISS be found that maximizes the payload capacity for a given set of flight-path and control constraints?*

Initially a Vertical Take-off Horizontal Landing (VTHL) Single Stage To Orbit (SSTO) RLV was found to be the most promising configuration in terms of turn around time, fuel efficiency and safety. However, given the similarity in the ascent trajectory, the Horizontal Take-off Horizontal Landing (HTHL) SSTO Winged Cone Configuration (WCC) was eventually selected for this research based on availability of aerodynamic and propulsive models.

[1] Torrance, P. (2010). The real mistakes of the space shuttle program. `http://www.thespacereview.com/article/1694/1` [Accessed: 8 March 2017]

[2] Galactic, V. (2016). Who we are. `http://www.virgingalactic.com/who-we-are/` [Accessed: 15 March 2017]

[3] XCOR (2016). Our hero: XCOR lynx. `spaceexpeditions.xcor.com/spacecraft/` [Accessed: 15 March 2017]

The main driver behind the development of RLVs is to have cheaper and more frequent access to space. The International Space Station (ISS) currently is the most frequently visited place in space. For that reason ISS was identified as a suitable objective destination for the ascent trajectory.

Different ascent trajectories have been identified for the WCC in earlier research performed by Powell et al. (1991), Lu (1991), Van Buren and Mease (1991), Hattis et al. (1991) and Mooij (1998). A typical take-off trajectory starts with a relatively steep pull-up with large flight-path angles γ of around 25°. Afterwards the flight levels out to comply with the upper bound constraints. For maximum efficiency of the air-breathing engine the vehicle will then follow the lower bound constraints of maximum dynamic pressure, heat rate and axial load while accelerating through the atmosphere. When enough velocity is gained a final pull-up will be performed to reach the final objective altitude of ISS.

In this research the focus will be on optimizing the payload capacity by flying a minimum fuel and heat-load trajectory. Every kilogram that can be saved on the fuel consumption and the Thermal Protection System (TPS) can be directly used for payload. Many trajectory optimization software programs exist, but the use of global optimization methods for trajectory optimization has gained a lot of attention over the last two decades. Global optimizers are easy to apply and are not dependent on a good initial guess of the trajectory. Different methods have been compared for interplanetary trajectory optimization (Castellini, 2008; Izzo, 2006), however, less research has been found on the application for ascent trajectory optimization. Since this trajectory has a different set of constraints and controls, the performance of the optimization methods might be different. This introduces an opportunity for further research.

Within the group of global optimization methods specifically the Multi-Objective (MO) methods are selected for this research. The ascent trajectory optimization problem has a number of conflicting objectives. Combining multiple objectives into a single objective function by applying weighted penalty functions is very problem specific and relies on trial and error (Parsopoulos and Vrahatis, 2002; Yang et al., 1997). MO global optimizers can handle different objectives in separate functions and are therefore thought to be better suited to handle the ascent trajectory optimization problem.

According to Wolpert and Macready (1997) the performance of every optimization method is different for each problem. Different optimization methods are therefore compared. MOEA/D (Zhang and Li, 2007) and NSGA-II (Deb et al., 2000) were identified as promising methods. Two implementations of NSGA-II with a tabu-list were also included (Tan et al., 2003), resulting in a total of four optimization algorithms for comparison. On the other hand, the definition of the ascent trajectory problem can also be changed to affect the performance of each optimizer. Taking the above into account, the following set of sub-questions is set up for this research:

SQ-1 What is the impact of different control parameters on the optimization process of the ascent trajectory?

SQ-2 Which MO global optimization method is most effective for the ascent trajectory optimization problem?

SQ-3 What is the best approach, in terms of handling constraints, objectives and different flight phases, to optimize the ascent trajectory of the WCC from take-off to circularized orbit?

To find the answers to the research questions a simulation model is set up. This simulation model propagates the ascent trajectory based on a guided angle of attack and throttle setting as a function of the normalized energy state of the vehicle. The guidance law is defined at a number of control nodes and is stored in a decision vector. This vector is randomly initialized and subsequently optimized with the different optimization algorithms.

Different control parameters can be included in the decision vector. The thrust can be guided by setting the equivalence ratio or the throttle factor as guidance parameters. The thrust can also be computed by a throttle law. Thrust is then computed by a numerical guidance law effectively excluding it from the set of optimization parameters. Finally thrust-vector control (TVC) can be included, allowing the direction of thrust to be deflected up and down.

The performance of different optimization methods and problem settings is assessed based on the convergence of the optimized populations with respect to a set of evaluation objectives. The best found problem settings, optimization method and approach are combined to optimize the final trajectory from take-off to an altitude of ISS at orbital velocity.

This report explains the theoretical background as well as the methodology and results of the research introduced above. Chapter 2 will first discuss a number of relevant projects of the past, present and future, resulting in the final definition of the chosen vehicle and trajectory for optimization. Chapter 3 presents the environment and vehicle models that are used in the simulation. Chapter 4 discusses the flight dynamics, including the definition of reference frames, equations of motion and the implemented guidance. Chapter 5 will show a trade-off of different available optimization methods, followed by a detailed explanation of the selected algorithms. The implementation and verification of the simulation software is then explained in Chapter 6. Finally the research results are discussed in Chapter 7 and final conclusions and recommendations are summarized in Chapter 8.

2

Ascent Trajectory Optimization Problem

Space-planes have been studied since the 1930s. They are still being studied today and are seen as a valuable concept for space flight in the future. The ascent trajectory optimization is an important part of the mission design. The goal of this chapter is to find a relevant vehicle and trajectory that can be used to study the performance of global optimization of the ascent trajectory of a space-plane. To do this, Section 2.1 will first discuss concepts and projects that have been studied in the past, are being studied today and will be of interest in the future. In Section 2.2 the information gathered will be used to select the most interesting vehicle and corresponding trajectory for this research. Section 2.3 will then discuss some of the trajectory optimization methods that have been used for past and present missions. It will also give a short introduction as to why global optimization is used for this research. A more in depth discussion of global optimization will follow in Chapter 5. Finally Section 2.4 combines all information to define a complete ascent mission with a set of mission requirements that can be used for optimization.

2.1. Mission Heritage

In this section past, present and future RLV projects will be discussed. Different concepts with varying ascent trajectories have been developed over time. A well known configuration is the Single Stage To Orbit (SSTO), which uses one single stage to reach orbital velocities. On the other hand, Two Stage To Orbit (TSTO) configurations consist of a first stage for initial take-off and acceleration. Subsequently, the second stage will accelerate further to orbital velocities. Finally, some configurations exist of a single stage that reaches space, but not at orbital velocities. In that sense those configurations can be seen as a TSTO where the second stage is either missing, or included in the form of a payload with a kick-stage that can reach orbital velocity. In this report those configurations will be referred to as Single Stage To Sub-Orbit (SSTSO).

2.1.1. Past and Present Projects

Many space-plane concepts have been studied in the past. Most projects have been cancelled before production. Some already in early stages of the design, but a few even made it to test flights. Today many reusable launcher systems are still being studied and developed. Ultimately the use of fully reusable space-planes could reduce the launch cost significantly and allow for more frequent access to space. This has already led to the emergence of space tourism, a new application for reusable space-planes. This section will discuss some of the important RLV projects of the past and present including details on the trajectories and the optimization of these trajectories.

Silverbird Bomber

The Silverbird was a design in the late thirties for a liquid-propellant rocket-powered sub-orbital bomber by Eugen Sänger and Irene Bredt. The design was further developed during the second world war as a long-range bomber to bomb the United States (Sänger and Bredt, 1944). It was supposed to be

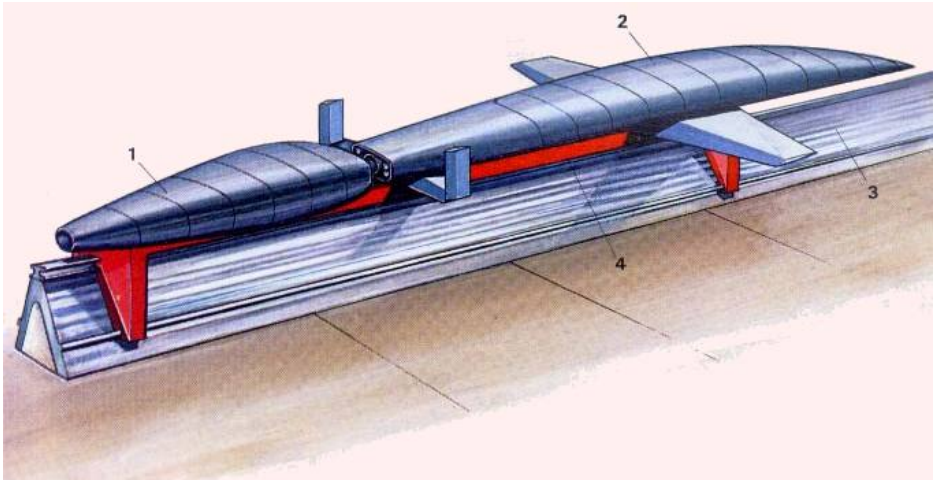


Figure 2.1: Silverbird.[1]

launched from a 3 km long rail track by a rocket-powered sled as shown in Figure 2.1. This sled would accelerate the vehicle to a take-off speed of 500 m/s. At this point its own rocket engine would be used to climb to a maximum altitude of approximately 150 km. The Silverbird would then re-enter the atmosphere using a skipping re-entry, which would extend its range far enough to drop a bomb on the American continent and continue flying to the Empire of Japan. The project was eventually cancelled, because it was too complex and expensive.

The Silverbird was the first concept for a SSTSO hypersonic vehicle. The design has since been very influential in the developments of later space-planes like the well known Space Shuttle (Hallion, 2005).

Space Shuttle

In 1968 the design of the Space Shuttle started to provide NASA with a reusable spacecraft able to carry astronauts to a permanent space station. It was first launched in 1981 and flew 135 mission until its retirement in 2013. The complete system consisted of 2 solid propellant reusable rocket boosters, an expendable external tank and the delta winged orbiter with its 3 main engines. The complete mission profile is shown in Figure 2.2. The Space Shuttle was launched vertically. The two solid rocket boosters burned out and jettisoned at approximately 125 seconds into the flight. Parachutes were used to slow both boosters to an impact velocity of 25 m/s. They could then be retrieved and refurbished for later use. The external tank was not reusable and after 8.5 minutes when it was burned out and jettisoned it would break up in the air and fall into the ocean. The main engines would give the orbiter its final boost into orbit and also powered it back to the Earth.(Darling, 2003).

The trajectory optimization of the Space shuttle for both the ascent and re-entry was done with the Program to Optimize Simulated Trajectories (POST). POST includes a number of direct gradient and shooting methods that will be further discussed in Section 2.3. Brauer et al. (1977) includes an extensive explanation of a trajectory optimization for the Space Shuttle. The optimization problem was defined as a constrained problem with 3 terminal equality constraints for the altitude (h_f = 92.6 km), the velocity (V_f = 7729 m/s) and the flight-path angle (γ_f = 0°) employed as penalty functions. Furthermore, a three-g acceleration limit was enforced after 60 seconds of flight. The set of control parameter to be optimized included the initial take-off mass, and the pitch rate at 8 control points. The objective of the optimization was to find values for the control parameters that maximized the payload mass while minimizing the error in the final state.

POST was able to find an optimal trajectory for the Space shuttle for both the ascent and the re-entry. Although the Space Shuttle completed many successful missions, it was not a fully reusable system. It also did not meet the desires of easy, cheap and frequent access to space[2].

[1]Source: `http://www.luft46.com/misc/sanger.html` [Accessed: 15 March 2017]
[2]Torrance, P. (2010). The real mistakes of the space shuttle program. `http://www.thespacereview.com/article/1694/1` [Accessed: 8 March 2017]
[3]Source: `https://en.wikipedia.org/wiki/Space_Shuttle` [Accessed: 15 March 2017]

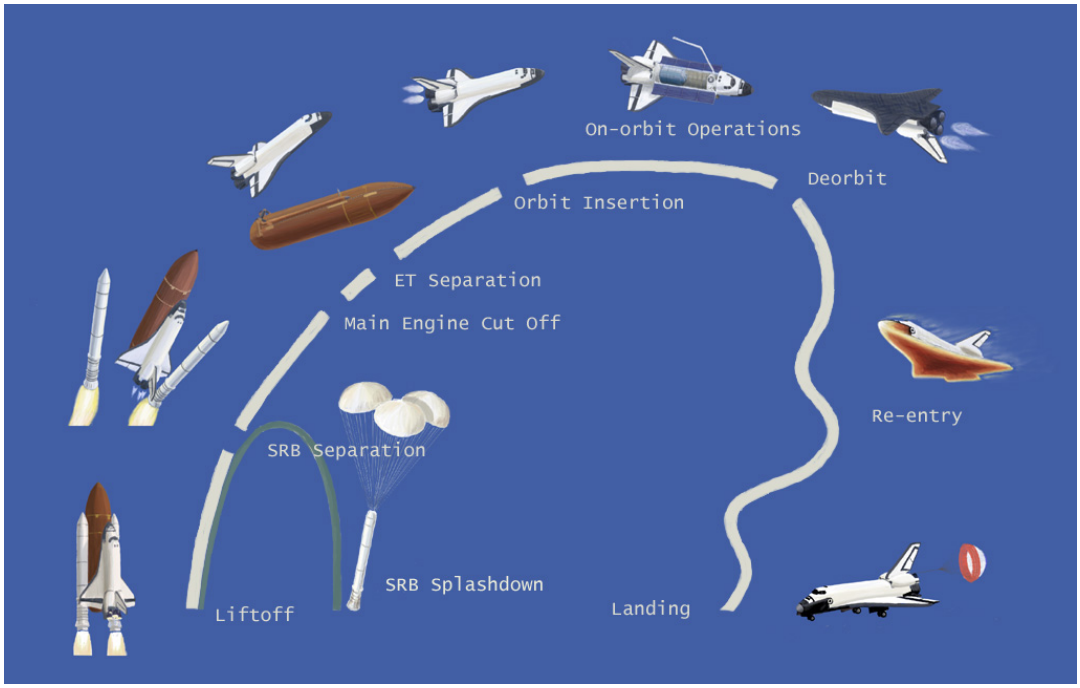


Figure 2.2: Space Shuttle and its mission profile.[3]

FESTIP

The Future European Space Transportation Investigations Program (FESTIP) started in 1994 to determine which launcher system concepts would become technologically feasible for Europe in the near future (Dujarric, 1999). Besides technical feasibility, the main selection criteria were the concept development cost, launch cost and adaptability to the launch market. Eight concepts, consisting of SSTO and TSTO with a combination of horizontal and vertical take-off and landing, were chosen for detailed design. The sub-orbital hopper (concept 15) and the semi-reusable TSTO (concept 16) seemed within Europe's technical reach and were further analyzed. The concepts are shown in Figure 2.3

The semi-reusable TSTO would consist of a reusable first stage with an expendable launcher as second stage. This expendable launcher could be the Ariane-5 core stage or a small-to-medium class expendable launcher. The reusable first stage would separate at low hypersonic velocities and return back to the landing site (Roenneke et al., 2003).

The sub-orbital horizontal take-off (HTO) Hopper would also consist of two stages. It would be launched horizontally using a rail-guided sled. It accelerates using rocket engines and at an altitude of 130 km it ejects the expendable second stage that further accelerates the payload. The Hopper stage then re-enters and glides back to the landing site (Spies, 2003). In 2004 tests have been done with

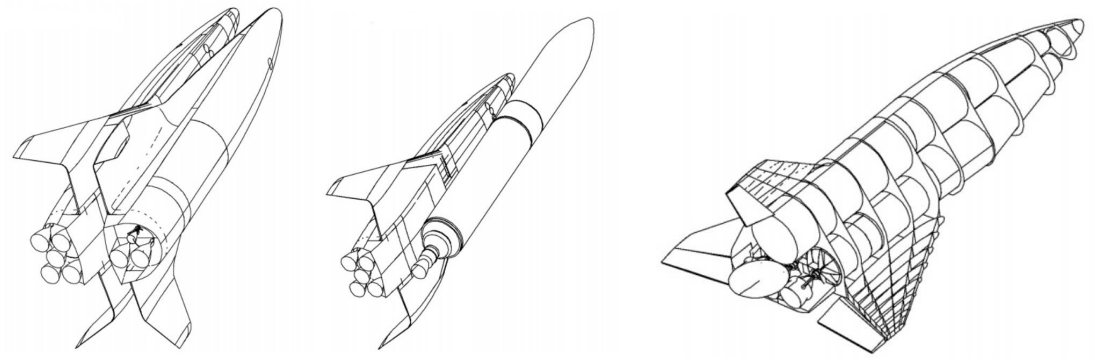
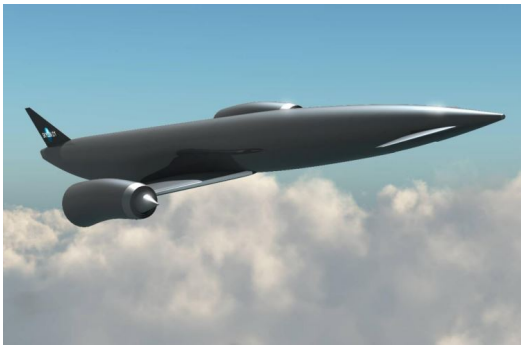


Figure 2.3: From left to right: FESTIP concept 16 fully reusable, semi-reusable and concept 15 Hopper (Dujarric, 1999).



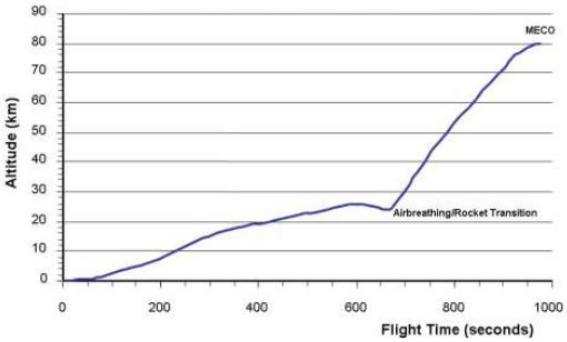


Figure 2.4: Skylon with its ascent trajectory (Longstaff and Bond, 2011; Varvill et al., 2014).

the Phoenix flight-test vehicle, which vehicle shape was derived from the Hopper concept (Jategaonkar et al., 2006).

As a part of the Future Launchers Preparatory Program (FLPP), a vertical take-off (VTO) Hopper was also designed. The reusable Hopper is launched vertically with an expendable second stage mounted on top. At an altitude of more than 90 km the engines are cut off and the second stage separates. The Hopper then follows a ballistic path followed by a gliding re-entry and a subsequent horizontal landing (Pezzella et al., 2009).

The trajectories of both the HTO and VTO versions have been optimized with the AeroSpace Trajectory Optimization Software (ASTOS)[4]. This software package includes both the direct multiple shooting and collocation methods. These local optimization methods will be further discussed in Section 2.3.

Skylon

Skylon (Longstaff and Bond, 2011) is a system still under development today. The first unmanned test flights should be performed from 2025 onwards[5]. It is an SSTO HTHL design using the specially developed SABRE engine. This engine can operate in air-breathing mode from take-off to approximately Mach 5. At this velocity and an altitude of 28.5 km the engine switches to pure rocket mode accelerating to a maximum velocity of Mach 27.8. The two stages of the engine are clearly visible in the altitude-time plot given in Figure 2.4. During the initial part of the flight, the vehicle has a shallow ascent, exploiting the dynamic pressure for its air-breathing engine. The maximum dynamic pressure during this section of the flight stays below 55,000 N/m^2. When the rocket engines are started, dynamic pressure is no longer an advantage and the ascent gets much steeper. The vehicle ascents and accelerates fast during this part of the flight. At the end of the flight the engines throttle back to keep the axial acceleration below the maximum constraint value of 3g. Once in orbit, the payload can be deployed in LEO. If needed to Skylon Upper Stage (SUS) could be used to place the payload in a different orbit. After the SUS is reunited with Skylon, the vehicle can perform a Space Shuttle like re-entry performing a gliding approach and landing (Varvill et al., 2014).

The trajectory optimization of Skylon is also performed with ASTOS. A maximum payload capacity of 17 tonnes is found for a launch from the Equator to an orbit at 160 km with a zero degree inclination. A maximum altitude of 600 km is defined for the vehicle. Launching from the Equator to a polar orbit at this altitude results in the minimum found payload capacity of 2.8 tonnes.

Ascender, SpaceCab and SpaceBus

Bristol Spaceplanes is a company founded in 1991 with a vision to decrease the cost of travelling to space by developing space-planes[6]. The designs of Ascender, SpaceCab and SpaceBus are shown in Figure 2.5 and were already proposed by Ashford in the early 1990s (Ashford, 1993).

The first design, Ascender, is a suborbital space-plane combining jet engines and rocket engines to reach a maximum altitude of 100 km and a velocity of Mach 3. The mission profile of Ascender is

[4]Astos Solutions (2016b). Launch vehicle application. `www.astos.de/solutions/space/launcher` [Accessed: 24 March 2017]

[5]BBC News (2015). BAE invests in space engine firm reaction engines. `http://www.bbc.com/news/business-34694935` [Accessed: 13 March 2017]

[6]Bristol Spaceplanes (2016). About bristol spaceplanes limited. `http://bristolspaceplanes.com/company/` [Accessed: 14 March 2017]

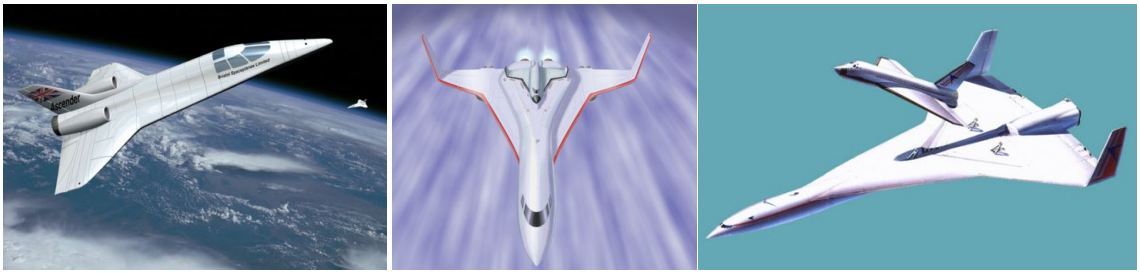


Figure 2.5: From left to right: Ascender, SpaceCab and SpaceBus.[7]

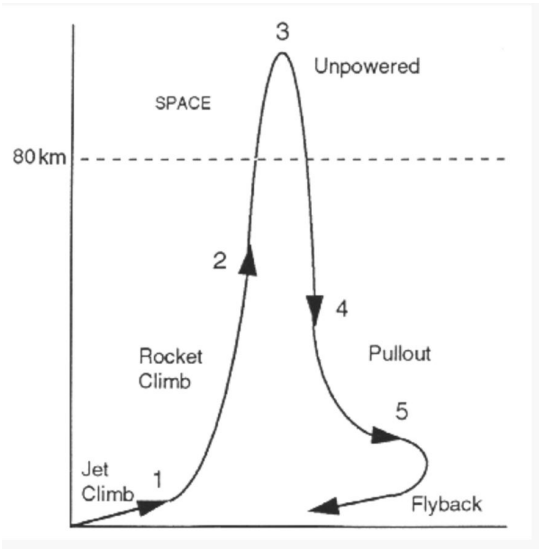


Figure 2.6: Mission profile of Ascender.[7]

shown in Figure 2.6. The jet engines would accelerate Ascender in the subsonic range to an altitude of 8 km. At this altitude the rocket engines would be started and a steep ascend follows, bringing the vehicle to an altitude of 64 km at a velocity of Mach 2.8. A coasting phase would follow during which the maximum altitude of 100 km is reached. An unpowered descend brings the vehicle back to the airfield.

The experience with Ascender should pave the way forward to the fully orbital SpaceCab. This is a TSTO HTHL design. The first stage would be a carrier aircraft able to accelerate to Mach 4 using first jet engines and subsequently rocket engines. After a steep climb the Ascender-like orbiter would separate and the carrier aircraft would re-enter and fly back to the landing site. The take-off mass of the complete configuration is 181 tonnes and a maximum payload of 750 kg could be transported to LEO.

Eventually SpaceBus would follow up. The design would be similar to SpaceCab but bigger and using turbo-ramjet engines and rocket engines to accelerate to Mach 6 before separation. The total take-off mass is 400 tonnes and a payload of 5 tonnes or up to 50 people could be taken to LEO. A next generation of SpaceBus would also be able to fly between Europe and Australia in 75 minutes. Bristol Spaceplanes is currently still seeking funding for the project (Ashford, 2009).

Virgin Galactic SpaceShipTwo

Virgin Galactic is a company trying to make space accessible for more people in the form of space tourism[8]. SpaceShipTwo is its sub-orbital vehicle that is currently still undergoing testing. It is air launched from a carrier aircraft, the WhiteKnightTwo. The mission profile is visualized in Figure 2.7. The carrier will take SpaceShipTwo to an altitude of 15 km. After separation, rocket engines will accelerate the SpaceShipTwo in a steep ascent to a velocity of 4000 km/h within 70 seconds. A parabolic flight is

[7]Source: `http://bristolspaceplanes.com/` [Accessed: 15 March 2017]
[8]Galactic, V. (2016). Who we are. `http://www.virgingalactic.com/who-we-are/` [Accessed: 15 March 2017]

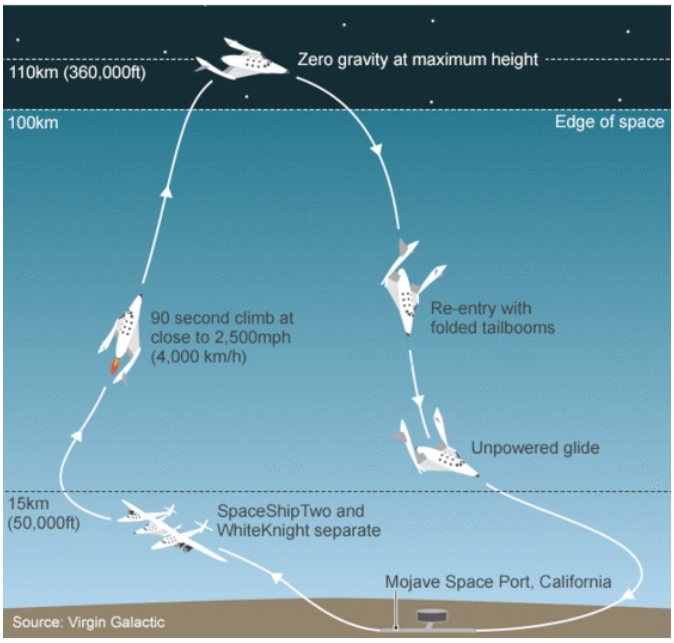


Figure 2.7: Mission profile SpaceShip 2.[9]

then followed, which allows people to experience weightlessness. After reaching altitudes higher than 100 km the vehicle can glide down and perform a horizontal landing (Seedhouse, 2015).

XCOR Lynx

XCOR is another company offering to fly people into space. The Lynx Mark II is a rocket propelled SSTO vehicle that reaches an altitude around 100 km. The mission profile is shown in Figure 2.8. It takes off horizontally and accelerates to a maximum velocity of Mach 2.9 in just 3 minutes. At an altitude of almost 60 km the engine shuts off and the vehicle coasts further to an altitude around 100 km. After the maximum altitude has been reached it glides back to the runway and performs a horizontal landing[10].

SpaceLiner

The SpaceLiner is a project that started in 2005 by the German Aerospace Center (DLR) and has since had a long line of concept developments. SpaceLiner aims at hypersonic long distance passenger transport, making it possible to fly from Australia to Europe in just 1.5 hours. The configuration uses two rocket propelled stages and launches vertically. At approximately Mach 12.5 the orbiter separates from the first stage. The first stage glides back to land. The orbiter accelerates further to a maximum velocity of 7.1 km/s at an altitude of 69 km (Sippel, 2015). At this point the flight-path angle is close to zero and a hypersonic gliding phase follows, meaning the vehicle will slowly descend and decelerate. The dynamic pressure is around 2,500 N/m^2 and the heat rate for a leading edge with a radius of R_n = 0.1 m would be 2,200 KW/m^2.

The baseline trajectory described above was found using ASTOS optimization with the objective to minimize the consumed propellant mass. The result is the black skipping trajectory shown in Figure 2.9. The green line indicates the final result and is a compromise that uses 0.3% more fuel than the optimal skipping trajectory, but reduces the heat load by 46% (Sippel et al., 2011). Reducing the heat load could also lead to an increasing payload capacity as the size of the thermal protection system (TPS) can be reduced, which saves mass.

[9]Source: `http://www.bbc.com/news/science-environment-29895140` [Accessed: 15 March 2017]
[10]XCOR (2016). Our hero: XCOR lynx. `spaceexpeditions.xcor.com/spacecraft/` [Accessed: 15 March 2017]
[11]Source: `http://www.missionmassimo.com/spaceflight/` [Accessed: 15 March 2017]

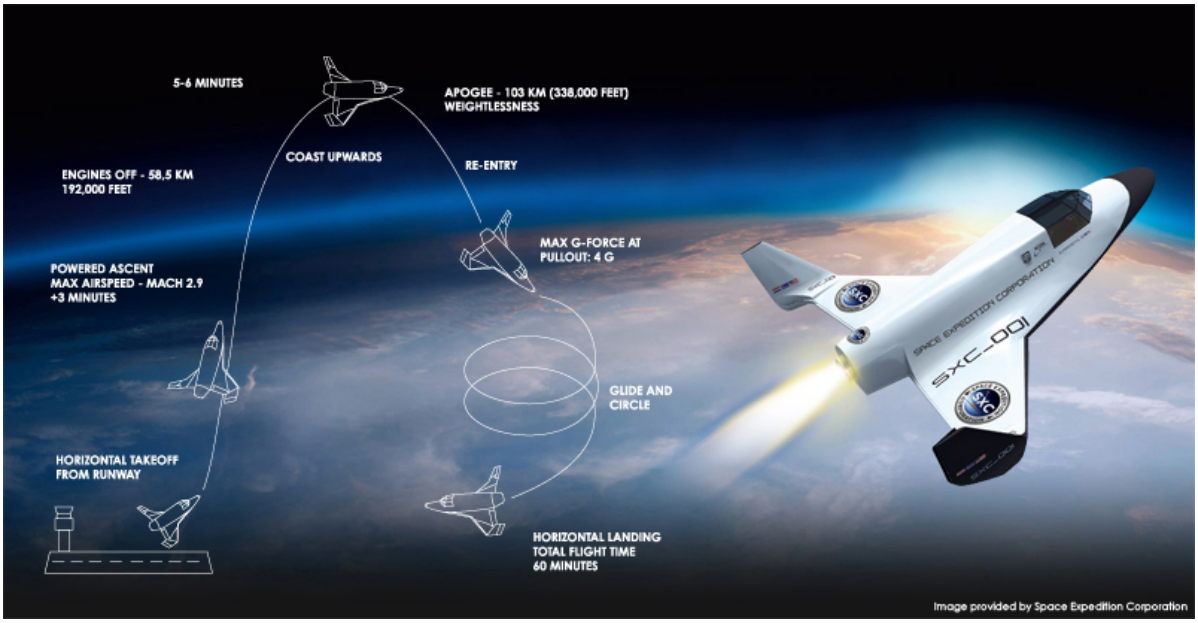


Figure 2.8: XCOR Lynx mission profile.[11]

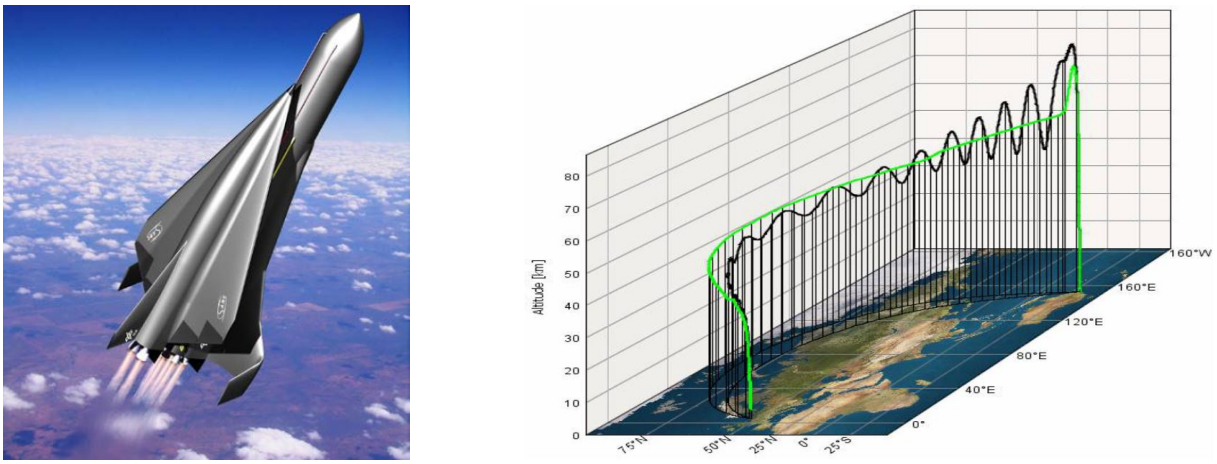


Figure 2.9: SpaceLiner and its trajectory (Sippel, 2015; Sippel et al., 2011).

2.1.2. Future Projects

A number of relevant RLVs were discussed in the previous section. Each of the projects used very different configurations. There are three important aspects in which design decisions need to be made. First of all a choice needs to be made whether one or two stages are used. These stages can be propelled using rocket or air-breathing engines or a combination of both. Finally, both take-off and landing can be done horizontally or vertically.

SSTO or TSTO

"There is little doubt that eventually the most economical space-planes will have only one stage" (Ashford, 2002). Having only one stage means no assembly of stages is required, saving time, crew and facilities on the ground. Furthermore, only one vehicle needs to be developed, parked and flown. Although this single-stage airplane-like vehicle seems ideal, experience has shown that developing such a vehicle is extremely difficult. The lack of technology readiness is also why the FESTIP study did not include air-breathing SSTO concepts (Dujarric, 1999). Furthermore, with TSTO configurations, the vehicle that finally goes into orbit is smaller. This reduces the fuel consumption and results in a larger payload capacity. TSTO concepts like SpaceCab and SpaceLiner could thus be interesting configurations for the near future. Meanwhile, experience with these concepts could pave the way

towards SSTO concepts later on.

Air-breathing or Rocket Engines
Rocket engines have been the major propulsion method for spacecraft up to now. With the prospect of SSTO concepts in mind, air-breathing engines will become more important. Air-breathing engines have the advantage of using oxygen from the air, thereby reducing the required fuel mass on board. In addition they have a higher specific impulse in the lower Mach number range than rocket engines (Kors, 1990). Rocket engines will still be required for final insertion and manoeuvrability in space, since air-breathing engines do not work outside the atmosphere. Having both engine systems will increase weight and size of the vehicle cancelling the positive effect of reduced fuel mass. The ideal configuration thus seems to be a hybrid engine combining both air-breathing and rocket propulsion in one system. The SABRE engine currently being developed for the Skylon SSTO vehicle is an example of such an engine (Longstaff and Bond, 2011; Varvill et al., 2014).

Horizontal or Vertical Take-off and Landing
The hope is that HTHl vehicles will be able to operate similar to aircraft, resulting in a number of advantages like: no extra costs for launch facilities, faster turn-around time and lower launch costs.

However, HTHL launch vehicles weigh approximately 4 times more during take-off than landing due to the large propulsion mass fraction (Dissel et al., 2006). Wings and landing gear for a HTHL vehicle will thus have to be designed for the gross take-off weight, while for a VTHL vehicle, those systems can be designed for the much lighter empty weight. The enormous mass and size reduction as a result of this, causes VTHL vehicles to outperform HTHL vehicles (Dissel et al., 2006; Livingston, 2008). According to research on TSTO configurations by Livingston (2008), the turn-around time of HTHL configurations is also not better, as the larger size of the vehicle increases the maintenance time.

Vertical launch thus seems to be more promising than horizontal launch after all. Those advantages, however, do not apply for landing. Horizontal landing requires less or no thrust while vertical landing comes with considerable safety issues including possible engine failure on landing (Ashford, 2002). A combination of vertical take-off and horizontal landing thus seems to be the most promising configuration.

2.2. Ascent Mission

The previous sections have explained the evolution of space-plane missions throughout the last decades. This resulted in the definition of a most promising configuration for the future. Based on these observations a relevant vehicle and trajectory for optimization will be selected in this section. This process is explained in Sections 2.2.1 and 2.2.2, respectively. A set of mission objectives and requirements will eventually be defined, based on the selected vehicle and trajectory. This will be discussed later in Section 2.4.

2.2.1. Vehicle

Following the observations from Section 2.1.2, an SSTO vehicle combining air-breathing and rocket propulsion will be chosen for the trajectory optimization. This choice is based on the fact that an SSTO air-breathing vehicle is the configuration strived for. Besides this, the availability of aerodynamic, propulsion and mass models is the most important selection criterion. Because models are available for the FESTIP concepts and the Winged Cone Configuration (WCC), a vehicle will be chosen from this selection.

Vehicle Trade-off
FESTIP did not study any relevant vehicles, taking the requirements stated above in mind. SSTO air-breathing concepts were eliminated on grounds of technical difficulty (Dujarric, 1999). The WCC is a simulation model for an SSTO air-breathing hypersonic vehicle that can be used for investigation of trajectory optimization (Shaughnessy et al., 1990). Multiple researchers have already used the vehicle for guidance and trajectory optimization studies. Van Buren and Mease (1991) used the Optimal Trajectories by Implicit Simulation (OTIS), which is based on a collocation method. Lu (1991) used an inverse dynamic approach and Hattis and Malchow (1992) used an algorithm based on a gradient/steepest descent method. Although the WCC is not the preferred VTHL configuration, it will follow

similar trajectory constraints. Comparison of global optimization methods on this horizontal take-off configuration could thus also be relevant for the vertical take-off configuration.

Vehicle Description

The original WCC and its aerodynamic, propulsive and mass models are described in Shaughnessy et al. (1990). The configuration is shown in Figure 2.10 and the geometric information converted to SI-units can be found in Table 2.1. The 61 meter long vehicle has a gross mass of 136,079 kg and a dry mass of 58,968 kg (Powell et al., 1991). It consists of an axisymmetric conical forebody with a half-angle of 5°, a cylindrical engine nacelle section and a cone frustrum engine nozzle section. Wings are mounted on the center-line of the fuselage. Elevons are placed on the trailing edges of the wings perpendicular to the fuselage center-line and can be controlled independently. Deflections are measured with respect to the hinge-line with zero degree deflections parallel to the vehicle center-line and positive deflections having the trailing edge down. On the center-line, a vertical tail is placed. It includes a full span rudder with its hinge-line at 25% of the chord length from the trailing edge. The deflections are again measured with respect to the hinge-line with zero degree deflections parallel to the vehicle center-line and positive deflections having the trailing edge left. At the front of the vehicle small canards are deployed at subsonic speeds. Deflections are measured with respect to the fuselage center-line with positive deflection having the trailing edge down.

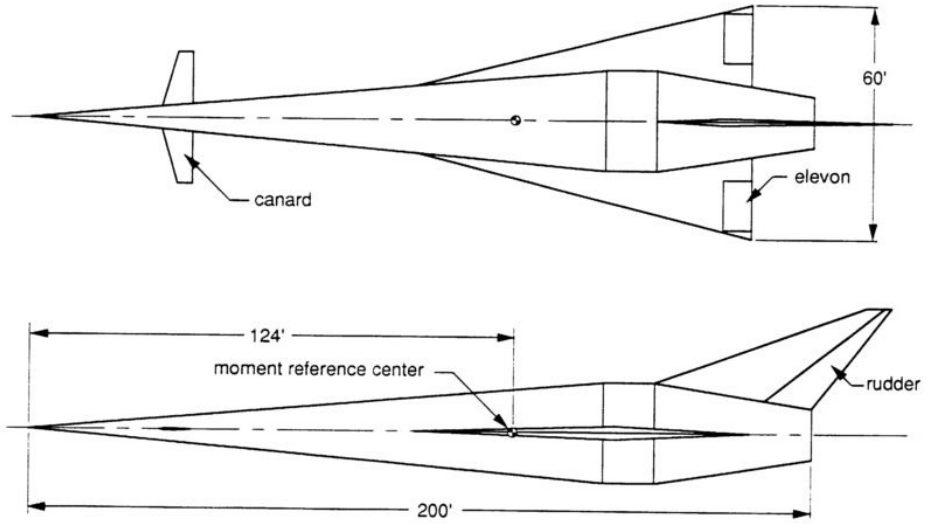


Figure 2.10: Winged-Cone Configuration (Shaughnessy et al., 1990).

The aerodynamic model is developed with the Aerodynamic Preliminary Analysis System (ASAP). Estimations for lift, drag and side force increment coefficients are given as a function of Mach number, surface deflection and angle of attack for the elevons, the canards and the rudder. Furthermore estimations of coefficients contributing to roll, yaw and pitch moments are given in graphs. A more detailed explanation on the aerodynamic model of the vehicle will be given in Section 3.2.3.

The propulsion model for the configuration was developed using a two-dimensional forebody, inlet and nozzle code with a one-dimensional combustor code. Estimations of the thrust coefficient and specific impulse are given as a function of Mach number, dynamic pressure and equivalence ratio. The nominal thrust acts along the body axis, meaning that for implementation of thrust-vector control (TVC) the angle will be measured with respect to this axis. A more in depth discussion on TVC will follow shortly. A more detailed explanation on the propulsion model of the vehicle will be given in Section 3.2.1.

Vehicle Controls

The vehicle is controlled by commanding the amount of thrust with the equivalence ratio (ϕ) and setting the angle of attack (α), which is in turn controlled with control surfaces like ailerons, elevators

fuselage length	60.96 m
cone half angle	5°
cylinder radius	3.92 m
cylinder length	3.93 m
boat-tail half angle	9°
boat-tail length	12.19 m
moment reference center from nose	37.80 m
wing reference area	334.73 m^2
aspect ratio	1.0
span	18.29 m
leading edge sweep angle	75.97 °
trailing edge sweep angle	0.0 °
mean aerodynamic chord	24.38 m
airfoil section	diamond
airfoil thickness to chord ratio	4.0 %
elevon area	8.57 m^2
chord	2.20 m
inboard section span location	4.57 m
outboard section span location	8.47 m
exposed area	60.00 m^2
theoretical area	116.02 m^2
span	9.90 m
leading edge sweep angle	70.0 °
trailing edge sweep angle	38.13 °
airfoil section	diamond
airfoil thickness to chord ratio	4.0 %
rudder area	14.99 m^2
span	6.95m
chord (relative to tail)	25%
canard area	14.33 m^2
theoretical aspect ratio	5.48
span	10.24 m
leading edge sweep angle	16.0 °
trailing edge sweep angle	0.0 °
airfoil section	NACA 65A006

Table 2.1: Winged-Cone Configuration geometry characteristics (Shaughnessy et al., 1990).

and elevons. The angle of attack influences the lift over drag ratio and with that the flight-path angle γ. The equivalence ratio gives the ratio of injected fuel to air. A value of unity gives the optimal efficiency, but thrust can be increased or decreased by respectively setting a higher or lower ϕ (Shaughnessy et al., 1990).

Studies on the WCC by Shaughnessy and Gregory (1991) have shown that at near orbital velocities, the center of aerodynamic pressure can move forward considerably. This causes instability that can be controlled by the elevons at a cost of extra drag. Active control of the center of gravity and the direction of thrust were found to reduce drag and fuel consumption. Mooij (1998) also investigated the use of thrust-vector control (TVC) and found that considerable fuel savings could be obtained.

TVC can be used only for trim, but also for active guidance control of the vehicle. In the former case, TVC is only used to stabilize the vehicle, with the result of smaller required elevon deflections. In the latter case, TVC will be used as a control parameter for guidance of the vehicle. This could be useful if limitations exist for the angle of attack, which might be a requirement to obtain the right inlet conditions for the air-breathing engine. Active TVC can reduce the required angle of attack by

providing extra lift with the thrust-elevation angle ϵ_T. A more in depth discussion on the guidance and control of the vehicle will follow in Section 4.4

2.2.2. Trajectory

The chosen WCC automatically leads to a horizontal take-off trajectory (Shaughnessy et al., 1990). From Powell et al. (1991), Lu (1991), Van Buren and Mease (1991), Hattis et al. (1991) and Mooij (1998) it is found that a typical take-off trajectory starts with a relatively steep pull-up with large flight-path angles γ of around 25°. Afterwards the flight levels out to comply with the constraints. When the vehicle has gained enough velocity, a final pull-up is initiated to perform a steep ascend.

The different studies mentioned earlier use different constraints. Van Buren and Mease (1991) found a near minimum fuel trajectory when tracking the constraints for dynamic pressure $\bar{q}$, axial acceleration n_a and heat rate $\dot{Q}$. The heat-rate constraint at the stagnation point was set approximately twice as high as for the Space Shuttle as space-planes are likely to include active cooling for the nose and leading edges. Powell et al. (1991) stated that $\bar{q}$ is the most demanding constraint. Lu (1991) used both $\bar{q}$ and $\dot{Q}$ as constraints and found that the minimum constraint value of $\bar{q}$, for which a trajectory can be found, was 71,772 N/m^2.

More constraints can be applicable to the ascending space-plane. However, this research will only consider the constraints mentioned above to be able to compare the result of the optimization applied here with those found in the earlier mentioned references. In particular the result found by Mooij (1998) will be used as a reference trajectory. In correspondence with his work the following constraints with typical values will be used throughout this research:

- Maximum dynamic pressure: $\bar{q}_c$ = 95,000 N/m^2
- Maximum heating rate: $\dot{Q}_c$ = 8,000 kW/m^2 at the leading edge of the wing (R_N = 0.1 m)
- Maximum axial load: $n_{a,c}$ = 1 g_0

Figure 2.11 shows the ascent trajectory that is found by Mooij (1998) after optimization. This trajectory will be used as a reference for the result found with the software developed for this research. The software architecture is discussed in Section 6.3 and a more extensive discussion on the resulting trajectory will follow in Section 6.4.2.

Because the air-breathing engine performs better at high dynamic pressure, a near optimal trajectory can be found by tracking the constraints. The result shows that the vehicle first closely track the

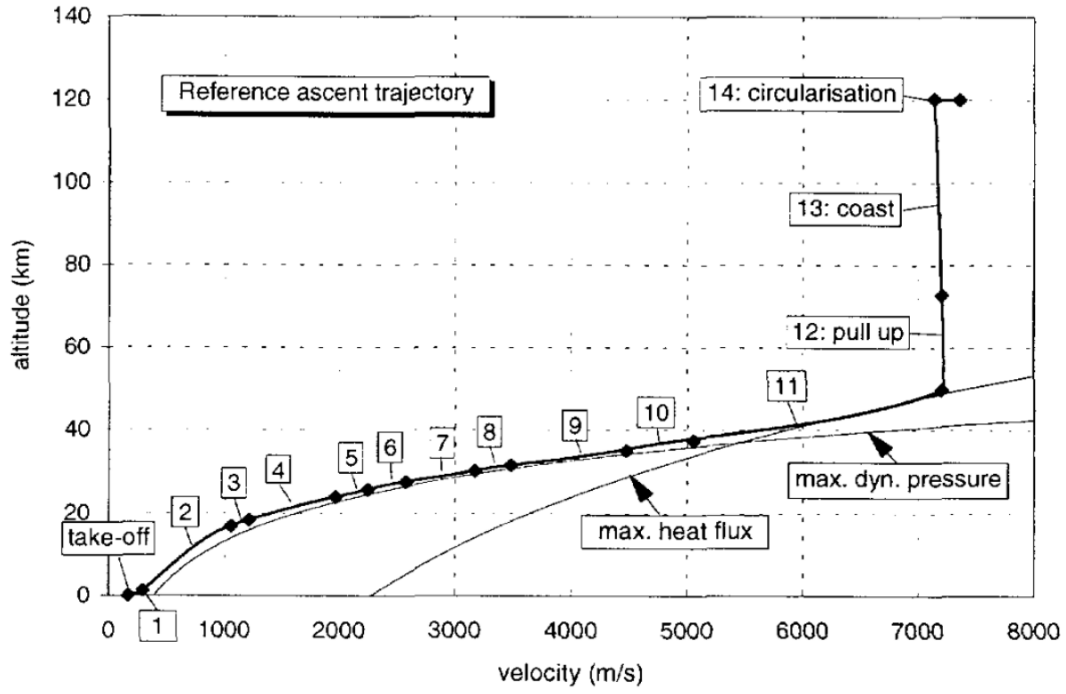


Figure 2.11: Trajectory and constraints of the reference trajectory obtained by Mooij (1998).

constraint of dynamic pressure and at velocities between 5,500 and 7,000 m/s the heat-rate constraint is tracked. At around 700 m/s the trajectory moves away from the dynamic-pressure constraint slightly to comply with the axial load constraint. When near orbital speeds are reached, a pull-up manoeuvre is performed for a final orbit insertion. Generally rocket engines are required for this final phase because of the absence of an atmosphere.

Mooij (1998) initially found a payload mass of 3412 kg for this trajectory reaching an altitude of 120 km. After further optimization with Taguchi's orthogonal arrays a maximum payload of 9146 kg was found. Lu (1991) used an inverse dynamic approach to simulate the trajectory up to the final pull-up and found a maximum payload of 8100 kg. Powell et al. (1991) used the POST optimization to find a minimum fuel trajectory reaching an altitude of 120 km and found a maximum payload capacity of 9100 kg. These examples show that currently used optimization methods are able to find feasible solutions for this ascent trajectory. However, the local optimization methods currently most widely used for trajectory optimization have a few disadvantages. In the next section these disadvantages will be shortly discussed, followed by an explanation of why global optimization might be an interesting alternative way to solve the ascent trajectory optimization problem.

2.3. Ascent Trajectory Optimization

The previous sections have mentioned some optimization methods that are currently successfully applied to solve ascent trajectory optimization problems. These methods are all local optimizers. Section 2.3.1 will discuss a number of the local methods. Subsequently, Section 2.3.2 will explain why global optimization methods could be an interesting alternative relevant for further research.

2.3.1. Local Optimization

Within the group of local optimizers a distinction can be made between indirect and direct methods. A number of advantages, disadvantages and real life applications for both will be discussed next.

Indirect Methods

Indirect methods first establish conditions for optimal control. These optimality conditions need to be satisfied in order to find an optimal trajectory. This leads to a boundary value problem that, in most cases, requires numerical solving. Common techniques are indirect collocation and (multiple) shooting (Betts, 1998). To obtain a good result, a good initial guess of the trajectory is essential and a priory knowledge of the constraint sequence is needed. Furthermore the user has to derive the adjoint differential equations (Betts, 1998; Von Stryk and Bulirsch, 1992). These preparations are all problem specific and can be very different for different trajectories. Although indirect methods can lead to good results, they require insight and a lot of work from the user.

However, indirect methods have been used for trajectory optimization. Indirect multiple shooting has been implemented in BNDSCO software that is widely used in Germany (Betts, 1998) and OPTAX that is frequently used by CNES and ARIANESPACE for Ariane future launchers and foreign launcher comparison (Talbot, 2003).

Direct Methods

Direct methods directly compare values of a cost function that needs to be minimized to find an optimal solution. Many different direct methods exist, but all use some form of parametrization of the control variables (Betts, 1998; Von Stryk and Bulirsch, 1992). The control parameters for optimization are defined at a number of control nodes. To obtain a good result, a relatively small number of optimization variables should be used (Betts, 1998). When a large number of parameters are used for optimization, the search space becomes too large to find a good solution and convergence can take a long time. Although convergence of direct methods generally takes longer, a less accurate initial guess of the optimization parameters is required and no adjoint differential equations need to be derived. This means less work and insight is required by the user.

Direct collocation, different gradient methods and the direct (multiple) shooting method are most commonly used. Especially for launch vehicles, direct shooting is most widely used. POST and ORAGE are examples of software that use direct shooting (Betts, 1998; Talbot, 2003). The Program to Optimize Simulated Trajectories (POST) is, among others, used for ascent optimization of the Space Shuttle and its capabilities are explained in Brauer et al. (1977). ORAGE algorithms have been implemented in

OPTAX-H. This combination can use ORAGE direct shooting during the atmospheric phase and OPTAX indirect shooting for the exo-atmospheric phase. This was especially developed for reusable launch vehicles (RLV), including horizontal take-off vehicles (Talbot, 2003).

Future Launcher Optimization Program (FLOP) is ONERA's latest trajectory optimization tool that uses the Olga optimization core also present in OPERA. It has implemented two different gradient techniques to find an optimal trajectory. The software has been especially designed to meet the needs of both expendable and reusable launchers (Jolly et al., 2003; Talbot, 2003).

AeroSpace Trajectory Optimization Software (ASTOS) is an extension to Graphical Environment for Simulation and Optimization (GESOP) that uses a direct collocation method (TROPIC) and a direct multiple shooting method (PROMIS)[12]. It has been widely used for launcher trajectory optimization. Cremaschi et al. (2009) show an example of ASTOS software being applied to a sounding rocket. Furthermore, the software has been used for trajectory optimization of the spaceliner Sippel et al. (2011), performance updates of the Ariane-5 family and VEGA, reference trajectories of the FESTIP Hopper, and feasibility studies for Skylon and many other vehicles[13].

2.3.2. Global Optimization

Compared to the local methods discussed above, global optimization methods have the advantage that they are easy to apply and not much insight in the optimization problem is required. The main draw-back is that computation times will be long complex problems. However, with the increasing computation power over the last decades, interest in global optimization methods has also been increasing.

Global methods can be divided into systematic and heuristic methods. Systematic methods will find a global optimum with a predetermined amount of work. Examples are branch and bound constraint problems that systematically search starting points for local optimization (Vasile, 2003). These methods need some insight in the problem at hand to be efficient. Examples of Heuristic methods are: Genetic Algorithms (GA), Evolutionary Programming (EP), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO) and Simulated Annealing (SA) (Izzo, 2006). These methods cannot be proven to find a minimum with a predictable amount of work. They generally pick random starting points and use specific methods to search the space in a 'careful' way (Vasile, 2003).

All methods mentioned above have been studied in a project carried out by the ESA, showing the potential of global optimizers for trajectory optimization. However, most applications of global optimization techniques so far have been on (low-thrust) interplanetary trajectories. As said by Betts (1998): "what works well for a launch vehicle guidance problem may be totally inappropriate for a low-thrust orbit transfer".

For ascent trajectories, so far, PSO has been applied on vertical launch trajectory optimization by Pagano and Mooij (2010) and a genetic algorithm has been applied by Al-Garni and Kassem (2005) to find the minimum-fuel-minimum-heat ascent control for an aerospace-plane. Al-Garni and Kassem (2007) also developed a hybrid optimization combining the better time performance of genetic algorithms with the better convergence of particle swarm optimization, resulting in a more robust system.

Even though research has indicated the potential of global methods for trajectory optimization in general, only very limited research has been done on the global ascent trajectory optimization of space-planes. In this research the performance of different global optimizers will therefore be applied to the ascent trajectory problem defined in the previous section. A more in depth discussion on global optimization methods including a trade-off and explanation of the used methods will follow in Chapter 5.

2.4. Mission Requirements

The final mission is defined based on the research objectives. As the main focus of this research is to compare different global optimizers and the optimization process, some simplifications are made. The control variables mentioned in Section 2.2.1 do not include banking, which leads to a vertical plane ascent trajectory. This trajectory is a simplification of the real world problem, but will be very suitable to study and compare different methods to optimize space-plane ascent trajectories. Because frequent access to space is a big advantage of space-planes, the International Space Station (ISS) would be a

[12] Astos Solutions (2016a). GESOP details. `www.astos.de/products/gesop/details` [Accessed: 24 March 2017]

[13] Astos Solutions (2016b). Launch vehicle application. `www.astos.de/solutions/space/launcher` [Accessed: 24 March 2017]

relevant goal. The ISS is frequently visited and space-planes would thus be a good means of transport. Specific mission trajectories require a guidance out of the vertical plane. However, the current research will not consider the inclination of the orbit and will only use the ISS altitude and its orbital velocity as the final conditions to the vertical plane ascent trajectory.

The simplifications are made because it allows for an easier initial optimization process. Furthermore, the resulting trajectory can be easily compared with the results found by Mooij (1998). The following set of mission requirements is set up based on the mission described in this section:

Mis-1 The vehicle shall be launched from the equator.
Mis-2 The vehicle shall follow a trajectory in the vertical plane along the equator ($\chi = 90°$).
Mis-3 The vehicle shall reach a final altitude of 408 km with a corresponding circular velocity.
Mis-4 The flight-path angle γ shall be equal to zero at the final state.
Mis-5 The trajectory shall comply with the constraint of maximum dynamic pressure: $\bar{q}_c = 95{,}000$ N/m^2.
Mis-6 The trajectory shall comply with the constraint of maximum heating rate: $\dot{Q}_c = 8{,}000$ kW/m^2.
Mis-7 The trajectory shall comply with the constraint of maximum axial load: $n_{a,c} = 1$ g_0.

3

Environment and Vehicle

The trajectory of any vehicle is determined by the external forces that are acting on it. In the case of an ascending space-plane these forces consist of gravitational, propulsive and aerodynamic forces. In this chapter the environment and vehicle models used to determine these forces will be discussed in Section 3.1 and Section 3.2, respectively. The models discussed here will later on be implemented in the simulation software that is discussed in Chapter 6.

3.1. Environment Models

The environment has a large influence on the performance of space-planes. Two important factors are the Earth's gravity field and the atmosphere. Picking the right models to simulate the environment is therefore important. It has to be noted that different objectives could require different models. For the comparison of the optimization process, which is the main objective of this research, less accurate models could be sufficient than required for the optimization of a specific mission trajectory design. However, the non-linearity of the problem means that different environmental models could lead to a much easier or harder to find optimal trajectory.

For this research a spherical Earth with constant rotation rate is used. The Earth Gravity Model 1996 (EGM96) is implemented, including only zonal harmonics up to J_2. For the atmosphere the tabulated US76 standard atmosphere is selected. Sections 3.1.1 to 3.1.3 shortly explain why these models are selected and describe their working.

3.1.1. Gravity Field

Many different gravity models exist. In combination with the different models a choice can be made for a rotating or non-rotating Earth. A number of different models is compared.

A constant gravity model uses a constant value for the gravitational acceleration regardless of the vehicles location. However, gravity decreases with altitude. For a launch trajectory high altitudes will eventually be part of the trajectory. Using the more accurate inverse squared law, a decrease in gravitational acceleration of 3% is already found at 100 km altitude.

The inverse squared law assumes a homogeneous distribution of the central body mass. However, this is not the case. The irregularities in Earth's mass distribution can be modelled with a spherical harmonics model. By far the most significant influence of the harmonics is the J_2 term. The largest effect is found at the surface of the Earth, which is just over 0.3% of the local gravitational acceleration. Higher spherical harmonics can be included but the effect of J_3 is already 1,000 times smaller than the effect of J_2.

Based on the previous discussion it is concluded that the constant gravity model lacks accuracy while the higher order spherical harmonics model is more complicated and computationally expensive than required for the current research. The J_2 zonal harmonic is not necessarily important for the currently studied trajectory. However, the Earth Gravitational Model 1996 (EGM96) is already implemented in Tudat[1] (see Section 6.1) and the additional computational effort is limited compared to using the inverse

[1]Tudat (2016). Tu delft astrodynamics toolbox. `http://tudat.tudelft.nl/` [Accessed: 1 June 2017]

square law. The spherical harmonics EGM96 including only the J_2 zonal harmonic is therefore selected in combination with a rotating Earth.

Central Gravity Including J_2
The irregularities in the mass distribution of Earth can be modelled with a spherical harmonics model. The gravity models implemented in Tudat use Cartesian components. The influence of the J_2 zonal harmonic is given by Wakker (2015, p. 531-132) in Cartesian components according to:

$$g_x = -\frac{3}{2}\mu J_2 \frac{R_E^2}{r^5} x\left(1 - 5\frac{z^2}{r^2}\right) \tag{3.1a}$$

$$g_y = -\frac{3}{2}\mu J_2 \frac{R_E^2}{r^5} y\left(1 - 5\frac{z^2}{r^2}\right) \tag{3.1b}$$

$$g_z = -\frac{3}{2}\mu J_2 \frac{R_E^2}{r^5} z\left(3 - 5\frac{z^2}{r^2}\right) \tag{3.1c}$$

where μ is the gravitational parameter of Earth, R_E is the mean equatorial radius of the Earth, r is the distance between the vehicle and the c.o.m. of the Earth and x, y and z indicate the Cartesian position of the vehicle with respect to the center of Earth.

3.1.2. Earth Shape

Although the Earth is often assumed to be a sphere, it can be better approximated by an oblate spheroid. This flattening of the poles is attributed to the centrifugal forces caused by rotation of the Earth around its axis. The mean radius at the poles R_P is therefore approximately 21 km smaller than the mean radius at the equator R_E. The radius at an arbitrary point on the surface of the Earth R_S can be approximated with the following expression (Mooij, 2015, p. 8):

$$R_S \approx R_E\left(1 - \frac{e}{2}\left(1 - \cos 2\delta\right)\right) = R_E\left(1 - e\sin^2\delta\right) \tag{3.2}$$

where $e = 1 - \frac{R_P}{R_E}$ represents the ellipticity of Earth and δ is the geocentric latitude. As δ will remain zero for the mission defined in Section 2.4, the altitude of the vehicle can be found with:

$$h = r - R_S = R - R_E\left(1 - e\sin^2\delta\right) = r - R_E \tag{3.3}$$

This is the same result as for a perfect sphere and the simpler spherical model is therefore used.

3.1.3. Atmosphere

Because the vehicle under consideration will accelerate to high velocities within the atmosphere, large aerodynamic forces will be present. The models discussed by Shaughnessy et al. (1990) show that lift, drag and thrust all depend on the dynamic pressure which in turn linearly depends on the atmospheric density. The atmospheric model used for the trajectory optimization can therefore have a strong influence on the results. A number of models will be discussed.

The US76 model is a standard atmosphere model, meaning that it gives a vertical distribution of the atmospheric properties that are roughly representative of year-round, mid-latitude conditions (NOAA et al., 1976). The exponential atmosphere is a simpler model, which states that density decreases exponentially with altitude according to. Although this model is commonly used, the offset of the exponential model with respect to the US76 standard model is significant. At 20 km altitude the US76 model, according to tabulated data from NOAA et al. (1976), predicts a 20% higher density than the exponential model.

Earth-GRAM and NRLMSISE-00 are both reference atmosphere models. This means that the model includes among others latitudinal, seasonal, geomagnetic and solar effects (Mooij, 2015). These models are therefore more accurate than the US76 when specific locations and time-dependent effects included.

Based on the previous discussion it is concluded that the exponential atmosphere model lacks accuracy. Both the Earth-GRAM and NRLMISE-00 models are more accurate than the US76 model. However, in this research latitudinal and time dependent effects like wind are not important as it is not based on a specific mission. The US76 atmospheric model is therefore good enough to comply with the requirements of this research. Although a strictly analytical model is more accurate, a tabulated version is used as it is already implemented in Tudat.

US76

The US76 model is an updated version of the US62 model. The total altitude range is divided in intervals with a linearly varying molecular scale temperature T_M in each interval. Both models are identical up to 50 km, and the same equations can be used up to 86 km. However, from 80 to 86 km the molecular mass is no longer constant for US76, but slowly decreasing (NOAA et al., 1976, tab. 8). The variation of the molecular scale temperature as a function of geopotential altitude for the first 6 layers is shown in Figure 3.1. The equations used to calculate the atmospheric properties in this altitude range can be found in NOAA et al. (1976) and (Mooij, 2015, p. 36-39).

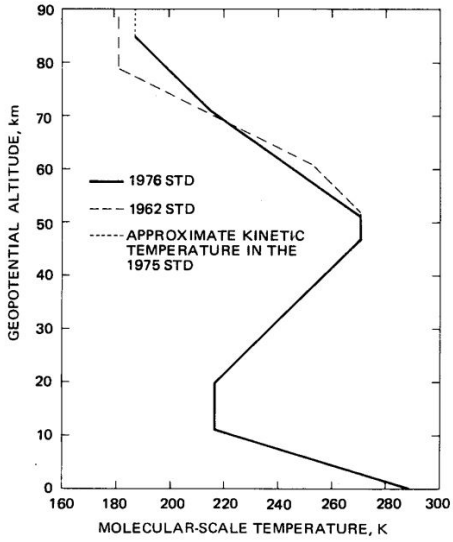


Figure 3.1: Molecular scale temperature as a function of geopotential altitude NOAA et al. (1976).

Above 86 km T_M is no longer defined and four successive functions are defined for the temperature T as a function of geometric altitude h. This is done in a way that the first derivative of T with respect to h is continuous over the complete interval from 86 to 1000 km. According to NOAA et al. (1976) these intervals represent:

- An isothermal layer from 86 to 91 km, given by:

$$T = 186.8673 \text{ K} = \text{constant} \qquad \text{and} \qquad \frac{dT}{dh} = 0 \text{ K/km} \tag{3.4}$$

- A layer in which $T(h)$ has the form of an ellipse, given by:

$$T = T_c + B\sqrt{1 - \left(\frac{h - h_8}{b}\right)^2} \qquad \text{and} \qquad \frac{dT}{dh} = -\frac{B(h - h_8)}{b}\frac{1}{\sqrt{1 - \left(\frac{h-h_8}{b}\right)^2}} \tag{3.5}$$

where T_c = 263.1905 K, B = -76.3232 K, b = 19.9429 km and h_8 = 91 km.

- A constant, positive-gradient layer from 110 to 120 km, given by:

$$T = T_9 + L_9(h - h_9) \qquad \text{and} \qquad \frac{dT}{dh} = L_9 = 12 \text{ K/km} \tag{3.6}$$

where T_9 = 240 K, L_9 = 12 K/km and h_9 = 110 km.

- A layer in which T increases exponentially towards an asymptote as h increases from 120 to 1000 km, given by:

$$T = T_\infty - (T_\infty - T_{10})e^{-\lambda\zeta} \qquad \text{and} \qquad \frac{dT}{dh} = \lambda(T_\infty - T_{10})\left(\frac{r_0 + h_{10}}{r_0 + h}\right)^2 e^{-\lambda\zeta} \tag{3.7}$$

where T_∞ = 1000 K, T_{10} = 360 K, h_{10} = 120 km, the effective Earth's radius for calculating geopotential r_0 = 6356.776 km, $\lambda = L_9/(T_\infty - T_{10} = 0.01875$ and $\zeta = \zeta(h) = (h - h_{10})\frac{r_0 + h_{10}}{r_0 + h}$.

The atmospheric pressure can now be found with the ideal gas law in the form of:

$$p = \frac{NR^*T}{N_A} \tag{3.8}$$

where N is the total density number in m^{-3}, R^* is the universal gas constant and N_A is Avogadro's constant. The total number density can be found through linear interpolation of a table from NOAA et al. (1976). Finally the density can be found using again the ideal gas law.

3.2. Vehicle Models

The performance of the vehicle depends strongly on the models used to simulate the vehicle. In this section these vehicle models, as described by Shaughnessy et al. (1990), will be discussed. Because the WCC is a theoretical vehicle, the models have limitations. These limitations will be addressed separately for each of the models. The propulsion model will first be discussed in Section 3.2.1. Subsequently the mass, aerodynamic and trim models will be studied in Section 3.2.2 to Section 3.2.4. Finally Section 3.2.5 gives a summary of the discussed vehicle parameters and their defined domains.

3.2.1. Propulsion

The WCC is modelled with a fully air-breathing engine. The model described by Shaughnessy et al. (1990) is obtained with a two-dimensional forebody, inlet, and nozzle code with a one-dimensional combustor code. The resulting propulsion model is described in this section including the implementation of TVC and the limitations of the model.

Model Implementation

The geometry of the propulsion system is not clearly documented, but the WCC is generally known to have engine modules all around the cylindrical engine nacelle section. The effects of angle of attack, sideslip, body angular rates and deflections of control surfaces on the thrust coefficient C_T and the specific impulse I_{sp} are neglected. These assumptions lead to an axisymetric thrust along the body x-axis (see Section 4.2.1).

Shaughnessy et al. (1990) give C_T and I_{sp} of the engine as a function of Mach number M, dynamic pressure $\bar{q}$ and fuel equivalence ratio ϕ. Linear interpolation can be used to find the right values from the graphs. Figure 3.2 shows the values as a function of Mach-number alone, using the nominal values for $\bar{q}$ and ϕ. For a given M, $\bar{q}$ and ϕ, the thrust can be calculated using:

$$T = \bar{q}C_T \tag{3.9}$$

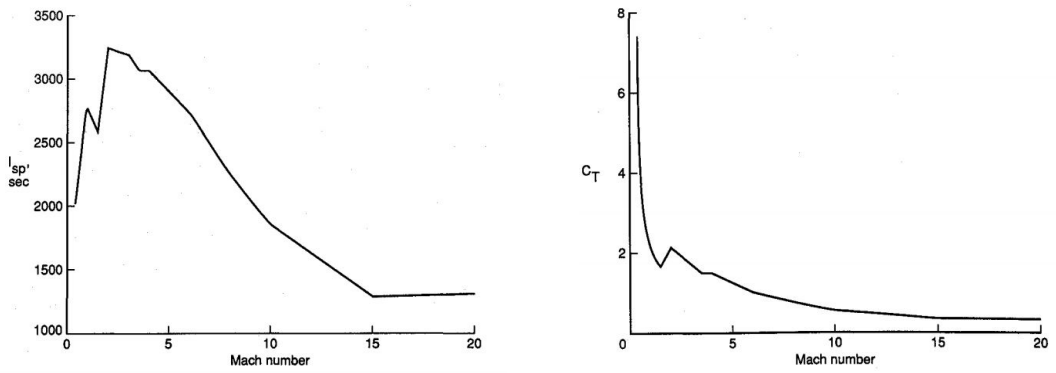


Figure 3.2: Engine characteristics at nominal dynamic pressure and fuel equivalence ratio (Powell et al., 1991).

with the thrust known, the fuel rate $\dot{m}$ can the be calculated with:

$$\dot{m} = \frac{T}{I_{sp}} \tag{3.10}$$

Engine throttling has been mentioned as a possible control variable. Throttling is important to make sure that the trajectory constraints are not violated. In the present model, throttling is simulated by varying the equivalence ratio ϕ. The nominal value is unity and corresponds to maximum efficiency. For more or less thrust, ϕ can be increased or decreased respectively. Thrust could also be varied by applying a scaling factor F_T to the maximum thrust for a given ϕ. This method could give different results due to the non-linear relation between T and ϕ.

The thrust-elevation angle ϵ_T is the other engine related control variable under consideration. TVC can be used to obtain a pitching moment that reduces the required deflection angle of the control surfaces required for trim. Changing the direction of the thrust-vector, in such a way that the exhaust gasses are deflected downwards, has the following consequences:

- The vertical component of the thrust-vector increases the lift force. This reduces the required angle of attack and therefore the drag .
- The horizontal component of thrust is decreased. This results in either a longer flight time before the pull-up velocity is reached, or a higher equivalence ratio is required. Both have a negative effect on the fuel consumption.

When the thrust-vector is no longer working along the body x-axis, the location of the center of thrust (c.o.t.) is important. The center of thrust is defined as the location where the thrust acts on the vehicle. Both the cylindrical engine nacelle and the cone frustrum nozzle are part of the propulsion system. The c.o.t. will therefore lie in the nozzle section. In accordance with Mooij (1998), the c.o.t. is estimated to be located at one third of the nozzle length from the aft of the fuselage. The angle of thrust deflection is measured from the body x-y plane, and is defined positive if the exhaust gasses are deflected upwards.

As stated before, TVC produces a part of the total pitch moment required for a trimmed flight. Instead of defining ϵ_T, it is also possible to state what fraction of the total required trim F_{TVC} should be produced by TVC. With the trim model discussed in Section 3.2.4 the required ϵ_T can be computed. Both ways of defining TVC could lead to very different results and are therefore interesting to compare. All thrust parameters discussed in this section will be included in the guidance for optimization discussed in Section 4.4.2. The effect that the different parameters have on the optimization process will be part of the research and results of the comparison will be given in Section 7.3.

Model Limitations

The theoretical propulsion model described is based on assumptions and has some limitations. Three important aspects will be discussed: the integration of the scram-jet engine in the vehicle, the implementation of TVC and the use of the propulsion system outside the atmosphere.

Generally a scram-jet engine is highly integrated in the vehicle design. The forebody is shaped to compress air for the inlet and the afterbody serves as an expansion surface (Curran, 2001). Figure 3.3 shows such an integrated design. Clearly the WCC is not shaped in this way and the geometry of the engine is not specified. The air-breathing propulsion model combined with this vehicle shape is thus purely theoretical.

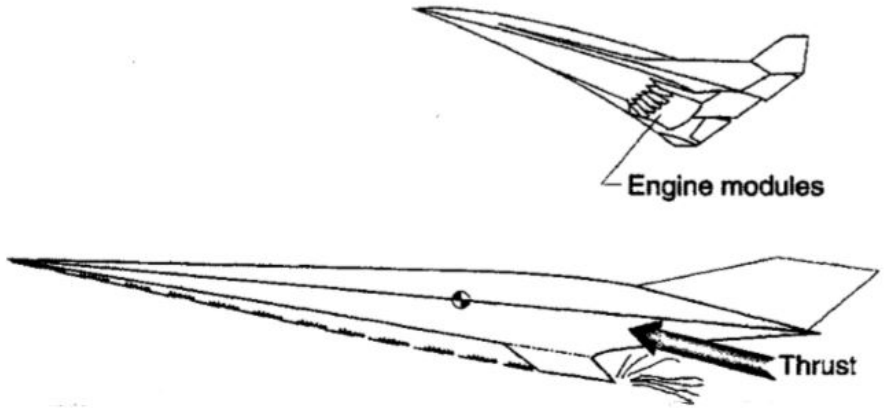


Figure 3.3: Scram-jet integrated in hypersonic vehicle design (Curran, 2001).

The implementation of TVC is also theoretical. Whether or not this is technically feasible is not taken into account. Furthermore, it is reasonable to assume that the implementation of TVC will increase the mass of the system, and affect the performance to some extent. These effects will be neglected, but are important to remember when conclusions are drawn.

The final part of the trajectories found by Mooij (1998) and Powell et al. (1991) include a coasting phase to reach the final altitude. Both studies required extra thrust once the final altitude was reached to circularize the orbit. Because the vehicle is now outside the atmosphere, the dynamic pressure is close to zero and conform Equation 3.9, the thrust delivered by the air-breathing engine is also close to zero. Rocket propulsion is required to reach the final state. In agreement with Mooij (1998) and Powell et al. (1991), a theoretical rocket engine is used with a specific impulse $I_{sp} = 465$ s. The mass of the rocket engine itself is initially neglected. However, a mass penalty for the final velocity increment ΔV required is added according to Tsiolkovsky's equation given by:

$$\Delta V = c_{eff} \ln \frac{m_f}{m} \tag{3.11}$$

where n_f is the final mass after the air-breathing engine is turned off, just before the rocket engine is started. m is the final mass after the rocket boost and c_{eff} is the effective exhaust velocity given by $c_{eff} = g_0 I_{sp}$.

3.2.2. Mass

The size and mass of the vehicle were obtained through an iterative method. Subsequently, the initial vehicle take-off mass, size, propulsion system and aerodynamics were estimated. POST simulations were used to find a trajectory to orbit, resulting in a final mass fraction. This mass fraction was used as an input to a vehicle sizing program that subsequently outputs new mass and size details for the vehicle. The process was repeated until converged. The final take-off mass found is 136,079 kg and the dry mass (vehicle mass without payload and fuel) equals 58,968 kg (Powell et al., 1991).

Model Implementation

The difference between the take-off mass m_0 and the dry mass m_{dry} is the mass that can be taken on board in the form of fuel or payload. The total take-off mass can thus be summed up according to:

$$m_0 = m_{dry} + m_{fuel} + m_{pl} \tag{3.12}$$

where m_{fuel} represents the total fuel mass on board and m_{pl} is the payload mass. Initially the assumption is made that all space is available for fuel. During flight, fuel is consumed and the mass of the vehicle will decrease. The fuel-mass rate can be calculated with Equation 3.10. When integrated over the flight time, the total mass of the consumed fuel for the ascent m_{cons} during flight can be calculated.

$$m_{cons} = \int_0^t \dot{m}\, dt \tag{3.13}$$

A fuel-mass penalty must be added to this value for the required ΔV to circularize the orbit. This mass penalty can be found with Equation 3.11 as described in Section 3.2.1 and will be referred to as m_{circ}. The vehicle mass at the final state can now be calculated with:

$$m = m_0 - m_{cons} - m_{circ} \tag{3.14}$$

The required fuel on board m_{fuel} must equal the total consumed fuel for ascent m_{cons} and circularization m_{circ}. To find the maximum payload mass that could be taken on board, Equation 3.12 can now be rewritten to the following form:

$$m_{pl} = m_0 - m_{dry} - m_{fuel}$$

When a positive mass is the result, the trajectory can be flown. When the result is negative, not enough fuel can be taken on board to fly the trajectory. The trajectory is therefore impossible to fly. Equation 3.12 shows that a minimum fuel trajectory is directly related to maximizing the payload capacity.

The change in vehicle mass has an effect on the center of mass (c.o.m) and the moments of inertia (I_{xx}, I_{yy} and I_{zz}) around the body axes. It is assumed that the c.o.m. only moves along the x-axis, and the location is measured positive aft from the moment reference center. As fuel is depleted, x_{com} increases and thus moves backwards, while the moments of inertia slowly decrease.

Model Limitations

The mass model has some limitations that are closely linked to the ones already mentioned for the propulsion model. Both the implementation of TVC and the rocket engine required to circularize the orbit are not present in the original configuration. The addition of these components in the mass model would most likely increase the dry weight of the vehicle and shift the center of gravity backwards.

The total take-off mass of 136,079 kg also seems to be optimistic compared to some of the projects discussed in Section 2.1.1. The take-off mass of Skylon, which has a similar design, is almost 2.4 times higher with 325 tonnes. Considering the fact that Powell et al. (1991), Lu (1991) and Mooij (1998) were all able to find trajectories with large payload masses quite easily, it can be concluded that the WCC is a very optimistic model. It might therefore be interesting to include mass penalties for the effects mentioned above. This will result in a more marginal payload capacity, which coincides better with the reality.

3.2.3. Aerodynamic

The aerodynamic characteristics of the WCC are described by Shaughnessy et al. (1990) and will be explained here. The Aerodynamic Preliminary Analysis System (APAS) was used to obtain the aerodynamic model. APAS is a subsonic/supersonic/hypersonic analysis code developed by NASA Langley and Rockwell International Inc (Sova et al., 1991). The Unified Distributed Panel (UDP) module (Bonner et al., 1991) was used up to velocities of Mach 1.5. It uses slender body theory and vortex panels to evaluate the contributions of the wings and tail. In the high supersonic and hypersonic range, the Hypersonic Arbitrary Body Program (HABP) (Cruz and Wilhite, 1989) is used to compute inviscid and viscous solutions on each of the finite elements of the model. Integration of the pressure and shear forces resulted in total force and moment coefficients. For pressure calculations on the impact surfaces, the tangent-cone and tangent-wedge methods were used for fuselage and wing components respectively. Prandtl-Myer expansion was used for pressure calculations of the shadow surfaces. Viscous shear forces were estimated with the reference enthalpy method.

Model Implementation
The data computed with APAS is visualized in graphs. The values of the coefficients are given as function of angle of attack α and Mach number M, and for the control surfaces also as a function of deflection angle δ. They can be used to calculate forces and moments relative to the moment reference center. With a small correction, the moments relative to the center of gravity can also be calculated. Implementations to calculate drag and lift forces and the pitching moment will be discussed here. Roll and yaw moments will not be used. Furthermore, the vehicle is assumed to fly at a zero angle of side-slip. The side force is therefore also left out.

Drag Force
The total drag force that acts on the moment reference center can be calculated with:

$$D = \bar{q} S_{ref} C_D \tag{3.15}$$

where S_{ref} is the reference area equal to the wing area and C_D is the total drag coefficient. The total drag coefficient is the sum of multiple increment coefficients as given by:

$$C_D = C_{D,f} + dC_{D,el} + dC_{D,r} + dC_{D,c} \tag{3.16}$$

where $C_{D,f}$ is the drag coefficient of the basic vehicle fuselage and the other drag increment coefficients represent the elevons $dC_{D,el}$, rudder $dC_{D,r}$, and the canards $dC_{D,c}$.

Lift Force
The total lift force acting on the moment reference center can be calculated in a similar way with:

$$L = \bar{q} S_{ref} C_L \tag{3.17}$$

where S_{ref} is still the reference wing area and C_L is the total lift coefficient. C_L is found by adding all lift increment coefficients according to:

$$C_L = C_{L,f} + dC_{L,el} + dC_{L,c} \tag{3.18}$$

where $C_{L,f}$ is the lift coefficient of the basic vehicle fuselage and $dC_{L,el}$, and $dC_{L,c}$ represent the lift increment coefficients of the elevons and canards respectively.

Pitching Moment
The pitching moment $\bar{M}_{mrc}$ about the moment reference center as a result of the total pitch moment coefficient C_m is given by:

$$\bar{M}_{mrc} = \bar{q} c S_{ref} C_m \tag{3.19}$$

where c is a longitudinal reference length, for which the mean aerodynamic chord is used. To find the pitching moment $\bar{M}$ about the center of gravity , the effects of the lift and drag forces need to be taken into account. Both forces are acting on the moment reference center. Their components acting along the body z-axis multiplied with the distance x_{com} between the m.r.c. and the c.o.m. result in an additional moment as indicated by:

$$\bar{M} = \bar{M}_{mrc} + x_{com}(D \sin\alpha + L \cos\alpha) \tag{3.20}$$

It can be seen that the lift and drag components along the z-axis depend on the angle of attack α. The total pitch moment coefficient is found by summing the pitch moment increment coefficients of

the basic vehicle fuselage $C_{m,f}$, the elevons $dC_{m,el}$, the rudder $dC_{m,r}$ and the canards $dC_{m,c}$, and the influence of the pitch rate dynamic derivative $C_{m,q}$, resulting in:

$$C_m = C_{m,f} + dC_{m,el} + dC_{m,r} + dC_{m,c} + C_{m_q}\left(\frac{qc}{2V}\right) \tag{3.21}$$

where $\left(\frac{qc}{2V}\right)$ is the non-dimensional pitch rate, with q being the pitch rate. The product of the pitch rate derivative and the non-dimensional pitch rate indicates the change in pitching moment as a result of the pitch rate. It is noted that the influence of the rotational motion is neglected and the last term will thus disappear from the equation.

Model Limitations
A few simplifying assumptions have been made in the aerodynamic model described. Shaughnessy et al. (1990) state that, ”changes in lift, drag and side force due to body angular rates and coupling between control surface deflections and sideslip are assumed negligible”. This assumption is accepted and not further examined. The effects of the rotational motion of the vehicle on the moment coefficients were also neglected. Since only a 3 DOF integration is performed to obtain the trajectory, the rotational rates will not be solved. The vehicle is assumed to be instantly trimmed at each new state and the effect of the angular rates on the aerodynamics is not taken into account. Furthermore, according to Oppenheimer et al. (2008), the slipstream from the trailing edge of the canards can have a significant interaction with the elevons, decreasing their efficiency. This change in elevon efficiency as a result of canard deflection is not included. However, it has to be noted that the canards are only deployed at subsonic speeds and the effect is thus only present at a very small part of the trajectory. Finally, a strong interaction exists between the air-breathing engine and the aerodynamics, and the addition of a system allowing TVC will have an influence on the aerodynamics of the vehicle as well. These effects are neglected for the current study.

3.2.4. Trim
The vehicle is said to be trimmed when no rotational moment around the c.o.m. is present. Three trim control parameters are present: the elevons, the canards and the rudder. Because the vehicle will stay in a vertical plane along the equator, yaw and roll moments are not required to trim and the rudder is therefore not used. TVC is added as an extra trim control parameter as an alternative method to produce a pitch moment. The implementation and limitations of all trim control parameters will be discussed next.

Elevons
The vehicle wings are positioned at the center-line of the fuselage and 2 elevons are located at the trailing edges of the wings, with their hinge line perpendicular to the vehicle center line. The configuration is shown in Figure 2.10. The elevons can be controlled independently. When both elevons are moved together, they act like elevators, producing a pitching moment. When both elevons are not deflected equally, they also act like ailerons, producing a rolling moment. As roll is not required for a vertical plane ascent, the latter option will not be used and both elevons were therefore represented by a single lift, drag and pitch coefficient.

Shaughnessy et al. (1990) give the influence of the elevon deflection on the moment coefficients. Values are shown for a deflection of -20, -10, 0, 10 and 20 degrees. The deflections of the elevons are measured with respect to the hinge line, and positive deflections have the trailing edge up. Because no information is available for deflections larger than $\pm$ 20°, the allowable elevon deflections for the configuration are assumed to lie in this range. Furthermore, the studies by Powell et al. (1991) and Mooij (1998) show that larger deflections are not required for the launch trajectory.

Canards
The canards are also mounted at the center-line of the fuselage near the front as is shown in Figure 2.10. The canards are only deployed at subsonic velocities for added stability. Shaughnessy et al. (1990) give the influence of the canards deflection on the moment coefficients. Values are shown for deflections of -10, -5, 0, 5 and 10 degrees. The deflections are measured with respect to the center-line of the fuselage and positive deflection have the trailing edge down. As with the elevons, both the left and right canard are always deflected at the same angle.

TVC
TVC will be added as an additional control parameter for the pitch moment. The implementation of TVC has already been discussed in Section 3.2.1. The c.o.t. is assumed to be located on the x-axis at one third of the nozzle section length from the aft of the vehicle. The thrust-elevation angle ϵ_T is measured with respect to the x-y plane, and is taken positive when the exhaust gasses are deflected upwards. The pitching moment about the c.o.m. caused by TVC can now be calculated with Equation 3.22.

$$\bar{M}_{TVC} = (x_{cot} - x_{com})\, T\, \sin\epsilon_T \tag{3.22}$$

The technical feasibility of implementing TVC is not taken into account. Information on the maximum deflection angle possible for this configuration is not available. Because this study is only concerned with the effect of TVC on the trajectory, feasibility is not a priority. For the sake of consistency, a maximum deflection angle of $\pm$ 20° is also used for ϵ_T.

It is noted that the azimuth angle of thrust ψ_T could also be used for TVC. However, this will only be useful to create a yawing moment to steer the vehicle. Since the vertical plane ascent does not require any yawing moments, this control parameter in not taken into account.

Trim Moments
Because no guidance system will be used for the trajectory optimization, the trim of the vehicle is modelled to be instantaneous. The sum of the moments around each body axis is set to zero at every time step. In this study, only pitch is used and the pitch moment coefficient should thus be set to zero. For the pitch moment, not only the moment due to the control surface deflections needs to be taken into account, but also the moment caused by TVC. From Equations 3.19 and 3.22 the pitch moment coefficient due to TVC $C_{m,TVC}$ about the m.r.c. can be determined with:

$$C_{m,TVC} = \frac{x_{cot}\, T\, \sin\epsilon_T}{\bar{q} c S_{ref}} \tag{3.23}$$

The sum of the total pitch moment coefficient should be zero at all time steps resulting in:

$$C_{m,TVC} + C_m = 0 \tag{3.24}$$

3.2.5. Vehicle Parameters
Table 3.1 summarizes the allowable range of the control parameters and states the minimum and maximum Mach, angle of attack and dynamic pressure for which the aerodynamic coefficients are defined. It is noted that maximum Mach number might be exceeded during the ascent trajectory. In these cases, the table is not extrapolated, but the maximum given table values will be used.

parameter	minimum value	maximum value
δ_{el} [deg]	-20.0	20.0
δ_c [deg]	-10.0	10.0
ϵ_T [deg]	-25.0	25.0
α [deg]	-1.0	12.0
M [-]	0.0	24.2
$\bar{q}$ [N/m^2]	0.0	239,401

Table 3.1: Minimum and maximum values of control parameters and independent variables.

4

Flight Dynamics

The previous chapter discussed the models that will be used to compute the external forces acting on the vehicle. In this chapter the resulting forces are used to set up the equations of motion (EOM). The EOM will eventually be propagated in the simulation model to find the trajectory, which is further explained in Chapter 6.

To set up the EOM, the state variables used to express the vehicles position, velocity and attitude need to be discussed first. This is done in Section 4.1. The external forces on the plane, found with the models discussed in the last chapter, and the resulting motions are defined in a number of different reference frames. The required reference frames and the transformation between them will be discussed in Section 4.2. The equations of motion are then given in Section 4.3. Finally the guidance methods used to find the desired ascent trajectories are discussed in Section 4.4.

4.1. State Variables

In this section the state variables will be discussed. For position and velocity, one can use orbital elements, Cartesian components or spherical components. For the attitude Quaternions, classical Euler angles or aerodynamic angles can be used. The position and velocity state variable choice will be justified here and the chosen set will be further explained. The same will then be done for the attitude variables.

4.1.1. Position and Velocity

As mentioned in the section introduction, three sets of state variables are available. Orbital elements are most commonly used for vehicles that are in orbit around a planet, moon or star. For this study, the vehicle will only reach orbit at the very end of the trajectory. Orbital elements are therefore not convenient for the study of an ascent trajectory. Orbital elements can be convenient if a specific orbit is targeted. Conversion to orbital elements will also be useful when the continuation of the mission in space is studied. However, for the current research this is not the case and orbital elements will for that reason not be further discussed

Cartesian components are easy to implement and fast for simulations. Furthermore, Cartesian components are more robust for simulations, because singularities do not occur. On the other hand, interpretation of the results is easier in spherical components (Mooij, 1994). For that reason it is decided to use Cartesian components for the simulation and convert to spherical components for the interpretation of the results. A more in depth discussion of this choice will follow in Section 4.3.

Cartesian components can be used to express position and velocity with respect to both the inertial and rotating frame. In both frames the position is given with variables x, y and z, and the velocity is expressed with variables $\dot{x}$, $\dot{y}$ and $\dot{z}$. Figure 4.1 shows the spherical state variables for position and velocity with respect to the rotating planetocentric frame. For the position we have:

- r: The distance from the c.o.m. of the central body to the c.o.m. of the vehicle.
- τ: The longitude measured from the zero-meridian and positive in eastern direction with $(-180^\circ \leq \tau < 180^\circ)$.

δ: The latitude measured from the equatorial plane and positive in northern direction with ($-90° \leq \delta \leq 90°$).

For the velocity the following variable are defined:

V: The magnitude of the velocity relative to the rotating frame.
χ: The heading angle defined as the direction of the velocity component projected on the local horizontal plane with respect to the local vector pointing north. The angle is measured positive from north towards the east and ($-180° \leq \chi < 180°$)
γ: The flight-path angle defined as the angle between the velocity vector $\boldsymbol{V}$ and the local horizontal plane with ($-90° \leq \gamma \leq 90°$) and a positive angle when the velocity vector points above the local horizon.

The mission defined in Section 2.4, which will be used throughout this research, always follows the equator in eastern direction. As a result, the latitude δ will always be equal to 0 degrees and the heading angle χ is always 90 degrees. This will allow for some simplifications in the transformation matrices and the equations of motion that will be discussed in the next sections.

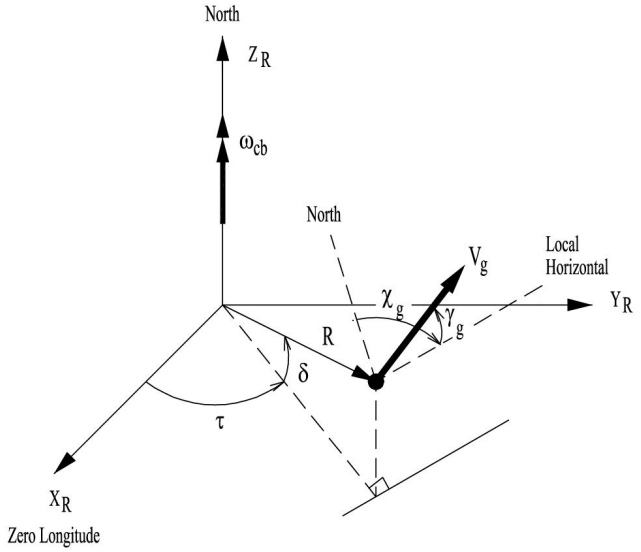


Figure 4.1: Definition of the six spherical flight parameters for position (r, τ, δ) and velocity (V, χ, γ), with the angles shown in positive direction (Mooij, 2015).

4.1.2. Attitude

To represent the attitude one can use Quaternions, classical Euler angles or aerodynamic angles. Aerodynamic angles will be used throughout this study, because they can be directly used for calculation of the aerodynamic forces, and they result in frame transformation matrices that are easily interpretable. The transformation matrices will be further discussed in Section 4.2.2. The aerodynamic angles are shown in Figure 4.2 and defined as:

α: The angle of attack defined as the angle between the X_B-axis and the aerodynamic $X_A Y_A$-plane, with ($-180° \leq \alpha < 180°$) and a positive angle of attack with the nose up.
β: The side-slip angle defined as the angle between the Y_B-axis and the Y_A-axis, with ($-90° \leq \beta \leq 90°$) and a positive angle of side-slip with the nose left.
σ: The bank angle defined as the Y_A- and the Y_T-axis, with ($-180° \leq \sigma < 180°$) and a positive angle when the right wing points down.

It is noted that α is used as a guidance parameter and will thus vary throughout the trajectory. Meanwhile, β and σ are only mentioned for completeness, and will always be set equal to zero. This will again allow for simplification in the transformation matrices and equations of motion discussed next.

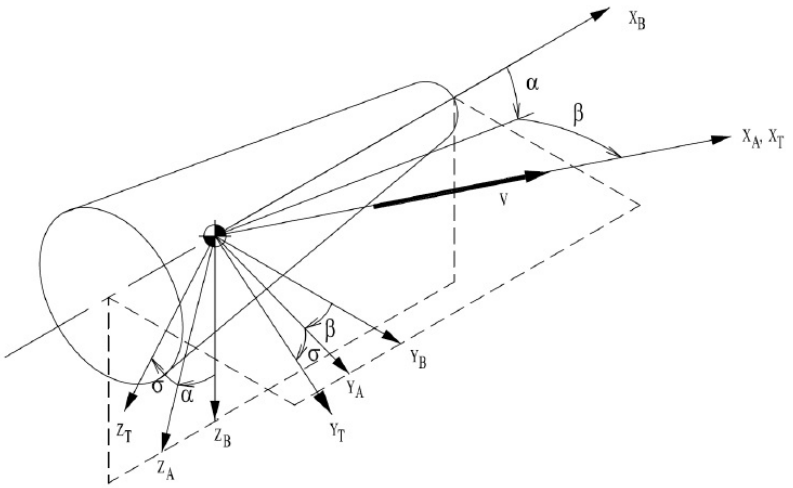


Figure 4.2: Relation between the body (B), aerodynamic (A) and trajectory (T) frames with aerodynamic angles α, β and σ shown in positive direction (Mooij, 2015).

4.2. Reference Frames and Frame Transformations

In this section the reference frames will be discussed. Section 4.2.1 will first shortly describe the relevant available frames given by Mooij (1994). Section 4.2.2 will then give the transformation matrices to be able to transform forces and moments expressed in one reference frame to another frame.

4.2.1. Reference Frames

Before the reference frames are discussed, it is noted that wind will not be a factor in this research. The distinction between ground-speed and airspeed made by Mooij (1994) will therefore not be made here. Furthermore, the wind frame will not be required. The following frames are relevant.

Inertial Planetocentric Reference Frame

The inertial planetocentric reference frame has its origin at the center of mass of the planet, which in this case is Earth. The frame is indicated with an index I. Multiple inertial frames are defined. A commonly used inertial frame for the Earth is the J2000-frame. The X_I- and the Y_I-axes lie in the equatorial plane, with the X_I-axis pointing to Aries at 12:00 on the first of January 2000. The Z_I-axis runs perpendicular to the equatorial plane pointing north and the Y_I axis completes the right handed system. Because the initial rotation of the frame does not influence the trajectory results, the X_I-axis is defined here to point at the zero-longitude meridian at time zero of the simulation. This is done for convenience.

It is noted that this reference frame is in reality a pseudo-inertial frame, because the Earth (or any planet) also rotates around the Sun. This rotation only has a marginal effect on the results and it is therefore safe to treat it as an inertial frame. The equations of translational motion are expressed in the inertial reference frame.

Rotating Planetocentric Reference Frame

The rotating frame coincides with the inertial planetocentric frame at time zero and is indicated with an index R. The frame is fixed to the planet and rotates around the Z_R-axis. Every full rotation it coincides again with the inertial frame.

The external force due to gravity is expressed in the rotating frame. Furthermore, the equations of translational motion are expressed in the rotating frame for interpretation of the results as will be explained shortly.

Body Reference Frame

The body reference frame has its origin at the center of mass of the vehicle and is indicated with an index B. The X_B-axis lies in the plane of symmetry and is pointing to the front of the vehicle. The

Z_B-axis lies in the same plane and is pointing down. The Y_B-axis completes the right handed system.

The aerodynamic moments, and the apparent forces and moments due to the mass-varying propulsion system are expressed in the body-frame.

Propulsion Reference Frame
The propulsion reference frame is indicated with an index P and is closely related to the body reference frame. The X_P-axis is collinear with the thrust-vector. The Y_P- and Z_P-axis follow from the rotations with respect to the body frame. It is noted that the propulsion frame coincides with the body frame when the thrust-vector is aligned with the X_B-axis.

Vertical Reference Frame
The vertical reference frame is indicated with an index V. The origin is located at the c.o.m. of the vehicle and the Z_V-axis is pointing towards the c.o.m. of the central body. The X_V-axis lies in a meridian plane perpendicular to the Z_V axis and is pointing north. The Y_Z-axis completes the right handed system.

No external forces or equations of motion will be expressed in the vertical reference frame. However, the vertical frame is a convenient intermediate step to transform from the vehicle centered frames to the planetocentric frames. These transformations will be further discussed in Section 4.2.2.

Aerodynamic Reference Frame
The aerodynamic reference frame is indicated with index A and has its origin at the c.o.m. of the vehicle. The X_A-axis is collinear with the X_T-axis. The Z_A-axis is collinear with the lift vector but in the opposite direction. The Y_A-axis completes the right handed system.

It is noted that the aerodynamic frame is identical to the trajectory frame when the bank angle is zero. This is the case for the currently defined mission and the trajectory frame is therefore not discussed. The aerodynamic lift, drag and side force will be expressed in the aerodynamic frame.

4.2.2. Frame Transformations

To set up the equations of motion it is important to be able to transform forces expressed in one frame to another frame. These transformations can be done through unit axis-rotations. This will be explained first, after which some important transformations will be given. Finally, it will be shown that combining these transformations will result in all important transformations required to set up the equations of motion in the inertial or rotating reference frame. The information in this section is based on Mooij (1994).

Unit Axis-rotations
Unit axis-rotations are used to rotate a reference frame around a single axis. In matrix form, the rotation of frame 1 around the unit axis X_1 by angle θ to obtain reference frame 2 looks like:

$$\begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix} \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} \tag{4.1}$$

The rotation matrix in Equation 4.1 can also be denoted by $\boldsymbol{C_1}(\theta)$:

$$\boldsymbol{C_1}(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix} \tag{4.2}$$

In a similar way the rotations around the Y- and Z-axes by an angle θ can be found with rotation matrices $\boldsymbol{C_2}(\theta)$ and $\boldsymbol{C_3}(\theta)$ respectively:

$$\boldsymbol{C_2}(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{4.3}$$

$$\boldsymbol{C_3}(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.4}$$

A complete rotation around multiple axes can be found by combining the unit axis rotations. For example: if the transformation from frame 1 to 2 requires a rotation -θ_1 around the Y-axis, θ_2 around the X-axis and θ_3 around the Z-axis, the total rotation matrix $\boldsymbol{C_{2,1}}$ is found with:

$$\boldsymbol{C_{2,1}} = \boldsymbol{C_3}(\theta_3)\boldsymbol{C_1}(\theta_2)\boldsymbol{C_2}(-\theta_1) \tag{4.5}$$

The transformation from reference frame B back to frame A can be easily found with the inverse rotation matrix:

$$\boldsymbol{C_{1,2}} = \boldsymbol{C_{2,1}^{-1}} = \boldsymbol{C_{2,1}^{T}} \tag{4.6}$$

In terms of the unit axis rotations, it is clear that the same rotations will need to be performed as in Equation 4.5, but now in opposite order and direction. The inverse transformation can be written as:

$$\boldsymbol{C_{1,2}} = \boldsymbol{C_2}(\theta_1)\boldsymbol{C_1}(-\theta_2)\boldsymbol{C_3}(-\theta_3) \tag{4.7}$$

When looking at the individual unit axis rotation matrices, it can be shown why $\boldsymbol{C_{1,2}} = \boldsymbol{C_{2,1}^{T}}$. The only difference between the transpose of these matrices and their original forms, is the changing sign in front of the $\sin$-terms. Because $\cos\theta = \cos -\theta$, it follows that:

$$\begin{aligned} \boldsymbol{C_1}^T(-\theta_2) &= \boldsymbol{C_1}(\theta_2) \\ \boldsymbol{C_2}^T(-\theta_1) &= \boldsymbol{C_2}(\theta_1) \\ \boldsymbol{C_3}^T(-\theta_3) &= \boldsymbol{C_3}(\theta_3) \end{aligned}$$

Taking the transpose of the individual matrices and multiplying them in opposite order will lead to the total transpose rotation matrix. This is exactly what is done in Equation 4.7 with respect to Equation 4.5, which indicates the validity of the statement in Equation 4.6.

The unit axis rotation matrices explained here will be used to find the rotation matrices required for transformation of the reference frames discussed in Section 4.2.1.

Rotating to Inertial Frame

The only difference between the rotating and the inertial reference frame is the rotation around the Z_R-axis. To transform from one frame to the other, the following variables are required:

ω_{cb} = rotational rate of the central body around its axis [rad/s]
t = time [s]

The variables with their positive directions are shown in Figure 4.3. The frames coincide at $t_0 = 0$, and the transformation matrix at an arbitrary time t only consists of a rotation around the Z_R-axis given by:

$$\boldsymbol{C_{I,R}} = \boldsymbol{C_3}(-\omega_{cb}t) = \begin{bmatrix} \cos(\omega_{cb}t) & -\sin(\omega_{cb}t) & 0 \\ \sin(\omega_{cb}t) & \cos(\omega_{cb}t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.8}$$

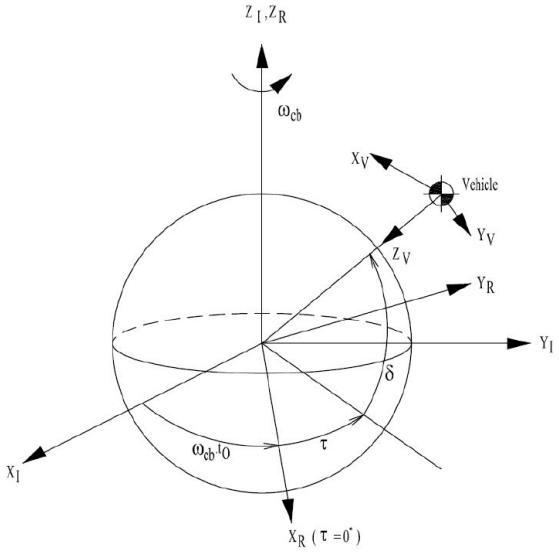


Figure 4.3: Relation between the inertial (I) and rotating (R) planetocentric frames and the vertical frame (V) (Mooij, 2015).

Vertical to Rotating Frame

Figure 4.3 shows the relation between the vertical and the rotating frames, including the transformation variables in positive direction. It is clear that the following variables are required for the transformation:

τ = planetocentric longitude [rad]
δ = planetocentric latitude [rad]

The transformation requires a rotation around the Y_V-axis first, followed by a rotation around the Z_V-axis. Knowing that δ = 0 throughout the flight, this results in:

$$\boldsymbol{C_{R,V}} = \boldsymbol{C_3}(-\tau)\boldsymbol{C_2}\left(\frac{\pi}{2}+\delta\right) = \boldsymbol{C_3}(-\tau)\boldsymbol{C_2}\left(\frac{\pi}{2}\right) = \begin{bmatrix} 0 & -\sin\tau & -\cos\tau \\ 0 & \cos\tau & -\sin\tau \\ 1 & 0 & 0 \end{bmatrix} \tag{4.9}$$

where the following relations were used:

$$\cos\left(\frac{\pi}{2}+\delta\right) = -\sin\delta$$
$$\sin\left(\frac{\pi}{2}+\delta\right) = \cos\delta$$

Aerodynamic to Vertical Frame

To transform from the Aerodynamic to the vertical frame, the following variables are required:

γ = flight-path angle [rad]
χ = heading angle [rad]

The flight-path angle is measured positive upwards from the local horizontal plane. The heading angle is measured with respect to a vector pointing north and positive in a clockwise direction. It is clear that a transformation from the trajectory to the vertical plane requires a rotation of -γ about the Y_T-axis and -χ about the Z_T-axis:

$$\boldsymbol{C_{V,A}} = \boldsymbol{C_3}(-\chi)\boldsymbol{C_2}(-\gamma) = \begin{bmatrix} \cos\chi\ \cos\gamma & -\sin\chi & \cos\chi\ \sin\gamma \\ \sin\chi\ \cos\gamma & \cos\chi & \sin\chi\ \sin\gamma \\ -\sin\gamma & 0 & \cos\gamma \end{bmatrix} \tag{4.10}$$

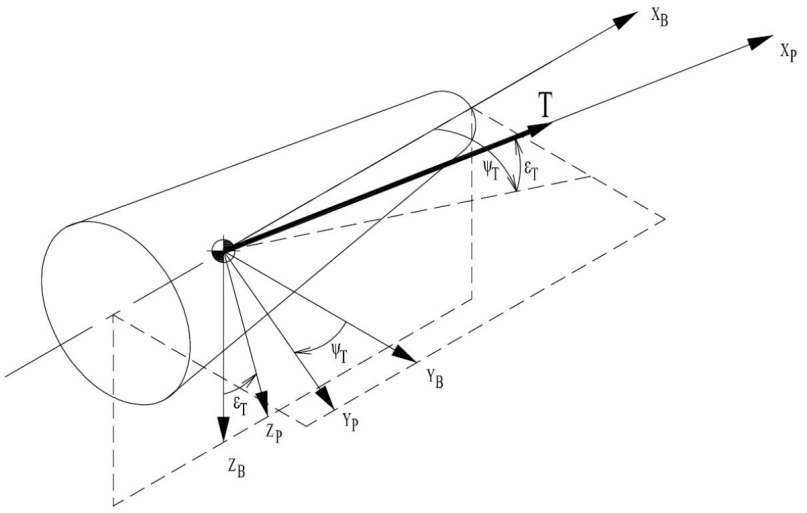


Figure 4.4: Relation between the body (B) and propulsion (P) frames with the indicated angles ϵ_T and ψ_T shown in positive direction (Mooij, 2015).

Body to Aerodynamic Frame
From Figure 4.2 it becomes clear that the body and aerodynamic reference frames are related by the variables:

α = angle of attack [rad]
β = side-slip angle [rad]

The transformation requires first a rotation of -α around the Y_B-axis and subsequently a rotation of β around the Z_B-axis. Knowing that β = 0, this results in :

$$\boldsymbol{C_{A,B}} = \boldsymbol{C_3}(\beta)\boldsymbol{C_2}(-\alpha) = \boldsymbol{C_2}(-\alpha) = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} \tag{4.11}$$

Propulsion to Body Frame
The relation between the body and the propulsion reference frames is shown in Figure 4.4. The required variables for the transformation are:

ϵ_T = elevation angle of thrust [rad]
ψ_T = azimuth angle of thrust [rad]

To transform from the propulsion frame to the body frame, a rotation of -ϵ_T around the Y_P-axis will be performed first. Subsequently a rotation of -ψ_T around the Z_P-axis is required. However, for the current research ψ_T will not be used resulting in:

$$\boldsymbol{C_{P,B}} = \boldsymbol{C_3}(-\psi_T)\boldsymbol{C_2}(-\epsilon_T) = \boldsymbol{C_2}(-\epsilon_T) = \begin{bmatrix} \cos\epsilon_T & 0 & \sin\epsilon_T \\ 0 & 1 & 0 \\ -\sin\epsilon_T & 0 & \cos\epsilon_T \end{bmatrix} \tag{4.12}$$

Combined Transformations
The transformation matrices obtained above can now be combined to find more complex transformations. Eventually, forces and moments expressed in the propulsion and aerodynamic frame will need to be transformed to the rotational or inertial frame for the equations of motion. The following combined transformations will be useful:

$$C_{R,A} = C_{R,V} C_{V,A} \tag{4.13}$$
$$C_{R,B} = C_{R,A} C_{A,B} \tag{4.14}$$
$$C_{R,P} = C_{R,B} C_{B,P} \tag{4.15}$$

To transform further to the inertial frame, $C_{I,R}$ should be applied to the the equations obtained above:

$$C_{I,A} = C_{I,R} C_{R,A} \tag{4.16}$$
$$C_{I,B} = C_{I,R} C_{R,B} \tag{4.17}$$
$$C_{I,P} = C_{I,R} C_{R,P} \tag{4.18}$$

4.3. Equations of Motion

In this section the equations of motion (EOM) will be set up. For the translational motion a set of three dynamic and three kinematic equations will be found. These equations can be defined in the inertial frame or rotating frame in both Cartesian and spherical coordinates.

For the simulation the EOM will be set up in the inertial frame in Cartesian components. The resulting state will need to be transformed to the rotational frame with the earlier discussed transformation matrices. This is the most robust form for programming as it is the simplest set of equations and no singularities are present. However, spherical components are better suited for interpretation of the results. Furthermore, implementing the EOM in the rotational frame will result in a set of equations that can be interpreted without any further transformations.

Based on the observations stated above, the EOM will be set up in Cartesian components in the inertial frame to be used in the simulation software. Subsequently the conversion between Cartesian components in the inertial frame and spherical components in the rotational frame is described for interpretation of the results.

External Forces

During its flight the vehicle is subject to a number of external forces. The external forces F_I include the gravitational, aerodynamic and propulsive forces. These forces will need to be transformed using the transformation matrices found in Section 4.2.2 to the inertial and rotational frame respectively for the EOM to be set up next. Furthermore, two apparent forces due to the varying mass of the system are also present. These forces can be attributed to the propulsion system and are the Coriolis force F_C and the relative force F_{rel}. The derivation can be found in Mooij (1994) resulting in:

$$F_C = -2\boldsymbol{\omega} \times (\dot{m}_{in} \boldsymbol{r_{in}} + \dot{m}_e \boldsymbol{r_e}) \tag{4.19}$$

$$F_{rel} = -(\dot{m}_{in} \bar{V}_{in} + \dot{m}_e \bar{V}_e) \tag{4.20}$$

where $\boldsymbol{\omega}$ is the rotational rate of the vehicle, $\dot{m}_{in}$ is the mass flow of the incoming air, $\dot{m}_e$ is the mass flow of the outgoing fuel-air mixture, $\bar{V}_{in}$ is the velocity vector of the incoming flow and $\bar{V}_e$ is the velocity vector of the outgoing flow. Because of the low rotation rates of a space-plane, F_C will be very small compared to the other external forces. This has been verified in the study by Mooij (1998) and the influence of F_C will therefore be neglected. F_{rel} represents the impulse thrust and is (together with the pressure term) the main thrust force already extracted from the propulsion model described in Section 3.2.1. The total external force can thus be represented by:

$$F_I = F_{A,I} + F_{G,I} + F_{T,I} \tag{4.21}$$

where $F_{A,I}$, $F_{G,I}$ and $F_{T,I}$ represent the aerodynamic, gravitational and propulsive forces expressed in the inertial frame, respectively.

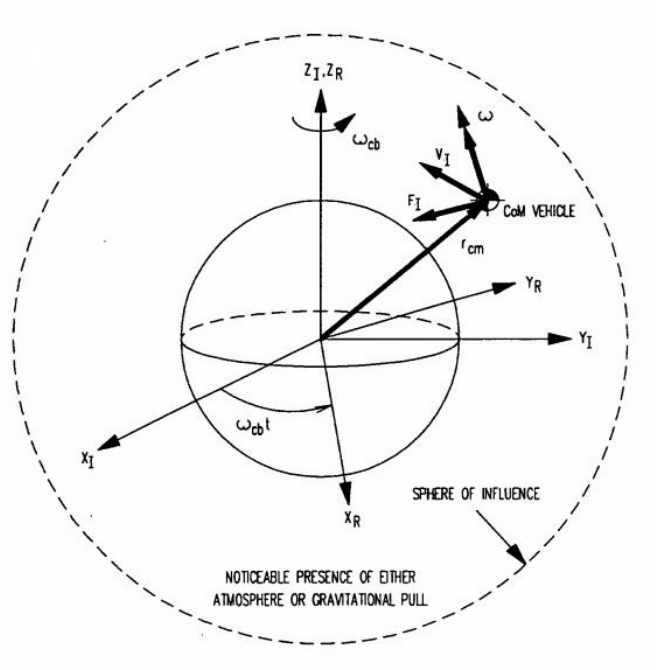


Figure 4.5: Definition of a vehicle moving in the sphere of influence of a celestial body w.r.t. the inertial (I) and rotation (R) planetocentric frames (Mooij, 1994).

Cartesian EOM in Inertial Frame

To set up the equations, the approach of Mooij (1994) is followed. It is assumed that the vehicle is non-elastic, and has a variable mass m due to the propulsion system. The vehicle has a distance $\boldsymbol{r_{cm}}$ from the c.o.m. of the central body and is moving with a velocity of $\boldsymbol{V_I} = (\dot{x}_I, \dot{y}_I, \dot{z}_I)$ with respect to the inertial planetocentric frame. The vehicle has a rotation of $\boldsymbol{\omega}$ and is subjected to the total external force $\boldsymbol{F_I}$. All of this is visualized in Figure 4.5.

For the translational motion with respect to the inertial frame, Newton's second law prescribes:

$$\boldsymbol{F_I} = m\frac{d^2\boldsymbol{r_{cm}}}{dt^2} \tag{4.22}$$

where we can substitute $\frac{d\boldsymbol{r_{cm}}}{dt} = \boldsymbol{V_I}$, and after reorganizing the resulting dynamic equation of motion is:

$$\frac{d\boldsymbol{V_I}}{dt} = \begin{pmatrix} \ddot{x}_I \\ \ddot{y}_I \\ \ddot{z}_I \end{pmatrix} = \frac{1}{m}\boldsymbol{F_I} \tag{4.23}$$

with the kinematic equation of motion given by:

$$\frac{d\boldsymbol{r_{cm}}}{dt} = \begin{pmatrix} \dot{x}_I \\ \dot{y}_I \\ \dot{z}_I \end{pmatrix} = \boldsymbol{V_I} \tag{4.24}$$

Conversion Between Spherical and Cartesian Components

To be able to interpret the results from the simulation, the state will have to be converted from Cartesian components in the inertial frame to spherical components in the rotating frame. Transforming the position is easy. First a transformation matrix is applied to transform from the inertial to the rotating frame:

$$\boldsymbol{X_R} = \boldsymbol{C_{R,I}}\boldsymbol{X_I} \tag{4.25}$$

Subsequently the following equations can be used to find the position in spherical components with respect to the rotating frame:

$$r = \sqrt{x_R^2 + y_R^2 + z_R^2} \tag{4.26a}$$

$$\tau = \tan^{-1}\left(\frac{y_R^2}{x_R^2}\right) \tag{4.26b}$$

$$\delta = \sin^{-1}\left(\frac{z_R^2}{\sqrt{x_R^2 + y_R^2 + z_R^2}}\right) \tag{4.26c}$$

The inverse relation is given by:

$$x_R = r\cos\delta\ \cos\tau \tag{4.27a}$$

$$y_R = r\cos\delta\ \sin\tau \tag{4.27b}$$

$$z_R = r\sin\delta \tag{4.27c}$$

Transforming the velocity is a little bit more complicated. First the velocity is transformed from the inertial frame to the rotating frame. Besides the frame transformation, the velocity caused by the rotating Earth has to be subtracted leading to:

$$\boldsymbol{V_R} = \boldsymbol{C_{R,I}}\left(\boldsymbol{V_I} - \boldsymbol{\omega_{cb}} \times \boldsymbol{X_I}\right) \tag{4.28}$$

With $\boldsymbol{V_R} = (\dot{x}_R, \dot{y}_R, \dot{z}_R)^T$, the magnitude of the velocity can easily be found with:

$$V = \sqrt{\dot{x}_R^2 + \dot{y}_R^2 + \dot{z}_R^2} \tag{4.29}$$

To find the direction angles γ and χ, the velocity will first be transformed to the vertical frame:

$$\boldsymbol{V_V} = \boldsymbol{C_{V,R}}\boldsymbol{V_R} \tag{4.30}$$

Knowing that γ has a restricted value between -90° and 90°, the direction angles now follow from:

$$\gamma = -\sin^{-1}\left(\frac{\dot{z}_V}{V}\right) \tag{4.31a}$$

$$\chi = \tan^{-1}\left(\frac{\dot{y}_V}{\dot{x}_V}\right) \tag{4.31b}$$

More details on the derivations can be found in Mooij (1994). The inverse relations are given by:

$$\dot{x}_V = V\cos\gamma\ \cos\chi \tag{4.32a}$$

$$\dot{y}_V = V\cos\gamma\ \sin\chi \tag{4.32b}$$

$$\dot{z}_V = -V\sin\gamma \tag{4.32c}$$

4.4. Vehicle Guidance

To fly a maximum payload trajectory, the vehicle has to be manoeuvred in such a way that flight-path constraints are not violated while the fuel mass and heat load throughout the flight are minimized. A guidance model is therefore required. The guidance model generates steering commands and computes deflections of control surfaces and thrust to ensure trimmed conditions throughout the flight.

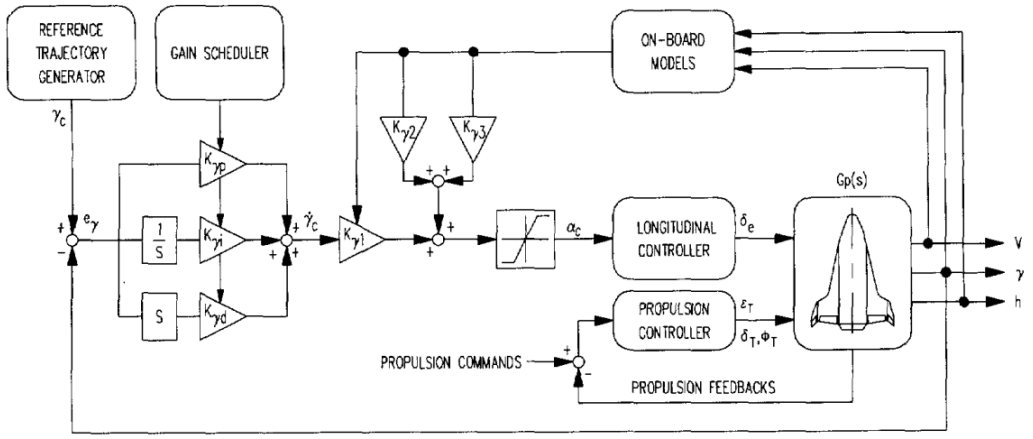


Figure 4.6: Inverse dynamic feedback guidance for vertical plane motion (Mooij, 1998).

However, as no rotational vehicle dynamics are included in the current work, deflections of control surfaces are not used as real time guidance and control parameters for steering the vehicle. Instead they are used to guarantee trim conditions at every vehicle state. The trim model used was already introduced in Section 3.2.4 and the implementation will be further discussed in Section 6.3.1.

The angle of attack and the thrust magnitude and direction are used as guidance parameters. Two different guidance methods will be used in this research. For verification of the simulation model, discussed in Section 6.4.2, the reference trajectory found by Mooij (1998) is reproduced. The guidance used to obtain this trajectory is therefore shortly discussed in Section 4.4.1. When optimizing the ascent trajectory, the guidance logic will follow from the optimization process. Section 4.4.2 will explain this in more detail.

4.4.1. Reference Guidance

The reference trajectory is based on the study performed by Mooij (1998). For this trajectory a so-called gamma-alpha steering was applied. A schematic overview of the method is shown in Figure 4.6. A short description of the method will be given here, for a more in depth discussion the reader is referred to Mooij (1998, Chapter 3.3).

Gamma-alpha Steering

Given a reference value for the flight-path angle γ_c, the required flight-path angle rate to drive the actual γ to the reference value can be given by:

$$\dot{\gamma}_c = K_{\gamma p} e_\gamma + K_{\gamma i} \int_0^t e_\gamma dt + K_{\gamma d} \frac{de_\gamma}{dt}, \qquad \text{with} \quad e_\gamma = \gamma_c - \gamma \tag{4.33}$$

where $K_{\gamma p} = 2$, $K_{\gamma i} = 1.8$ and $K_{\gamma d} = 0.0$ as selected by Mooij (1998). The commanded angle of attack can now be computed with:

$$\alpha_c = K_{\gamma 1} \dot{\gamma}_c + K_{\gamma 2} + K_{\gamma 3} \tag{4.34}$$

where

$$K_{\gamma 1} = \frac{mV}{T + C_{L\alpha} \bar{q} S_{ref}}$$

$$K_{\gamma 2} = -\frac{T\epsilon_T - mg\cos\gamma + C_{L0} \bar{q} S_{ref}}{T + C_{L\alpha} \bar{q} S_{ref}}$$

$$K_{\gamma 3} = \frac{2\omega_{cb} + \frac{V}{r}\cos\gamma}{T + C_{L\alpha}\bar{q}S_{ref}}$$

Setting the current angle of attack in the simulator to this value will lead to a new state with a corresponding flight-path angle. Returning to Equation 4.33, the new flight-path angle error e_γ with respect to the current reference flight-path angle can be computed resulting in the new flight-path angle rate and commanded angle of attack.

Throttle Law

The thrust magnitude is controlled with a throttle law. The used throttle law is extensively explained by Mooij (1998, Chapter 3.3). Only a short description of the resulting equations will be given here.

As the air-breathing engine is most efficient when tracking the constraint boundaries, the main goal of throttle law is to fly as close to the flight-path constraints as possible without violating any of the constraints. For each constraint a maximum commanded thrust T_c is computed as a function of the constraint error:

$$T_{c,\bar{q}} = D + \frac{mV}{2H_s}\dot{h} + K_{\bar{q}p}e_{\bar{q}} + K_{\bar{q}i}\int_0^t e_{\bar{q}}dt, \qquad \text{with} \quad e_{\bar{q}} = \bar{q} - \bar{q}_c \tag{4.36}$$

where H_s is the density scale height, $\dot{h}$ is the rate of change in altitude and the error $e_{\bar{q}}$ is the difference between the actual dynamic pressure and the maximum constraint value. The gains can be computed with:

$$K_{\bar{q}p} = -\frac{2m}{\rho V}\zeta_{\bar{q}}\omega_{\bar{q}}$$

$$K_{\bar{q}i} = -\frac{m}{\rho V}\omega_{\bar{q}}^2$$

where the damping coefficient $\zeta_{\bar{q}} = 0.7$ and the natural frequency $\omega_{\bar{q}} = 0.75$ rad/s, were selected by Mooij (1998). The commanded heat-rate-dependent thrust can be found in a similar way:

$$T_{c,\dot{Q}} = D + \frac{mV}{2C_{\dot{Q},2}}\frac{\dot{h}}{H_s} + K_{\dot{Q}p}e_{\dot{Q}} + K_{\dot{Q}i}\int_0^t e_{\dot{Q}}dt, \qquad \text{with} \quad e_{\dot{Q}} = \dot{Q} - \dot{Q}_c \tag{4.38}$$

where $C_{\dot{Q},2} = 3$ is a constant from Chapman's equation and the error $e_{\dot{Q}}$ is the difference between the actual heat rate and the maximum constraint value. The gains can be computed with:

$$K_{\dot{Q}p} = -\frac{2mV}{C_{\dot{Q},2}\dot{Q}}\zeta_{\dot{Q}}\omega_{\dot{Q}}$$

$$K_{\dot{Q}i} = -\frac{mV}{C_{\dot{Q},2}\dot{Q}}\omega_{\dot{Q}}^2$$

where the damping coefficient $\zeta_{\dot{Q}} = 0.7$ and the natural frequency $\omega_{\dot{Q}} = 0.075$ rad/s, were selected by Mooij (1998). The commanded axial-acceleration-dependent thrust can also be found in a similar way. However, the resulting equation depends on the thrust rate, drag rate and mass rate. These rates are locally very steep as a result of the instantaneous trim that allows for discrete changes in the elevon deflections and thus drag. The proposed axial-acceleration-dependent throttle control is

therefore highly unstable for the used simulation model. An alternative, simpler controller is therefore used instead:

$$T_{c,n} = T + K_{np} e_{n_a} + K_{ni} \int_0^t e_{n_a} dt, \qquad \text{with} \quad e_{n_a} = n_a - n_{a,c} \tag{4.40}$$

where T is the current guided thrust and the error e_{n_a} is the difference between the actual axial acceleration and the maximum constraint value. Good values for the gains have been determined by means of trial and error. The resulting values are:

$$K_{np} = -262,500 \qquad K_{ni} = -140,625$$

Finally a small note is made with respect to the integrated errors. If a constraint is not limiting for a long time, the integrated error can reach very large values. This will cause instabilities when the constraint does become the limiting. To prevent this, the integrated error will only be computed for the active constraint.

Based on this set of equations a maximum commanded thrust will be computed for each of the constraints.The lowest resulting value is limiting and will be the actual guided thrust. If the lowest value is higher than the maximum thrust that can be produced by the engine the guided thrust will be set to the maximum possible thrust.

4.4.2. Guidance for Optimization

When optimizing the ascent trajectory, the guidance logic will follow from the optimization process. Each individual of the optimization process is defined by a decision vector that states values for the guidance parameters for optimization at a number of points along the trajectory. Hermite spline interpolation is used to find values for the guidance parameters in-between the given points, resulting in a smooth guidance logic. This section will discuss the different parameters that will be included in the guidance. Furthermore, the definition of the control nodes and the decision vector are explained.

Guidance Parameters

The vehicle is guided by setting a number of control parameters. For the current mission those parameters are the angle of attack α and the thrust settings. The thrust settings include both the magnitude and direction of thrust. Because one of the research objectives is to find the influence of different control parameters on the optimization process, two alternatives were given in Section 3.2.1 for each of the thrust settings.

The direction of thrust can be controlled by either setting the thrust-elevation angle ϵ_T, or the thrust-vector trim fraction F_{TVC}. The thrust magnitude can be controlled with the equivalence ratio ϕ, the throttle factor F_T, or the throttle law described in the previous section. In the latter case neither ϕ nor F_T are included in the set of optimization parameters. Table 4.1 gives an overview of the possible combinations of thrust settings that can be used. The final column shows the complete set of optimization parameters included in the decision vector, which will be discussed shortly. It is noted that the angle of attack α and the normalized energy state $\hat{E}$ (discussed next) are always part of the set of optimization parameters.

Control Node Definition

The control nodes represent points along the trajectory where values for the guidance parameters are defined. Time is usually a good independent variable to define the control nodes, but the final time of the ascent trajectory is unknown at the start of the optimization process. Time can therefore not be used as an independent variable to define the location of the control nodes. A commonly used alternative for trajectory optimization problem is the normalized energy state $\hat{E}$ (Hänninen, 2009; Papp, 2014), where:

$$E = gh + \frac{1}{2}V^2, \qquad \text{with} \quad \hat{E} = \frac{E}{E_{final}} \tag{4.41}$$

thrust magnitude setting	thrust direction setting	parameters in decision vector
throttle law	no TVC	$\hat{E}, \alpha$
	ϵ_T	$\hat{E}, \alpha, \epsilon_T$
	F_{TVC}	$\hat{E}, \alpha, F_{TVC}$
ϕ	no TVC	$\hat{E}, \alpha, \phi$
	ϵ_T	$\hat{E}, \alpha, \phi, \epsilon_T$
	F_{TVC}	$\hat{E}, \alpha, \phi, F_{TVC}$
F_T	no TVC	$\hat{E}, \alpha, F_T$
	ϵ_T	$\hat{E}, \alpha, F_T, \epsilon_T$
	F_{TVC}	$\hat{E}, \alpha, F_T, F_{TVC}$

Table 4.1: Possible variations of thrust settings with resulting optimization parameters in the decision vector.

The energy state is known for both the initial and final state and will be monotonically increasing throughout the trajectory as the vehicle will be both ascending and accelerating. The control nodes can be distributed uniformly or randomly between the initial and final energy state. Dijkstra (2012) has studied a number of different methods and found that non-uniform node control performed best in finding a guidance logic. Non-uniform node control is therefore also applied here. Energy state values at which the nodes are defined are initialized randomly. They are then included as optimization parameters. This means the optimization itself will find the best locations at which to define the guidance parameters.

Decision Vector

The standard definition of the optimization problem is used as an example here and includes the normalized energy state $\hat{E}$, the angle of attack α and the equivalence ratio ϕ as optimization parameters. The decision vector used as input for the optimization with these parameters is defined as:

$$\boldsymbol{x_v} = \left[\hat{E}_1 \cdots \hat{E}_{n-1}, \hat{E}_n, \alpha_1 \cdots \alpha_{n-1}, \alpha_n, \phi_1 \cdots \phi_{n-1}, \phi_n\right] \tag{4.42}$$

where n indicates the number of control nodes for each guidance parameter. Every guidance parameter will be given a value at each node by the optimizer within the domain set by the user. In-between the nodes guidance values are computed by interpolating with a Hermite spline interpolator, resulting in a smooth guidance logic. An example of such a guidance logic is given in Figure 4.7, where 8 control nodes were used. The control nodes, indicated with circles, are positioned closer together at the right side of the figure than in the middle. This shows the non-uniform distribution of the control nodes. It is noted that different numbers of control nodes will be used. The resulting effects are discussed in Section 7.4.

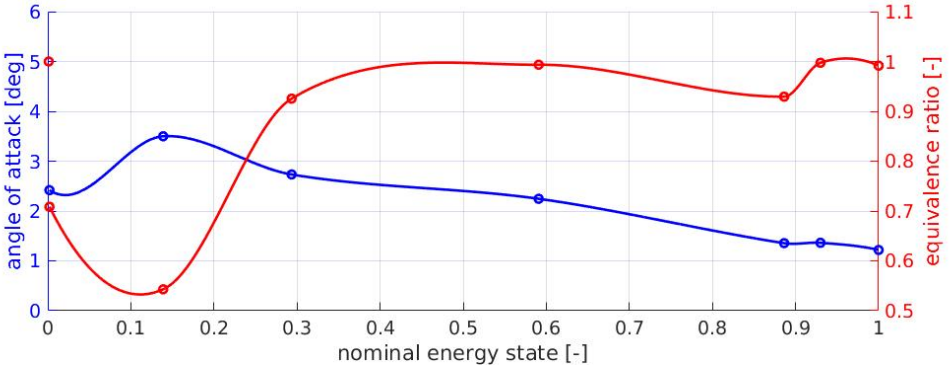


Figure 4.7: Guidance logic as a function of the normalized energy state.

5

Optimization

The previous chapters have discussed all models that are used to simulate the ascent trajectory of the WCC. In the scope of this research, a number of global optimizers will be compared with respect to their performance on optimizing this trajectory. Furthermore, the effect of additional constraints and control parameters on the performance of the optimizers will be evaluated. This chapter will therefore discuss the general optimization process.

The state of the art of ascent trajectory optimization was already introduced in Section 2.3. Although local optimization is currently most widely used for trajectory optimization, global optimization shows promise for the future and was therefore selected for this research. Section 5.1 will given an overview of the different categories and methods within the set of global optimizers. Throughout the section a number of methods will be selected for the current research. Section 5.2 will then explain how each of the selected methods work. Finally Section 5.3 gives some insight in the trajectory optimization problem itself, including information on the constraints, objectives and optimization parameters.

5.1. Global Optimization

With the fast increase of computer power in recent years interest in global optimization methods has been increasing as well. Within the set of global optimizers a distinction can be made between the deterministic methods and the stochastic methods. Deterministic methods do not include any randomness. Most are systematic and will produce a solution with a predictable maximum amount of work. On the other hand, stochastic methods do include randomness and most stochastic methods are heuristic, meaning that they cannot be proven to find a global optimum with a predictable amount of work (Vasile, 2003).

Although systematic methods are usually more reliable than heuristic methods, they have the disadvantage of requiring insight in the problem at hand to find a solution with a reasonable amount of work. Heuristic methods can handle problems as a black box and have been used for many applications over the past decades. Stochastic and heuristic methods seem for this reason very promising to find initial pseudo-optimal trajectory solutions. This research will therefore only focus on this group of methods.

According to the 'No Free Lunch Theorem' by Wolpert and Macready (1997), the averaged performance of all search algorithms over all problems is equal. Or in other words, if an optimizer performs well on one problem, it will perform poorly on another. For that reason it is important to apply the right optimization techniques to the right problems.

The most important groups of stochastic algorithms are: simulated annealing (SA), ant colony optimization (ACO), particle swarm optimization (PSO) and evolutionary algorithms (EA). Within these groups, many different forms of the algorithms exist. EA, for example, includes genetic algorithms (GA), genetic programming (GP) and differential evolution (DE). Some comparative studies of global optimizers have been performed on benchmark problems (Vesterstøm and Thomsen, 2004), and space trajectory problems (Myatt et al., 2004; Vinkó et al., 2007). These studies included GA, PSO, MPSO (Multiple PSO), DE and ASA (Adaptive SA). Based on the earlier mentioned studies, the most promising global optimizers seemed to be MPSO, ASA and DE.

Single-Objective Optimization

A distinction has to be made between single-objective (SO) optimization and multi-objective (MO) optimization. All methods described above are in their original form single-objective optimizers. The general optimization problem can be mathematically described as:

$$\min f(\boldsymbol{x}) \tag{5.1}$$

where $\boldsymbol{x}$ represents the state vector with parameters to optimize:

$$\boldsymbol{x} = (x_1, x_2, \dots, x_n)^T \tag{5.2}$$

and where the problem is subjected to the following equality and inequality constraints:

$$h_i(\boldsymbol{x}) = 0 \qquad \text{for } i = 1, 2, \dots, m \tag{5.3}$$
$$g_i(\boldsymbol{x}) \leq 0 \qquad \text{for } i = m, m+1, \dots, p \tag{5.4}$$

All methods aim at minimizing the objective function $f(\boldsymbol{x})$. For a single objective optimization, all constraints and objectives need to be described by one single function. In this study the objective is to maximize the payload capacity by minimizing the fuel consumption and the heat load. First, only the fuel consumption will be taken into account. Because the vehicle design is set, the objectives for maximizing payload and minimizing fuel consumption are interchangeable: every kilogram saved on fuel is directly available for payload. It can thus be seen as a single-objective optimization problem in the form:

$$f(\boldsymbol{x}) = m_{cons} \tag{5.5}$$

where m_{cons} is the consumed fuel mass as a result of the trajectory flown. However, the integrated heat load Q is also important in the trajectory optimization. Minimizing Q could significantly reduce the weight of the thermal protection system (TPS) and could thus result in a higher payload capacity. This objective could be included in the single objective by estimating the mass of the TPS m_{TPS} as function of the integrated heat load, and adding that to Equation 5.5:

$$f(\boldsymbol{x}) = m_{cons} + m_{TPS} \tag{5.6}$$

By expressing both objectives as masses, scaling is already included. The objective function can also be defined in alternative ways, but proper scaling of both terms of the function is crucial to success. Scaling is often an ambiguous process, which can result to complications when including multiple objectives in a single function.

The trajectory is also subjected to constraints and a final target position and velocity must be reached. The constraints can be added to the cost function in the form of penalty functions to transform the problem in an unconstrained single objective optimization problem:

$$f(\boldsymbol{x}) = m_{cons} + m_{TPS} + P_{constraints} + P_{target} \tag{5.7}$$

where $P_{constraints}$ and P_{target} represent the penalty functions for constraint violations or not reaching the final state. Penalty functions are problem dependent and rely on proper scaling as well (Parsopoulos and Vrahatis, 2002; Yang et al., 1997). Applying large penalties may restrict the optimizer from finding better solutions, which can cause the optimizer to get stuck in a local optimum. Choosing penalties too low, may allow the optimizer to select unfeasible solutions. Finding suitable penalty functions will therefore be a problem on its own.

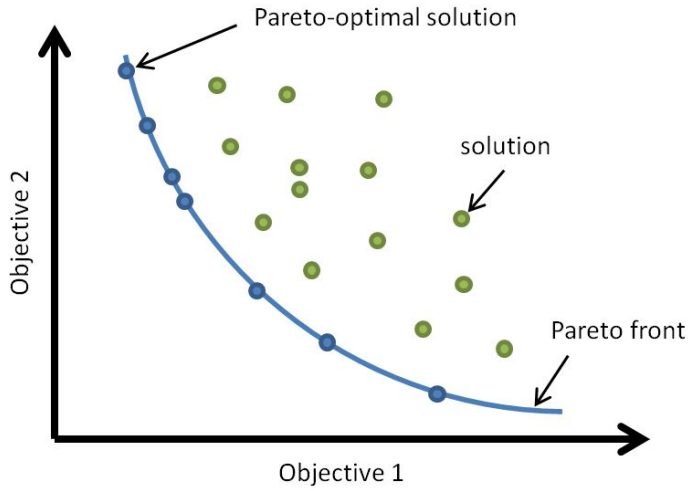


Figure 5.1: Visualization of the Pareto front for a two-dimensional problem.

Multi-Objective Optimization

Combining multiple objectives and constraints in one objective function is not unambiguous. The result therefore depends heavily on how the objective function is formed. To avoid this, multi-objective optimization can be applied. All objectives and constraints will then be evaluated separately in the form of m objectives, resulting in the following mathematical representation:

$$\min \boldsymbol{f}(\boldsymbol{x}) = \min \left[f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \dots, f_m(\boldsymbol{x})\right]^T \tag{5.8}$$

where the problem is subjected to the following equality and inequality constraints:

$$\boldsymbol{h}(\boldsymbol{x}) = \left[h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \dots, h_i(\boldsymbol{x})\right]^T = 0 \qquad \text{for } i = 1, 2, \dots, j \tag{5.9}$$

$$\boldsymbol{g}(\boldsymbol{x}) = \left[g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), \dots, g_i(\boldsymbol{x})\right]^T \leq 0 \qquad \text{for } i = j, j+1, \dots, p \tag{5.10}$$

The solution will now be found in the form of a set of points that together forms a Pareto front. Points in the Pareto front cannot improve one of their objective functions without degrading at least one of the other objectives. This is visualized in two dimension in Figure 5.1. To find the set of Pareto optimal solutions, the notion of dominance is important. This is among others explained by Zitzler and Thiele (1998). A solution $\boldsymbol{x_1}$ is said to dominate another solution $\boldsymbol{x_2}$ if:

$$f_i(\boldsymbol{x_1}) \leq f_i(\boldsymbol{x_2}) \qquad \forall i = 1, 2, \dots, m \tag{5.11}$$

$$f_i(\boldsymbol{x_1}) < f_i(\boldsymbol{x_2}) \qquad \text{for at least one } i \tag{5.12}$$

This is denoted by $\boldsymbol{x_1} \prec \boldsymbol{x_2}$. A set of solutions will form a Pareto front if none of the solutions is dominated by another solution of the set.

The Pareto front gives a representation of the relation and compromise that exists between the different objectives. Multi-objective optimization has thus the advantage that decisions on compromising objectives can be made after a better understanding is obtained. Since multi-objective optimization seems better suited for the problem at hand, the focus of this study will be further narrowed down to global multi-objective optimizers.

Castellini (2008) did an extensive comparison of multi-objective global optimizers. Both the self developed dual-grid multi-objective particle swarm optimization (DGMOPSO) and the improved fast and elitist non-dominated sorting genetic algorithm (NSGA-II) first introduced by Deb et al. (2000) showed good performance in terms of robustness and convergence speed.

Tan et al. (2003) also studied the addition of a tabu list as an improvement to the NSGA-II algorithm. It can be incorporated in the existing algorithm relatively easily. Tabu restriction has shown to avoid

premature convergence, which is a known weakness of genetic algorithms. As the search space of the trajectory optimization problem is very large, premature convergence could become a problem when using the NSGA-II algorithm.

MOEA/D is a multi-objective evolutionary algorithm based on decomposition. This is a relatively new method first introduced by Zhang and Li (2007). Studies (Li and Zhang, 2009; Zhang and Li, 2007) also compared the performance of this method with, among others, NSGA-II, which showed equal or better performance for most problems. Adaptations to the method are also proposed, which showed even better results. Qi et al. (2014) compared Adaptive-MOEA/D, paλMOEA/D and MOEA/D-AWA, with the last one showing the best performance.

A final selection of the methods is based on their performance and availability of the software. PaGMO is an open source software platform developed by ESA (Biscani et al., 2010), which includes a number of optimization algorithms. Because limited time is available for this research, a restriction is made to methods already implemented in PaGMO. Of the methods discussed in this section, both NSGA-II and MOEA/D are implemented in PaGMO and are therefore selected for comparison. A number of improvements were discussed for both NSGA-II and MOEA/D. Because tabu restriction is relatively easy to implement within the NSGA-II algorithm, it will be included in this research. The other methods are deemed too time consuming to fit within the current scope and are therefore left for future work.

The selected methods will be discussed in Section 5.2. The NSGA-II method will be explained in Section 5.2.1. The implementation of the tabu-restriction will then be given in Section 5.2.2. Because the update of the MOEA/D method is based on DE, Section 5.2.3 will discuss the working of DE, before Section 5.2.4 will follow with a more in depth explanation of the MOEA/D method.

Parallel Optimization

The optimization process could be further improved by using a distributed approach, which takes advantage of the ability to run multiple processes parallel. There are a number of ways in which this distribution can be achieved (Van Veldhuizen et al., 2003). The simplest way to use parallelization is to distribute unrelated computing tasks over different processors. This will not influence the final result, but speed up the process.

Another option is the island model where multiple populations are optimized parallel to each other. At certain times the populations are allowed to communicate and exchange individuals. This method maintains diversity by evolving populations independently. This allows the optimizer to better cover different areas of the search space. Migration between populations will introduce different individuals into the population. New individuals could lead to jumps towards better solution and have a positive effect on the convergence.

The way in which the communication between individuals is implemented will influence the performance of the optimizer. Different migration and replacement policies have been studied by Cantú-Paz (1999). Replacing bad individuals from one population with good individuals from another population has shown to considerably increase the convergence speed of the optimization. However, the quality of the results was not taken into account, and premature convergence could be a potential trap.

The island model shows great potential as it can be implemented in many different ways. First of all one could combine multiple optimization methods. Each method has its strengths and weaknesses. If done right, combining multiple optimizers in an island model could therefore lead to a complete model that takes advantage of the strengths of each individual method. Because NSGA-II-tabu and MOEA/D are already being studied for this research, it is relatively easy to implement a parallel configuration of the two methods. This will show whether the individual optimizers could complement each other when used in parallel. The implementation of this method is further explained in Section 5.2.5.

5.2. Methods

This section will explain the working of each of the previously selected optimization algorithms. It will also give some insight in how the parameters affect the diversity and convergence speed of the algorithm.

5.2.1. NSGA-II

The NSGA-II algorithm is based on a basic genetic algorithm. Both real coded and binary coded GAs can be used. Because Castellini (2008) evaluated the real coded version and the search space of the

trajectory problem is a continuous space, this version will also be used here. The real coded NSGA-II method available in PaGMO uses simulated binary crossover (SBX) and real polynomial mutation. The algorithm works as follows:

1: At the start of the optimization, the algorithm randomly initializes n parameter vectors in a D-dimensional space, that represent the population. Every individual $\boldsymbol{x_i}$ consists of D parameter values to be optimized.

2: The individuals are evaluated with the objective functions $\boldsymbol{f}(\boldsymbol{x})$

3: Apply fast non-dominated sorting as explained by Deb et al. (2000). The first solution is stored in a temporary set P'. From there on, every next solution p is temporarily added to P' and compared with all the solutions q already present in the set P'.

if $p \prec q$	remove solution q from P'
if $q \prec p$	remove solution p from P'
else	keep both p and q in P'

When all solutions are evaluated the set P' will represent the first Pareto-optimal set. The procedure is now repeated without the solutions of set P' to form the next Pareto front, until all solutions are stored in a set. The first set has rank 0, the next rank 1 etc.

4: To evaluate the rank of the solutions within a front, the crowding-distance is calculated. This gives an estimate of the density of the solutions surrounding a point. In each front, the solutions are sorted in ascending order for each objective M. All solutions are initially assigned a distance $i_{dist} = 0$ except for the lowest and highest, which have $i_{dist} = \infty$. For l individuals in front F, the crowding-distance can now be found for $i = 2$ to $(l - 1)$ with:

$$F[i]_{dist} = F[i]_{dist} + \sum_{m=0}^{M} \Big(F[i+1].m - F[i-1].m \Big) \tag{5.13}$$

where $F[i]_{dist}$ is the crowding distance of individual i in front F, and $F[i].m$ represents the m^{th} objective function value of the i^{th} individual of front F. To achieve a good spread of the Pareto front, an individual i with a larger crowding distance than individual j is said to perform better, which is indicated with: $i \prec_n j$.
If both non-dominated ranking i_{rank} and local crowding distance i_{dist} are included, a general rule for the ranking order of individuals can be expressed as:

$$i \prec_n j \quad \text{if } (i_{rank} < j_{rank}) \quad \text{or } \big((i_{rank} = j_{rank}) \text{ and } i_{dist} > j_{dist}\big) \tag{5.14}$$

5: Perform a binary tournament selection of the initial population to form the parent group of the current generation. Two randomly chosen individuals will enter a tournament, of which the best performing individual according to Equation 5.14 will become a parent (Goldberg and Deb, 1991). This process is repeated until the parent pool is full; i.e., has n individuals.

6: Apply simulated binary crossover (SBX) with a crossover probability of cr. The value of cr can be anything between zero and one, but is usually close to one. SBX is explained by Deb and Kumar (1995) and will be shortly described here. First a random number $u \in U[0,1]$ will be created. Subsequently β' can be found with:

$$\int_0^{\beta'} \mathcal{P}(\beta) d\beta = u \tag{5.15}$$

where $\mathcal{P}(\beta)$ is described by:

$$\mathcal{P}(\beta) = \begin{cases} 0.5(\eta_c + 1)\beta^{\eta_c} & \text{if } \beta \leq 1 \\ 0.5(\eta_c + 1)\frac{1}{\beta^{\eta_c + 2}} & \text{if otherwise} \end{cases} \tag{5.16}$$

with η_c a constant that determines how distant the children can be from the parent solutions. Common values lie in the range [1,100], where low values of η_c result in children close to the parents and high values of η_c allow for children further away from the parents. β indicates the ratio of the spread of the children c_1 and c_2 and the parents p_1 and p_2 according to:

$$\beta = \frac{c_1 - c_2}{p_1 - p_2} \tag{5.17}$$

With the value of β' known, the children can be found with:

$$c_1 = 0.5\big((p_1 + p_2) - \beta'|p_2 - p_1|\big)$$
$$c_2 = 0.5\big((p_1 + p_2) + \beta'|p_2 - p_1|\big)$$

If the parameters are bound in the form of $x_i^{(L)} < x_i < x_i^{(U)}$, the spread factors β_1' and β_2' can be found by adapting the distribution function of Equation 5.16 to $\mathcal{P}(\beta)/\mathcal{P}_1'$ and $\mathcal{P}(\beta)/\mathcal{P}_2'$, where:

$$\mathcal{P}_1' = \int_0^{\beta^{(L)}} \mathcal{P}(\beta)d\beta \tag{5.18}$$

$$\mathcal{P}_2' = \int_0^{\beta^{(U)}} \mathcal{P}(\beta)d\beta \tag{5.19}$$

with

$$\beta^{(L)} = \frac{p_1 + p_2 - 2x_i^{(L)}}{|p_2 - p_1|} \quad \text{and} \quad \beta^{(U)} = \frac{2x_i^{(U)} - p_1 - p_2}{|p_2 - p_1|}$$

The children now follow from:

$$c_1 = 0.5\big((p_1 + p_2) - \beta_1'|p_2 - p_1|\big)$$
$$c_2 = 0.5\big((p_1 + p_2) + \beta_2'|p_2 - p_1|\big)$$

7: After crossover, the resulting children are mutated with a mutation probability of m, according to the real polynomial mutation scheme explained by Deb and Deb (2014). For a child c_1 a random number $u \in U[0,1]$ is created and the mutated child c_1' follows according to:

$$c_1' = \begin{cases} c_1 + \bar{\delta}_L\big(c_1 - x_i^{(L)}\big) & \text{for } u \le 0.5 \\ c_1 + \bar{\delta}_U\big(x_i^{(U)} - c_1\big) & \text{for } u > 0.5 \end{cases} \tag{5.20}$$

where the two parameters $\bar{\delta}_L$ and $\bar{\delta}_U$ can be calculated with:

$$\bar{\delta}_L = \big(2u\big)^{1/(1+\eta_m)} - 1 \qquad \text{for } u \le 0.5$$
$$\bar{\delta}_U = 1 - \big(2(1-u)\big)^{1/(1+\eta_m)} \qquad \text{for } u > 0.5$$

where η_m is a constant in the range [1,100] that controls the influence of the perturbation caused by the mutation. A low value will only cause small perturbations and thus mutated solutions close to the initial solution. High values can result in mutated solutions further away from the initial solution.

8: Evaluate the objective functions $\boldsymbol{f}(\boldsymbol{x})$ for each of the child solutions.
9: Create an intermediate population by combining the parents and the children together.
10: Apply fast dominated-sorting and find the local-crowding distance, as was done in step 3 and 4. Order the individuals according to their rank as can be found with Equation 5.14.
11: Pick the first n (best performing) individuals from the intermediate population for the next generation. This can be seen as an application of elitism.
12: As long as the stopping criterion has not been reached, the algorithm will re-iterate starting again with step 5.

The main control parameters of the NSGA-II algorithm are the population size n, the number of generations G, the crossover probability cr, the mutation probability m, the crossover distribution constant η_c and the mutation distribution constant η_m. A higher value of n, m, η_c and η_m increases randomness and therefore increases the exploration power of the algorithm. This comes at a cost of slower convergence. Lowering these values and increasing cr will result in a faster convergence that might come at a cost of finding local minima.

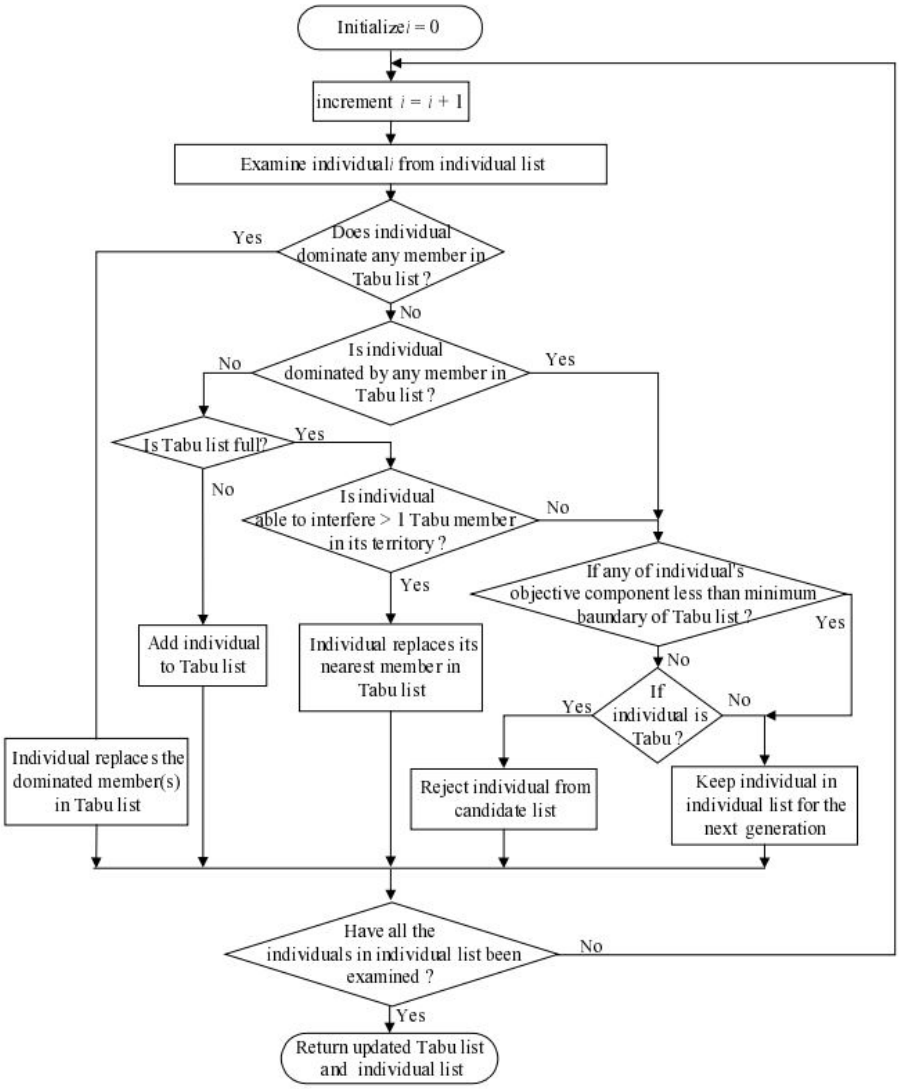


Figure 5.2: Heuristic reasoning of the individual examination rule for the tabu list (Tan et al., 2003).

5.2.2. NSGA-II-tabu

The implementation and the benefit of adding a tabu restriction to evolutionary algorithms is explained by Tan et al. (2003). The tabu list will keep the best individuals found so far. After genetic operators have been applied to the population and cost functions have been evaluated, the individuals of the new generation will be checked against the individuals in the tabu list. The heuristic reasoning for this examination can be found in Figure 5.2. Note that the territory interference mentioned in the figure is a similar method to the crowding distance explained in Section 5.2.1, which has the aim to evenly spread the members of the Pareto front. The individuals in the individual list after examination will be returned as the next generation.

Keeping a tabu list avoids repetition of currently found good individuals. As explained by Tan et al. (2003), standard genetic algorithms are specifically sensitive to getting stuck in local optima. When an optimal schema is found, this individual is more likely to be chosen for reproduction. This can lead to premature convergence to this optimal schema. If cross-over and mutation cannot find a better schema, the algorithm might get stuck in a local optimum. The tabu restriction makes sure that the individuals in the tabu list are only allowed to appear once in the population. This method reserves diversity in the population and thus increases the chance of finding different and better schemas. Tabu restriction can be easily incorporated in the NSGA-II algorithm.

5.2.3. Differential Evolution

Because MOEA/D is based on differential evolution, this algorithm will be explained separately here. Differential evolution was first proposed by Storn and Price (1997). The method includes many aspects of evolutionary strategies like crossover and mutation. The DE algorithm described here is also denoted as the DE/rand/1/bin method. This method performs well according to the tests done by Storn and Price (1997) and works as follows:

1: At the start of the optimization, the algorithm randomly initializes n parameter vectors in a D-dimensional space. Every parameter vector $\boldsymbol{x}_i(G)$ consists of D values and represents an individual of the current generation G. The individuals are called 'target vectors'.

2: Mutation is applied to each target vector $\boldsymbol{x}_i(G)$ in the generation, resulting in a mutant vector $\boldsymbol{v}_i(G+1)$:

$$\boldsymbol{v}_i(G+1) = \boldsymbol{x}_{r1}(G) + F\big(\boldsymbol{x}_{r2}(G) - \boldsymbol{x}_{r3}(G)\big) \tag{5.21}$$

with r_1, r_2, r_3 integer mutually different random indices $\in 1, 2, \dots, n$, that are not equal to i. As a result, the minimum size of the population is four. F is a real constant factor $\in [0, 2]$ controlling the influence of the differential variation. A two-dimensional visualization of the generation of the mutant vector can be found in Figure 5.3.

3: Crossover introduces diversity in the perturbed vectors. The resulting vectors after crossover are called trial vectors. The trial vector $\boldsymbol{u}_i(G+1)$ consists of D parameter values and can be represented as:

$$\boldsymbol{u}_i(G+1) = \big(u_{1,i}(G+1), u_{2,i}(G+1), \dots, u_{D,i}(G+1)\big) \tag{5.22}$$

Each parameter in the trial vector is picked according to:

$$u_{j,i}(G+1) = \begin{cases} v_{j,i}(G+1) & \text{if } rand(j) \le cr \quad \text{or } j = r_{index}(i) \\ x_{j,i}(G) & \text{if } rand(j) > cr \quad \text{or } j \ne r_{index}(i) \end{cases} \tag{5.23}$$
$$j = 1, 2, \dots, D$$

where $rand(j)$ is the j^{th} element of a vector with D random numbers from a uniform distribution $\in [0, 1]$, and $r_{index}(i)$ is a random index $\in 1, 2, \dots, D$. This random index makes sure that at least one parameter value is taken from the trial vector. cr is the crossover constant with a value $\in [0, 1]$ and determines the likelihood with which the parameters are picked from the different vectors. It is noted that several variants of crossover exist. A different method is explained by Storn (1996) and Storn (1999).

4: Both the performance of the target vector $\boldsymbol{x}_i(G)$ and the trial vector $\boldsymbol{u}_i(G+1)$ are evaluated with the objective function in the form of Equation 5.7. The greedy principle is used, meaning that the best performing individual will continue to the next generation.

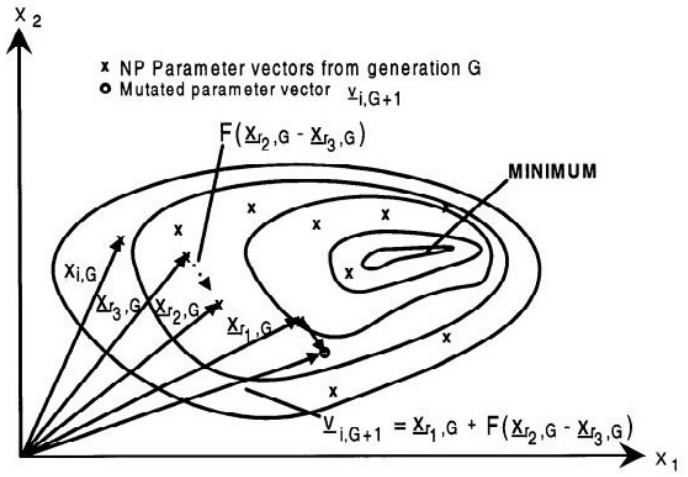


Figure 5.3: Two-dimensional example of generating the mutant vector $\boldsymbol{v}_i(G+1)$ in DE (Storn and Price, 1997).

5: Steps 2 to 4 are repeated for all individuals in the current generation until a next generation is formed with an equal number of individuals.
6: As long as the stopping criterion has not been reached, the algorithm will re-iterate starting again with step 2.

The main control parameters in the DE algorithm are the constant F, the population size n and the crossover constant cr. Storn and Price (1997) include recommendations for the settings of these control variables. A reasonable choice for the population size is found to be $5D < n < 10D$. The value of F is generally chosen between 0.4 and 1 with 0,5 being a good initial guess. A good choice for the crossover constant cr is 0.1. However, a higher value of cr often leads to faster convergence. For that reason it is useful to first try if a value of 0.9 or 1.0 will work.

5.2.4. MOEA/D

The MOEA/D algorithm was first introduced by Zhang and Li (2007) and is based on an evolutionary algorithm. The multi-objective optimization problem is decomposed in a number of scalar optimization sub-problems that are optimized simultaneously. The objective of the sub-problems is an aggregation of all the f_is in Equation 5.8. An approximation of the Pareto front can be found by optimization of the sub-problems.

Multiple methods exist for constructing the aggregation functions. The most common ones are explained by Zhang and Li (2007) and include the weighted sum approach, the Tchebycheff approach and the boundary intersection methods. The code available in PaGMO includes the Tchebycheff method which will therefore be explained here.

Suppose a number of evenly spread weight vectors $\boldsymbol{\lambda} = [\lambda_1, \dots, \lambda_m]^T$, where m is the number of objective functions f_i, and all components of $\lambda_i \geq 0$ add up to 1. The scalar optimization problem is now written as:

$$\min\left[g\left(\boldsymbol{f}(\boldsymbol{x})|\boldsymbol{\lambda},\boldsymbol{z}\right)\right] = \min\left[\max_{i=1,\dots,m}\left(\lambda_i|f_i(\boldsymbol{x}) - z_i|\right)\right] \tag{5.24}$$

where $\boldsymbol{x}$ is the parameter vector with values within the limited range of the controls. $g\left(\boldsymbol{f}(\boldsymbol{x})|\boldsymbol{\lambda},\boldsymbol{z}\right)$ indicates a scalar objective function that depends on the objectives $\boldsymbol{f}(\boldsymbol{x})$, the respective weights $\boldsymbol{\lambda}$ and reference objective vector $\boldsymbol{z}$. This last vector is given by:

$$\boldsymbol{z} = [z_1, \dots, z_m]^T = \min\left[f_i(\boldsymbol{x})\right] \qquad \text{for } i = 1, \dots, m$$

For every Pareto optimal point $\boldsymbol{x}*$ a weight vector $\boldsymbol{\lambda}$ exists that will produce $\boldsymbol{x}*$ as optimal solution of Equation 5.24 (Zhang and Li, 2007). This means that the solutions of the scalar objective functions

are Pareto optimal solutions. By repeating this for a number of weight vectors, a Pareto front can be found. To obtain a good approximation of the Pareto front, n should be reasonably large and the weight vectors should be properly selected (Li and Zhang, 2009).

For the general algorithm the number of sub-problems evaluated depends on the chosen number of weight vectors $\lambda^1, \dots, \lambda^n$. The j^{th} sub-problem is given by:

$$\min\left[g\left(\boldsymbol{f}(\boldsymbol{x})|\boldsymbol{\lambda^j}, \boldsymbol{z}\right)\right] = \min\left[\max_{i=1,\dots,m}\left(\lambda_i^j|f_i(\boldsymbol{x}) - z_i|\right)\right] \tag{5.25}$$

where $\boldsymbol{\lambda^j} = [\lambda_i^j, \dots, \lambda_m^j]$. New sub-solutions are only influenced by their neighborhood, which is defined by the Euclidean distance of the weight vectors. The code in PaGMO uses DE polynomial mutation to find the next generation as described in Section 5.2.3, however, it is noted that other update schemes are possible. The algorithm goes through the following steps (Li and Zhang, 2009):

1: Set the external population (EP) to zero. This is an array used to store non-dominated solutions during the search
2: Initialize n evenly spread spread weight vectors $\boldsymbol{\lambda^i}$. Calculate the Euclidean distance between all pairs of weight vectors. For each sub-problem $i = 1, \dots, n$, create a vector $\boldsymbol{B}(i) = [i_1, \dots, i_T]$, where $i_1, \dots, i_T$ indicate the indices of the T closest weight vectors to λ^i.
3: Randomly generate an initial population consisting of n parameter vectors $\boldsymbol{x^1}, \dots, \boldsymbol{x^n}$. Set values for $\boldsymbol{F^1}, \dots, \boldsymbol{F^n}$, where $\boldsymbol{F^i} = \boldsymbol{f}(\boldsymbol{x^i}) = [f_1(\boldsymbol{x^i}), \dots, f_m(\boldsymbol{x^i})]^T$ for each $i = 1, \dots, n$.
4: Initialize the reference objective vector $\boldsymbol{z} = [z_1, \dots, z_m]^T$. This vector stores the best value z_i found so far for objective $f_i(\boldsymbol{x})$.
5: for $i = 1, \dots, n$, do step 6 to 9:
6: Create a uniformly distributed random number $u \in [0, 1]$ and compare with a selection constant $0 \leq p \leq 1$. Then set:

$$\boldsymbol{P} = \begin{cases} \boldsymbol{B}(i) & \text{if } u < p \\ [1, \dots, n] & \text{otherwise} \end{cases}$$

Use the DE operations of step 2 and 3 as explained in Section 5.2.3 to generate a new solution $\boldsymbol{y}$, making sure to pick vectors with indices in $\boldsymbol{P}$. Polynomial mutation is then applied as was explained in step 8 of Section 5.2.1. If a parameter of $\boldsymbol{y}$ is outside its range, it will be reset randomly.
7: While $c < n_r$
Update the reference objective vector $\boldsymbol{z}$ for $j = 1, \dots, m$. If $f_j(\boldsymbol{y}) < z_j$, then $z_j = f_j(y)$.
$c = c + 1$.
8: Update the neighboring solutions. For each index $j \in \boldsymbol{B}(i)$, if $g\left(\boldsymbol{f}(\boldsymbol{y})|\boldsymbol{\lambda^j}, \boldsymbol{z}\right) \leq g\left(\boldsymbol{f}(\boldsymbol{x^j})|\boldsymbol{\lambda^j}, \boldsymbol{z}\right)$ then $\boldsymbol{x^j} = \boldsymbol{y}$ and $\boldsymbol{F^j} = \boldsymbol{f}(\boldsymbol{y})$.
9: Update EP by removing all vectors that are dominated by $\boldsymbol{F}(\boldsymbol{y})$, and add $\boldsymbol{F}(\boldsymbol{y})$ to EP if it is not dominated by any vectors currently in EP.
10: When the stopping criterion has been reached, stop and output EP. Otherwise the algorithm will re-iterate starting again with step 5.

The most important control parameters of the MOEA/D method are n, T, p and the choice of the weight vectors $\boldsymbol{\lambda^i}$. Furthermore, η_m, cr and F are important constants that have already been discussed in Sections 5.2.1 and 5.2.3. The study by Li and Zhang (2009) showed that the method is not very sensitive to the values of cr and F. The values used in Li and Zhang (2009) are $cr = 1.0$, $F = 0.5$ and $\eta_m = 20$. For the weight vectors, values are taken from a set

$$\frac{0}{H}, \frac{1}{H}, \dots, \frac{H}{H}$$

where H is a constant. The number of weight vectors n depends on the constant H and the number of objectives m and can be calculated according to (Murata et al., 2001):

$$
\begin{aligned}
n_2(H) &= H + 1 \\
n_3(H) &= \sum_{i=0}^{H} n_2(i) = \sum_{i=0}^{H} (i+1) \\
&\vdots \qquad\qquad\qquad \vdots \\
n_m(H) &= \sum_{i=0}^{H} n_{m-1}(i)
\end{aligned}
$$

In the problems solved by Zhang and Li (2007) and Li and Zhang (2009), the population size was chosen in the order of a few hundred. Important is that $n_r \ll T \ll n$. Li and Zhang (2009) used $T = 20$, $n_r = 2$ and $p = 0.9$. Choosing T or p too large will cause the algorithm to ignore the neighborhood relationship, resulting in slow convergence. A low value of n_r makes sure that not the whole neighborhood will be replaced by a child solution. This keeps the diversity in the algorithm.

5.2.5. Parallel NSGA-II-tabu with MOEA/D

The parallel optimization can be seen as a set of two completely separate optimization processes following the algorithms as described before. The only difference is that the populations will now communicate and exchange information with each other at certain moments. Each population follows the following steps:

1: Evolve the population. If the current generation number is a multiple of the migration interval, continue with step 2. Else, repeat step 1.
2: Select individuals for migration according to the migration policy and store these individuals in a buffer accessible by the target population.
3: Wait until the individuals for replacement are made available by the other population. Place these individuals in the current population following the replacement policy.
4: Update the Pareto information and return to step 1.

Different methods for the migration and replacement policies are described by Cantú-Paz (1999). The following migration and replacement methods are also available and PaGMO and will be evaluated:

best migration: This method picks the best performing individuals for migration, while also leaving these individuals in the current population.
best-kill migration: This method will pick the best performing individuals for migration, while deleting these individuals from the current population.
random migration: This method picks random individuals for migration, while also leaving these individuals in the current population.

worst replacement: This method replaces the worst individuals in the current population with the migrated individuals.
fair replacement: This method replaces the worst individuals in the current population with the migrated individuals, but only if the migrated individuals perform better.
random replacement: This method randomly replaces individuals in the current population with the migrated individuals.

Each method has its advantages. As stated by Cantú-Paz (1999), keeping the best individuals and replacing the worst individuals from one population with the best individuals from the other population will lead to faster convergence. However, this method will be very likely to converge prematurely. The best-kill migration method already adds some diversity by making sure that the best individuals are not present in both populations. Using random migration and replacement will result in slower convergence, but better diversity and therefore is more likely to actually find the global optimum. For each migration method the number of individuals to migrate and the interval at which to do this can be set by the user. In general, the higher the migration rate, the faster the convergence will be.

5.3. Problem

The general MO optimization problem aims at minimizing a number of objectives $\boldsymbol{f}$ as was already described in Section 5.1. The mathematical representation is repeated here:

$$\min \boldsymbol{f}(\boldsymbol{x}) = \min \left[f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \dots, f_m(\boldsymbol{x})\right]^T$$

where $\boldsymbol{x}$ represents the decision vector containing the parameters to be optimized and the problem is subjected to the following equality and inequality constraints:

$$\boldsymbol{h}(\boldsymbol{x}) = \left[h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \dots, h_i(\boldsymbol{x})\right]^T = 0 \qquad \text{for } i = 1, 2, \dots, j$$
$$\boldsymbol{g}(\boldsymbol{x}) = \left[g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), \dots, g_i(\boldsymbol{x})\right]^T \leq 0 \qquad \text{for } i = j, j+1, \dots, p$$

The decision vector contains a set of guidance parameters. After optimization, a final guidance law is found that results in a pseudo-optimal trajectory with respect to the objectives. Section 4.4.2 already discussed how the decision vector is formed and what guidance parameters can be included. Section 5.3.1 will explain the effect of different numbers of parameters and control nodes on the problem dimension. Subsequently Sections 5.3.2 and 5.3.3 will describe the objectives and constraints applicable to the trajectory optimization. For each objective and constraint a cost function is set up. To improve the optimization process, some additional penalty functions are created. The implementation and the effect of these functions is discussed in Section 5.3.4 . Section 5.3.5 combines the previously discussed cost and penalty functions of all objectives and constraints into a complete set of objectives functions that can be used for the constrained optimization problem. Finally Section 5.3.6 discusses a number of different approaches in which the objective function can be used in the optimization process.

5.3.1. Problem Dimension

The dimension of the optimization problem affects the optimization process. A larger problem dimension will result in a larger search space, which generally complicates the optimization process. The convergence speed will be slower, as it take more computational effort to search a larger space. Furthermore, the optimization is more likely to get stuck in a local optimum.

The dimension D of the problem is defined by:

$$D = m\left(n - n_{fixed}\right) \tag{5.26}$$

where m represents the number of optimization parameters, n the number of control nodes and n_{fixed} the number of control nodes for which the guidance parameters are fixed from the start. Throughout this research only the first control node will be fixed, but the final node could also be fixed if the guidance parameters at the final state have to satisfy certain conditions.

Equation 5.26 shows that reducing the number of guidance parameters or control nodes is a good way to reduce the dimension of the problem. Section 4.4.2 has already given an overview of the different combinations of guidance parameters that can be included in the decision vector. The minimum number of parameters used is two, in which case only $\hat{E}$ and α are included. When the magnitude and direction of thrust are included in the guidance for optimization, this number can go up to four. The number of control nodes at which each of the parameters is defined will be varied between 6 and 12.

The effect of the problem dimension on the ascent trajectory optimization will be included in this research. A comparison of different optimization parameters is discussed in Section 7.3. Results of a similar comparison for different numbers of control nodes follow in Section 7.4.

5.3.2. Objectives

To find a trajectory that is able to maximize the payload capacity to a target state, the following objectives are identified:

- Minimize the error in the final state.
- Minimize the fuel consumption

- Minimize the integrated heat load.

These objectives are optimized, while satisfying the constraints, which will be discussed shortly. For each of the objectives stated above a cost function is set up. These cost functions will indicate to what extent the objectives are satisfied. All cost functions are normalized so that different cost and penalty functions can be combined to form a single objective function.

Energy-state Cost Function
The mission does not target a specific orbit, as was explained in Section 2.4. The final state can therefore be defined by the velocity at a certain altitude. The normalized energy state as defined in Equation 4.41 will be a good reference point to evaluate the error in the final state. As the normalized energy state at the target state is equal to 1, the resulting cost function can be defined as:

$$c_E = 1 - \hat{E} \tag{5.27}$$

Fuel-mass Cost Function
The consumed fuel mass m_{cons} can be found by subtracting the final vehicle mass m_f from the take-off mass m_0. To normalize the result, m_{cons} is divided by the initial mass, which leads to the following fuel-mass cost function:

$$c_{fm} = \frac{m_0 - m_f}{m_0} \tag{5.28}$$

Heat-load Cost Function
The final total heat load Q can be computed by integrating the heat rate $\dot{Q}$ over the flight time. To normalize this value the total heat load is divided by the heat-rate constraint $\dot{Q}_{max}$ integrated over the flight time. This leads to the following heat-load cost function:

$$c_Q = \frac{\int_0^{t_f} \dot{Q} dt}{t_f \, \dot{Q}_c} \tag{5.29}$$

5.3.3. Constraints

The ascent trajectory is subject to a number of constraints. These constraints were discussed in Section 2.2.2 and are repeated here:

- Maximum dynamic pressure: $\bar{q}_c$ = 95,000 N/m^2
- Maximum heating rate: $\dot{Q}_c$ = 8,000 kW/m^2 at the leading edge of the wing (R_N = 0.1 m)
- Maximum axial load: $n_{a,c}$ = 1 g_0

It is noted that setting more or less stringent values for these constraints will have a significant effect on the resulting trajectory. Lu (1991) already varied $\bar{q}_c$ and found that the minimum value for which trajectory solution can be found was 71,772 N/m^2. Higher constraint values will allow the vehicle to accelerate at lower altitudes where the engine is more efficient. The optimization will be able to find solutions that satisfy all constraints more easily, when the constraint values are less stringent. Although interesting to study, varying constraint values will not be included in this research.

Penalty functions are set up to evaluate the extent to which constraint violation take place. They are derived in a similar way for all constraints. Each constraint is set to prevent failure of the vehicle. The flight is considered safe as long as the vehicle remains within the boundaries of the constraints. Therefore, as long as no violations of the constraints take place, no penalty is given.

From a physical point of view, only the maximum constraint violation is relevant as the duration of the actual violation does not contribute to the potential failure. However, it is reasonable to assume that a trajectory with one constraint violation is closer to a feasible solution than a trajectory with ten constraint violations, even if the maximum violation is equal. So for the optimization process, the number of violations and the duration of each violation is relevant.

The resulting constraint penalty functions are therefore a summation of the normalized maximum constraint violation and the normalized integrated constraint violation. All penalty functions are normalized so that different cost and penalty functions can be combined to form a single objective function. The resulting functions are given by:

$$p_{\bar{q}} = \begin{cases} 0 & \text{if } \bar{q}_{max} < \bar{q}_c \\ \frac{\bar{q}_{max} - \bar{q}_c}{\bar{q}_c} + \frac{\sum_{i=1}^{n} \int_{t_{0,i}}^{t_{f,i}} \left(\bar{q}_i(t) - \bar{q}_c\right) dt}{t_f \, \bar{q}_c} & \text{if } \bar{q}_{max} > \bar{q}_c \end{cases} \tag{5.30}$$

$$p_{\dot{Q}} = \begin{cases} 0 & \text{if } \dot{Q}_{max} < \dot{Q}_c \\ \frac{\dot{Q}_{max} - \dot{Q}_c}{\dot{Q}_c} + \frac{\sum_{i=1}^{n} \int_{t_{0,i}}^{t_{f,i}} \left(\dot{Q}_i(t) - \dot{Q}_c\right) dt}{t_f \, \dot{Q}_c} & \text{if } \dot{Q}_{max} > \dot{Q}_c \end{cases} \tag{5.31}$$

$$p_{n_a} = \begin{cases} 0 & \text{if } n_{a,max} < n_{a,c} \\ \frac{n_{a,max} - n_{a,c}}{n_{a,c}} + \frac{\sum_{i=1}^{n} \int_{t_{0,i}}^{t_{f,i}} \left(n_{a,i}(t) - n_{a,c}\right) dt}{t_f \, n_{a,c}} & \text{if } n_{a,max} > n_{a,c} \end{cases} \tag{5.32}$$

where the first term in each equation gives the penalty for the maximum constraint violation and the second term represents the summation of n constraint violations integrated over the duration of the violation.

5.3.4. Trajectory Penalties

To increase the effectiveness of the objective functions, a few additional penalty functions will be studied:

- Monotonic energy-state penalty
- Flight-path oscillation penalty
- No-flight penalty
- Fuel-mass penalty

Each of the penalty functions will be discussed separately.

Monotonic Energy-state Penalty

Up until the final pull-up the vehicle should be ascending and accelerating throughout the whole trajectory. As the initial decision vectors are set randomly, it is very likely that the guidance laws produced by the first generations do not result in trajectories with a monotonically ascending and accelerating vehicle.

Because the guidance is a function of the energy state, a decreasing energy state can lead to a repetition of part of the guidance. As a result the vehicle can get stuck in a guidance cycle. Assume, for example, the following very simple decision vector:

$$\boldsymbol{x_v} = \left[\hat{E}_1,\, \hat{E}_2,\, \alpha_1,\, \alpha_2,\, \phi_1,\, \phi_1\right] = \left[0.1,\, 0.2,\, 3°,\, 0°,\, 1,\, 0\right] \tag{5.33}$$

At an energy state of 0.1 the guidance law prescribes $\alpha = 3.0°$ and $\phi = 1.0$, resulting in an accelerating and ascending vehicle. The energy state will increase towards the value of 0.2 and both α and ϕ will drop. If both values drop enough, the energy state will decrease again towards a value of 0.1. The values of α and ϕ will increase again and the whole cycle repeats.

Because these repeating cycles are costing valuable computation power without leading anywhere, a penalty will be included to penalize the non-monotonic behavior of the energy state. The penalty is defined by the following function:

$$\Delta p_{E_{mon},i} = \begin{cases} 0 & \text{if } \hat{E}_i > \hat{E}_{i-1} \\ \hat{E}_{i-1} - \hat{E}_i & \text{if } \hat{E}_i < \hat{E}_{i-1} \end{cases} \tag{5.34a}$$

$$p_{E_{mon}} = \sum_{i=0}^{n} \Delta p_{E_{mon},i} \tag{5.34b}$$

where n is the total number guidance evaluations.

No-flight Penalty
As mentioned before, the initial solutions found by the optimizer will not result in trajectories reaching the final state. In fact, the majority of the initial solutions will crash into the Earth very soon after take-off. Of course, a vehicle that does not fly will not come close to the final state and will thus perform badly on this objective. However, by not flying, the vehicle will not use any fuel, the heat load will be close to zero and no boundary constraints are violated. The optimizer will see non-flying solutions as very good solutions on all accounts but one, and as a result the optimizer will converge to only non-flying solutions very fast.

This effect is prevented by including a no-flight penalty. For the penalty function to have the desired effect, it should comply to the following requirements:

- The effect of the penalty should be larger than the constraint penalties and the cost functions found for marginally flying trajectories.
- No-flight trajectories with a higher final energy state should outperform no-flight trajectories with a lower final energy state.

where a no-flight trajectory is defined as a trajectory with a final normalized energy state below 0.1. This value is determined empirically and is based on the fact that solutions with a final energy state above 0.1 were found to converge easily towards solutions reaching the final state. The resulting penalty function is now given by:

$$p_{nf} = \begin{cases} 0 & \text{if } \hat{E}_f > 0.1 \\ K\left(0.1 - \hat{E}_f\right)^2 & \text{if } \hat{E}_f < 0.1 \end{cases} \tag{5.35}$$

where $\hat{E}_f$ is the final normalized energy state and the weight factor is set empirically to $K = 1000$. Higher values can be selected without a significant change in performance. Much lower values should be avoided, as the first requirement stated above will not be met in that case.

Flight-path Oscillation Penalty
Initial results have shown that the found trajectories often have a strongly oscillating flight-path angle at the second part of the trajectory. The result is a phugoid motion where kinetic and potential energy are interchanged. Theory predicts that the vehicle will perform optimal when tracking the boundary constraints. Figure 2.11 shows that the boundary constraints do not show any oscillations in the velocity-altitude graph. The vehicle will therefore be able to better track the boundary constraints (without violating any) when this phugoid motion is not present.

The oscillation is seen as an undesirable effect and a penalty function is created to discourage this behavior. Up until the pull-up the flight-path angle should comply with the following requirements in accordance with the reference trajectory provided by Mooij (1998):

- The flight-path angle is always positive ($\gamma > 0$).
- The flight-path angle rate is always negative ($\dot{\gamma} < 0$).

It is noted that the second statement is only valid for the atmospheric acceleration phase. The flight-path oscillation penalty should therefore only be applied to this phase. Based on these requirements the flight-path angle penalty can be defined by:

$$\Delta p_{\gamma,i}(t) = \begin{cases} 0 & \text{if } \gamma_i(t) > 0 \\ \gamma_i(t) & \text{if } \gamma_i(t) < 0 \end{cases} \tag{5.36a}$$

$$\Delta p_{\gamma} = \int_{t_0}^{t_f} \Delta p_{\gamma,i}(t)\, dt \tag{5.36b}$$

where the angles are given in radians. The flight-path angle rate penalty can be defined in a similar way by:

$$\Delta p_{\dot{\gamma},i}(t) = \begin{cases} 0 & \text{if } \dot{\gamma}_i(t) < 0 \\ \dot{\gamma}_i(t) & \text{if } \dot{\gamma}_i(t) > 0 \end{cases} \tag{5.37a}$$

$$\Delta p_{\dot{\gamma}} = \int_{t_0}^{t_f} \Delta p_{\dot{\gamma},i}(t)\, dt \tag{5.37b}$$

where the angular rates are given in radians per second. finally the total normalized flight-path angle penalty function is given by:

$$p_\gamma = \frac{\Delta p_\gamma + \Delta p_{\dot{\gamma}}}{2\pi} \tag{5.38}$$

Fuel-mass Penalty
When the fuel-mass cost function is taken as an optimization parameter, a better result will be found for no-flight trajectories. A penalty function is therefore created to account extra fuel mass for a lack of altitude and velocity at the final state. Tsiolkovsky rocket equation is used to estimate the additional fuel mass.

First an effective final velocity V_{eff} at the objective altitude h_{obj} is computed. The final energy state is computed with:

$$E_f = gh_f + \frac{1}{2}V_f^2 \tag{5.39}$$

The effective final velocity is now defined as the velocity the vehicle would have if the final energy state was obtained at the objective altitude:

$$V_{eff} = \sqrt{2\left(E_f - gh_{obj}\right)} \tag{5.40}$$

Using Tsiolkovsky's equation with a specific impulse I_{sp} = 465 seconds, the additional fuel-mass penalty can now be computed according to:

$$m_{penalty} = m_f\left(1 - \frac{1}{e^{\frac{\Delta V}{g_0 I_{sp}}}}\right) \tag{5.41}$$

where m_f represents the mass of the vehicle at the final state and $\Delta V = V_f - V_{eff}$. Finally the normalized fuel-mass penalty is defined by:

$$p_{fm} = \frac{m_{penalty}}{m_0} \tag{5.42}$$

5.3.5. Objective Functions

Cost and penalty functions defined in the previous section can be combined to obtain specific objective functions. These objective functions are used to guide the optimization towards the desired result. The objective functions can be split into functions that lead to feasible solutions, and function that lead to optimal solutions. Both subsets will be discussed separately.

Feasibility Objectives
Trajectories that reach the desired final state without violating any of the boundary constraints are called feasible trajectories. The optimization results are guided to these solutions by a set of feasibility objective functions. Sets of feasibility objective functions can be formed in multiple ways. Different options will be discussed here with the potential advantages and disadvantages of each. The following functions will be used:

$$O_{f,1a} = c_E + p_{E_{mon}} \tag{5.43a}$$
$$O_{f,2a} = p_{\bar{q}} + p_{\dot{Q}} + p_{n_a} + p_{nf} \tag{5.43b}$$
$$O_{f,3a} = p_{\gamma} \tag{5.43c}$$

where $O_{f,1a}$ represents the energy-state objective, $O_{f,1b}$ the constraint objective and $O_{f,1c}$ flight-path angle objective, with:

$$\boldsymbol{O_{f,a}} = \begin{pmatrix} O_{f,1a} \\ O_{f,2a} \\ O_{f,3a} \end{pmatrix}$$

The more objectives are included in the optimization problem, the more complicated the problem becomes. This often leads to a poorer performance. Two alternatives are therefore given that only include two objectives. The first alternative simply does not take the flight-path angle objective into account resulting in the set:

$$O_{f,1b} = c_E + p_{E_{mon}} \tag{5.44a}$$
$$O_{f,2b} = p_{\bar{q}} + p_{\dot{Q}} + p_{n_a} + p_{nf} \tag{5.44b}$$

where

$$\boldsymbol{O_{f,b}} = \begin{pmatrix} O_{f,1b} \\ O_{f,2b} \end{pmatrix}$$

The second alternative combines objective $O_{f,2a}$ and $O_{f,3a}$ into a single objective function resulting in the following set:

$$O_{f,1c} = c_E + p_{E_{mon}} \tag{5.45a}$$
$$O_{f,2c} = p_{\bar{q}} + p_{\dot{Q}} + p_{n_a} + p_{nf} + p_{\gamma} \tag{5.45b}$$

where

$$\boldsymbol{O_{f,c}} = \begin{pmatrix} O_{f,1c} \\ O_{f,2c} \end{pmatrix}$$

To test the no-flight penalty function both $\boldsymbol{O_{f,b}}$ and $\boldsymbol{O_{f,c}}$ are also ran without the inclusion of p_{nf}. This results in the following two sets:

$$O_{f,1d} = c_E + p_{E_{mon}} \tag{5.46a}$$
$$O_{f,2d} = p_{\bar{q}} + p_{\dot{Q}} + p_{n_a} \tag{5.46b}$$

where

$$\boldsymbol{O_{f,d}} = \begin{pmatrix} O_{f,1d} \\ O_{f,2d} \end{pmatrix}$$

and

$$O_{f,1e} = c_E + p_{E_{mon}} \tag{5.47a}$$
$$O_{f,2e} = p_{\bar{q}} + p_{\dot{Q}} + p_{n_a} + p_{\gamma} \tag{5.47b}$$

where

$$\boldsymbol{O_{f,e}} = \begin{pmatrix} O_{f,1e} \\ O_{f,2e} \end{pmatrix}$$

The performance of each of the sets of feasibility objectives will be compared and results are given in Section 7.2.

Optimality Objectives
Trajectories that reach the desired final state with a minimal heat load and fuel consumption during the flight are called optimal trajectories. The optimization results are guided to these solutions by a set of optimality objective functions. The following set of functions will be used:

$$O_{o,1} = c_{fm} + p_{fm} \tag{5.48a}$$
$$O_{o,2} = c_Q + c_E \tag{5.48b}$$

where

$$\boldsymbol{O_{o,}} = \begin{pmatrix} O_{o,1} \\ O_{o,2} \end{pmatrix}$$

The energy-state cost function c_E and the fuel-mass penalty are included in the objective functions to stimulate the optimality objectives not to guide the solutions towards a no-flight trajectory.

5.3.6. Optimization Approach

Different approaches can be used to find a trajectory that is both feasible and optimal. A number of approaches will be discussed here.

The first approach simply optimizes both the feasibility objectives and the optimality objectives at the same time. This is the simplest approach, but leads to a problem definition with 4 or 5 objective functions. This large number of objectives will complicate the problem and reduce the performance of the optimizer. A schematic overview of this approach is given in Figure 5.4.

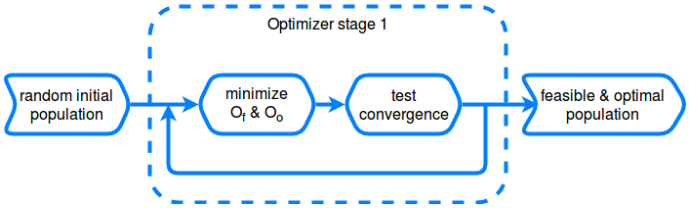


Figure 5.4: Optimization approach 1.

Another approach is to first optimize for the feasibility objectives. Once a population with feasible solutions is found, a second optimization stage is started that further evolves the feasible population with respect to the optimality objectives. However, the second stage could push the population back into the unfeasible space. This problem can be resolved by either adding the feasibility objectives to the second stage as is shown in Figure 5.5. A second option is to iterate over the two stages until the final population is both feasible and optimal. This approach is shown in Figure 5.6.

A final approach that can be used, optimizes with respect to the feasibility objectives first. A second stage is then used to optimize locally with respect to the optimality objectives around the found feasible solution. This local search is implemented by setting boundary values for each parameter in the decision vector according to:

$$\boldsymbol{lb_i} = \boldsymbol{x_{v,i}} - C\,\boldsymbol{x_{v,i}} \tag{5.49a}$$
$$\boldsymbol{ub_i} = \boldsymbol{x_{v,i}} + C\,\boldsymbol{x_{v,i}} \tag{5.49b}$$

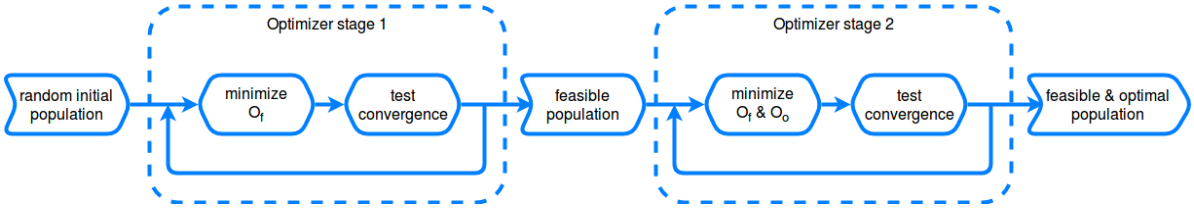


Figure 5.5: Optimization approach 2.

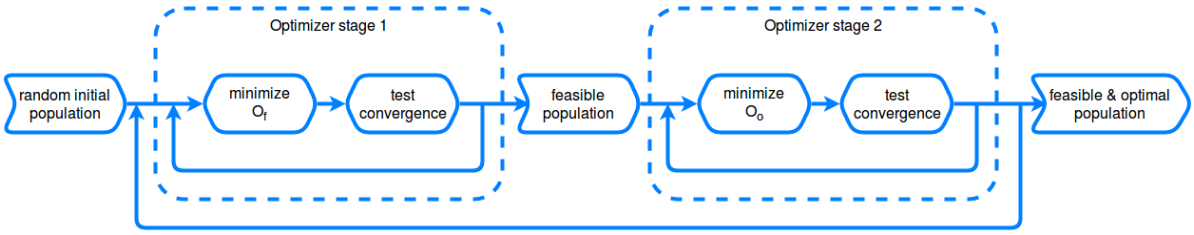


Figure 5.6: Optimization approach 3.

where C is a very small constant number that depends on the sensitivity of the problem. If C is chosen too high the population might evolve towards non-feasible solutions. If C is too small a restricted search space will be the result, which limits the possible improvement with respect to the optimality objectives. An initial value of $C = 1 \cdot 10^{-3}$ will be used. A schematic overview of this method is given in Figure 5.7.
The different approaches proposed in this section will be compared in this research. The results can be found in Section 7.6.

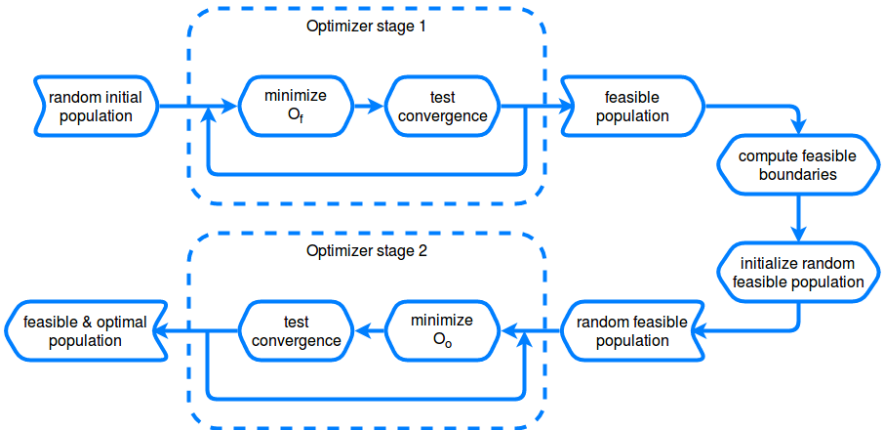


Figure 5.7: Optimization approach 4.

6

Simulation and Optimization Software

The trajectory simulation and optimization described in the previous chapters are implemented in the form of a software model. This chapter will explain the implementation. Section 6.1 will first describe the external software packages that have been used. Subsequently Section 6.2 discusses the trade-off and working of a number of numerical methods that are present in the simulation software. Finally, Section 6.3 explains the software architecture of both the trajectory simulation and the optimization model, followed by a the verification of the obtained software model in Section 6.4.

6.1. External Software

The software model is created using external software packages. The existing software is extended to create the software architecture that will be further discussed in Section 6.3. The complete software architecture can be split in the trajectory simulation model and the optimizer. The trajectory simulation model is created using the Tudat software. This toolbox is further discussed in Section 6.1.1. The optimization model uses already implemented optimizers from the PaGMO library. This software toolbox is further discussed in Section 6.1.2.

6.1.1. Tudat Simulation Software

The TU Delft Astrodynamics Toolbox (Tudat) is an open source software toolbox developed and maintained by students and staff of the Astrodynamics & Space missions section of the faculty of Aerospace Engineering at Delft University of Technology. The repository includes models and functions programmed in C++ that can be used to set up a simulation. The following models from Tudat are used for the trajectory simulation:

- Atmosphere models
- Gravity models
- Earth shape and rotation models
- Interpolation models
- Integration models
- The general trajectory simulation structure

The existing Tudat software is extended with problem-specific models for the vehicle, guidance and external forces. The architectures for the separate models and complete framework are discussed in Section 6.3.

In each of the separate models a number of numerical methods are used for computations, including methods for integration, interpolation and data search. Different methods are available in Tudat for each. Section 6.2 gives a trade-off of the methods and explains how all the selected methods work.

6.1.2. PaGMO Optimization Software

Parallel Global Multi-objective Optimizer (PaGMO) is a software toolbox developed within the Advanced Concepts Team at ESA. Its purpose is to tackle high-dimensional global optimization problems (Biscani et al., 2010). The PaGMO library contains a number of global and local optimization algorithms coded in C++ including the NSGA-II and MOEA/D methods used for this research. The implementation in PaGMO also provides automatic parallelization of the optimization process via a coarse-grained approach based on the island model (Biscani et al., 2010). This means advantage can be taken of the capabilities of modern multi-core processors, without requiring knowledge of parallelizing algorithms.

The optimizers of PaGMO that will be used require an optimization problem definition as input. This problem should always be derived from the PaGMO 'base' class, which represents a box-bounded, multi-objective, mixed-integer, constrained optimization problem defined by [1]:

- the dimension of the global search space,
- the dimension of the integral (or combinatorial) part of the problem,
- the lower and upper bounds of the search space,
- the total number of constraints,
- the number of inequality constraints,
- the constraint computation functions,
- the objective functions,
- a fitness dimension, and
- a constraints tolerance vector.

The output of the optimizers is a control history represented by a vector with values of the control variables at different control nodes.

6.2. Numerical Methods

This section will give a trade-off for numerical methods that are used in the different simulation sub-models. Finally a more in depth explanation of the selected methods is given. Integration, interpolation and data-search will be discussed

6.2.1. Integration

To determine whether the control history found with the optimization process results in a good trajectory, the equations of motion will need to be integrated. The integration will be performed with a numerical integrator. The general form of a numerical integration can be written as:

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\, A(t_n, h, \boldsymbol{y}_n) \tag{6.1}$$

with

$$\dot{\boldsymbol{y}} = f(t, \boldsymbol{y}) \tag{6.2}$$

where h is the step-size, A is an increment function depending on the integration method, t_n is the current time, and $\boldsymbol{y}_n$ and $\dot{\boldsymbol{y}}_n$ represent the current state and its derivative respectively. With the equations of motion expressed in the inertial frame in Cartesian coordinates, $\boldsymbol{y}_n$ and $\dot{\boldsymbol{y}}_n$ look like:

$$\boldsymbol{y}_n = \begin{pmatrix} x_I \\ y_I \\ z_I \\ \dot{x}_I \\ \dot{y}_I \\ \dot{z}_I \end{pmatrix} \quad \text{and} \quad \dot{\boldsymbol{y}}_n = \begin{pmatrix} \dot{x}_I \\ \dot{y}_I \\ \dot{z}_I \\ \ddot{x}_I \\ \ddot{y}_I \\ \ddot{z}_I \end{pmatrix}$$

In this section different integration methods will be assessed. A restriction is made to the methods available in Tudat, where the Euler method is neglected as it is unstable. This leaves the following set:

[1] Biscani, F. (2010). pagmo::problem::base class reference. `http://esa.github.io/pagmo/classpagmo_1_1problem_1_1base.html` [Accessed: 1 June 2017]

- fixed step Runge-Kutta of order 4
- variable step Runge-Kutta-Fehlberg of order 4-5
- variable step Runge-Kutta-Fehlberg of order 5-6
- variable step Runge-Kutta-Fehlberg of order 7-8
- variable step Runge-Kutta-Dormand-Prince of order 8-7

A trade-off of the methods listed above will be made based on the accuracy and efficiency of the methods. Finally one integration method will be selected.

Integrator Selection
Research by Jackson et al. (1978) showed that all variable step-size RKF integration methods outperform the simple RK-4 method, on both linear and non-linear problems. For non-linear problems the Dromand-Prince RK-87 method was found to be even better (Prince and Dormand, 1981).

Variable step-size integrators have the advantage of being able to adapt the step-size of the integrator based on the local truncation error. This allows the integrator to use much larger step-sizes on parts of the problem while still using a small step-size for the more demanding parts of the problem. However, for the problem at hand the maximum integration step-size is limited by the guidance step-size. The variable step-size will therefore only be advantageous if many integration steps will be used within each guidance step.

A test has been performed to see if a variable step-size would be advantageous for the current simulation. For this test the simulation is run with different guidance and integration step-sizes. Appendix B explains the tests and shows results. It was concluded that multiple integration steps are only required if large guidance step-sizes are selected. However, a small guidance step-size, below 0.2 seconds, is required to guarantee accuracy of the results. Using a variable step-size integrator is therefore not advantageous and the RK-4 integrator is selected for this research.

RK-4 Integrator
In this section the RK-4 method will be discussed. The method is well known and a good explanation of the algorithm can be found for example in Burden and Faires (2001). Four estimations are made for the change of the solution in one integration step. A weighted average of these approximations is computed to find the final solution. This is expressed in the following equations:

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} \tag{6.3}$$

with the coefficient equations for k given by:

$$\begin{aligned}
k1 &= h\, f\left(t_n, \quad \boldsymbol{y}_n\right) \\
k2 &= h\, f\left(t_n + \frac{h}{2}, \quad \boldsymbol{y}_n + \frac{k_1}{2}\right) \\
k3 &= h\, f\left(t_n + \frac{h}{2}, \quad \boldsymbol{y}_n + \frac{k_2}{2}\right) \\
k4 &= h\, f\left(t_n + h, \quad \boldsymbol{y}_n + k_3\right)
\end{aligned}$$

where h indicates the integration step-size and f is a function that gives an approximation of the derivative at a certain location.

6.2.2. Interpolation

The optimizers discussed in Section 5.2 will optimize a set of control parameters at certain control points. However, the values between the control points are also required for integration of the trajectory. Furthermore, the data of the vehicle models and the atmospheric model is given in tabular form. Data between these tabulated values will be required for the proper calculations of forces on the vehicle. To find all these in-between values, interpolation is required. In this section different interpolation methods will be discussed and different methods are chosen for a variety of implementations. The methods under consideration are:

- linear interpolation
- Lagrange polynomial interpolation
- Cubic spline interpolation
- Hermite spline interpolation

A trade-off will be made first to see which methods are most suited for the different purposes. Subsequently the chosen methods will be explained in more detail. A more in depth explanation of the different methods can be found in Burden and Faires (2001).

Interpolation Selection

Using Lagrange polynomial, cubic splines or Hermite splines has the advantage of producing continuous functions that can be differentiated at the control points. However, Lagrange polynomials have the disadvantage that with an increasing number of control points, higher degree polynomials are required. These higher degree polynomials tend to oscillate. Figure 6.1 shows an example of a Lagrange interpolating polynomial of degree 20 trying to fit data points of the back side of a duck figure. It is clear that this would not give reliable results for the trajectory optimization. The Lagrange interpolation is therefore rejected.

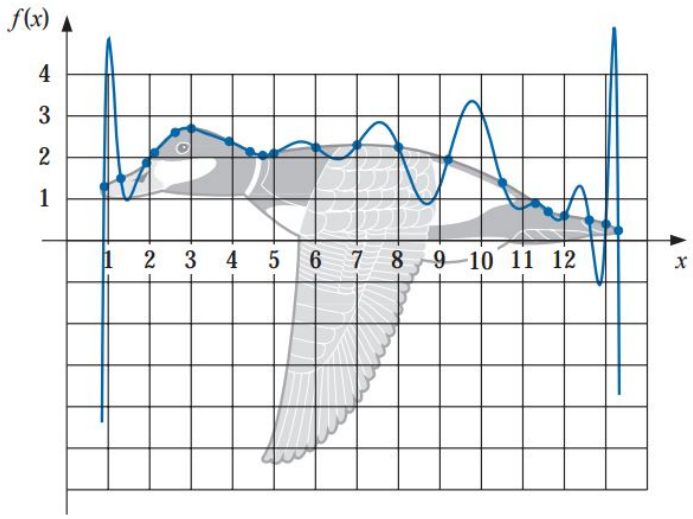


Figure 6.1: Example of a Lagrange interpolating polynomial of degree 20 (Burden and Faires, 2001).

Spline interpolation uses piecewise polynomial interpolation that creates a different polynomial between each set of points. Hermite splines are of order 3 and demand a continuous first derivative in the points. Cubic splines are of order 4 and demand continuity of both the first and second derivatives in the points. Figure 6.2 shows an example of a piecewise cubic-spline interpolation.

Figure 6.3 shows the piece-wise linear interpolation of the same set of data points. It is clear that the results are close, but the linear interpolation does not result in a smooth function. It is noted that linear interpolation of the controls could lead to sudden changes in, for example, the angle of attack and bank angle. These sudden changes might not be possible in reality, meaning that continuous results of a spline interpolation would be preferred. Because the controls do not need to be continuous in the second derivative, Hermite spline interpolation is chosen for the controls. Furthermore, Hermite splines allow for enforcing monotonicity. This makes sure that controls cannot overshoot between the control points (Fritsch and Carlson, 1980).

The aerodynamic and propulsive coefficients do not need to be continuous with time. Since the vehicle models are based on many assumptions, the error made by linear interpolation is assumed to be relatively small. Because linear interpolation is less computationally expensive, this method will be used to extract data from the tabulated aerodynamic and propulsive models. It is noted that coefficients depend on a number of variables requiring multi-dimensional linear interpolation.

Cubic-spline interpolation will be used to extract the density of the atmosphere. This method has already been implemented in the used atmospheric models available in Tudat. Cubic splines are required

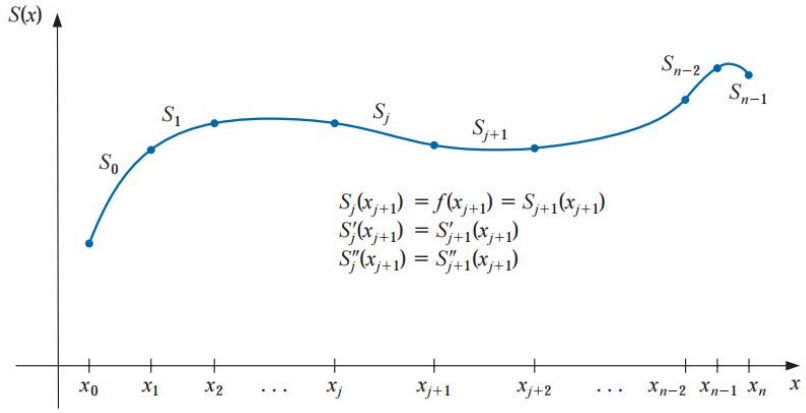


Figure 6.2: Example of a piecewise cubic spline interpolation (Burden and Faires, 2001).

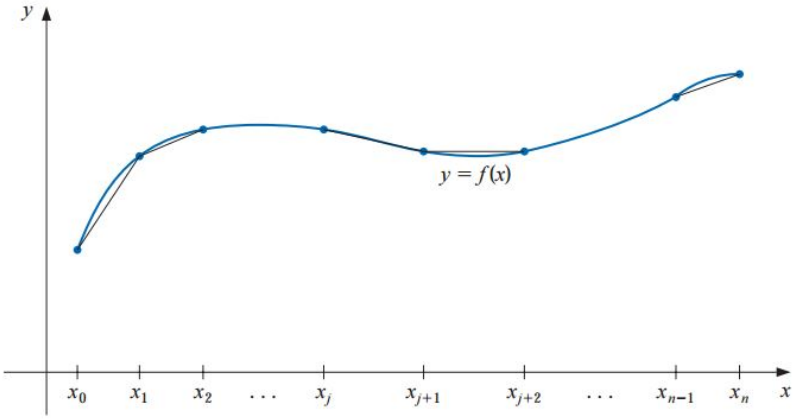


Figure 6.3: Example of a piecewise linear interpolation (Burden and Faires, 2001).

because discontinuities in the derivatives will cause unnatural spikes in the calculated forces and thus the accelerations.

Linear Interpolation

One-dimensional linear interpolation will be shortly explained in this section with help of Figure 6.4. Imagine points A and B are known function values, and the function value y_2 of point C at a known x_2 is requested. The function value y_2 can now be found with:

$$y_2 = y_0 + \frac{y_1 - y_0}{x_1 - x_0}(x_2 - x_0) \tag{6.4}$$

With this equation the function value y_i for any x_i with $x_0 \leq x_i \leq x_1$ can be found. To extend to multi-linear interpolation, Equation 6.4 is rewritten in the following form:

$$y_2 = y_0 \frac{x_1 - x_2}{x_1 - x_0} + y_1 \frac{x_2 - x_0}{x_1 - x_0} = y_0 N_a + y_1 N_b \tag{6.5}$$

where N_A and N_B represent the normalized distances along the x-axis.

The form of Equation 6.5 can be extended to any dimension. Multi-dimensional linear interpolation will be used for extraction of coefficients of the aerodynamic and propulsion model. Because most of the coefficients of the aerodynamic model are a function of 3 parameters, tri-linear interpolation will be shown as an example. Three dimensions also allow for a visual representation as is shown in Figure 6.5.

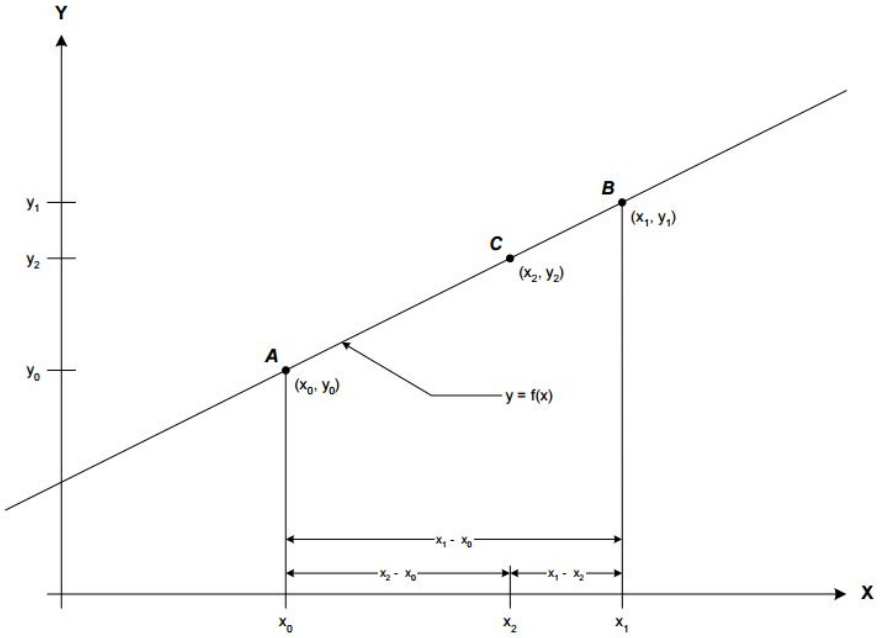


Figure 6.4: Example of one-dimensional linear interpolation.

If the function value $v(x, y, z)$ is known at points A to H, the function value $v_8(x_2, y_2, z_2)$ at point I can now be found with:

$$v_8 = v_0 N_A + v_1 N_B + v_2 N_C + v_3 N_D + v_4 N_E + v_5 N_F + v_6 N_G + v_7 N_H \tag{6.6}$$

where N_A to N_H now represent the normalized volumes between the known points A to H and the requested point I. Their values are given by:

$$N_A = \frac{(x_1 - x_2)(y_1 - y_2)(z_2 - z_0)}{(x_1 - x_0)(y_1 - y_0)(z_1 - z_0)} \qquad N_E = \frac{(x_1 - x_2)(y_1 - y_2)(z_1 - z_2)}{(x_1 - x_0)(y_1 - y_0)(z_1 - z_0)}$$
$$N_B = \frac{(x_1 - x_2)(y_2 - y_0)(z_2 - z_0)}{(x_1 - x_0)(y_1 - y_0)(z_1 - z_0)} \qquad N_F = \frac{(x_1 - x_2)(y_2 - y_0)(z_1 - z_2)}{(x_1 - x_0)(y_1 - y_0)(z_1 - z_0)}$$
$$N_C = \frac{(x_2 - x_0)(y_1 - y_2)(z_2 - z_0)}{(x_1 - x_0)(y_1 - y_0)(z_1 - z_0)} \qquad N_G = \frac{(x_2 - x_0)(y_1 - y_2)(z_1 - z_2)}{(x_1 - x_0)(y_1 - y_0)(z_1 - z_0)}$$
$$N_D = \frac{(x_2 - x_0)(y_2 - y_0)(z_2 - z_0)}{(x_1 - x_0)(y_1 - y_0)(z_1 - z_0)} \qquad N_H = \frac{(x_2 - x_0)(y_2 - y_0)(z_1 - z_2)}{(x_1 - x_0)(y_1 - y_0)(z_1 - z_0)}$$

Cubic Spline Interpolation

As already mentioned before, cubic spline interpolation will be used in the tabulated atmospheric model. Cubic splines are continuous in the first and second derivative over the whole interval in all points. Assuming a tabulated function $y_i = f(x_i)$, the interpolating spline at an interval $[x_j, x_{j+1}]$ is given by:

$$y = A y_j + B y_{j+1} + C y_j'' + D y_{j+1}'' \tag{6.7}$$

where A, B, C and D are defined as:

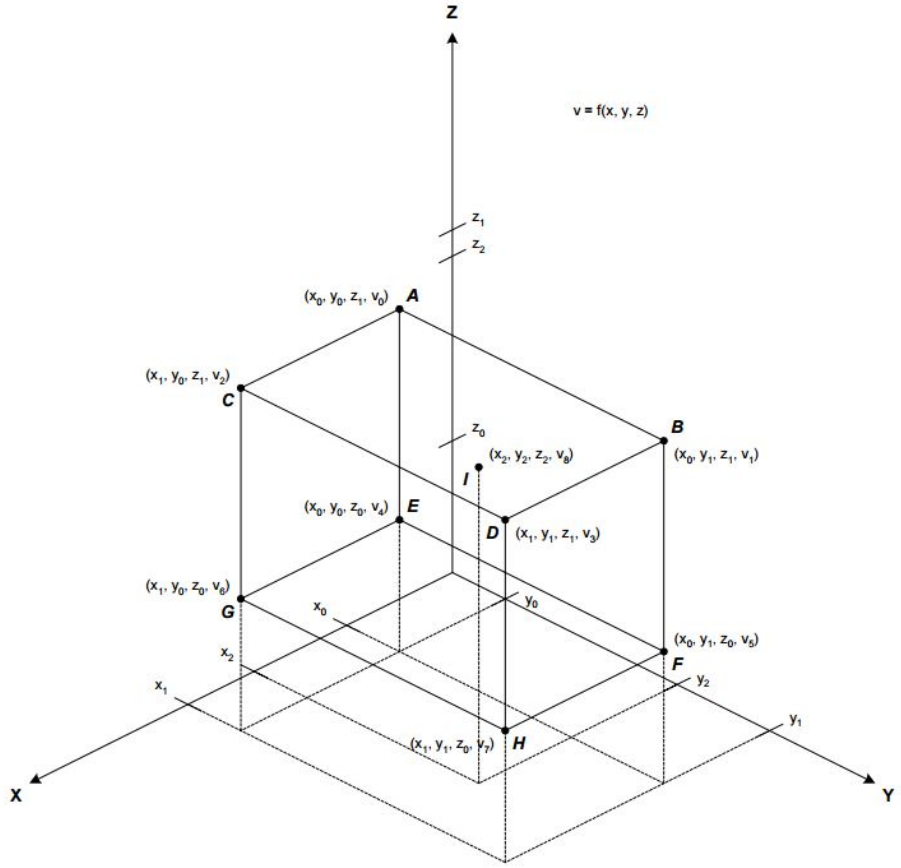


Figure 6.5: Example of tri-linear interpolation.

$$A \equiv \frac{x_{j+1} - x}{x_{j+1} - x_j}$$
$$B \equiv 1 - A = \frac{x - x_j}{x_{j+1} - x_j}$$
$$C \equiv \frac{1}{6}(A_3 - A)(x_{j+1} - x_j)^2$$
$$D \equiv \frac{1}{6}(B_3 - B)(x_{j+1} - x_j)^2$$

The method is further explained in Press et al. (2002). Because the values of the second derivatives at the nodes are not known, they will need to be calculated. The implementation in Tudat assumes the first derivative to be continuous at the nodes and curvature to be zero at the endpoints. Press et al. (2002) gives a more in depth explanation on the working of this method.

Hermite Spline Interpolation

Hermite splines are specified by the function values and the first derivatives at the endpoints of the domains. This results in a continuous first derivative over all domains. For an arbitrary interval $[x_j, x_{j+1}]$ the interpolated value $p(x)$ can be found with:

$$p(x) = H_1(t)p_j + H_2(t)(x_{j+1} - x_j)m_j + H_3(t)p_{j+1} + H_4(t)(x_{j+1} - x_j)m_{j+1} \tag{6.8}$$

where p_i and m_i are the function values and their derivatives at the nodes respectively. $H(t)$ represent the Hermite basis functions given by:

$$H_1(t) = 2t^3 - 3t^2 + 1$$
$$H_2(t) = t^3 - 2t^2 + 1$$
$$H_3(t) = -2t^3 + 3t^2$$
$$H_4(t) = t^3 - t^2$$

and

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

6.2.3. Data Search

To interpolate table values, the right values will first need to be found. The interpolation methods implemented in Tudat already include a binary search as data search method. This method will therefore be applied. Binary search is implemented as described in Press et al. (2002). The method eliminates half of the table at each step , until the final value is found. A visualization is shown in Figure 6.6

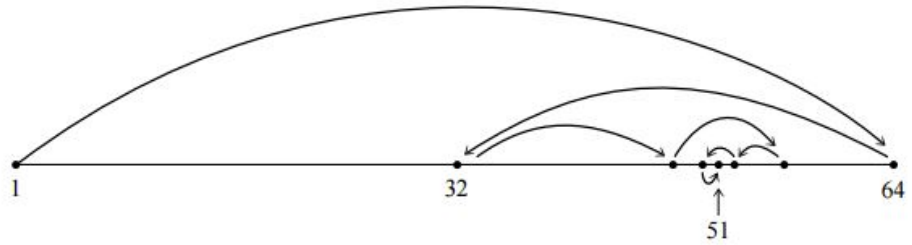


Figure 6.6: Example of binary search (Press et al., 2002).

6.3. Software Architecture

The complete software architecture is split in the trajectory simulation and the optimization, where the result of the trajectory simulation forms the basis of the objective functions for the optimization. The trajectory simulation model and its sub-modules will be discussed in Section 6.3.1. The optimization architecture is shown in Section 6.3.2.

6.3.1. Simulation model

The complete simulation model combines the following modules:

- environment
- guidance
- vehicle
- external forces
- propagation

All of the modules will be explained separately. The environment is easy to implement. Only the right model needs to be selected. The selection of the environmental models used for this simulation is described in Section 3.1. The resulting architecture for the environment is given in Section 6.3.1 and is more or less the same for any application implemented in Tudat. The vehicle models described in Section 3.2 are problem specific and will be implemented from scratch. Finally, all models are tied together in a simulation framework that is already present in Tudat.

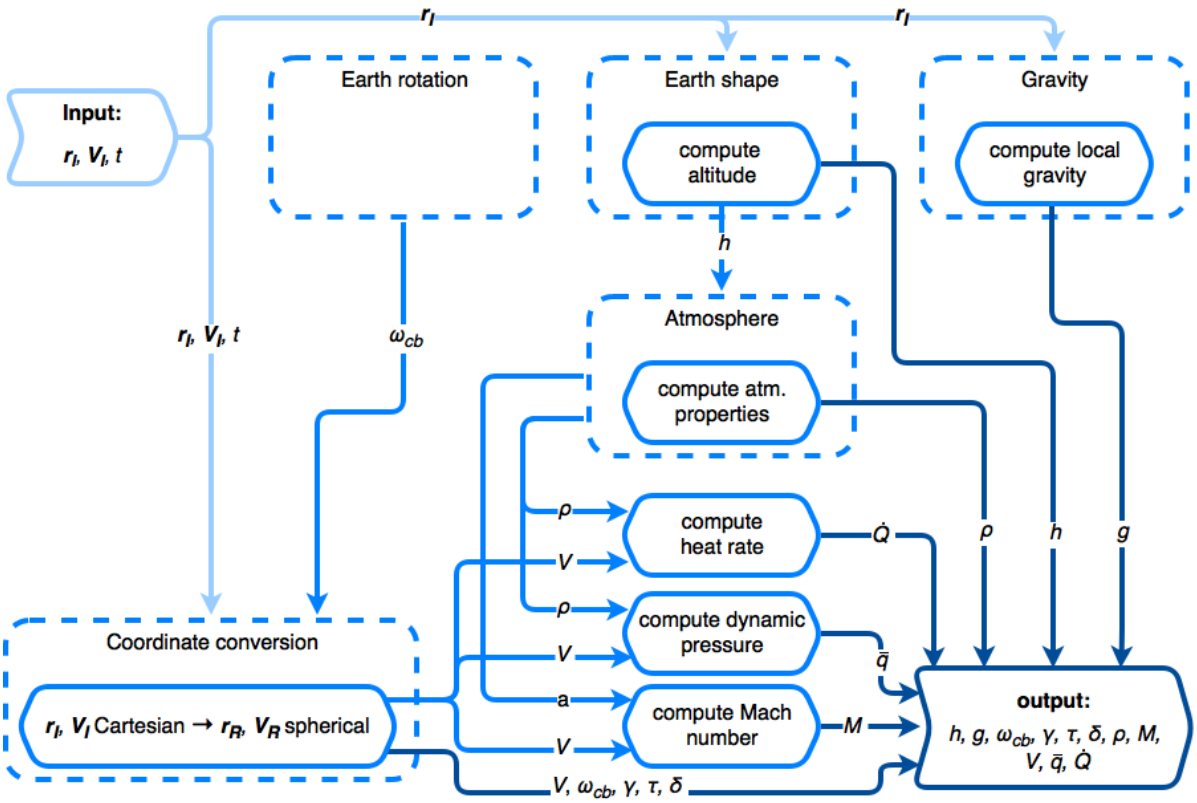


Figure 6.7: Architecture of the environment model.

Environment

The environment includes an atmosphere model, gravity model and Earth shape and rotation models. It is relatively easy to implement as the framework and the individual models are already present. Only the right model needs to be selected. The resulting architecture for the environment is shown in Figure 6.7 and will be more or less the same for any application implemented in Tudat.

The input of the environment consists of the state including velocity and position at a given time. The state is converted from the inertial Cartesian frame to a rotational spherical frame. Subsequently the environmental models compute and output the local gravitational acceleration g, the rotational rate of the Earth ω_{cb}, the local atmospheric density ρ, the current Mach number M, the dynamic pressure $\bar{q}$ and the current velocity V.

Guidance

The guidance module takes the decision vector ($\boldsymbol{x_v}$) from the optimization as input. This decision vector gives values for guidance parameters at a number of control nodes, where the location of the control nodes is given by the normalized energy state ($\hat{E}$). In order to find the guidance at the current time the normalized energy state is first computed. A Hermite spline interpolator is then used to compute values of each guidance parameter as a function the current normalized energy state. Figure 6.8 shows a schematic visualization of this architecture.

It is conventional to include trim in the guidance model. However, the guidance described here is only used to compute values of the guidance parameters as a function of time. Because rotational dynamics are not included, the effect of trim on the guidance is not taken into account. Including trim within the guidance model is therefore not a strict requirement. It was found to be more convenient to place trim within the vehicle model, as different sets of guidance parameters can be used in the simulation. Each setup requires a different interaction with the vehicle models as will be further explained shortly.

The different sets of guidance parameters that can be used were already explained in Section 4.4.2. The angle of attack (α) is the main guidance parameter. For the standard problem either the equivalence ratio (ϕ) or the throttle factor (F_T) will be used to control the magnitude of the thrust. When F_T is used,

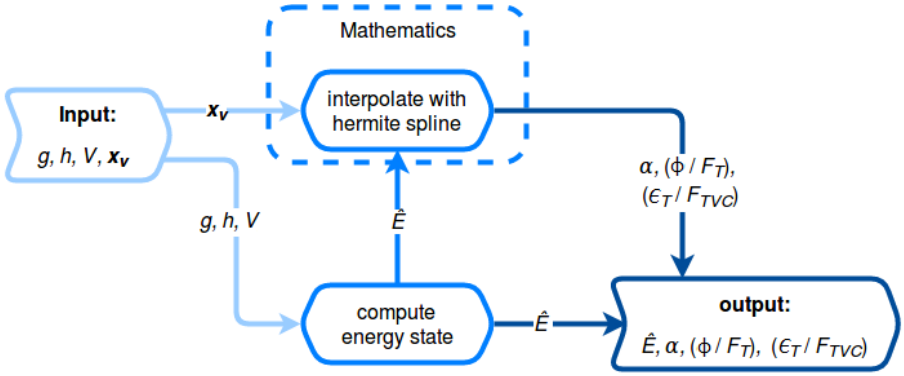


Figure 6.8: Architecture of the guidance model.

ϕ is set to the nominal value of 1.0. Thrust can also be controlled with the throttle law, which tracks the constraints. Thrust settings will then be determined in the propulsion model, which outputs either F_T or ϕ depending on the user settings. As both F_T and ϕ are, in this case, not used as active guidance parameters, this reduces the number of optimization parameters in the decision vector. If TVC is used an additional guidance parameter is be included. TVC can either be guided by setting the thrust-elevation angle (ϵ_T) or the TVC trim fraction (F_{TVC}).

The guidance described above is used for the trajectory simulation during optimization. For the simulation verification the guidance algorithm used by (Mooij, 1998) is implemented. This architecture was already explained in Section 4.4.1.

Vehicle

The vehicle module calculates and stores a number of vehicle parameters. It is divided into four separate sub-modules that are closely linked together:

- Propulsion
- Mass
- Aerodynamic
- Trim

Each of these sub-modules is discussed below.

The propulsion sub-module takes the thrust guidance parameters as input. These parameters include the equivalence ratio (ϕ) or the throttle factor (F_T) and, if TVC is used as guidance parameter, respectively either the thrust-elevation angle (ϵ_T) or the TVC trim fraction (F_{TVC}). Multi-linear interpolation is used to calculate the thrust magnitude and the specific impulse as function of the equivalence ratio, the dynamic pressure and the Mach number. If throttle control is used an equivalence ratio of 1.0 is taken as a default value after which the resulting thrust magnitude is multiplied with the throttle factor.

With the thrust magnitude and specific impulse known, the mass rate is calculated. Each integration time step the mass is propagated with the calculated mass rate. The center of gravity x_{com} is then calculated using linear interpolation of a table giving x_{com} as a function of the total current mass.

The aerodynamic sub-module sets the angle of attack (α) dictated by the guidance module. Using bi-linear interpolation, the basic vehicle aerodynamic coefficients for drag ($C_{D,f}$), lift ($C_{L,f}$) and pitch ($C_{m,f}$) are calculated as a function of the independent variables α and M. The total pitch moment coefficient C_m should be zero after a trim correction. $C_{m,f}$ is sent to the trim sub-module as the moment coefficient that needs to be counteracted by trim. The trim sub-module returns a set of aerodynamic increment coefficients that are added to the basic vehicle aerodynamic coefficients. The total aerodynamic coefficients are set in the list of current vehicle parameters.

The trim sub-module takes the basic body pitch moment coefficient $C_{m,f}$ and, if TVC is used, the thrust-elevation angle (ϵ_T) or the TVC trim fraction (F_{TVC}) as input. The moment coefficient needs to be counteracted with trim. This trim is obtained with control surface deflections when TVC is not used. If TVC is used, this will produce a certain pitch moment coefficient as well. Only the remaining pitch moment coefficient will now have to be counteracted by use of the control surfaces. Bi-linear

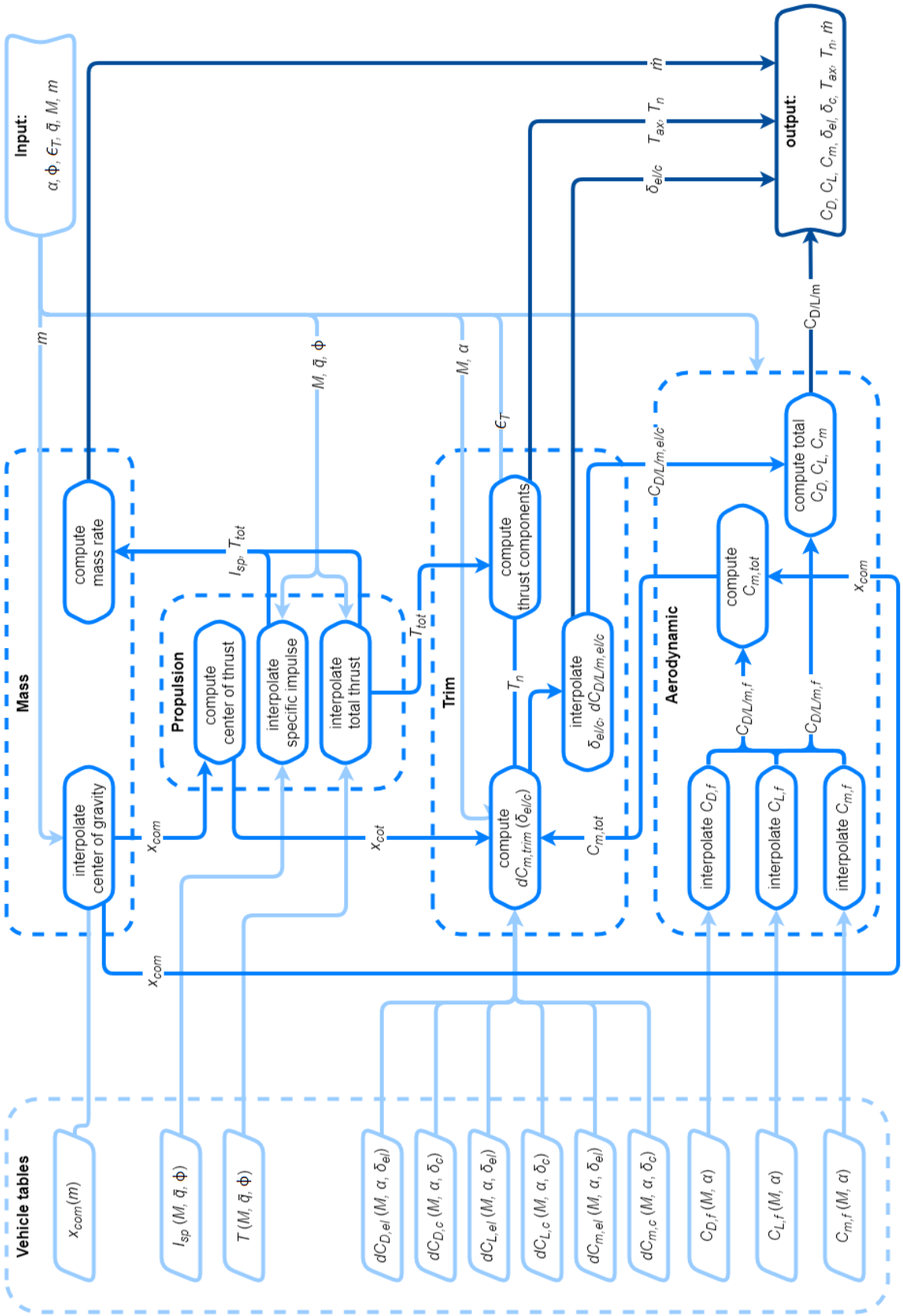


Figure 6.9: Architecture of the vehicle model.

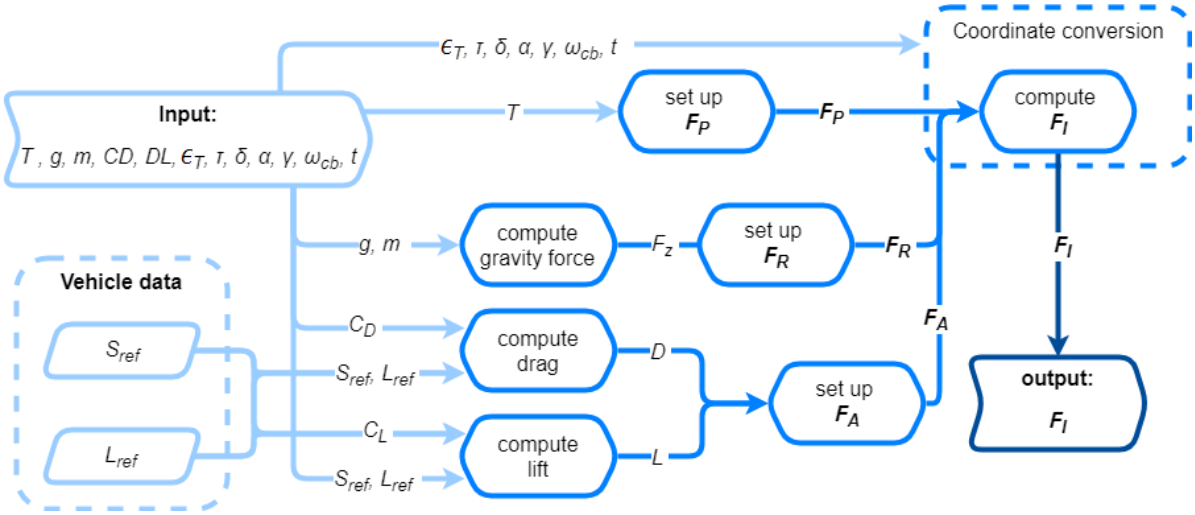


Figure 6.10: Architecture of the external forces model.

interpolation will first be used to calculate the increment pitch moment coefficient as function of angle of attack and Mach number at each deflection angle for which tabulated values are given. Linear interpolation is then applied to find the deflection angle that counteracts the (remaining part of the) body pitch moment coefficient. The resulting aerodynamic increment coefficients are returned to the aerodynamic sub-module. The control surface deflection angles are set in the list of current vehicle parameters.

Figure 6.9 shows a schematic visualization of this vehicle model architecture. This architecture is created for a guidance including α, ϕ and ϵ_T in the decision vector. It is noted that including F_T, F_{TVC} or using the throttle law will result in some minor changes in the architecture.

External Forces

In this model environment and vehicle parameters are used as input to calculated the external forces acting on the space-plane at the current time. The following forces are present:

- thrust expressed in the propulsion frame,
- gravity expressed in the rotating frame,
- drag and lift expressed in the aerodynamic frame.

The thrust is already computed in the propulsion model because it was required for calculation of the TVC trim. The gravity is a function of the local gravitational acceleration and the current mass of the vehicle. For the current mission only drag and lift are present as aerodynamic forces. Their values are computed as a function of the total aerodynamic coefficients, the dynamic pressure $\bar{q}$ and the vehicle reference length L_{ref} and reference area S_{ref}.

The forces are expressed in their respective frames and subsequently transformed to the inertial frame in Cartesian components. The total force vector expressed in the inertial frame is sent to the output. This force vector is used to setup the state derivative that is subsequently used for the integration of the state in the propagation model. The architecture of the external forces model is shown in Figure 6.10.

Propagation

The propagation model takes the external forces, mass rate, heat rate and the current state and time as input. From the input parameters the state derivative is formed. The state and state derivative are given by:

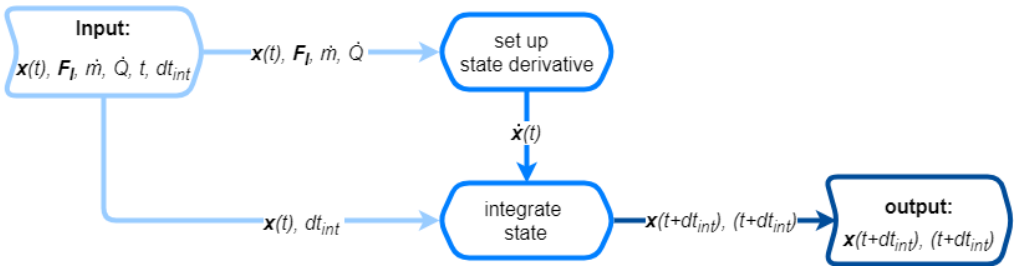


Figure 6.11: Architecture of the propagation model.

$$\boldsymbol{x} = \begin{pmatrix} x_I \\ y_I \\ z_I \\ \dot{x}_I \\ \dot{y}_I \\ \dot{z}_I \\ m \\ Q \end{pmatrix} \quad \text{and} \quad \dot{\boldsymbol{x}} = \frac{d\boldsymbol{x}}{dt}$$

where $\boldsymbol{x}$ is the current state including the position and velocity in the inertial frame, and the mass (m) and heat load (Q). The state derivative $\dot{\boldsymbol{x}}$ includes the velocity and acceleration in the inertial frame and the mass rate ($\dot{m}$) and heat rate ($\dot{Q}$). Integration is then performed resulting in an output of the next state and time. The architecture of the propagation model is shown in Figure 6.11.

Complete simulation

The complete simulation model combines the previously discussed models to perform a complete ascent trajectory simulation from start to finish. At the start of the simulation some choices can be specified for a number of parameters and models to be used. The following parameters need to be set:

- initial state ($\boldsymbol{x_0}$)
- decision vector ($\boldsymbol{x_v}$) (input from optimization)
- initial time (t_0)
- integration time-step (dt_{int})
- guidance update time-step (dt_g)
- use of TVC (yes/no)
- implementation of TVC (ϵ_T/F_{TVC})
- use of throttle law (yes/no)
- implementation of thrust control (ϕ/F_T)
- final time (t_f)
- target altitude (h_f)
- target velocity (V_f)

All these variables are set in the simulation settings. The simulation starts by running the environment model. Given the current known energy state of the vehicle at the current time, the guidance module then computes the current guidance parameters if the current time is a multiple of the guidance update time-step. The vehicle module then computes the propulsion, mass, aerodynamic and trim parameters. Subsequently the accelerations are calculated and the state is propagated to the next time. The state and a number of parameters will be saved in the saved trajectory data. As long as the final time or state is not reached, the simulation model will repeat this cycle. The complete simulation model architecture is given in Figure 6.12.

6.3.2. Optimization model

In this section the general framework of the optimization model will be discussed. The focus lies on the integration of the simulation model within the optimization. For more details on the working of specific optimization methods and approaches, the reader is referred to Chapter 5.

The optimization model can be split in the problem and the algorithm sub-models. Before the optimization can be run, the user has to specify the problem settings. These settings include:

- set optimization parameters (Opt_{param})
- set optimization parameter bounds (Opt_{bounds})
- number of control nodes (n_{cont})
- set objectives (Obj)

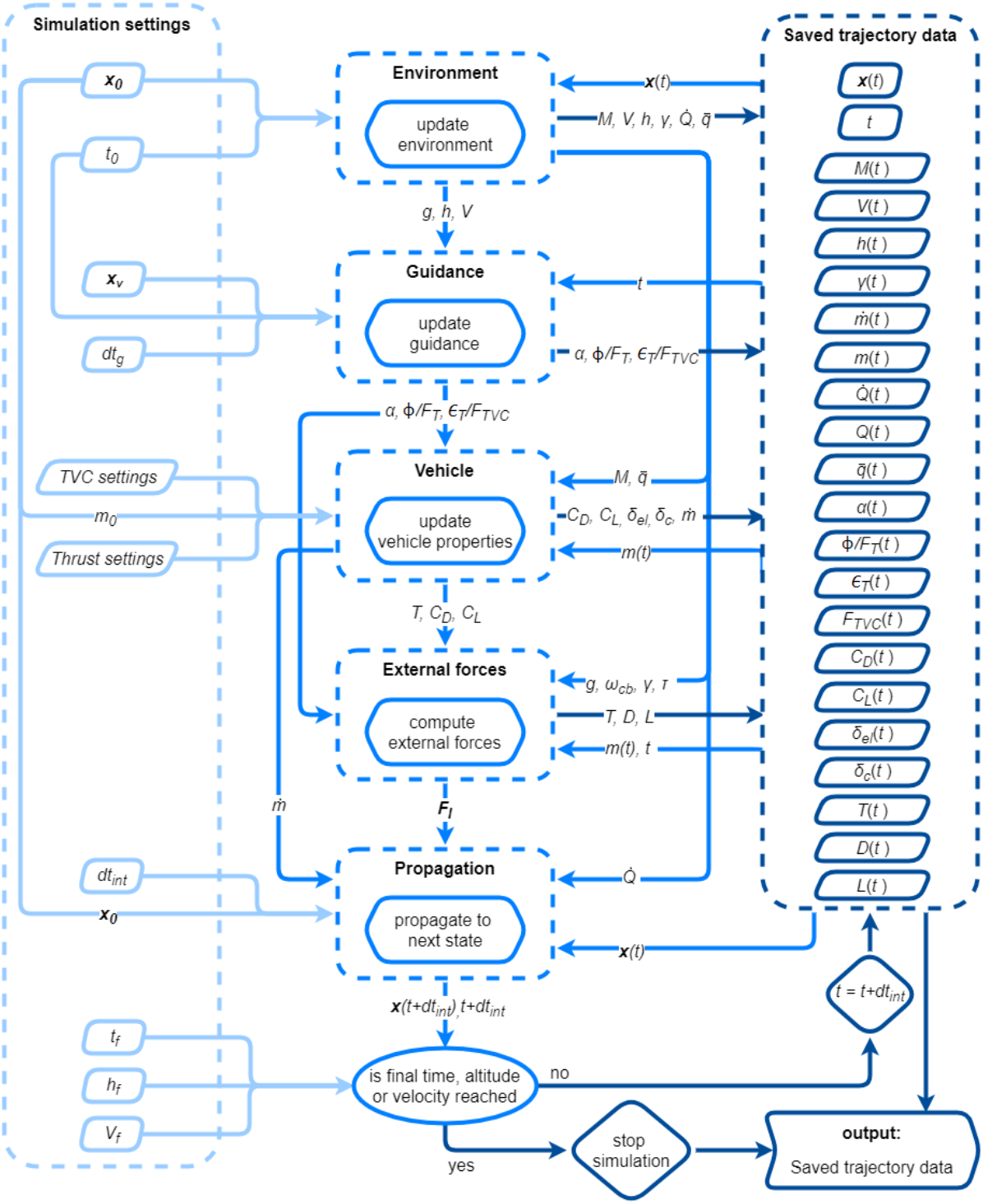


Figure 6.12: Architecture of the complete trajectory simulation model.

The objectives are functions that give a measure of how well a trajectory, simulated with the model discussed in the previous section, performs. A number of objective functions have been discussed in Section 5.3.5.

Subsequently an algorithm is selected. Each algorithm has some specific parameters that need to be set. These settings are explained for each algorithm in Chapter 5. The general algorithm settings that need to be set include:

- set algorithm ($Algo$) and algorithm specific settings
- set population size (n_{pop})
- set maximum number of generations (n_{gen})
- set convergence criterion (C_{conv}) (optional)

When both the problem and the algorithm have been set the optimization starts by initializing a random population. For each individual the simulation is run and the objective functions are evaluated. The selected optimization algorithm then dictates how to evolve to the next generation. The process will then be repeated with this new population. The cycle is ended when the maximum number of generations has been reached or a certain convergence criterion has been met. A schematic overview of the optimization implementation is shown in Figure 6.13.

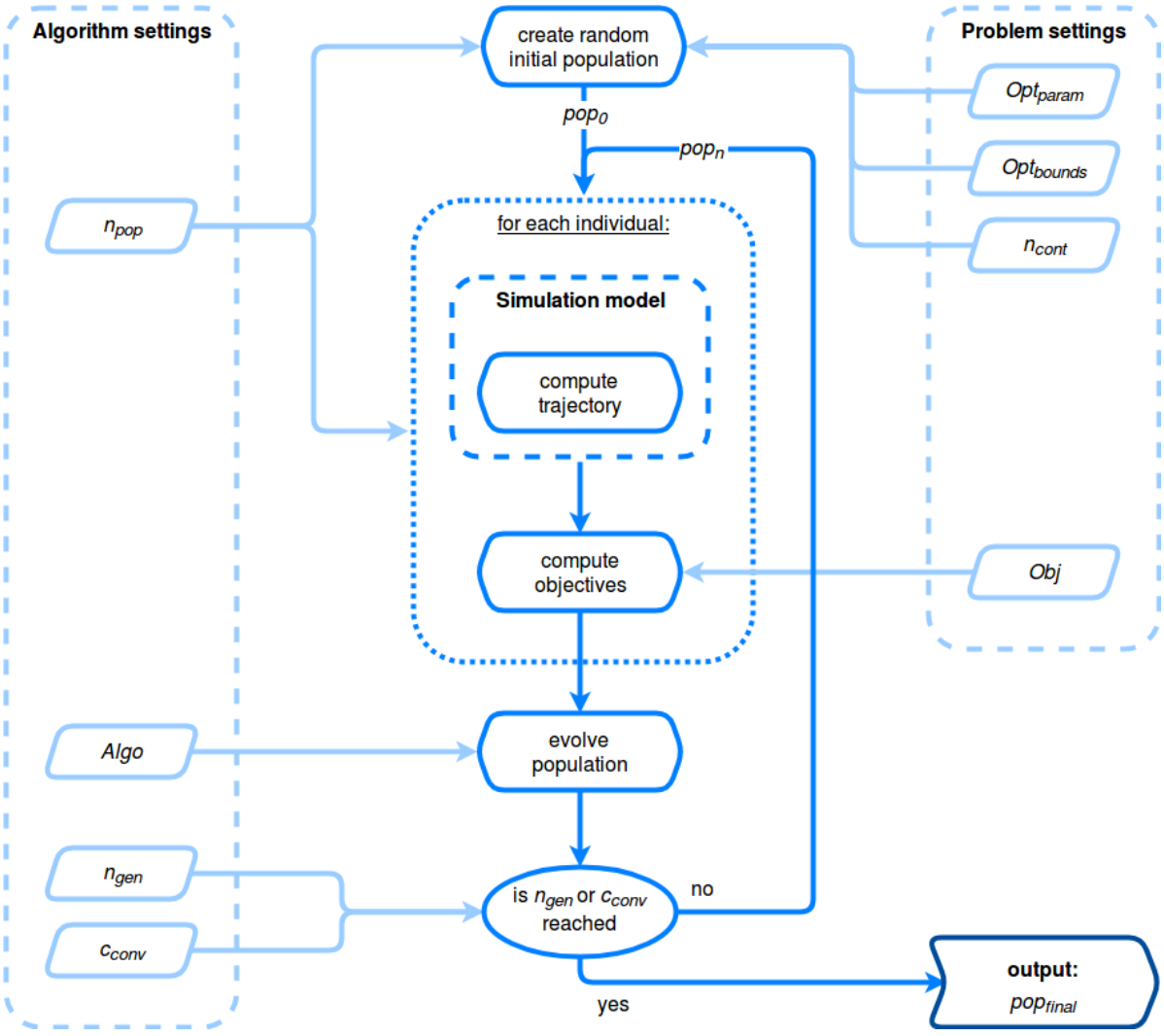


Figure 6.13: Architecture of the complete optimization model.

6.4. Software Verification

This section explains the software verification procedure. Verification is done at two levels. Each sub-module is verified separately first. This is further discussed in Section 6.4.1. Finally the complete simulation model is verified by comparing the results with the reference trajectory obtained by Mooij (1998). The results of this verification are discussed in Section 6.4.2.

6.4.1. Module Verification

All software components are verified individually by conducting unit-tests. A distinction is made between software that is already implemented in either Tudat or PaGMO, and software that will be coded from scratch. All categories will be discussed in this section.

Existing Software

Software that is already implemented in Tudat is assumed to be free of errors. These pre-existing models include all environment models and all mathematical methods. To make sure that no errors have appeared in later updates, the individual models will still need verification. For this purpose, Tudat includes unit-tests for its software components [2]. Each Tudat model will be verified with its respective unit-test.

Vehicle Models

The vehicle models including propulsion, mass, aerodynamics and trim are self implemented based on the vehicle data provided by Shaughnessy et al. (1990). The mathematical basis on which these models rely is explained in Section 3.2. To verify the implementation a manual check of the computation is done for a number of states. The simulation verification will subsequently verify that the mathematical basis on which the implementation relies is correct. This verification is further explained in Section 6.4.2.

Guidance, Objectives and Constraints

The guidance computes the control parameters at the current time with Hermite spline interpolation. The interpolation method is already verified with an existing unit-test as explained above. Only a visual inspection will follow to guarantee the correctness of the output values.

Constraints are given for the flight-path, but also for a number of guidance parameters. The values of the parameters are computed in the previously discussed models and their correctness is already verified. Visual inspection of the trajectory output will make sure that non of the parameters exceeds its constraint value.

The objective functions are checked manually for a number of values to see if the implementation produces the expected result. The verification does not say anything about the performance of the objective functions. The performance of different objectives will be compared as part of the study and results are given in Chapter 7.

Optimization Methods

The optimization methods are already available in PaGMO and are expected to be free of errors. However, adaptations to the standard algorithms were suggested in Section 5.2. Implementing these adaptations could introduce errors in the algorithm. The optimization algorithms therefore need to be verified to guarantee proper working of the code.

PaGMO includes multiple test problems to which the optimizers can be applied. The Zitzler–Deb–Thiele (ZDT) functions are a well known test bed for multi-objective optimization algorithms. Both the NSGA-II method and the MOEA/D method have been tested with these functions (Chase et al., 2009; Qi et al., 2014). The working of the optimizers is therefore verified by applying the algorithms to the ZDT test bed and comparing the optima found with the real optima and the results found by the studies mentioned earlier. All methods converge to the real optimum for all ZDT problems. However, differences can be seen in the convergence speed and the distribution of the individuals in the fronts.

Figure 6.14 shows the results after 50, 100, 300 and 500 generations for ZDT-1. A population size of 24 individuals was used. For NSGA-II-tabu, both the individuals of the evolving population and the tabu population are included in the figure. For the parallel method, also the MOEA/D population was included resulting in a total of 3 populations in the figure for this method. These methods therefore

[2] Tudat (2016). Tu delft astrodynamics toolbox. `http://tudat.tudelft.nl/` [Accessed: 1 June 2017]

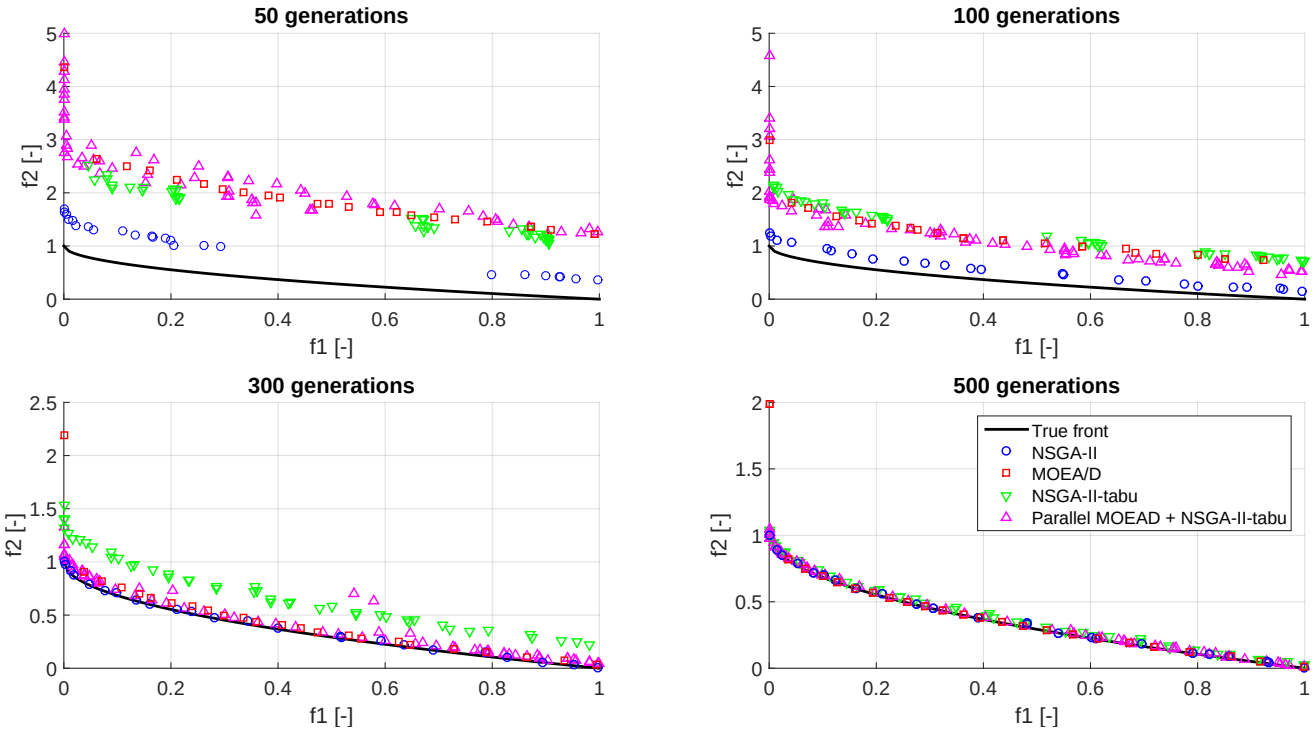


Figure 6.14: PaGMO methods applied to ZDT-1 test problem with n = 24.

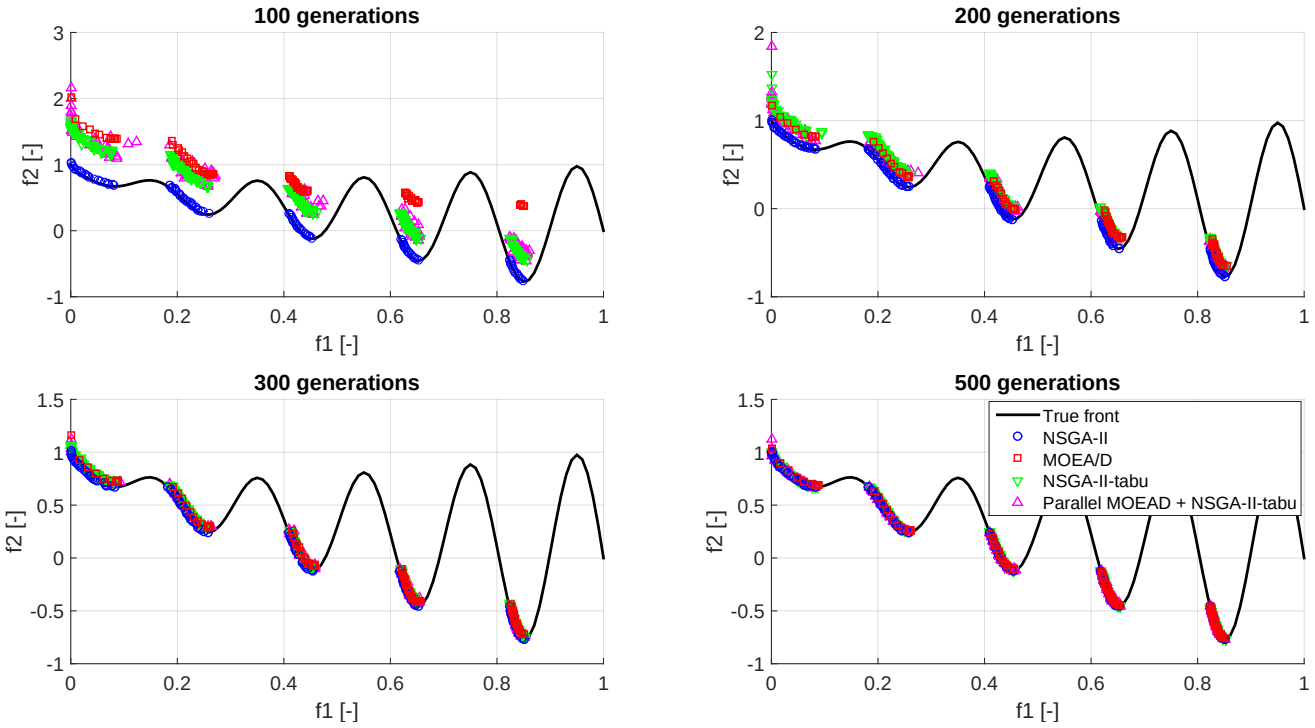


Figure 6.15: PaGMO methods applied to ZDT-1 test problem with n = 100.

show more individuals than the original NSGA-II and MOEA/D methods. The figure shows that NSGA-II converges fastest to the real optimum. At 50 generations both NSGA-II and NSGA-II-tabu show some clustering of the individuals towards the left and the right of the figure, while no individuals are present in the middle section of the front. Throughout the optimization, individuals in the populations are better distributed over the front. At 300 generations the strongly improved distribution over the complete front can be seen for NSGA-II-tabu. Both MOEA/D and the parallel method show better distribution of the population from the start of the optimization as can be seen in the top-left figure. At generation 300 NSGA-II-tabu is found to stay behind in terms of convergence with respect to the other methods. As the best individuals are taken out of the evolving population, it is expected that this method converges slower. After 500 generations, also NSGA-II-tabu has converged completely to the real optimum.

It is noted that with a population size of 100, NSGA-II-tabu shows similar convergence speed to MOEA/D and the parallel method for this problem. Furthermore, although clustering is still found for NSGA-II and NSGA-II-tabu at the start of the optimization, a good distribution of the population is found much faster.

Figure 6.14 shows similar result for the ZDT-3 problem with a population size of 100 individuals. The output is given at 100, 200, 300 and 500 generations. Again NSGA-II is found to converge fastest and is already almost completely converged after 100 generations. NSGA-II is found to converge faster for this problem, with a now comparable convergence speed to MOEA/D. Observations made on the distribution of individuals for ZDT-1 are not applicable for ZDT-3. Both NSGA-II and NSGA-II-tabu show no signs of clustering at the start. The difference could be attributed to the larger population size that is used and the fact this is a different problem. However, even with the larger population size, MOEA/D shows a worse distribution for this problem than it did for ZDT-1. Where all the other methods have almost perfectly distributed their individuals over the different sections of the front, MOEA/D only has very few individuals on the left part of the front, while many individuals are found on the right. This can be best observed at generations 300 and 500 in the bottom figures.

The results summarized above show that all implemented methods are working as they are expected to. It also gives an indication that convergence speed and distribution of the population over the front are problem specific. This reinforces the objective to compare different optimization methods for the ascent trajectory optimization problem. The results of the performance comparison of the different methods are given in Section 7.5.

6.4.2. Simulation Verification

The complete simulation model is verified with respect to the reference trajectory obtained by Mooij (1998). This trajectory and its constraints were already introduced in Section 2.2.2. The trajectory was obtained following the guidance that was described in Section 4.4.1. For the verification run, the same environment, vehicle and guidance models were used and also the same guidance was applied. Only two discrepancies are known in the implementation of the throttle law:

- The constraint error is only integrated for the active constraint.
- Constant gains are applied for the axial acceleration

Both changes were explained in Section 4.4.1. A comparison of the reference and verification trajectory results is given in Figure 6.16. Both trajectories overlap perfectly. However, the verification run only reaches an altitude of 86 km instead of the objective altitude of 120 km.

To find out why the final altitude is not reached, the resulting time-histories of a number of flight parameters are compared in Figures 6.17 to 6.24. The flight begins with a large angle of attack of 12° to reach a flight-path angle of 25°. After this steep take-off section the flight levels out.

Figure 6.21 shows that the dynamic-pressure constraint is reached quickly as the vehicle accelerates. The engine is throttled back to not violate the constraint, as can be seen in Figure 6.20. Around 900 seconds into the flight a dip in dynamic-pressure constraint allows for a short moment of full thrust. 100 seconds later the dynamic-pressure and heat-rate constraint intersect. The vehicle now follows the heat-rate constraint as can be seen in Figure 6.22.

When a velocity of 7,000 m/s is reached, the pull-up manoeuvre is initiated. The angle of attack is set to 6°. Figures 6.23 and 6.24 show that both lift and drag increase as a result. To make sure that the extra drag does not slow down the vehicle too much, extra thrust is given for this section.

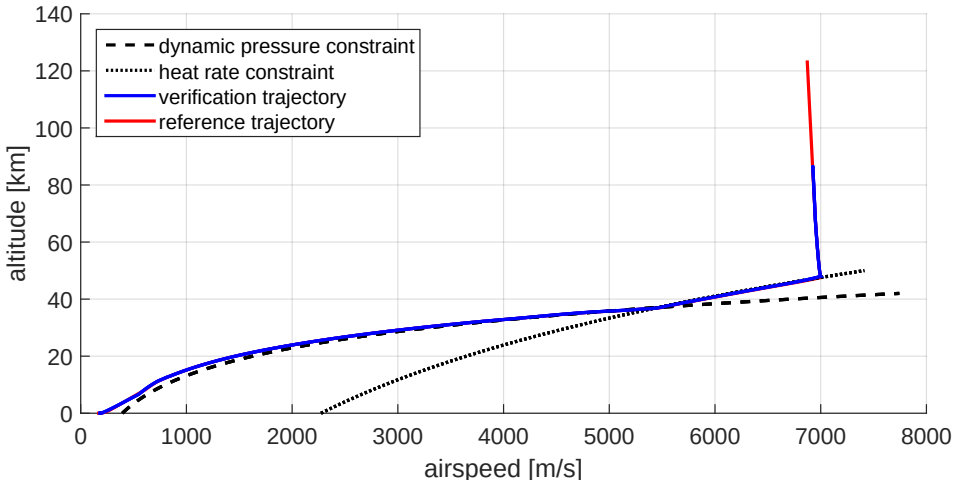


Figure 6.16: Comparison of the reference trajectory found by Mooij (1998) and the verification trajectory obtained with the developed software.

The fligh-path angle increases as a result of the extra generated lift as is visible on the right side of Figure 6.18. When a dynamic pressure drops below 2,500 N/m^2 the engine is turned off and the angle of attack and deflection angles are set to zero. During this coasting phase the flight-path angle slowly decreases and the simulation is ended when it finally drops below zero.

Figures 6.25 and 6.26 show the thrust-elevation angle and the elevon-deflection angle of the reference trajectory when 50% of the trim is trim is produced with TVC. The same results are given for the verification in Figures 6.27 and 6.28. When TVC is included, the elevons can be deflected less, which reduces drag. A good comparison is found between the reference and the verification run. In the middle section of the flight, the verification seems to require a slightly lower thrust-elevation angle. This can be explained when comparing with the throttle factor results in Figure 6.20. At the dip around 1,100 seconds into the flight, the reference produces less thrust. As a result, a larger thrust-elevation angle is required to produce the same pitch moment with TVC.

The main difference between the reference and verification for all figures is the shift in time. Although a very similar pattern is found, the verification takes an extra 120 seconds to reach the pull-up velocity. The difference could be caused by the discrepancies in the throttle law mentioned before. Figure 6.21 shows that the reference slightly overshoots the dynamic-pressure constraint, while the verification moves towards the constraint value more slowly. As a result, the reference can give more thrust during this initial part of the trajectory as is visible in Figure 6.20. The reference trajectory accelerates faster at the start of the trajectory, which could be a reason why the pull-up velocity is reached earlier.

Figure 6.18 shows why the final altitude of 120 km is not reached. During the pull-up, the verification run does not reach the same flight-path angle as the reference. Furthermore, the flight-path angle decreases much faster during the coasting phase. Even though the pull-up manoeuvre of the verification is initiated 120 seconds later, flight-path angle drops back to zero earlier.

This last observation indicates a significant discrepancy between the simulation model and the reference. The differences between the reference and the verification throughout the flight discussed before, result in some differences at the initial state of the pull-up. The vehicle is 1500 kg lighter and starts the pull-up around 400 meters higher. To assess the effect of these changes, the pull-up is simulated separately with the initial pull-up conditions of the reference trajectory:

$$V_0 = 7000 \text{ m/s}, \qquad h_0 = 47,542 \text{ m}, \qquad m_0 = 71,174 \text{ kg}, \qquad \gamma_0 = 0.12^\circ$$

The resulting final velocity, altitude, flight-path angle and duration of the flight-phase are given on the first line of Table 6.1. These results match with the result of the pull-up for the complete verification trajectory discussed earlier. Discrepancies in the pull-up phase can therefore not be attributed to slightly different guidance throughout the take-off and acceleration phases.

changed parameters	V_f [m/s]	h_f [m]	γ_{max} [deg]	duration [s]
-	6,925	86,566	2.46	219
α = 10°	6,810	107,954	3.64	247
F_T = 2.0	6,986	90,669	2.55	235
m_0 = 65,000 kg	6,919	90,170	2.64	225
V_0 = 7,200 m/s	7,075	96,205	2.61	259
α = 10°, F_T = 2.0, V_0 = 7,200 m/s	7,005	120,012	3.84	220

Table 6.1: Final state results for a number of changed parameter values during the pull-up

It remains unclear what does cause the discrepancy. Time-histories of all flight parameters that influence thrust, drag, lift and gravitational forces on the vehicle have been compared. Each of the parameters stays within a few percent of the reference results. The sensitivity of each of the parameters has been evaluated, which indicated that none of these parameters directly cause the fast drop in the flight-path angle.

To see if the current simulation model can reach final state of the reference, the effect of the initial pull-up conditions is further investigated. Table 6.1 first shows results using the reference conditions. Subsequently the effect of changing the pull-up angle of attack, throttle factor, mass and velocity is evaluated. Increasing the angle of attack results in a higher maximum flight-path angle. It will therefore take longer for the flight-path angle to drop back to zero and the vehicle therefore gains more altitude. Increasing the pull-up velocity will not lead to a much higher flight-path angle , but the vehicle will gain more altitude at a similar angle as it is moving faster. Increasing by 200 m/s has led to a gain of almost 10 km in the final altitude. The throttle factor and initial mass have smaller effects on the resulting altitude.

All changes have led to an increased final altitude. However, decreasing the mass is not a convenient method as the goal of the optimization is to take as much mass (in the form of payload) as possible. This method is therefore not taken as a viable option. The other three changes are combined for which results are shown in the last row of Table 6.1. This combination of methods resulted in a pull-up that reached the reference altitude.

A final complete trajectory verification run was performed with the changed pull-up conditions. Because the vehicle accelerates further before the pull-up and more thrust is given during pull-up, more fuel is used during the ascent. However, the final velocity is 173 m/s higher than found by the reference. This saves more than 3,000 kg of fuel during the final circularization boost. The verification found a total payload mass capacity of 2,424 kg compared to 3,412 kg found by Mooij (1998) for the reference trajectory.

The current model is used for throughout this research, even though the exact cause of the discrepancy between the verification and reference simulation models has not been found. This is allowed, because the main goal of this research is to study the effect of different trajectory constraints, objectives and parameter settings on the optimization process. This section has shown that the same constraints are followed in similar order. Furthermore, all control parameters were found to affect the trajectory in the same way as for the reference. Resulting effects found for the optimization are therefore also applicable to the reference model.

The current simulation does result in a higher fuel consumption and therefore a lower payload mass. It will therefore be harder for the optimizer to find a solution with this simulation model, as will be discussed Section 7.7.3. However, as was mentioned already in Section 3.2.2, the WCC is a very optimistic model. (Mooij, 1998) and Powell et al. (1991) found high payload masses quite easily. The current simulation results in a more marginal payload mass, which might actually be more realistic when comparing with real space-plane projects.

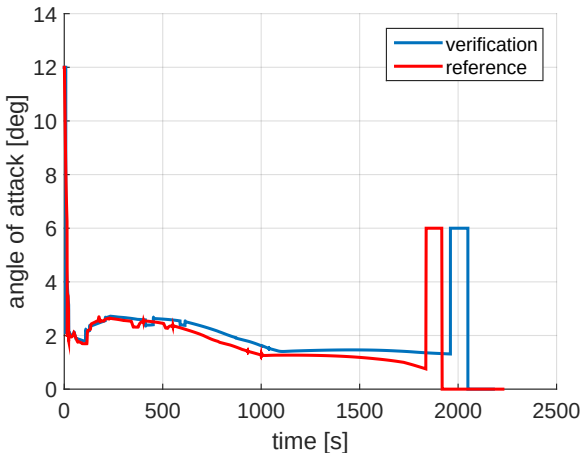


Figure 6.17: Comparison of angle of attack versus time for reference and verification trajectories.

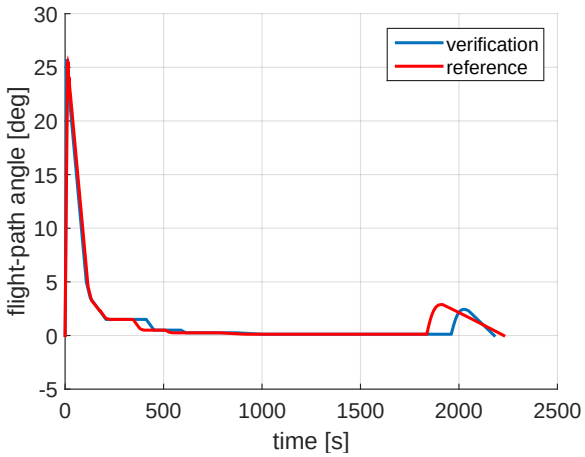


Figure 6.18: Comparison of flight-path angle versus time for reference and verification trajectories.

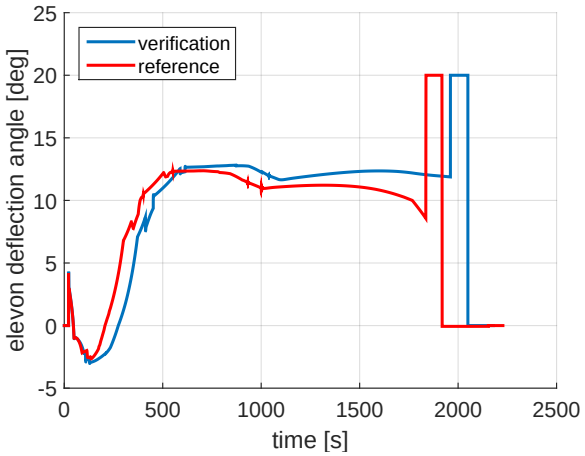


Figure 6.19: Comparison of elevon deflection versus time for reference and verification trajectories.

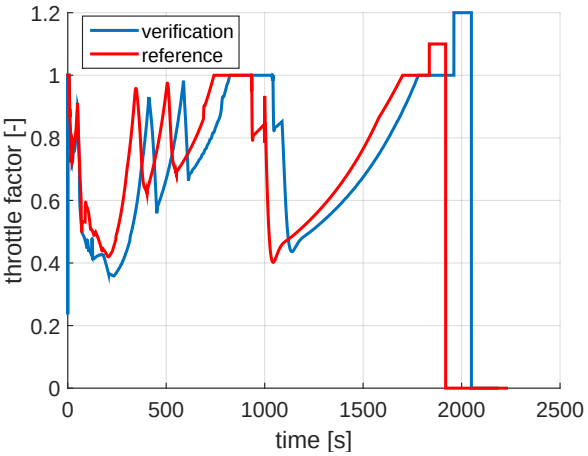


Figure 6.20: Comparison of throttle factor versus time for reference and verification trajectories.

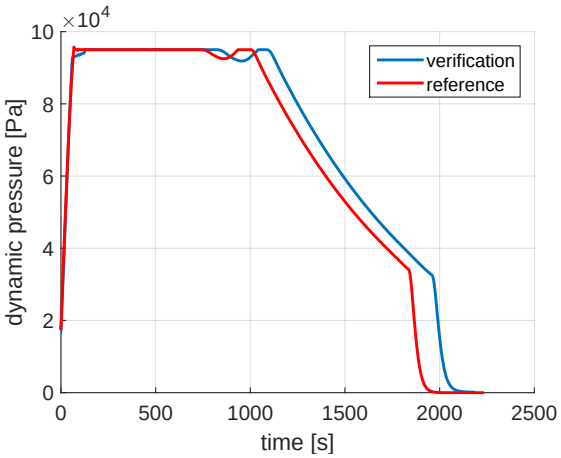


Figure 6.21: Comparison of dynamic pressure versus time for reference and verification trajectories.

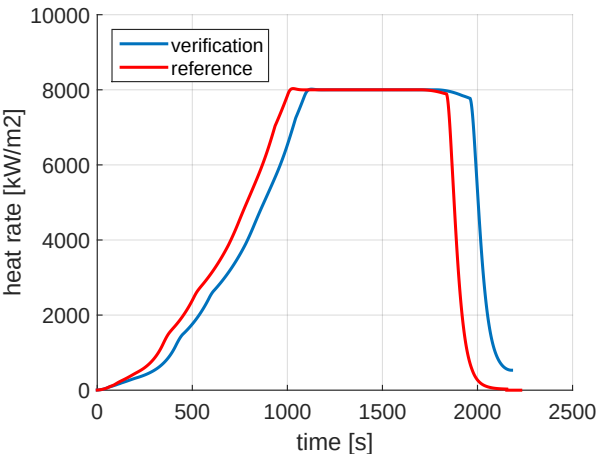


Figure 6.22: Comparison of heat rate versus time for reference and verification trajectories.

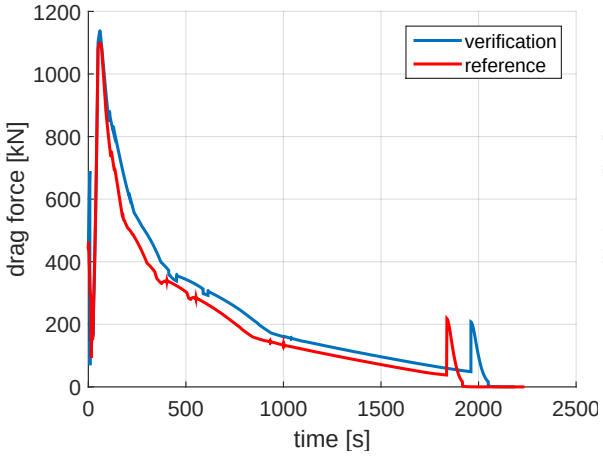


Figure 6.23: Comparison of drag versus time for reference and verification trajectories.

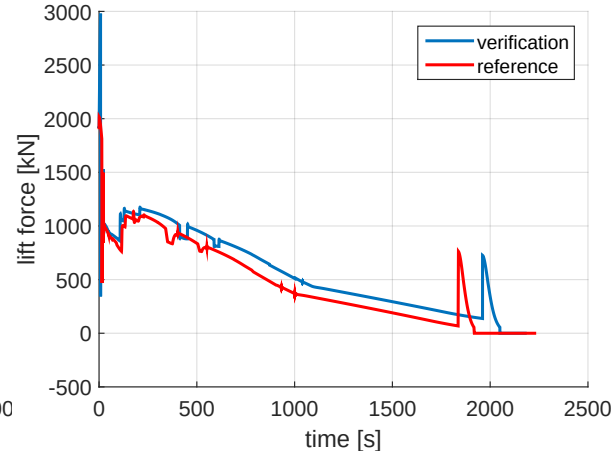


Figure 6.24: Comparison of lift versus time for reference and verification trajectories.

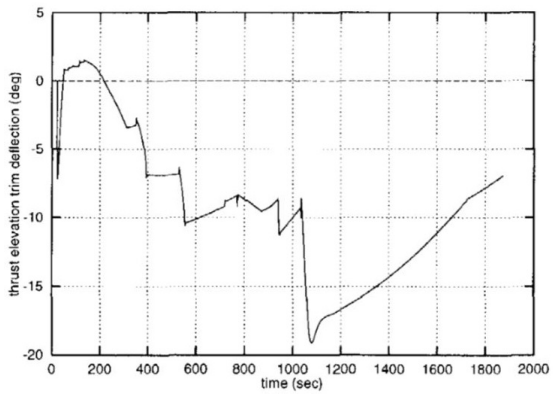


Figure 6.25: Thrust-elevation angle versus time for reference trajectory with 50% TVC (solid) and 0% TVC (dashed) Mooij (1998).

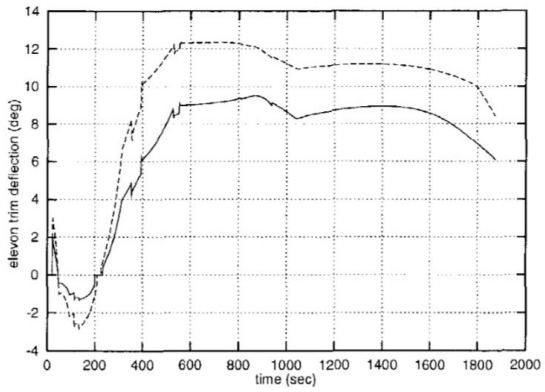


Figure 6.26: Elevon deflection versus time for reference trajectory with 50% TVC (solid) and 0% TVC (dashed) Mooij (1998).

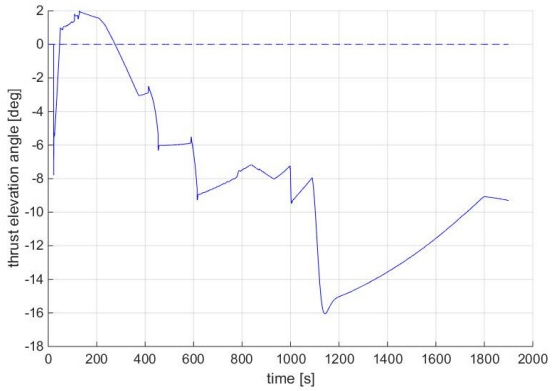


Figure 6.27: Thrust-elevation angle versus time obtained from verification simulation with 50% TVC (solid) and 0% TVC (dashed).

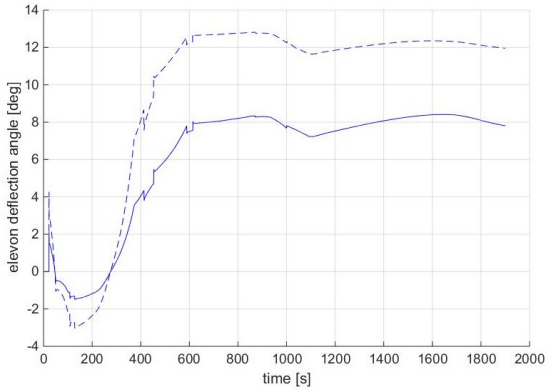


Figure 6.28: Elevon deflection versus time obtained from verification simulation with 50% TVC (solid) and 0% TVC (dashed).

7

Results

This chapter includes an extensive discussion of the results. In Section 7.1 the general methodology used will be given first. The subsequent sections will discuss all found results in more or less chronological order. Finally a sensitivity analysis will be performed in Section 7.8.

7.1. Methodology

This section will shortly discuss the general method used to obtain the results shown in the following sections. First an overview is given of steps taken to obtain all results, after which the general settings for the standard trajectory optimization problem are given. Finally the objectives for evaluation of the results will be discussed.

Research Approach

The current research revolves around the following main question:

> *How can the payload capacity of an HTHL SSTO launch vehicle to ISS be maximized given the flight-path and control constraints?*

To find the answer to this question the problem is divided in three sub-questions:

SQ-1 What is the impact of different control parameters on the optimization process of the ascent trajectory?

SQ-2 Which MO global optimization method is most effective for the ascent trajectory optimization problem?

SQ-3 What is the best approach, in terms of handling constraints, objectives and different flight phases, to optimize the ascent trajectory of the WCC from take-off to circularized orbit?

The answer to each of these questions very much depends on the answer to the other questions. The impact of certain control parameters, for example, might be different for various optimization methods. At the same time, the performance of the optimizers will most likely be dependent on the general optimization approach that is taken. To accommodate these dependencies as much as possible the following step by step approach is taken to obtain the required results:

1: Compare the different sets of constraint objective functions and find which set is best able to find a trajectory that does not violate any flight-path constraints.

2: Compare runs with different control parameters (equivalence ratio, throttle factor, throttle law, TVC). Find the impact of the control parameters on the ability of the optimizer to find an ascent trajectory without constraint violations.

3: Compare different numbers of control nodes (6, 8, 10, 12) for the ascent problem with throttle law, the standard problem and the standard problem with TVC, where the standard problem includes ϕ as the thrust magnitude control parameter for optimization. These three problems include 2, 3 and 4 control parameters for optimization, respectively. An overview of the standard problem settings will follow shortly.

4: Compare the four optimization methods for a number of promising settings found previously in steps 1 to 3.
5: Compare different approaches to handle the combined optimization of constraints and objectives (approach 1 to 4 described in Section 5.3.6).
6: Use the best found optimization approach, method and combination of controls to optimize the ascent mission defined in Section 2.4. Study the difference in performance of optimizing the complete trajectory at once or dividing the trajectory into different phases (take-off, atmospheric acceleration, pull-up).

Steps 1 through 5 are all performed on the atmospheric acceleration phase starting at an altitude of 350 m with a velocity of 220 m/s, and ascending to 48 km at a velocity of 7 km/s. The results of steps 1 to 6 are discussed in Sections 7.2 to 7.7 respectively

Standard Optimization Settings
To be able to compare the obtained results a standard set of settings is used for each optimization run. Unless otherwise specified the following settings have been used:

- optimizer = MOEA/D
- population size = 100
- number of generations = 500
- number of free/total control nodes = 7 / 8
- guidance update time-step = 0.2 s
- integration step-size = 0.2 s
- initial/final altitude = 350 / 48000 m
- initial/final velocity = 220 / 7000 m/s
- initial/boundary angle of attack = 5.0 / [0.0, 3.5] deg
- initial/boundary equivalence ratio = 1.0 / [0.4, 1.0]
- initial random seeds 1 / 2 / 3 = 3176319168 / 3747925739 / 1501949093

where the default MOEA/D method is selected as the standard optimizer, because initial test runs showed the best performance for a simple ascent trajectory optimization.

Because heuristic optimization methods are used, the results will be dependent on the initial random seed. Starting with a different initial seed could potentially lead to very different results. Each optimization is therefore repeated with 3 different initial seeds given in the settings list above. A larger number of repetitions is preferable, but the runs are time consuming and the time-constraint to this research does not allow for more repetitions. However, running each optimization 3 times should at least give an indication of the consistency of the results and the sensitivity to the initial seed.

Evaluation Objectives
The results will be compared based on the final objective function values. For a good comparison, each result is evaluated with respect to the same set of objective functions, even though different objectives might be used for the actual optimization. The used evaluation objectives are given in Table 7.1. For a detailed explanation of the objective functions the reader is referred to Section 5.3.2. A note on the interpretation and sensitivity of the resulting function values is given next.

number	name	composition
f[1]	energy-state objective	$c_E + p_{E_{mon}}$
f[2]	constraint objective	$p_{\bar{q}} + p_{\dot{Q}} + p_{n_a} + p_{nf}$
f[3]	flight-path angle objective	p_γ
f[4]	fuel-mass objective	$c_{fm} + p_{fm}$
f[5]	heat-load objective	$c_Q + c_E$

Table 7.1: Objective functions used for evaluation and comparison of the results.

Objective Interpretation and Sensitivity

In general, the lower the function value, the better the performance. For the feasibility objectives f[1], f[2] and f[3] the final values should approach zero. The optimality objectives f[4] and f[5] will never reach zero, but lower values are also considered better.

The energy-state objective value f[1] is very sensitive to a change in velocity, but less sensitive to a change in altitude. For a final state of $h = 48$ km and $V = 7000$ m/s an offset of 1 m or 1 m/s would result in an approximate function value of $4{\cdot}10^{-7}$ and $3{\cdot}10^{-4}$, respectively. An energy-state objective function value of $5{\cdot}10^{-5}$ should therefore give a final state with an altitude and velocity offset in the order of 100 m and 0.1 m/s respectively. It is noted that f[1] can also become negative if either the final altitude or velocity exceeds the objective value. Because the objective altitude and velocity are also set as a stop condition for the simulation, a large negative value will never appear.

The constraint objective function f[2] is equally sensitive to each constraint violation. A function value of 0.01 indicates a maximum constraint violation of 1%, or a combination of smaller violations. A trajectory without flight-path constraint violations should have a constraint objective function value of zero.

The flight-path oscillation objective f[3] should also be zero for a trajectory with a monotonically decreasing flight-path angle. Generally values below 0.01 tend to give relatively smooth trajectories, whereas values above 0.2 show significant oscillations. The results discussed in Section 7.2 will confirm this.

The fuel-mass objective f[4] is zero if no fuel is used. Based on the maximum available space for fuel on board, a value of 0.5666 indicates that all fuel possibly available has been used. The objective values should therefore stay below this value. To reach the altitude of the ISS at orbital speed, part of the fuel should be saved for the pull-up phase. Based on initial tests a maximum fuel-mass objective value of around 0.47 is estimated to be available for the atmospheric acceleration phase.

The heat-load objective f[5] generally ranges around the 0.5. A value of 1 indicates that the average heat rate during the trajectory is equal to the heat-rate constraint. Although a lower value of f[5] is preferred, a higher value is not problematic as long as the heat-rate constraint is not violated.

7.2. Constraint Handling

A number of different feasibility objective functions were defined in Section 5.3.5. The performance of these sets of feasibility objectives are compared in this section. The constraint handling cases compared are defined in Table 7.2:

Each case is run with three different initial seeds. Table 7.3 gives the resulting objective function values for one competitive individual of each run. The different constraint handling cases are compared based on this table.

Case CH2 is the same as CH1, but with the no-flight penalty included in the constraint objective. Similarly, case CH4 is equal to CH3, but with the addition of the no-flight penalty. Comparing case CH2 with CH1, and case CH4 with CH3 gives insight of the effect of the no-flight penalty. The function value of f[2] has decreased considerably for run CH2.2 with respect to CH1.2. Also for cases CH4.1 and CH4.3, f[2] is significantly lower than for cases CH3.1 and CH3.3, respectively. This indicates the effectiveness of the no-flight parameter.

Figure 7.1 gives better insight in how the no-flight penalty works. On the left side, populations are given for some initial generations of an optimization without the no-flight penalty included in the

case	generations	obj-1	obj-2	obj-3
CH1	0 to 500	$c_E + p_{E_m on}$	$p_{\bar{q}} + p_{\dot{Q}} + p_{n_a}$	-
CH2	0 to 500	$c_E + p_{E_m on}$	$p_{\bar{q}} + p_{\dot{Q}} + p_{n_a} + p_{nf}$	-
CH3	0 to 500	$c_E + p_{E_m on}$	$p_{\bar{q}} + p_{\dot{Q}} + p_{n_a} + p_{\gamma}$	-
CH4	0 to 500	$c_E + p_{E_m on}$	$p_{\bar{q}} + p_{\dot{Q}} + p_{n_a} + p_{nf} + p_{\gamma}$	-
CH5	0 to 500	$c_E + p_{E_m on}$	$p_{\bar{q}} + p_{\dot{Q}} + p_{n_a} + p_{nf}$	p_{γ}
CH6	0 to 300	$c_E + p_{E_m on}$	$p_{\bar{q}} + p_{\dot{Q}} + p_{n_a} + p_{nf}$	-
	301 to 500	$c_E + p_{E_m on}$	$p_{\bar{q}} + p_{\dot{Q}} + p_{n_a} + p_{nf} + p_{\gamma}$	-

Table 7.2: Definition of constraint handling cases.

Case	Seed	f[1]	f[2]	f[3]	CPU time [s]
CH1	1	0.00962	0.24044	0.01814	74,771
	2	0.00898	0.29404	0.99075	65,853
	3	-0.00004	0.00000	0.10303	80,259
CH2	1	-0.00013	0.24264	0.02992	92,389
	2	-0.00009	0.01701	0.28894	90,140
	3	-0.00009	0.01423	0.16863	86,981
CH3	1	-0.00006	0.97648	0.01250	46,166
	2	-0.00011	0.23850	0.00374	66,728
	3	-0.00002	1.15069	0.00365	42,328
CH4	1	-0.00010	0.38312	0.00410	96,371
	2	-0.00010	0.27429	0.00306	85,527
	3	-0.00006	0.28806	0.00298	93,334
CH5	1	0.00376	0.42508	0.03700	92,851
	2	0.01611	0.33205	0.05811	91,780
	3	0.00007	0.25648	0.00405	91,425
CH6	1	-0.00012	0.24269	0.00314	90,318
	2	-0.00006	0.01625	0.00392	94,748
	3	-0.00010	0.01632	0.00387	87,417

Table 7.3: Optimization results for different sets of constraint objective functions applied to the optimization. (n=100, gen=500)

constraint objective. A no-flight solution is indicated by an energy-state objective value close to 1, in which case the vehicle is neither accelerating nor ascending. During the first 10 generations almost the whole population converges towards this no-flight condition. Only two individuals have found an energy-state objective function value lower than 0.9. The optimization process relies on these individuals for further improvement. After 50 generations, the population starts moving towards better energy-state solutions, but almost half of the individuals are still no-flight solutions. These individuals are of little help for the optimizer and are effectively slowing the optimization process down. Besides the slower optimization process, a risk for premature convergence also exists. If none of the individuals is able to escape the convergence towards no-flight solutions during the first few generations, the optimizer might get trapped in the local no-flight optimum.

On the right side of Figure 7.1, populations are shown for the initial generations of a run that does include the no-flight penalty. Initially the constraint objective values are higher, because of the given penalty. However, after 10 generations, the whole population has moved towards better constraint and energy-state objectives. None of the individuals has converged to a no-flight solution. The result

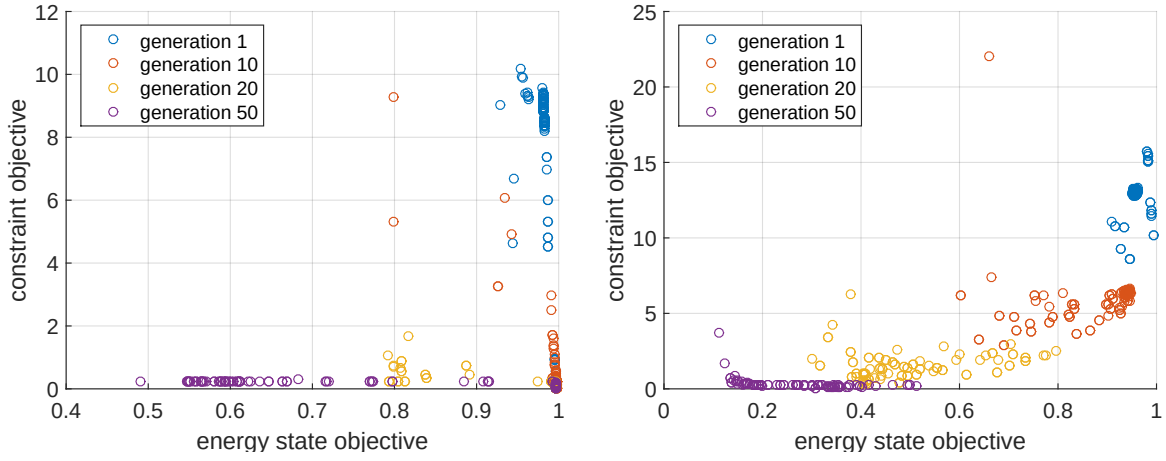


Figure 7.1: Comparison of initial convergence of the population with (right figure) and without (left figure) no-flight penalty.

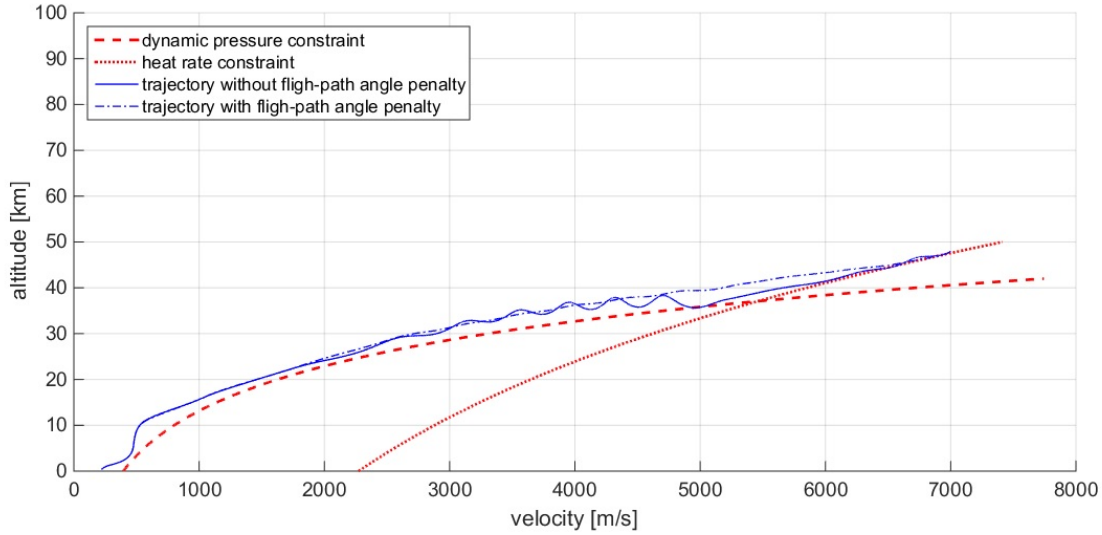


Figure 7.2: Comparison of trajectory result with and without flight-path angle oscillation penalty.

is a much better diversity of the population, where all the individuals contribute to the optimization process in terms of exploration and convergence. This already results in a much better converged and distributed population after 50 generations. The improved diversity also protects the population from getting trapped in a local optimum, which increases the likelihood of finding a good solution.

Finally it is noted that CH1 and CH3 run much faster than the other cases. Although this seems positive, it is actually a direct result of the worse performance. The large part of the population that converges to a no-flight solution is computationally inexpensive, because their trajectories are very short. For these solutions the program only has to simulate a few seconds as opposed to around 2000 seconds of flight time for a good trajectory. The shorter computation time is therefore in this case not a positive sign.

When comparing the results of CH3, CH4 and CH5 with the results of CH1 and CH2, the effect of the flight-path angle oscillation penalty can be seen. The function value f[3] has dropped considerably for almost all of the runs. However, this has come at the cost of more constraint violations indicated by the increased function value f[2].

This is, of course, not the desired behavior. CH6 has therefore been included in the comparison. This test case will first optimize according to CH2 for 300 generations. From generation 301 to 500 the flight-path angle oscillation penalty is then included and the optimization process will now follow CH4.

The results for this case combine the better function values for f[2], obtained by CH2, with the better function values for f[3], obtained by CH4. The trajectory plots of runs CH2.2 and CH6.2 are shown in an altitude-velocity diagram in Figure 7.2. This comparison shows that although both trajectories reach the objective altitude and velocity without violating the constraints, the inclusion of the flight-path angle oscillation penalty results in a much smoother trajectory.

The oscillations found in the resulting trajectories show a periodic exchange between potential and kinetic energy, also known as a phugoid motion. This oscillation was expected to be a characteristic motion of the trajectory. Appendix A explains how the characteristic motion of the ascent trajectory can be found, and summarizes the results.

The phugoid motion was found to be the predominating eigenmotion, but the computed frequencies do not match those seen in the trajectory optimization results. Figure 7.3 shows the flight-path angle versus time for the trajectory found in CH2.2. The oscillation has a period of around 200 seconds and occurs at velocities between 2,000 and 5,000 m/s. From the results computed in Appendix A a period of approximately 1,000 seconds is found at velocities of 2,000 m/s, whereas this period increases to around 2,200 seconds at a velocity of 4,000 m/s.

The computed period of the characteristic phugoid motion is a factor 5 to 10 higher than the period found in the optimization results. Although this discrepancy seems to be too large to attribute to

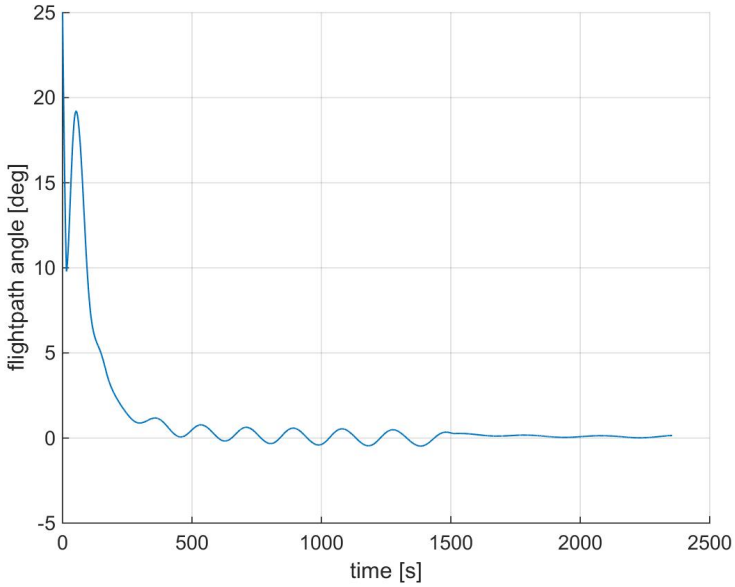


Figure 7.3: Flight-path angle versus time for trajectory with strong phugoid motion.

the linearization, no other causes have been identified, and the characteristic motion is therefore still thought to be the most likely explanation for the oscillation. Further research in the vehicle guidance and its response should be performed to confirm this expectation.

Finally, it should be noted that reducing the flight-path angle oscillation generally did not result in better fuel-mass and heat-load objective values. Although a trajectory without this strong phugoid motion is expected to perform better, the currently implemented guidance is not able to effectively remove the oscillations. The flight-path oscillation objective will therefore not be used when optimizing for the final optimality objectives.

7.3. Control Parameters

Different control parameters can be included in the decision vector for optimization, as was explained in Section 4.4.2. The standard problem uses the angle of attack α and the equivalence ratio ϕ. However, ϕ can be replaced by the throttle factor F_T. The thrust can also be controlled with the throttle law explained in Section 4.4, meaning either ϕ or F_T will still be used, but not as optimization parameter. Furthermore, the thrust-elevation angle ϵ_T or the thrust-vector trim fraction F_{TVC} can be added as optimization parameter. An overview of the studied throttle and TVC settings can be found in Table 7.4.

The different throttle settings are tested for constraint handling case CH2 defined in Table 7.2. A comparison of the throttle setting results is shown in Figure 7.4. The figure is zoomed in to the lower

case	throttle law	throttle parameter		TVC parameter	
Thr-S1	no	ϕ	[0.4 1.0]	-	
Thr-S2	no	F_T	[0.4 1.0]	-	
Thr-S3	yes	ϕ	[0.4 1.0]	-	
Thr-S4	yes	F_T	[0.4 1.0]	-	
TCV-S1	yes	ϕ	[0.4 1.0]	F_{TVC}	[0.0 0.5]
TVC-S2	yes	ϕ	[0.4 1.0]	F_{TVC}	= 0.5
TVC-S3	yes	ϕ	[0.4 1.0]	ϵ_T	[-25.0 5.0]
TVC-S4	no	ϕ	[0.4 1.0]	ϵ_T	[-25.0 5.0]

Table 7.4: Definition of throttle and TVC settings with the domain of each parameter.

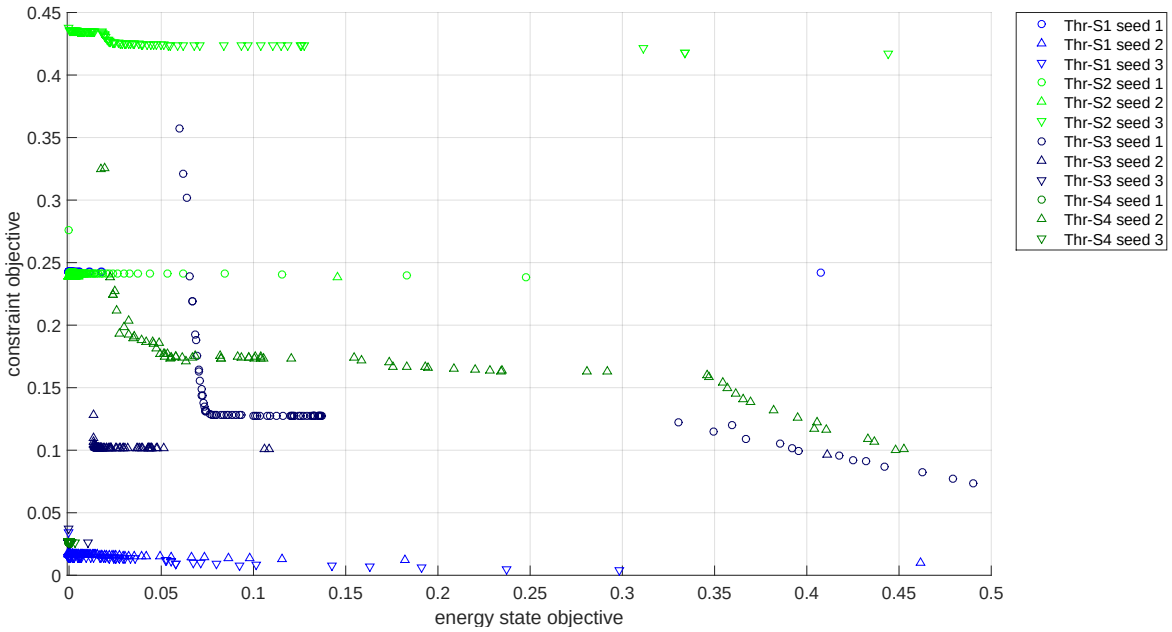


Figure 7.4: Comparison of throttle setting results (n = 100, gen = 500).

objective values to see more detail. As a result some individuals of Thr-S1, Thr-S2 and Thr-S3 with higher energy-state objectives lie outside the figure. However, these individuals are not important for the comparison of the results. The different seeds of each case are sometimes hard to distinguish. It is noted that the focus lies on the location of the different fronts per case, and differentiating the seeds is therefore not crucial. This is generally the case for the discussion of the results throughout this chapter.

The best solutions are found in the lower left corner of the figure. A few fronts are explicitly pointed out, because they are hard to see in the figure. Behind the front of Thr-S2 with a constraint objective value of approximately 0.24 a front of Thr-S1 is also present. Furthermore, the runs of Thr-S4 with seed 1 and 3 are completely converged and are visible as a single point in the lower left corner of the figure. Also run Thr-S3 with seed 3 has converged around that area.

Thr-S1, Thr-S3 and Thr-S4 all showed similar performance. Each case has one or two runs performing very well, with energy-state objectives close to zero and constraint objectives below 0.05. The remaining runs perform worse for both objectives. Only Thr-S2 performed badly all runs, with its best found constraint objective value of 0.24. This value indicates a maximum possible constraint violation of 24% or multiple smaller violations.

Not enough data is currently available to state that one setting is best, but some conclusions can still be drawn. For the runs that do not include the throttle law, the optimizer performs better with the equivalence ratio (Thr-S1) than with the throttle factor (Thr-S2). When the throttle law is included, both Thr-S3 and Thr-S4 perform similarly. Based on these observations, the equivalence ratio seems to slightly outperform the throttle factor in general. Because the use of the equivalence ratio is physically more realistic than the throttle factor, the equivalence ratio is used throughout the rest of the runs.

No definitive conclusion can be drawn with respect to the throttle law. However, using the throttle law will reduce the number of control parameters in the decision vector. With the throttle law, more control nodes could be included without over-sizing the problem dimension. This is further investigated in Section 7.4. TVC can also be added as an additional parameter for optimization. This is discussed next.

Figure 7.5 shows a similar comparison, but now for the TVC setting results. Again it is noted that the figure is slightly zoomed in. For all settings a few individuals with constraint and energy-state objective values between 0.3 and 0.5 have been cut off from the view. These individuals are not important to the comparison.

TVC-S1 to 3 all perform consistently well for each initial seed. Although TVC-S3 with seed 2 has some individuals with a constraint objective of around 0.24, the same run also produced better performing

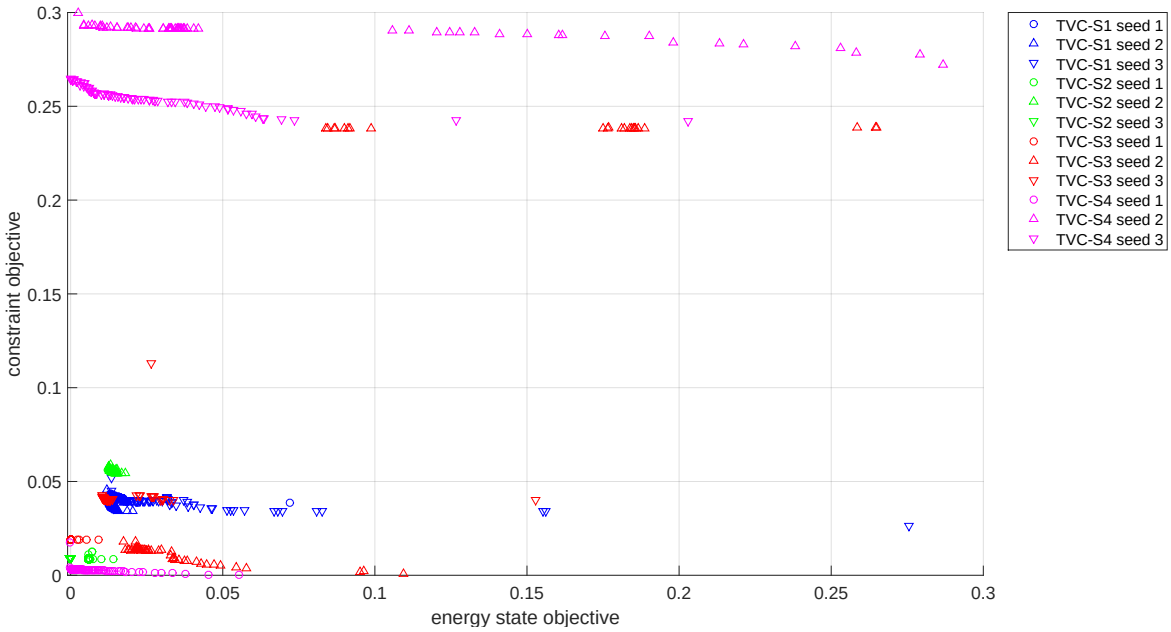


Figure 7.5: Comparison of TVC setting results (n = 100, gen = 500).

individuals. Leaving the simulation to run for more generations would eventually lead to all individuals of this population converging to the best solution. Comparing TVC-S1 with TVC-S3, no significant difference in optimization performance is found between using F_{TVC} or ϵ_T.

The populations of TVC-S2 almost completely converged to a single point for each of the 3 initial seeds. For this case, the value of F_{TVC} was fixed, effectively leaving the optimizer with only 2 control parameters in the decision vector. This reduces the dimension of the problem. The smaller resulting search space could explain why this setting shows a better convergence for the same number of generations.

Leaving out the throttle law in TVC-4 results in 4 control parameters in the decision vector, which increases the problem dimension and complexity. At the same time the flexibility of the possible guidance also increases when more control parameters are included. It is expected that the optimization process should be able to find a better guidance when given more flexibility. However, this better solution will be harder to find, as the search space is now much larger. This could explain why TVC-S4 with seed 1 has found the best solution of all runs, while the runs with seed 2 and 3 have not (yet) found a good solution at all.

When comparing these results with Figure 7.4, TVC-S1 to S3 show more consistent convergence to good solutions. The inclusion of TVC therefore seems to have a positive result on the optimization process as long as the problem dimension is not over-sized. TVC allows for the generation of more lift with the same amount of thrust. The vehicle will therefore not have to accelerate as fast to gain altitude, which could explain why the inclusion of TVC makes it easier to stay within the flight path constraints. This helps the optimization process, as it will be easier to minimize the constraint objective.

7.4. Control Nodes

The results discussed in the previous section have already indicated that the dimension of the optimization problem has an effect on the optimization performance. The problem dimension depends on the number of control parameters included in the decision vector and the number of control nodes at which each control parameter is defined. To study the effect of the problem dimension on the optimization process, both of these aspects are varied in a number of test cases defined in Table 7.5.

Figures 7.6 to 7.8 show comparisons of the different numbers of control nodes for 2, 3 and 4 control parameters, respectively. Each of the figures compares runs with 6, 8, 10 and 12 control nodes. The figures are again zoomed in to show more detail leaving some individuals with higher objective values out. These individuals are not important for the comparison.

case	control nodes	control parameters
CN2.6	6	$\hat{E}$, α
CN2.8	8	$\hat{E}$, α
CN2.10	10	$\hat{E}$, α
CN2.12	12	$\hat{E}$, α
CN3.6	6	$\hat{E}$, α, ϕ
CN3.8	8	$\hat{E}$, α, ϕ
CN3.10	10	$\hat{E}$, α, ϕ
CN3.12	12	$\hat{E}$, α, ϕ
CN4.6	6	$\hat{E}$, α, ϕ, ϵ_T
CN4.8	8	$\hat{E}$, α, ϕ, ϵ_T
CN4.10	10	$\hat{E}$, α, ϕ, ϵ_T
CN4.12	12	$\hat{E}$, α, ϕ, ϵ_T

Table 7.5: Definition of control node comparison cases.

Figure 7.6 compares the runs with 2 control parameters in the decision vector. The best performing solutions are found in the lower left corner of the figure. The most consistently good performance is found when 10 control nodes are used, as all of the seeds have converged to very low objective function values. However, the runs using 6 and 12 control nodes have also found good results for two of the three initial seeds with only one run staying behind in performance. Given the limited number of runs, no definitive conclusion can be drawn with respect to this comparison.

The runs with 3 control parameters in the decision vector are compared in Figure 7.7. In the lower left corner of the figure, only one blue point is visible with a value of zero for both objective function values. This point represents the completely converged result of case CN3.6 for both runs with initial seed 2 and 3. Under the blue dot, also a purple dot is present, which is the converged population of run CN12 with seed 3. Seed 2 and 3 of CN3.8 also perform well with values for both objectives close to zero. The fronts of CN3.6-1, CN3.8-1 and CN3.10-3 largely overlap in the figure and are all located around a constraint objective value of 0.24. All runs of case CN3.10 perform worse and also

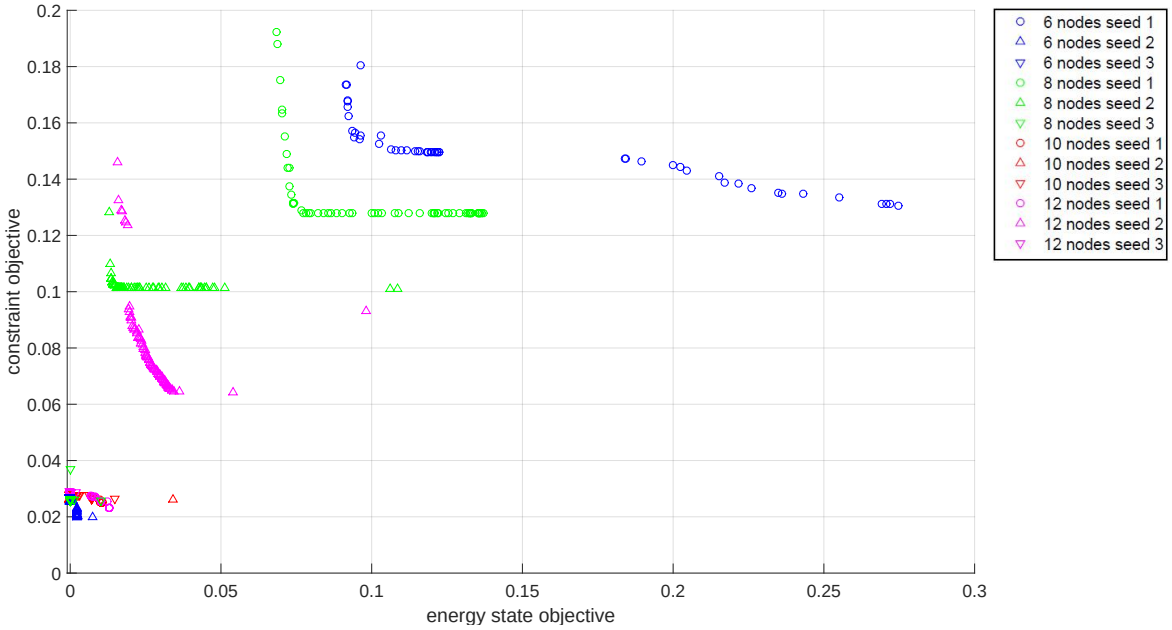


Figure 7.6: Comparison of optimization results using 2 control parameters and 6, 8, 10 or 12 control nodes (n = 100, gen = 500).

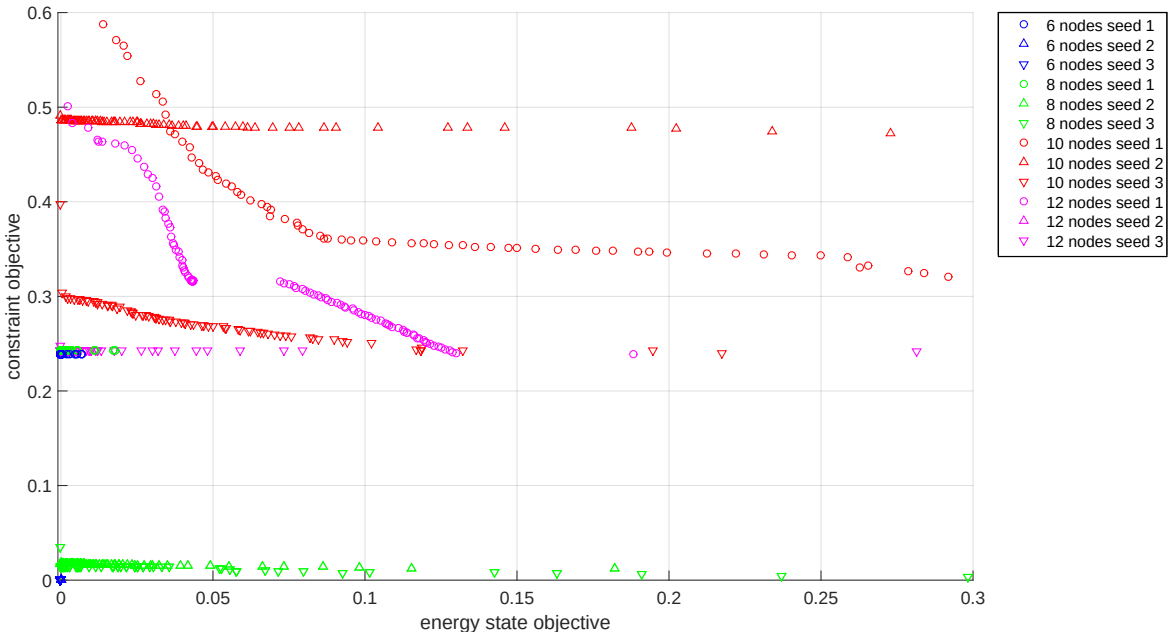


Figure 7.7: Comparison of optimization results using 3 control parameters and 6, 8, 10 or 12 control nodes (n = 100, gen = 500).

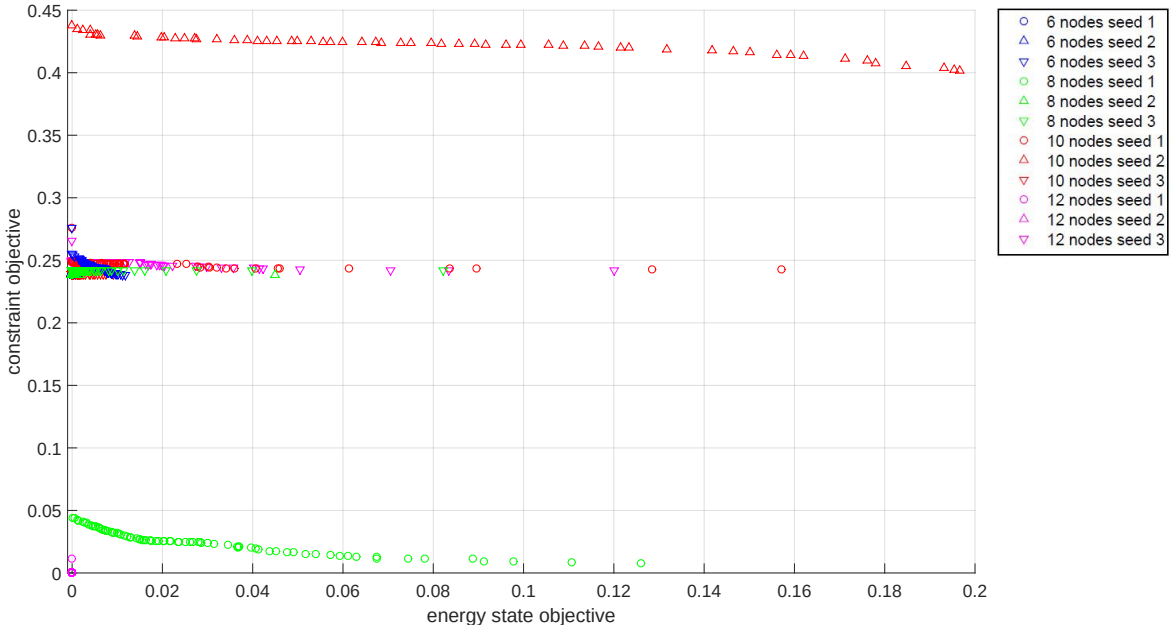


Figure 7.8: Comparison of optimization results using 4 control parameters and 6, 8, 10 or 12 control nodes (n = 100, gen = 500).

the remaining run of CN3.12 has higher objective function values. Based on these results, it appears that the optimizer performs better when less control nodes are used. Case CN3.12 seems to be an exception to this statement, as it performed better than CN3.10 and one of its runs even performed slightly better than CN3.8.

Figure 7.8 compares the runs with 4 control parameters. In this set all runs seem to perform similar, with most fronts located around a constraint objective value of 0.25 and energy-state objectives close to zero. Cases CN4.8 and CN4.12 have one run with a significantly lower constraint objective value, while CN4.10 has one run that performs worse. Based on this limited available data, the conclusion is drawn that with 4 control parameters in the decision vector, the number of control nodes within the

tested range has no significant influence.

When comparing the three figures with each other, some observations can be made with respect to the optimal problem dimension. When 2 optimization parameters are used, good solutions were found for all tested numbers of control nodes, but the most consistent performance was found for 10 control nodes. When an extra optimization parameters was included, the best performance was found with 6 control nodes. The problem dimension is computed with Equation 5.26. As the values of the initial control nodes are fixed, the problem dimension of both CN2.10 and CN3.6 is equal to 18. This indicates that a dimension of 18 is optimal for the currently studied problem. When 4 optimization parameters are included, all tested numbers of control nodes result in a larger problem dimension. This could explain why none of the tested cases resulted in a consistently good performance. The dimension can be reduced by using less than 6 control nodes. Although this is not tested here, it is expected that using fewer control nodes will limit the flexibility of the control history over the complete trajectory to an extent that it will have a negative influence on the optimum trajectory that can be obtained with the resulting guidance.

A larger problem dimension can still lead to good results as shown by run CN4.12-1 in Figure 7.8, but the performance seems to be less consistent. A larger problem dimension should theoretically be able to result in an equal or better performance, because more flexibility is possible in the control history. However, if the search space is too large the optimization problem is more likely to converge to a local optimum before the real optimum is found. The convergence of most fronts around a constraint objective value of 0.25 in Figure 7.8, indicates that this premature convergence happens when using 4 control parameters. This was confirmed by continuing the runs of CN4 up to generation 1200. In these extra 700 generations none of the populations was able to break out of the found local optimum.

Some limitations to the above found results and conclusion should be mentioned. Beside the number of control parameters also the implementation of the used control parameters is important. The search space can be influenced considerably, by setting the domain of each control parameter. When the domains of the control parameters are limited, the total search space is reduced. The optimizer should therefore be able to find good solutions for larger problem dimensions. The above found optimum problem dimension is therefore only valid for this specific implementation of the control parameters.

Furthermore, the optimizer settings also influence the convergence. These settings include the size of the population, and various settings that influence the reproduction and selection procedures of the optimizers. All these settings influence the exploration power, the convergence speed and the total computation time of the optimizer. Changing the optimization method or settings in a way that increases the diversity and reduces the convergence speed, will reduce the likelihood of premature convergence. Although this will slow down the optimization process, this means that good solutions might be found for a higher problem dimension when the optimization is run for more generations. With the current settings, running the optimization for 1,000 generations instead of 500 generations will generally not improve the results considerably. Based on those results it is concluded that most populations will converge within 500 generations to a solution that is close to their final (local) optimum. The conclusions drawn in this section are therefore valid for the current optimization settings.

Finally Figures 7.9 and 7.10 show examples of the resulting control history as a function of the energy state for a competitive individual of cases CN-3.6 to CN3.12. The control nodes at which the control parameters are defined are indicated with the circles. Figures 7.11 and 7.12 show the resulting control history as a function of time. All control histories have a similar shape. The larger number of control nodes primarily leads to fluctuations around the general control history shape found by the run with 6 control nodes.

Based on the results discussed in this section cases CN2.10 and CN3.6 are selected to be used throughout the rest of this research.

7.5. Optimizer Comparison

All results so far are obtained with the default MOEA/D optimization method. In this section the performance of this optimizer is compared with three other methods. The other methods studied are NSGA-II, NSGA-II-tabu and NSGA-II-tabu-reintro. The last method is an additional version of NSGA-II-tabu that reintroduces one random tabu individual back into the individual list every 5 generations. All used methods are explained in Section 5.2.

Each optimization is tested for cases CN2.10 and CN3.6 defined in the previous section. Figures

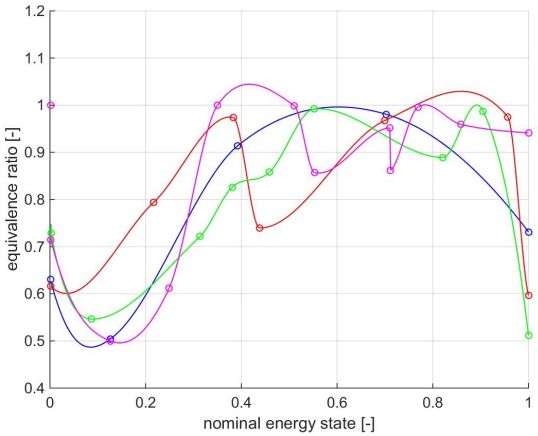


Figure 7.9: Comparison of equivalence ratio versus normalized energy state for control history with 6, 8, 10 and 12 control nodes.

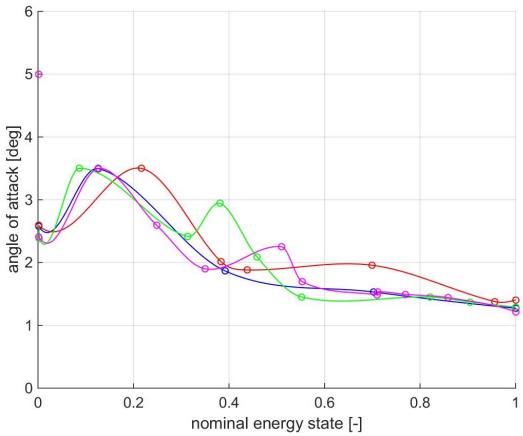


Figure 7.10: Comparison of angle of attack versus normalized energy state for control history with 6, 8, 10 and 12 control nodes.

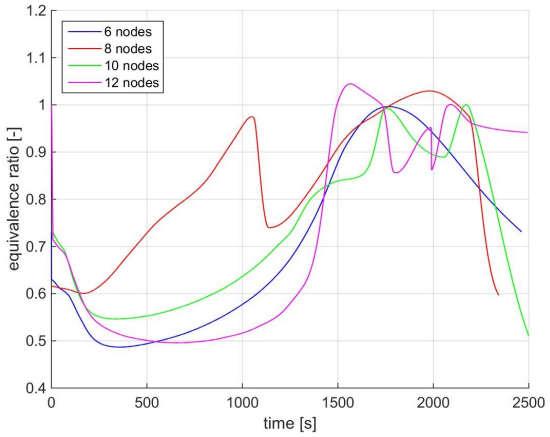


Figure 7.11: Comparison of equivalence ratio versus time for control history with 6, 8, 10 and 12 control nodes.

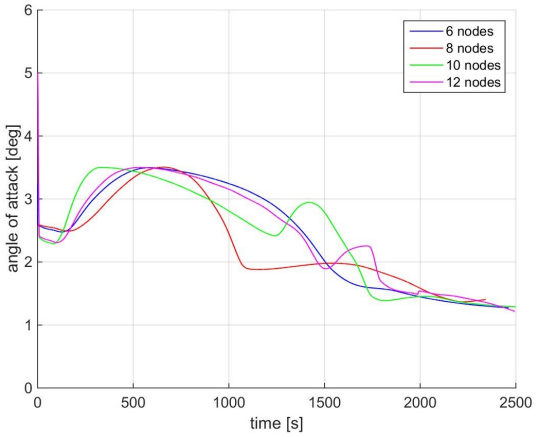


Figure 7.12: Comparison of angle of attack versus time for control history with 6, 8, 10 and 12 control nodes.

7.13 and 7.14 compare the results of the methods for each case, respectively. The figures are again slightly zoomed in to the lower objective function values where the better performing individuals are located. The individuals that lie outside of the figure scope are not relevant for the comparison.

Figure 7.13 compares the performance of the different optimizers for case CN3.6. Both MOEA/D and NSGA-II perform similarly with respect to the energy state objective. However, where MOEA/D converged to solutions with constraint objective values close to zero, NSGA-II did not find constraint objective values below 0.24. Both implementations of NSGA-II-tabu show nicely distributed Pareto fronts, but none of the runs have converged to a good solution.

Figure 7.14 compares the performance of the different optimizers for case CN2.10. With the throttle law included NSGA-II performs similar to MOEA/D. Also both NSGA-II-tabu implementations seem to perform better, specifically with respect to the constraint objective. However, it is noted that the populations of NSGA-II-tabu with initial seeds 1 and 3, and NSGA-II-tabu-reintro with seed 1 have converged to a no-flight solution. These solutions have a constraint objective value of 0 and an energy-state objective value of 0.997, and are not visible in Figure 7.14.

Figure 7.15 shows how quickly a population can converge to such a no-flight solution. After 10 generations, the population is almost completely converged. The lack of diversity in the population restricts exploration in other areas of the search space. After 100 generations, the population is converged to a single no-flight solution. A problem definition that includes the throttle law is found to be more sensitive to get stuck in such a no flight-solution. Also the NSGA-II-tabu methods are more likely

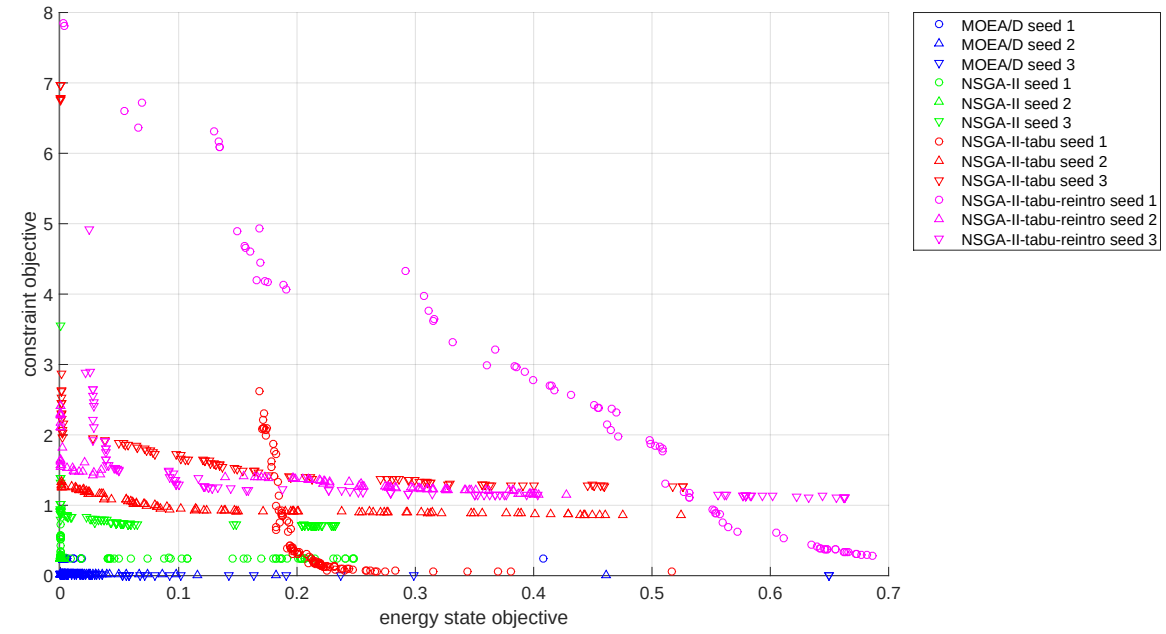


Figure 7.13: Comparison of optimization results for case CN3.6 using 4 different optimization methods (n = 100, gen = 500).

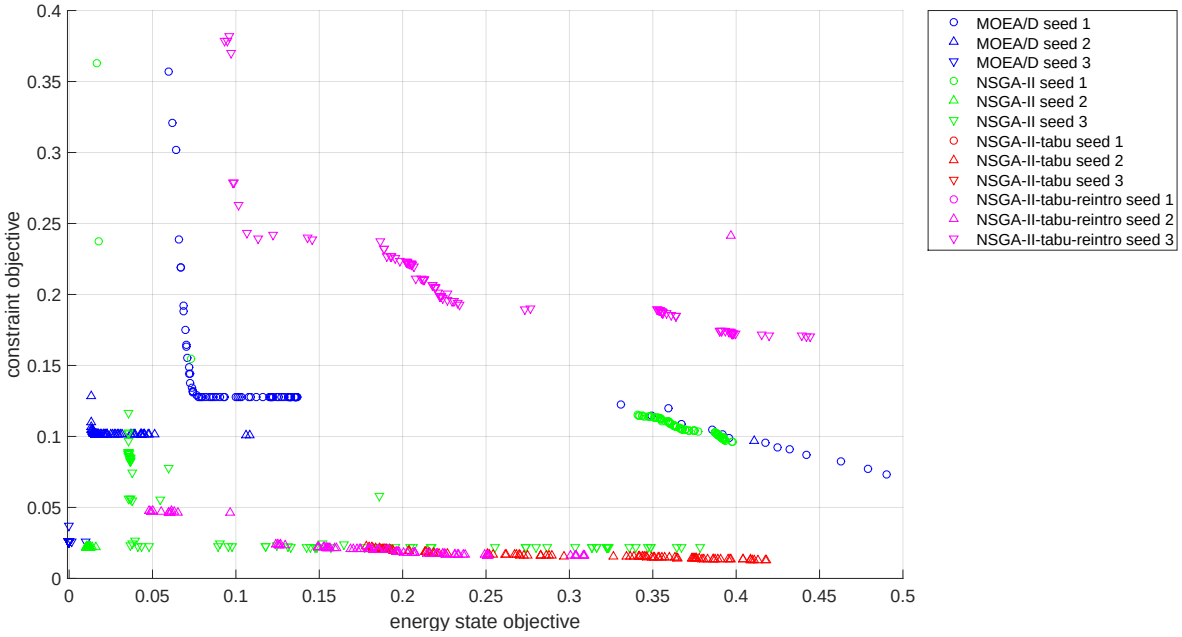


Figure 7.14: Comparison of optimization results for case CN2.10 using 4 different optimization methods (n = 100, gen = 500).

to find a no-flight solution.

The sensitivity to no-flight solutions for NSGA-II-tabu can be explained by the initial random population. In general the ascent trajectory optimization struggles to find initial solutions that actually fly. During the initial generations the optimizer relies on a few 'lucky' individuals that do not crash straight away. Once a few individuals have escaped the set of no-flight solution, the rest will quickly follow. The no-flight penalty helps this process as was already discussed in Section 7.2. However, when the tabu-list is included, those initial good individuals are separated from the evolving population and stored in the tabu-list. During the initial generations of the optimization this increases the likelihood of the optimizer to converge to a no-flight solution, as is found in the results. Reintroduction of a tabu individual into the individual list can help to get out of this local optimum, as is illustrated by run NSGA-II-tabu-reintro with seed 3.

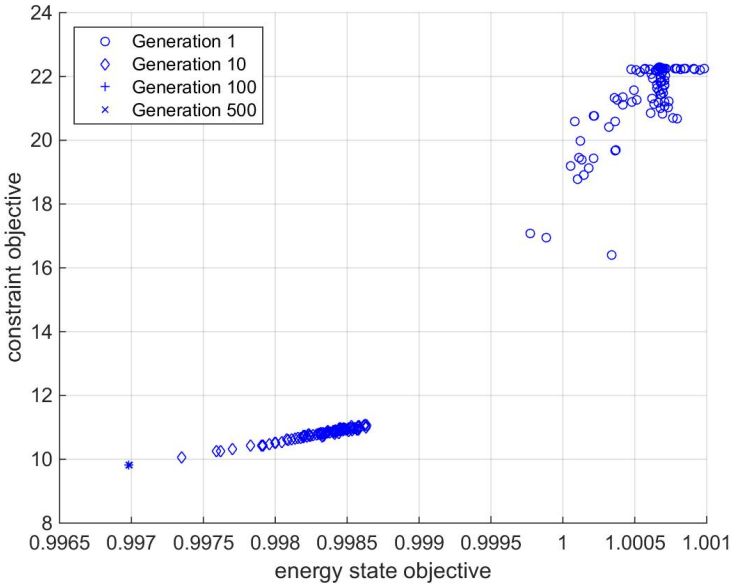


Figure 7.15: Example of premature convergence to no-flight trajectory (n = 100).

Although the no-flight solutions are a problem that currently occurs with the NSGA-II-tabu methods, this should not define the performance of this method. The convergence to no-flight solutions is an issue specific to the initial stage of the ascent trajectory optimization problem. Inclusion of a tabu-list makes the optimization more sensitive to this problem, but it could potentially be solved by further improving the no-flight penalty. Only initiating the tabu-list for the first time after the 10th generation would also be an easy solution.

Overall MOEA/D seems to be the most consistent and best performing optimization method tested here. However, some limitations to the found results and conclusions should be mentioned.

All methods have been tested with the default settings provided by Tudat. Although these settings generally work well, the performance of the optimization process is very problem specific. For each method a better combination of settings might exist for the ascent trajectory optimization problem. At this stage it is therefore only possible to state that MOEA/D seems to be the most promising method studied here. Further research should prove if this is true for different combinations of settings.

Although both implementations of NSGA-II-tabu did not result in good solutions, that does not mean that the methods did not work. In fact, the methods showed more diversity and a better distribution of the Pareto fronts, which is exactly what the intended result was. However, the increase in diversity goes hand in hand with a slower convergence. Running these methods for more generations might result in solutions just as good or even better than the solutions currently found with MOEA/D. However, with the currently available computation power and time, these methods are not competitive for the ascent trajectory problem.

Finally, a note is made on parallel optimization discussed in Section 5.2.5. The current implementation did not allow for a proper comparison of the results. However, initial tests were performed with a parallel implementation of MOEA/D and NSGA-II-tabu. The results indicated a more robust method with a convergence speed similar to MOEA/D while maintaining the diversity found with NSGA-II-tabu. Further research is required to confirm these findings.

7.6. Objective Handling

The feasibility and optimality objectives each drive the optimization to different directions of the search space. This fact complicates the optimization considerably, and the way objectives are handled is therefore very important. Different approaches were tested to handle the objectives:

approach 1: Optimize feasibility and optimality objectives together.

approach 2: First optimize feasibility objectives and continue optimization with both feasibility and optimality objectives
approach 3: First optimize feasibility objectives and continue optimization by alternating optimality and feasibility objectives.
approach 4: First optimize feasibility objectives and continue with optimization of optimality objectives around the best found feasible individual.

The different objectives and optimization approaches were explained in Section 5.3.5. In this section the results of each approach will be discussed. It is noted that the following guidelines will be kept in mind when assessing the results:

- Any individual with a combined energy-state and constraint objective value over 0.1 is considered not feasible.
- A fuel-mass objective of around 0.47 is required for the currently evaluated atmospheric acceleration phase to reach the final state.
- A low heat-load objective is desirable, but not a requirement, as long as the heat-rate constraint is not violated.

Approach 1
This approach optimizes both the feasibility and optimality objectives together. The approach is tested with and without using the throttle law for cases CN2.10 and CN3.6, respectively. Furthermore, the optimality objectives have been evaluated together as a single objective function and as two separate objective functions, resulting in a total number of three and four objective functions, respectively. Each of the test cases is defined in Table 7.6.

case	nodes	throttle law	generations	population size	Obj 1	Obj 2	Obj 3	Obj 4
Appr1.1	6	no	0 to 500	105	f[1]	f[2]	f[4] + f[5]	-
Appr1.2	10	yes	0 to 500	105	f[1]	f[2]	f[4] + f[5]	-
Appr1.3	6	no	0 to 500	120	f[1]	f[2]	f[4]	f[5]
Appr1.4	10	yes	0 to 500	120	f[1]	f[2]	f[4]	f[5]

Table 7.6: Definition of cases for simultaneous optimization of feasibility and optimality objectives.

Figure 7.16 shows a comparison between the runs using a single optimality objective function, or two separate functions for test case CN3.6. The data points are projected on the three planes. It is noted that the combined feasibility objective function value on the vertical axis is cut off at 0.5, leaving out a number of non-feasible individuals with a feasibility objective value over 0.1.

Separating the optimality objectives shows better results. In fact, Appr1.1 has not found any feasible solutions. Another observation can be made when comparing the red data points on the right vertical plane with those on the bottom plane. Individuals with lower fuel-mass objective values also perform relatively well on the feasibility objective. The individuals that perform well on the heat-load objective, tend to perform worse on the feasibility objective.

This last observation could explain why Appr1.1 has only found non-feasible solutions. When looking at the bottom plane of the Figure 7.16, the individuals of Appr1.1 are located more towards a lower heat-load objective value than the indiviudals of Appr1.3. When combining objectives in one objective function each objective should be given a proper weight factor. In the current implementation, the heat-load objective slightly dominates the fuel-mass objective. The population will therefore converge more towards the heat-load objective than the fuel-mass objective. which results in a worse performance of the feasibility objective.

Figure 7.17 shows the same comparison but now for test case CN2.10, which includes the throttle law. The combined feasibility objective function value on the vertical axis is again cut off at 0.5. When comparing Appr1.2 with Appr1.1 in Figure 7.16, combining the optimality objectives into one function seems to work better when the throttle law is included. However, only the run with initial seed 3 has found feasible solutions. The dark green front visible on the top side of the horizontal plane looks promising, but consists of only non-feasible solutions. Also the solutions of Appr1.2 at the right side of the horizontal plane are not feasible.

When all the non-feasible solutions are excluded, separating the optimality objectives also outperforms the combined objective function when the throttle law is included. However, it should be noted that in both cases the throttle law seems to have a slight positive effect on the results.

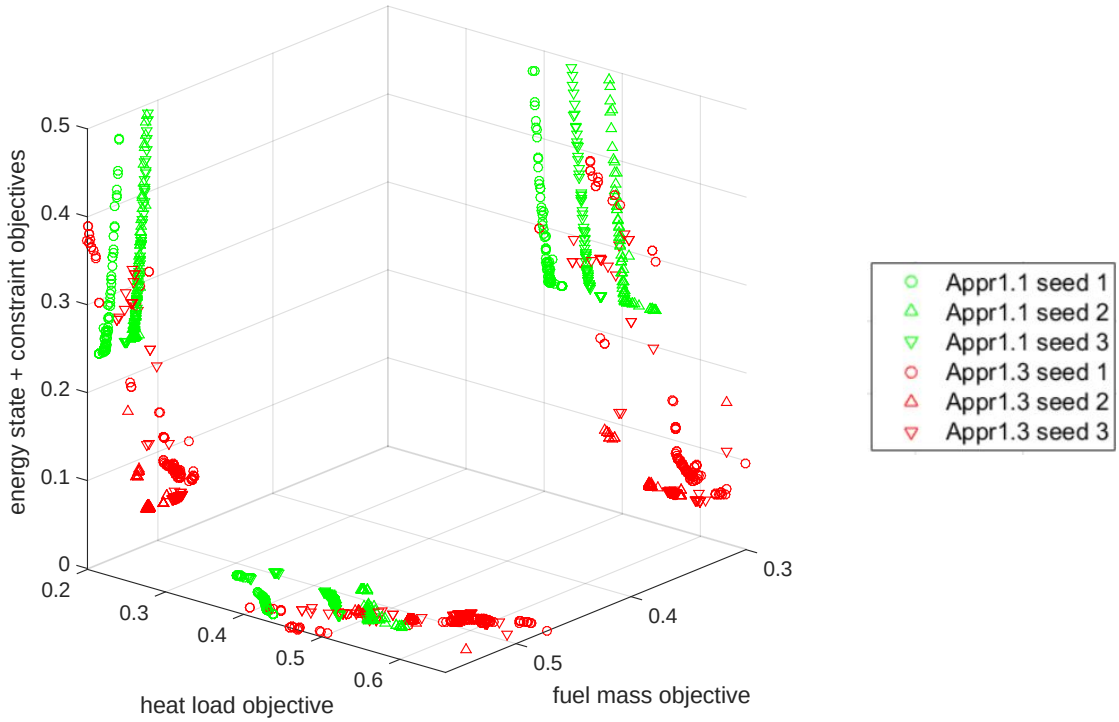


Figure 7.16: Comparison of optimization results for Appr1.1 and Appr1.3 without throttle law projected on planes (gen = 500).

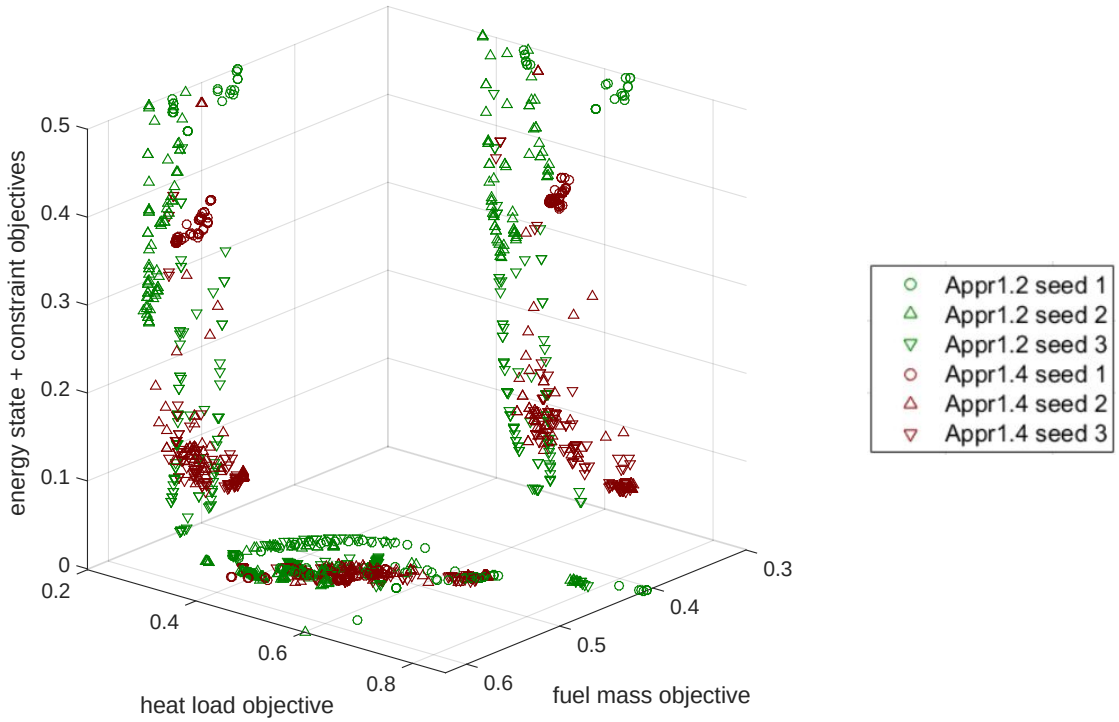


Figure 7.17: Comparison of optimization results for Appr1.2 and Appr1.4 with throttle law projected on planes (gen = 500).

Approach 2
This approach first optimizes the feasibility objectives. Subsequently a combined optimization of the feasibility and optimality objectives is performed. The approach is tested with and without using the throttle law for cases CN2.10 and CN3.6, respectively. Furthermore, both feasibility and optimality objectives have been evaluated together as one objective function and as two separate objective functions. The resulting test cases are defined in Table 7.7.

The obtained results show strong similarity to those discussed for approach 1. The same conclusions also apply. Separating all objectives in separate functions showed the best results. Furthermore, including the throttle law seemed to have a positive influence on the results as well. No other significant difference in performance was found between approach 1 and 2.

case	nodes	throttle law	generations	population size	Obj 1	Obj 2	Obj 3	Obj 4
Appr2.1	6	no	0 to 500	100	f[1]	f[2]	-	-
			501 to 700	100	f[1] + f[2]	f[4] + f[5]	-	-
Appr2.2	10	yes	0 to 500	100	f[1]	f[2]	-	-
			501 to 700	100	f[1] + f[2]	f[4] + f[5]	-	-
Appr2.3	6	no	0 to 500	105	f[1]	f[2]	-	-
			501 to 700	105	f[1] + f[2]	f[4]	f[5]	-
Appr2.4	10	yes	0 to 500	105	f[1]	f[2]	-	-
			501 to 700	105	f[1] + f[2]	f[4]	f[5]	-
Appr2.5	6	no	0 to 500	105	f[1]	f[2]	-	-
			501 to 700	105	f[1]	f[2]	f[4] + f[5]	-
Appr2.6	10	yes	0 to 500	105	f[1]	f[2]	-	-
			501 to 700	105	f[1]	f[2]	f[4] + f[5]	-
Appr2.7	6	no	0 to 500	120	f[1]	f[2]	-	-
			501 to 700	120	f[1]	f[2]	f[4]	f[5]
Appr2.8	10	yes	0 to 500	120	f[1]	f[2]	-	-
			501 to 700	120	f[1]	f[2]	f[4]	f[5]

Table 7.7: Definition of cases that first optimize the feasibility objectives and subsequently the optimality and feasibility objectives combined.

Approach 3
This approach first optimizes the feasibility objectives. Subsequently an alternate optimization of the optimality and feasibility objectives is performed. The approach is again tested for both cases CN2.10 and CN3.6. For Appr3.3 and Appr3.4 the energy state is also included as an optimality objective function. The resulting test cases are defined in Table 7.8. Results for Appr3.1 and Appr3.2 are shown in Figures 7.18 and 7.19. The results of Appr3.3 and 3.4 are very similar and will therefore not be discussed separately.

Figure 7.18 shows the results of Appr3.1, which does not include the throttle law. The green points indicate the final population after 50 generations of optimizing for the feasibility objectives. The blue points show the populations after optimizing 50 generations for the optimality objectives. It is noted that different seeds are hard to distinguish. However, only the different location of the green points with respect to the blue points is relevant for the discussion that follows. Furthermore, it is important to realize that the green points located near the intersection of horizontal plane with the vertical planes, are actually positioned on the vertical plane only.

When looking at the bottom plane, the blue points show good performance with respect to the heat-load and fuel-mass objectives. This is not surprising as these populations were optimized with respect to these objectives. The green individuals have been optimized with respect to the feasibility objectives and perform worse with respect to the optimality objectives. When looking at the vertical planes, the green individuals do show good performance with respect to the feasibility objectives. The blue individuals perform well with respect to the optimality objectives, but none of the solutions is feasible, as the feasibility objective value of all individuals is much larger than 0.01. These results

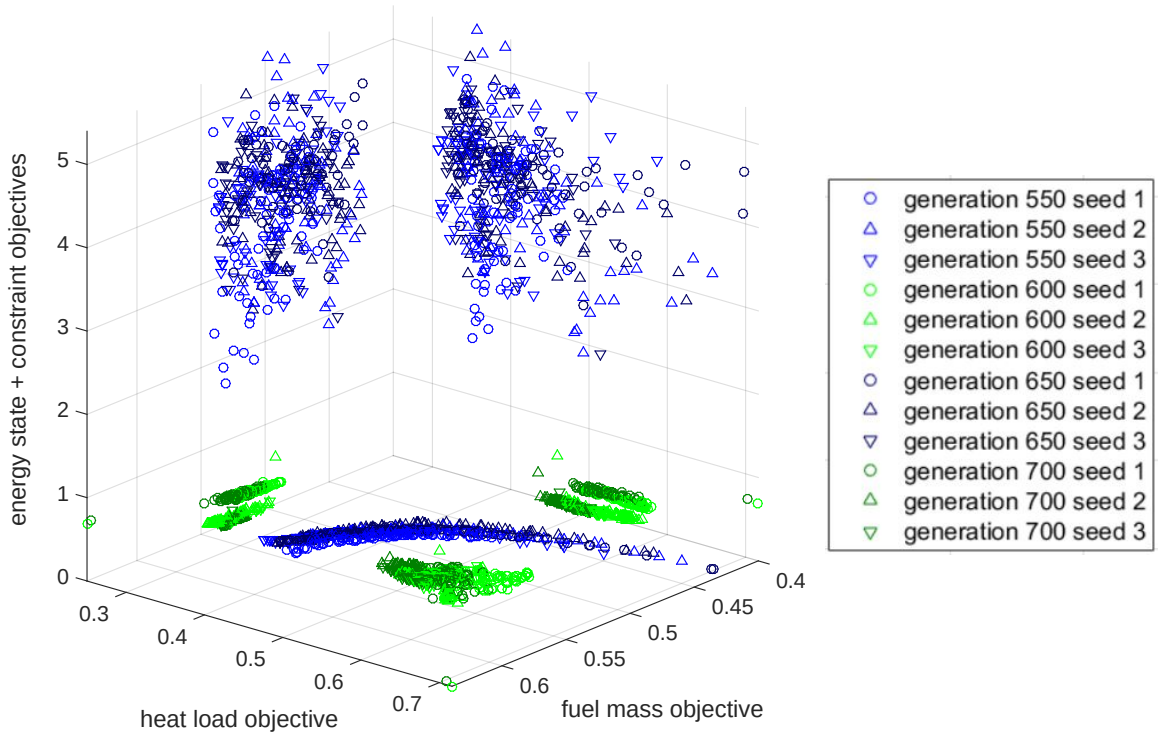


Figure 7.18: Results of Appr3.1 with alternate optimization of feasibility and optimality objectives projected on planes (n = 100, gen = 500).

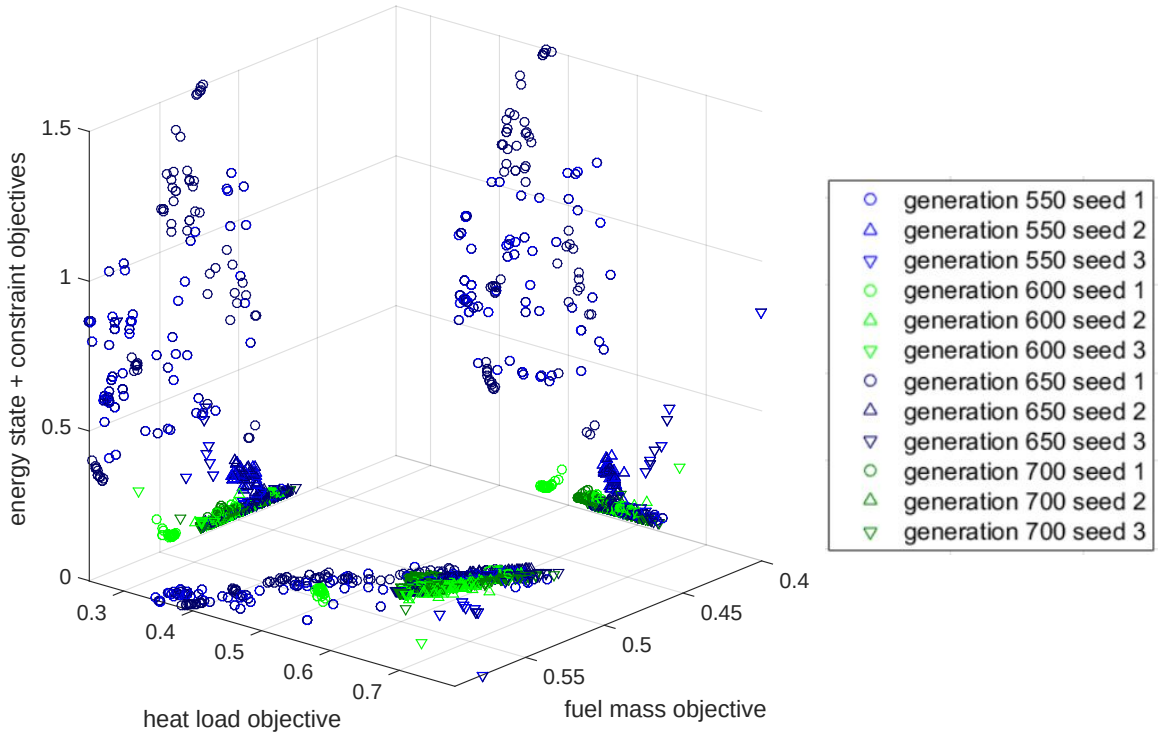


Figure 7.19: Results of Appr3.2 with alternate optimization of feasibility and optimality objectives, projected on planes (n = 100, gen = 500).

case	nodes	throttle law	generations	population size	Obj 1	Obj 2	Obj 3
Appr3.1	6	no	0 to 500	100	f[1]	f[2]	-
			501 to 550	100	f[4]	f[5]	-
			551 to 600	100	f[1]	f[2]	-
			601 to 650	100	f[4]	f[5]	-
			651 to 700	100	f[1]	f[2]	-
Appr3.2	10	yes	0 to 500	100	f[1]	f[2]	-
			501 to 550	100	f[4]	f[5]	-
			551 to 600	100	f[1]	f[2]	-
			601 to 650	100	f[4]	f[5]	-
			651 to 700	100	f[1]	f[2]	-
Appr3.3	6	no	0 to 500	105	f[1]	f[2]	-
			501 to 550	105	f[1]	f[4]	f[5]
			551 to 600	105	f[1]	f[2]	-
			601 to 650	105	f[1]	f[4]	f[5]
			651 to 700	105	f[1]	f[2]	-
Appr3.4	10	yes	0 to 500	105	f[1]	f[2]	-
			501 to 550	105	f[1]	f[4]	f[5]
			551 to 600	105	f[1]	f[2]	-
			601 to 650	105	f[1]	f[4]	f[5]
			651 to 700	105	f[1]	f[2]	-

Table 7.8: Definition of cases that first optimize the feasibility objectives and subsequently alternately the optimality and feasibility objectives.

show the conflict that exists between both sets of objectives. The desired solution is a compromise between both sets of objectives. To obtain this desired solution, the problem should be optimized with respect to both sets of objectives simultaneous.

Figure 7.19 shows the results of Appr3.2, which includes the throttle law. The color representation is the same as for Figure 7.18. The results show a similar pattern, but the contrast is not as strong. The populations found after optimization with the optimality objectives, indicated in blue, still have some individuals moving into the non-feasible space, but a large part of the populations remains in the feasible space. The difference between the results of Appr3.1 and Appr3.2 can be entirely attributed to the throttle law.

Thrust has a negative influence on the function values of the optimality objectives. The more thrust is given, the higher the fuel consumption and the heat load. When the feasibility objectives are replaced by the optimality objectives, the optimizer no longer pushes for solutions that reach the final state and do not violate the constraints. The optimizer will therefore slowly converge to solutions that use less thrust. However, these solutions are not feasible. When the throttle law is used, thrust is not directly controlled by an optimization parameter. A problem case with the throttle law included will therefore be able to better maintain thrust, and will therefore not converge to non-feasible solutions as fast.

Approach 4

This approach first optimizes the feasibility objectives. A good performing feasible solution is then used for further optimization of the optimality objective. For each optimization parameter in the final decision vector a domain is set within which the parameter must remain. The range is given as a percentage of the initial range of the respective parameter. The resulting optimization can be seen as a local optimization around the found feasible solution. The solutions used as a starting point for the optimization are given in Table 7.9. The used test cases are defined in Table 7.10.

Figure 7.20 shows the results for Appr4.1, where the throttle law was not used. Giving more freedom to a control parameter will result in an improvement of the optimality objective. For a range of $\pm$ 1% the fuel-mass objective improved by a maximum value of 0.007. This coincides with a fuel mass saving of approximately 1,000 kg. However, at the same time, the feasibility objective value increases to 0.25.

Solutions	nodes	throttle law	f[1]	f[2]	f[4]	f[5]
Sol1	6	no	-0.00007	0.00000	0.51227	0.55594
Sol2	10	yes	-0.00010	0.02557	0.50297	0.55600

Table 7.9: feasible solution used for approach 4.

case	domain	nodes	throttle law	generations	population size	Obj 1	Obj 2
Appr4.1.1	± 1%	6	no	0 to 200	100	f[4]	f[5]
Appr4.1.2	± 0.1%	6	no	0 to 200	100	f[4]	f[5]
Appr4.1.3	± 0.01%	6	no	0 to 200	100	f[4]	f[5]
Appr4.2.1	± 1%	10	yes	0 to 200	100	f[4]	f[5]
Appr4.2.1	± 0.1%	10	yes	0 to 200	100	f[4]	f[5]
Appr4.2.1	± 0.01%	10	yes	0 to 200	100	f[4]	f[5]

Table 7.10: Definition of cases that optimize the optimality objectives within a set domain around a feasible solution.

The resulting solution is therefore not feasible. Even an offset of less than ± 0.01% already caused the feasibility objective value to increase from -0.00007 to 0.1.

Figure 7.20 shows the same result but now for Appr 4.2, which does include the throttle law. As was the case with approach 3, the throttle law seems to have a positive effect on the results. The throttle law helps to keep the feasibility objective at a set level, while a small improvement on the optimality objectives is found. The maximum improvement of the fuel-mass objective was 0.004, resulting in an equivalent fuel saving of 550 kg.

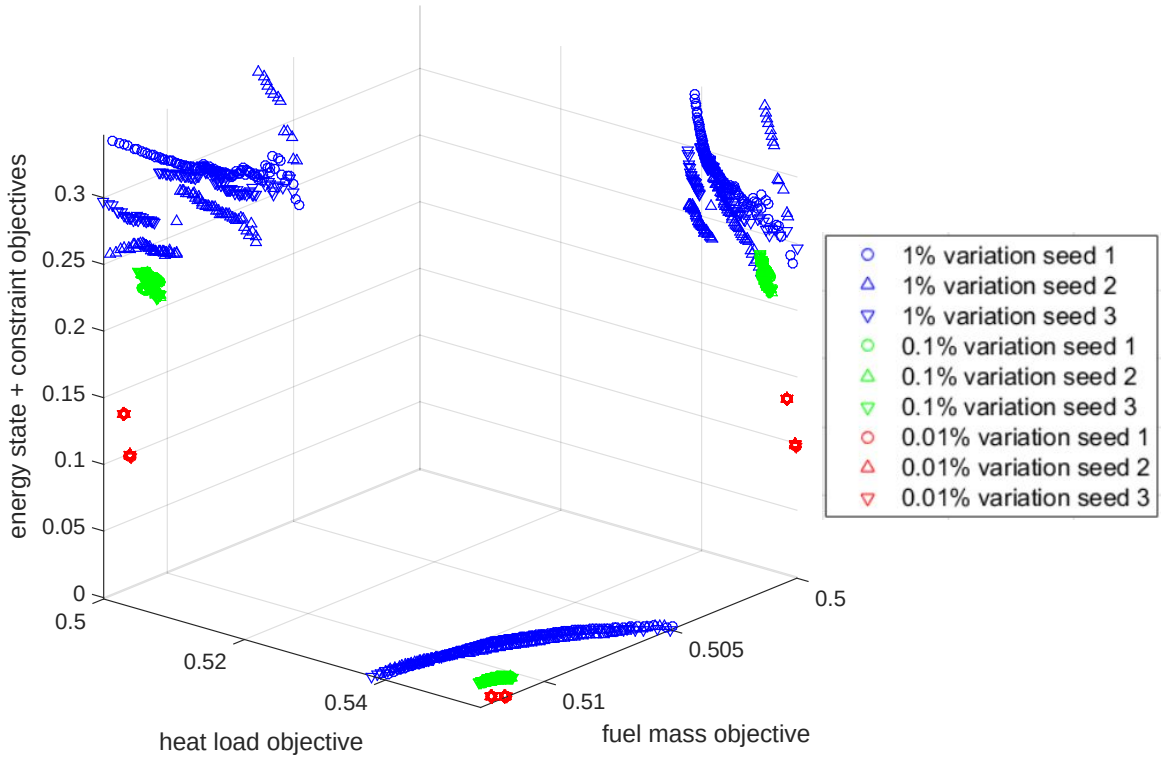


Figure 7.20: Results of Appr4.1 without throttle law projected on planes for 3 different parameter domains (n = 100, gen = 200).

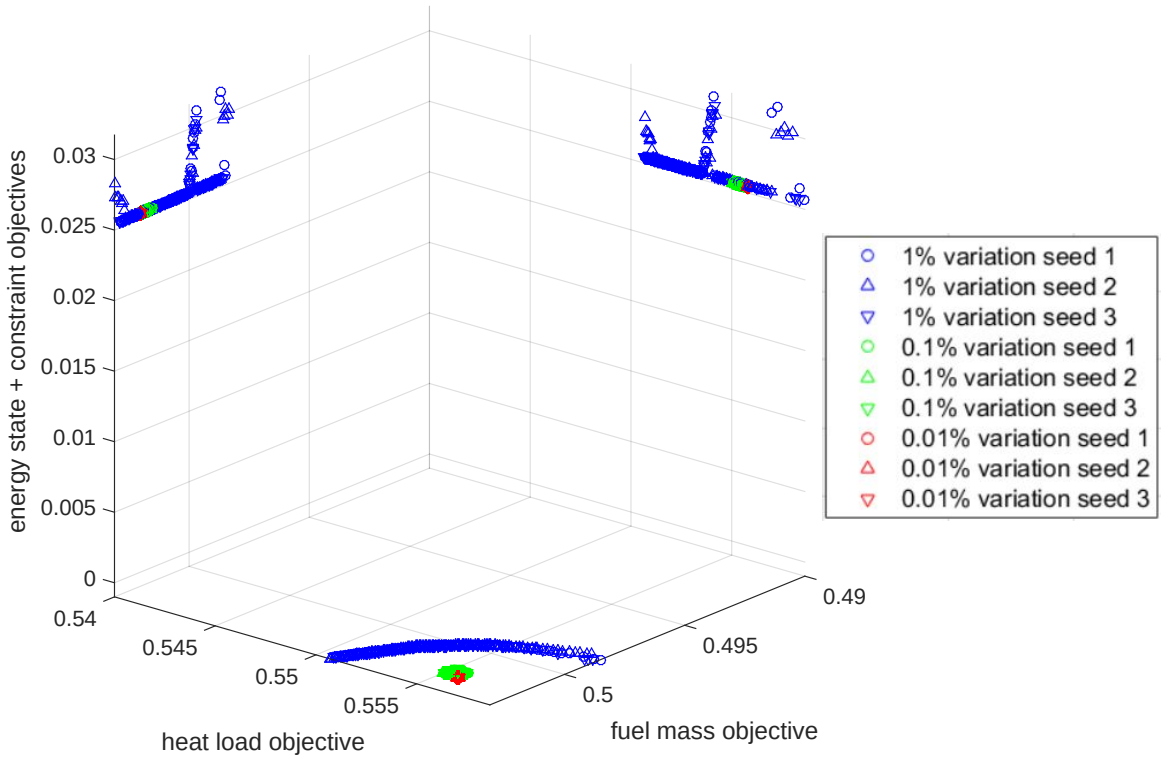


Figure 7.21: Results of Appr4.2 with throttle law for projected on planes 3 different parameter domains (n = 100, gen = 200).

Trade-off

Given all the results found in this section, a simple trade-off between the four approaches can be performed. Although the trade-off is based on the current problem and implementation, some side notes will be made that apply to a more general case.

Approach 3 and 4 both do not give the desired result. Although they are successful in optimizing the optimality objectives, the found solutions are mostly non-feasible. It is noted that for a re-entry problem, optimality and feasibility objectives both drive the solutions in a similar direction. Hänninen (2009) showed that a method similar to approach 3 was able to give good results for a re-entry problem. However, for the current investigated problem, the conflict between optimality and feasibility objectives is to large too optimize them separately.

Approach 1 and 2 performed similarly. However, approach 2 requires an extra optimization stage, which makes approach 1 easier to apply. For the current implementation, including this extra stage did not benefit the process and approach 1 is therefore preferred.

For both approach 1 and 2 it was found that separating all objectives into different functions will lead to the best results. This will therefore also be applied for the final mission optimization. However, reaching the required value of approximately 0.47 for the fuel-mass objective was found to be extremely difficult. Furthermore, the solutions that do have a good fuel-mass objective value, perform badly in terms of the heat-load objective. Because the heat-load objective is not a hard requirement, this objective will be excluded in the final mission optimization. This should make it easier to find solutions with the required fuel-mass objective value.

7.7. Optimizing the Mission

This section will discuss the results of the final optimized ascent mission. A complete set of mission requirements was defined in Section 2.4.

Optimizing the mission from start to end was found to be impossible for the current implementation within an acceptable time frame. The boundaries of each control parameter are based on the maximum and minimum values that are expected to occur. When the complete trajectory is optimized as a single problem, only one set of extreme values is set for the whole trajectory. This can result in a considerably

increased search space of the optimization problem, which has a negative influence on the optimization performance as was found in Section 7.4.

Figure 6.17 show why implementing variable boundary for the optimization parameters conditions could be beneficial. The figure shows the angle of attack versus time for the reference trajectory. The maximum angle of attack is 12 degrees, but this state only lasts for about 10 seconds. For the majority of the flight, the angle of attack does not exceed 3 degrees. Currently a variable boundary condition is implemented by splitting the trajectory in different phases. Separating the trajectory into stages means that the majority of the flight can be optimized with a much lower maximum boundary condition for the angle of attack. This helps to reduce the search space of the optimization process. Based on the control history of the reference trajectory, the trajectory is split in three phases. The separate phases are defined in Table 7.11

phase	control parameter	domain
take-off	α	[0.0 12.0]
	ϕ	[0.0 2.0]
atmospheric acceleration	α	[0.0 3.5]
	ϕ	[0.4 1.0]
pull-up	α	[0.0 12.0]
	ϕ	[0.0 2.0]

Table 7.11: Definition of flight phases with corresponding control parameter boundaries.

Results for the combined optimization of different phases are discussed in Appendix C.1. The following combinations were used:

- take-off + atmospheric acceleration
- atmospheric acceleration + pull-up
- take-off + atmospheric acceleration + pull-up

For none of the combinations above a feasible trajectory was found within a reasonable amount of time using the current implementation. Each of the phases is therefore optimized separately and the results are discussed in Sections 7.7.1 to 7.7.3. Finally, Section 7.7.4 combines the results of the separate optimization processes to form the final complete ascent trajectory.

It is noted in advance that results of the different flight phases discussed next do not seamlessly merge together. In order to obtain a smooth continuation of the control history over consecutive flight phases, the initial as well as the final node should be defined prior to starting the run. Furthermore, the derivative of the final control value of a given flight phase should match the derivative of the initial control value of its succeeding flight phase. This is currently not implemented in the software.

The initial state of each flight phase should also match the final state of its preceding phase. This could be obtained with the current implementation when optimizing in series. However, due to time constraints the phases were optimized in parallel. Although successive states do not match, values for initial mass, velocity and altitude were chosen conservatively, which should also lead to a conservative final payload mass.

7.7.1. Take-off

The definition of the take-off phase is based on the reference trajectory found by Mooij (1998). The vehicle starts at zero altitude with a velocity of 170 m/s. A steep ascent is initiated until a flight-path angle of 25 degrees is reached. In the reference trajectory this condition was met at a velocity of approximately 220 m/s and an altitude of about 350 meters.

Reaching the flight-path angle of 25 degrees was found to be an essential start condition to be able to find a feasible solution for phase 2. The final flight-path angle was therefore included as an extra objective. The objective was added to the set of evaluation objectives $\boldsymbol{f}$ and is defined by:

$$f[6] = \left| \frac{\gamma_{obj} - \gamma_f}{\gamma_{obj}} \right| \tag{7.1}$$

parameter	start value	objective value	control specifications
m [kg]	136,079	-	-
V [m/s]	170	220	-
h [m]	0	350	-
γ [deg]	0.1	25	-
α [deg]	12	-	4 nodes, domain [0.0 12.0]
ϕ [-]	1	-	4 nodes, domain [0.5 2.0]
objectives	f[1], f[2], f[4], f[6]		

Table 7.12: Definition of settings, boundary conditions and objectives of the take-off flight phase.

with γ_{obj} representing the flight-path angle objective value and γ_f the flight-path angle at the final state of the found trajectory. Table 7.12 summarizes the settings, boundary conditions and objectives used for optimization of the take-off.

The resulting control histories and the final trajectories found by the optimizer for the 3 different initial seeds used are shown in Figures 7.22 to 7.24. The reference trajectory by Mooij (1998) used a constant α = 12 degrees and ϕ = 1. The control history found by the optimizer is similar in magnitude. Some fluctuation are visible, especially for the angle of attack, but the different control histories result in very similar final trajectories, as shown in Figure 7.22. Some difference can be seen in the resulting trajectories. As expected, the larger initial angle of attack found by the run with initial seed 1 results in a slightly steeper initial ascent. The run with seed 3 starts by accelerating and ends with a steeper ascent.

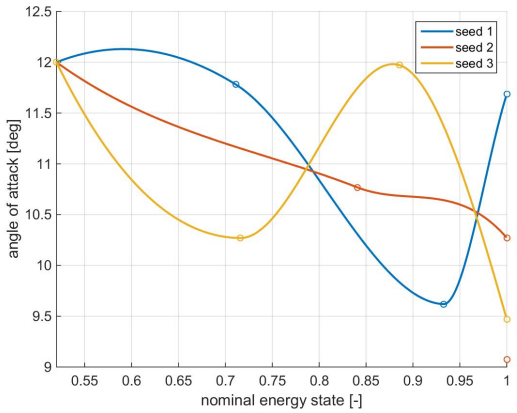


Figure 7.22: Control history of angle of attack versus the normalized energy state for the take-off phase.

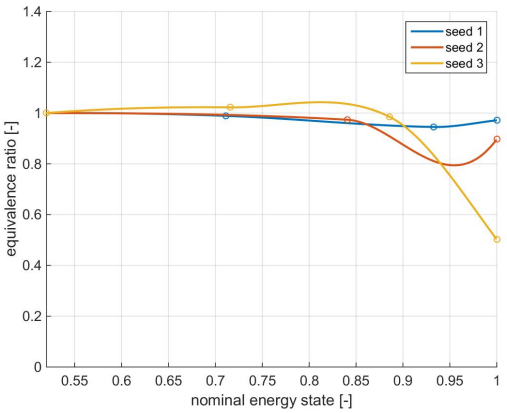


Figure 7.23: Control history of equivalence ratio versus the normalized energy state for the take-off phase.

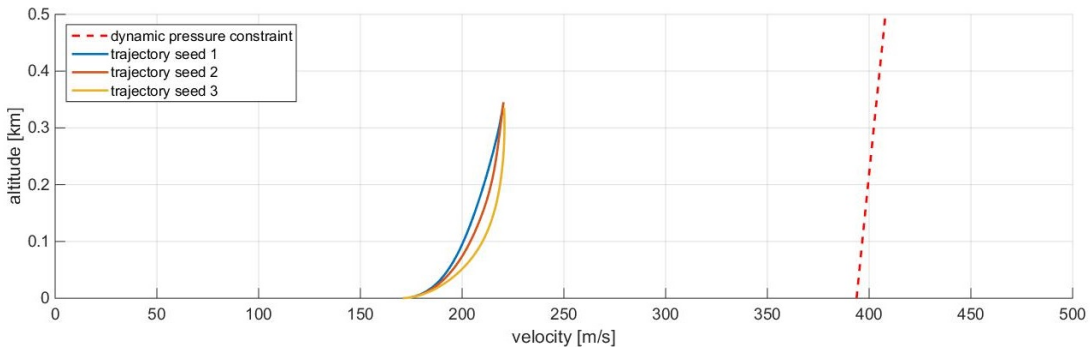


Figure 7.24: Final trajectories found for the take-off phase.

	seed 1	seed 2	seed 3
m [kg]	135,506	135,506	135,512
V [m/s]	361.7	361.6	352.0
h [m]	221.0	221.1	220.5
γ [deg]	25.00	25.00	25.00
α [deg]	11.95	10.27	9.30
ϕ [-]	0.98	0.90	0.48
used fuel mass [kg]	573	573	567

Table 7.13: Parameter values at the final obtained state of the take-off trajectory phase.

The final objective values of the different runs are very similar. None of the runs violates any trajectory constraints and the fuel-mass objective values are almost identical. A complete overview of the final states is given in Table 7.13.

It is noted that the vehicle overshoots the final velocity by almost 4% of the objective value. This is caused by the relatively large integration time step of 0.2 seconds. The lower fuel-mass consumption found with seed 3, can primarily be attributed to the relatively small overshoot in altitude and velocity of the trajectory. When the guidance time step is reduced, the final state will be closer to the objective. However, this will come at a cost of an increased computation time.

7.7.2. Atmospheric Acceleration

The atmospheric acceleration starts at the final state of the lift-off and accelerates the vehicle to a final velocity of 7,000 m/s at an altitude of 48 km. This state is again extracted from the reference trajectory found by Mooij (1998). It is noted that the initial state of the atmospheric acceleration for the current optimization does not match the final state summarized in Table 7.13 as was previously discussed, but the values for the selected initial mass, velocity and altitude are conservative.

The settings, boundary conditions and objectives for the atmospheric acceleration are defined in Table 7.14 for 3 different guidance settings. The cases match with the previously defined cases CN3.6, CN2.10 and TVC-S2, with the last case here having 10 instead of 8 control nodes. The results of the different cases are compared in Figure 7.25.

When focusing on the bottom plane of the figure, case CN3.6 shows a well distributed front. Improving the fuel-mass objective too much will push the solutions into the unfeasible space with high constraint objectives, located on the right side of this plane. When looking at the vertical planes,

parameter		start value	objective value	control specifications
m [kg]		135,563	-	-
V [m/s]		220	7,000	-
h [km]		350	48,000	-
γ [deg]		25	-	-
α [deg]	CN3.6	5.0	-	6 nodes, domain [0.0 3.5]
	CN2.10	5.0	-	10 nodes, domain [0.0 3.5]
	TVC-S2	5.0	-	10 nodes, domain [0.0 3.5]
ϕ [-]	CN3.6	1	-	6 nodes, domain [0.4 1.0]
	CN2.10	1	-	-
	TVC-S2	1	-	-
F_T [-]	CN3.6	-	-	-
	CN2.10	-	-	-
	TVC-S2	1	-	10 nodes, F_{TVC} = 0.5
objectives		f[1], f[2], f[4]		

Table 7.14: Definition settings, boundary conditions and objectives of the atmospheric acceleration flight phase.

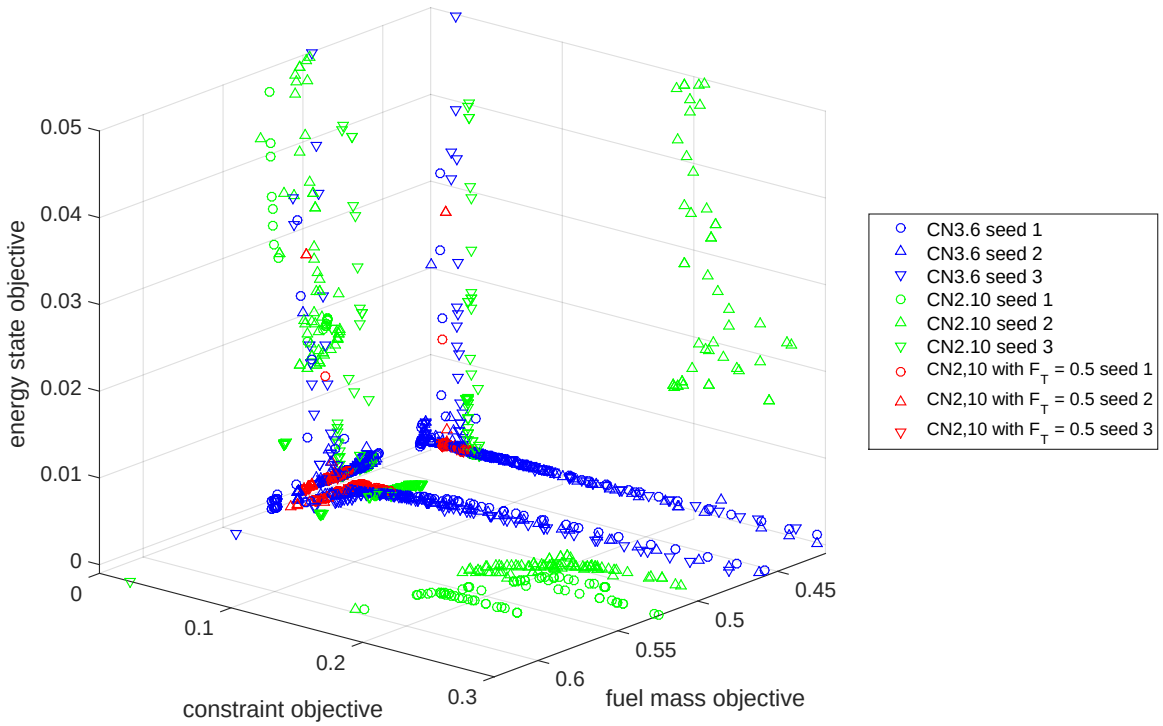


Figure 7.25: Objective function values of populations (n = 105 gen = 500) projected on planes for a comparison of the optimization cases applied to the atmospheric acceleration phase.

the majority of the blue individuals is found to have energy-state objective values very close to zero, indicating that the final objective state is reached.

A large part of the individuals of case CN2.10 has constraint objective values larger than 0.1 and fuel-mass objective values over 0.49. A small group of green points with better constraint and fuel-mass objective values can be found near the top of the bottom plane. However, a large part of this group has energy-state objective values higher than 0.01 and is therefore also not feasible. This indicates that this case struggles to find a feasible solution in general.

However, when a fixed TVC fraction is added to this case (TVC-S2), all the individuals are found on the top-left side of the bottom plane, indicating good constraint and fuel-mass objective values. Also the energy-state objective is very close to zero. The populations have also converged better as can be seen by the fact that the resulting individuals are located very closely together. The performance has thus improved considerably with this addition.

The optimizer easily finds solutions with a constraint objective value below 0.05, which indicates a maximum violation of 5% or a combination of smaller violations. Although this is not bad, it is still a violation. It seems as if the constraint objective is hard to satisfy completely.

It was deemed possible that the optimization method struggles to get the objective value to zero, more so, than that it was impossible to find a trajectory without constraint violations. In an attempt to remove the constraint violations from the trajectory, the maximum constraint values were decreased by 5%. This would mean that, for this new case, a resulting constraint objective value of 0.05 would not violate the original constraint values. A comparison for cases CN3.6 and TVC-S2 is shown in Figures 7.26 and 7.27, respectively.

When looking at the bottom plane of Figure 7.26, it can be seen that the complete front has moved towards a higher constraint objective value by an approximate value of 0.05. For case TVC-S2 shown in Figure 7.27, the constraint objective value remains below 0.05, meaning that the original constraint values are not violated. However, both the energy-state objective and the fuel-mass objective deteriorated significantly. A large part of the solutions now has energy-state objective values over 0.01, meaning the target state is not reached. Furthermore, the fuel-mass objective values are all above 0.47, meaning that not enough fuel will be left for the pull-up phase. Lowering the objective constraint values will thus not help the optimizer to find feasible trajectories that stay withing the original constraints. It

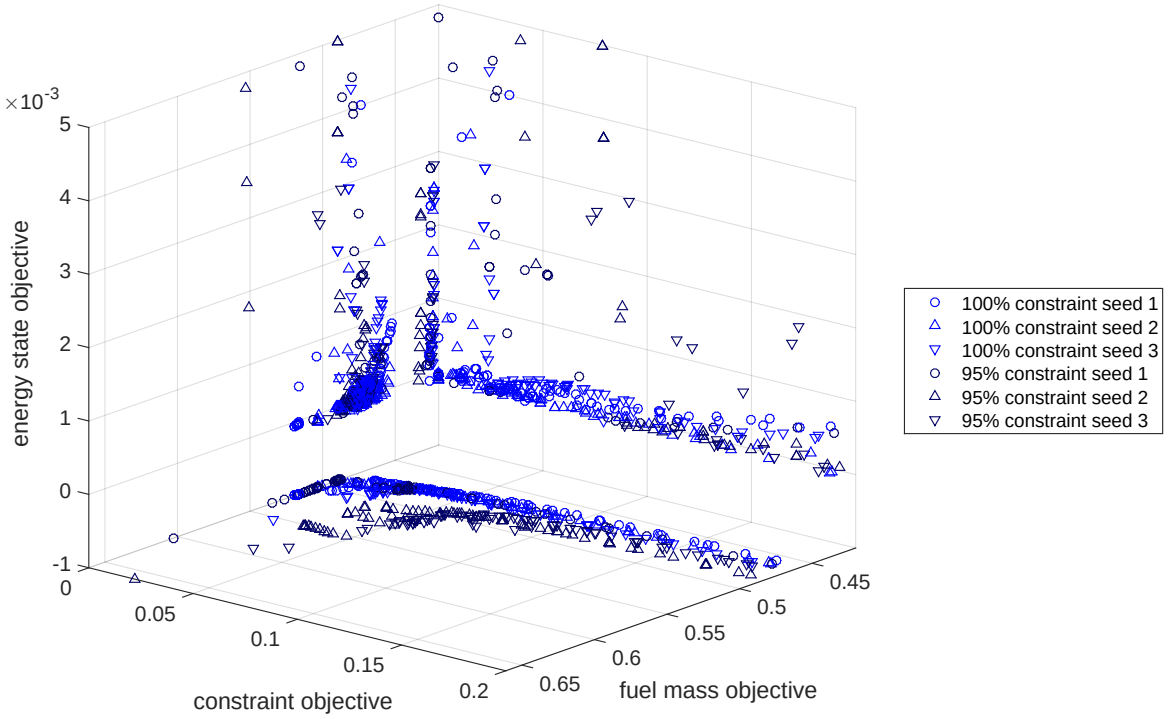


Figure 7.26: Objective function values of populations (n = 105 gen = 500) for a comparison of optimization result for atmospheric acceleration phase with 95% and 100% constraint values for case CN3.6.

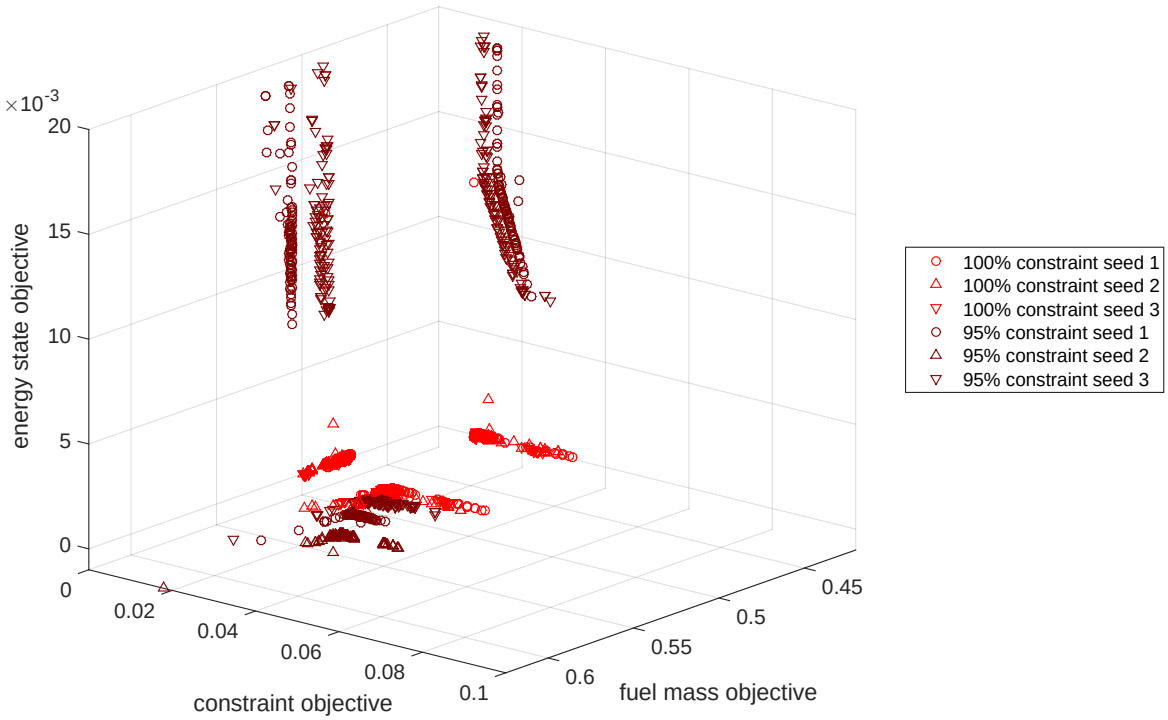


Figure 7.27: Objective function values of populations (n = 105 gen = 500) for a comparison of optimization result for atmospheric acceleration phase with 95% and 100% constraint values for case TVC-S2.

can therefore be concluded that the originally set constraints are indeed hard to satisfy.

The results obtained with TVC-S2 are shown in Figures 7.28 to 7.37 and will be discussed here. The results for CN3.6 are very similar and are further discussed in Appendix C.2. The trajectory found by TVC-S2 reaches the final state almost 300 seconds before the trajectories found by CN3.6. The difference can primarily be attributed to the larger acceleration throughout the first 300 seconds of the ascent as a result of the throttle-law implementation. Other than the duration of the flight, the results of both optimization cases show a similar behavior versus time.

The results found by both methods also show strong similarities with the result found by Mooij (1998) for the reference trajectory shown in Figure 6.17 to Figure 6.28. Figure 7.30 shows the control history of the angle of attack. Just like the reference, the angle first drops to around 2 degrees, before it increases to a maximum value of approximately 3 degrees. As the vehicle accelerates, the angle of attack slowly reduces to around 1.5 degrees. Figure 7.31 shows that the energy state almost increases linearly with time. The angle of attack versus time will therefore have a very similar pattern as shown in Figure 7.30.

Figure 7.29 shows that the flight-path angle also decreases fast during the first part of the trajectory. When the flight-path angle approaches zero, some small oscillation are visible. This results in the light phugoid motion visible by the oscillations in the trajectories shown in Figure 7.28. Even with the oscillations, the trajectories do track the constraints, as was found for the reference. Figures 7.34 and 7.36 show that the axial-acceleration and dynamic-pressure constraint are limiting during the first part of the trajectory, while the heat-rate constraint is limiting during the last part as shown in Figure 7.37. The similarity with the reference, indicates that the optimizer converges to the expected solution.

Figure 7.30 shows the control history of the angle of attack found by each run of case TVC-S2. Each run follows a similar line, but up to an energy-state value of 0.5, considerable differences can still be seen. This confirms that the trajectory optimization is quite sensitive to the initial seed for the current implementation.

The largest constraint violations are found for the axial accelerations. The PI controller implemented in the throttle law should minimize this constraint violation, but the inertia of the system causes a small overshoot to occur. Increasing the damping of the system should reduce this overshoot. Reducing all maximum allowable constraint values was tested before and not found to be effective. However, only reducing the maximum axial acceleration by about 2% could potentially delete the axial-acceleration error completely without the negative effects found before.

The final objective values of the different runs are very similar, with none of the results finding a solution without any constraint violations. A complete overview of the final state for all three runs of both CN3.6 and TVC-S2 is given in Table 7.15.

The objective final state was defined by a final velocity and altitude. Table 7.15 shows that the final state is very close to both objectives. The largest offset for the altitude is 0.094% of its objective value, while the largest offset of the velocity is only 0.004% of its objective value. It is noted that these results could be further improved by using a smaller integration time step, but this would result in an increased computation time. Also the final values of the variables m, γ, α and ϕ are in the same order of magnitude for both cases and all three initial seeds, indicating the consistency of the optimization results. The average mass of the vehicle after the acceleration phase is 71,006 kg. This is almost

	CN3.6			TVC-S2		
	seed 1	seed 2	seed 3	seed 1	seed 2	seed 3
m [kg]	71,524	70,918	71,034	71,048	70,837	70,678
V [m/s]	7,000	7,000	7,000	7,000	7,000	7,000
h [m]	47,973	47,985	47,999	47,999	47,955	47,997
γ [deg]	0.02	0.11	0.08	0.09	0.13	0.01
α [deg]	1.24	1.36	1.38	1.39	1.45	1.20
ϕ [-]	0.98	0.98	0.96	1.0	1.0	1.0
used fuel mass [kg]	64,039	64,645	64,529	64,515	64,726	64,885

Table 7.15: Parameter values at the final obtained state of the atmospheric acceleration trajectory phase.

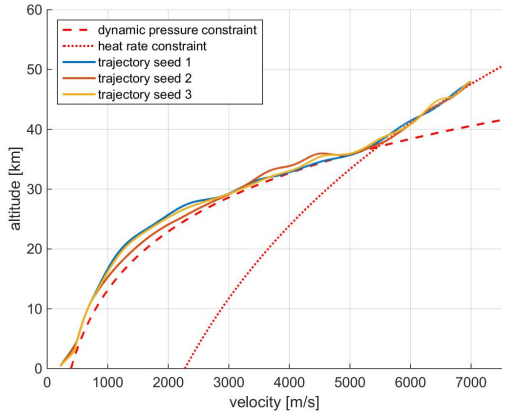


Figure 7.28: Final trajectory for atmospheric acceleration phase optimized with case TVC-S2.

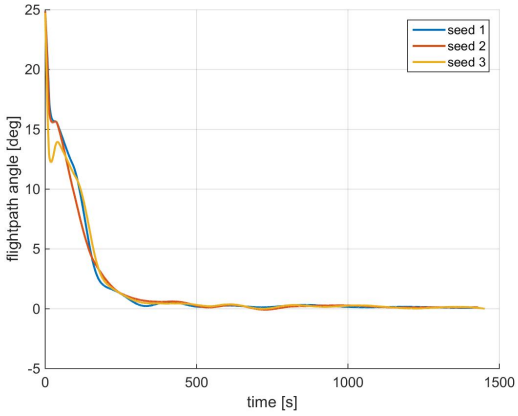


Figure 7.29: Flight-path angle versus time for atmospheric acceleration phase optimized with case TVC-S2.

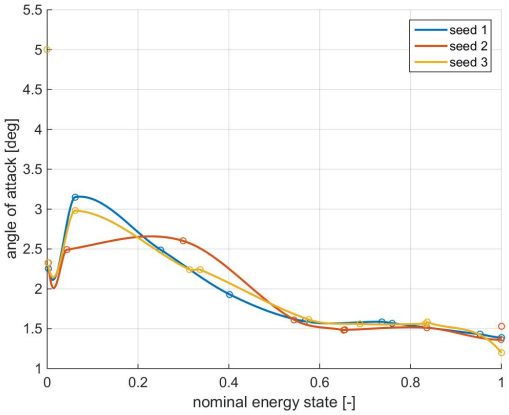


Figure 7.30: Control history of angle of attack versus the normalized energy state for acceleration phase optimized with case TVC-S2.

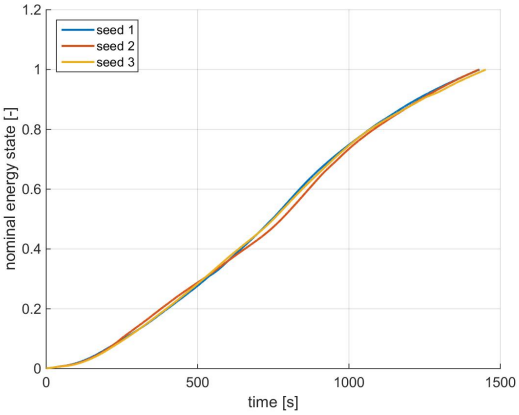


Figure 7.31: Energy state versus time for acceleration phase optimized with case TVC-S2.

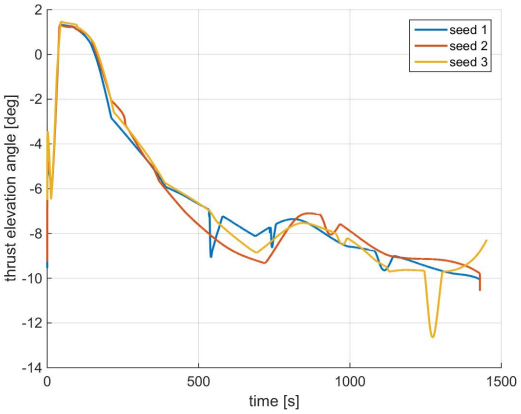


Figure 7.32: Thrust-elevation angle versus time for acceleration phase optimized with case TVC-S2.

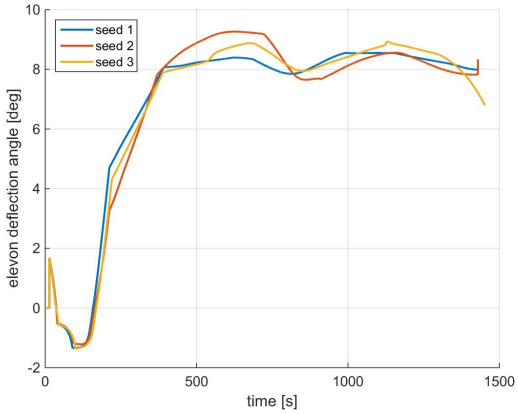


Figure 7.33: Elevon deflection versus time for acceleration phase optimized with case TVC-S2.

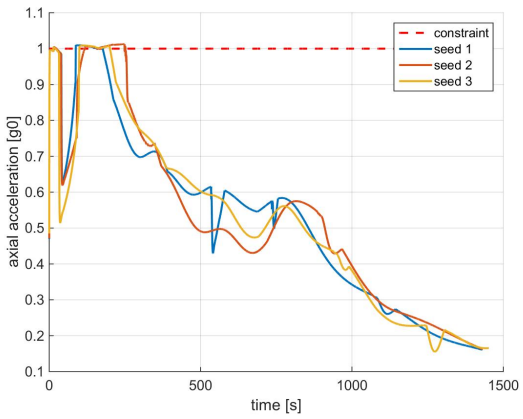


Figure 7.34: Axial acceleration versus time for acceleration phase optimized with case TVC-S2.

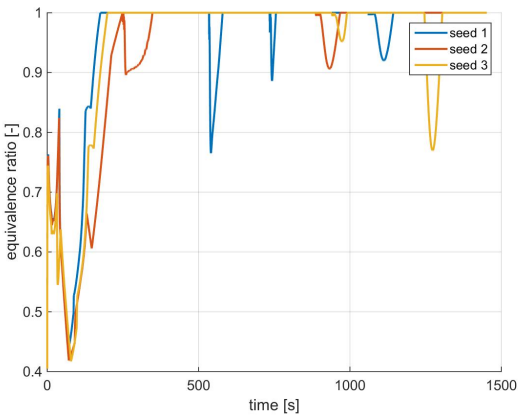


Figure 7.35: Equivalence ratio versus time for acceleration phase optimized with case TVC-S2.

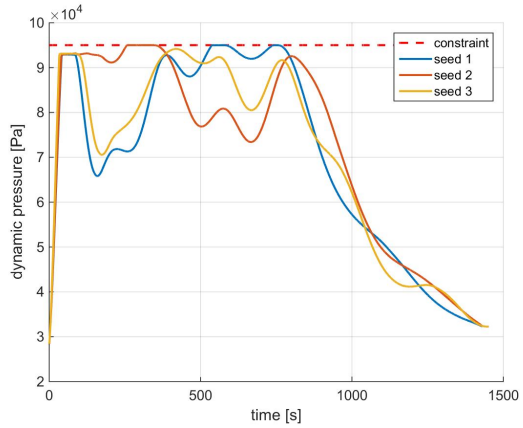


Figure 7.36: Dynamic pressure versus time for acceleration phase optimized with case TVC-S2.

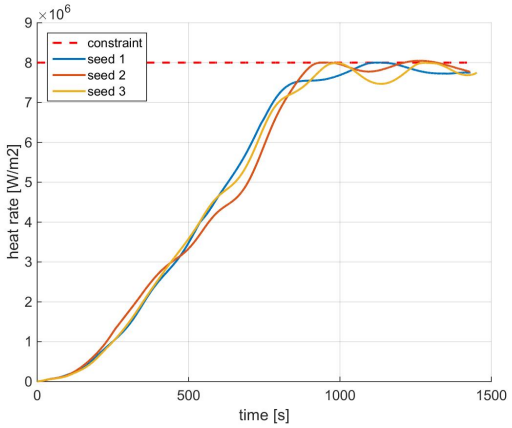


Figure 7.37: Heat rate versus time for acceleration phase optimized with case TVC-S2.

identical to the value of 71,150 kg found by Mooij (1998) for the reference trajectory. However, with the currently used simulation model the vehicle consumed 3,000 kg more fuel for the same reference trajectory, due to small discrepancies in the models. It can therefore be conclude that the optimizer has reduced the fuel consumption with respect to the reference trajectory.

7.7.3. Pull-up

The definition of the pull-up phase is also based on the reference trajectory found by Mooij (1998). The pull-up follows the atmospheric acceleration and its initial state should therefore match the final state of the acceleration phase. As explained previously, for the currently optimized phases this last requirement is not met. However, the initial state is chosen conservatively starting with a higher initial mass and lower initial energy state than any of final states found for the atmospheric acceleration phase summarized in Table 7.15.

The objective of the pull-up phase is to reach the altitude of ISS of 408 km at a corresponding orbital velocity. It is noted that the inertial circular velocity of 7,668 m/s was used, where the relative circular velocity of 7,175 m/s should have been used. The optimizer was not able to find a feasible solution for the lower objective velocity with the same optimization settings. Unfortunately the error was discovered very late in the process of writing this report, and not enough time was left to further study the effect of different settings. The old results with the wrong circular velocity will therefore be discussed here, as relevant observations can still be made. The different final velocity does have an effect on the resulting trajectory, the found payload mass and on the optimization performance. The section will therefore end with an extensive explanation of these effects.

During the pull-up the vehicle will ascend fast. At high altitudes the atmosphere is almost non-existing and the air-breathing engine produces no thrust. The optimizer therefore struggles to find a trajectory that reaches the objective velocity at the objective altitude, as no thrust can be given at the end of the trajectory. A final boost should therefore be given by a rocket engine. A penalty for the extra fuel mass required by the rocket engine is already included in the fuel-mass objective in the form of Tsiolkovsky's rocket equation as explained in Section 5.3.4.

However, the velocity dominates the altitude in the energy-state objective computation. When optimizing for the energy-state objective, the optimizer will find trajectories that do not reach the objective altitude but rather reach higher velocities to compensate for this loss in energy. For the take-off and atmospheric acceleration phases this problem can be solved by setting the objective velocity as a stop condition for the simulation, meaning the vehicle can never go faster than its objective velocity. This is allowed, because a monotonic increase in velocity is desired for these phases. For the pull-up phase this cannot be done, as the vehicle will have to accelerate to velocities higher than the objective velocity before exchanging this extra kinetic energy to potential energy when climbing to the objective altitude. The energy-state objective is therefore not effective for this flight phase. To resolve this, the energy-state objective is replaced by an altitude objective function:

$$f[7] = \left| \frac{h_{obj} - h_f}{h_{obj}} \right| \tag{7.2}$$

where h_{obj} represents the objective altitude and h_f the altitude at the final state of the found trajectory. The pull-up is optimized for case CN3.6. Table 7.16 summarizes the settings, boundary conditions and objectives used for optimization of the pull-up.

parameter	start value	objective value	control specifications
m [kg]	71,525	-	-
V [m/s]	7,000	7,668	-
h [m]	48,000	408,000	-
γ [deg]	0.1	-	-
α [deg]	1.4	-	6 nodes, domain [0.0 12.0]
ϕ [-]	1	-	6 nodes, domain [0.0 2.0]
objectives	f[2], f[4], f[7]		

Table 7.16: Definition settings, boundary conditions and objectives of the pull-up flight phase.

The results are visualized in Figures 7.38 to 7.45. Some interesting behaviors are found for this flight phase. The trajectory of the pull-up phase is shown in Figure 7.38 for all three runs. The trajectory of the different runs overlap perfectly. The vehicle first accelerates to around 8,025 m/s, following the heat-rate constraint, before performing the final pull-up. during this pull-up the extra kinetic energy gained is transformed into potential energy. The final velocity stocks at around 7,638 m/s which is 30 m/s short of the objective orbital velocity. This final velocity boost should be given by a rocket engine as explained earlier.

Figure 7.45 shows that the normalized energy state increases fast towards a value of 1, as long as the vehicle is accelerating. But during the actual pull-up, the normalized energy state drops. This progression of the energy state versus time explains why the objective velocity cannot be obtained. If the final energy state is not equal to 1, either the velocity or altitude objective is not reached. The velocity objective could potentially be reached, if the maximum energy state is not defined by the final energy at the objective state, but by a higher energy state. This would allow the vehicle to accelerate further, before performing the final pull-up.

The behavior of the control history can also be explained by the course of the energy state. Many control nodes are defined at a nominal energy state of 1 as visualized by the many points on the right side of Figures 7.40 and 7.42. This happens because the nominal energy state is close to 1 for the majority of the flight time. Furthermore, the fact that the nominal energy state is decreasing from

about 800 seconds into the flight onwards, results in the last part of the guidance logic being repeated in reverse for the remaining part of the flight. This causes some weird behaviors in the resulting angle of attack and the equivalence ratio versus time as shown in Figures 7.41 and 7.43 respectively.

The values of the points on the right side of Figures 7.40 and 7.42 seem very random and, in fact, they are. This is explained by the dynamic pressure dropping to zero as the vehicle exits the atmosphere as shown in Figure 7.44. With no dynamic pressure, both the angle of attack and the equivalence ratio have no effect and both parameters could be set to zero with effectively the same results. This was done for the reference trajectory given by Mooij (1998), where this segment of the trajectory was defined as a coasting phase. The large differences in the angle of attack and equivalence ratio between the three runs found for the exo-atmospheric phase therefore do not result in any differences in the flight-path angle of the final trajectory.

The final objective values of the different runs are very similar, with all of the results finding a solution without any constraint violations. A complete overview of the final state for all three runs is given in Table 7.17.

	seed 1	seed 2	seed 3
m [kg]	59,420	59,394	59,420
V [m/s]	7,638	7,638	7,638
h [m]	408,038	408,013	408,066
γ [deg]	2.51	2.50	2.50
α [deg]	0.72	0.69	0.72
ϕ [-]	1.65	1.64	1.73
used fuel mass [kg]	12,104	12,131	12,105

Table 7.17: Parameter values at the final obtained state of the pull-up trajectory phase.

As explained before, none of the runs reach the velocity objective. For the current solutions a rocket engine is required for the last boost. The altitude objective is reached with a small overshoot of less than 0.02%. However, currently the final flight-path angle is around 2.5 degrees, meaning the vehicle will still be ascending when the final state is reached. To remain at the final objective altitude, the final flight-path angle should be zero as was also stated in the mission requirements. Future runs should therefore include the flight-path angle objective f[6]. This addition is not expected to have a large influence on the found fuel consumption. Initial tests have indicated an additional fuel consumption of 300 kg, but the exact impact should be further investigated.

When the objective velocity is lowered to the relative circular velocity of 7,175 m/s, no feasible solutions are found by the optimizer. The verification, discussed in Section 6.4.2, already showed that a higher pull-up velocity has a positive effect on the maximum altitude that is reached. The maximum

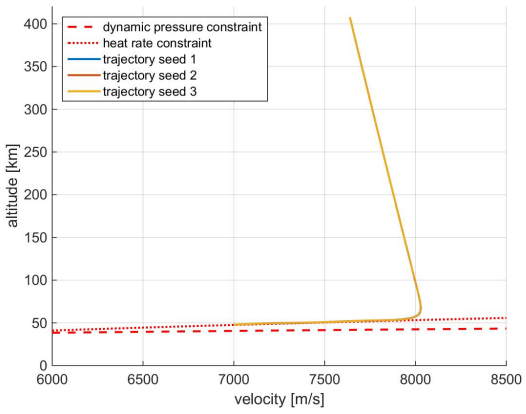


Figure 7.38: Final trajectory for pull-up phase optimized with case CN3.6.

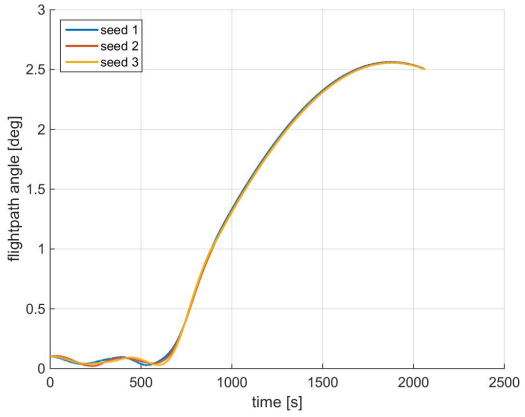


Figure 7.39: Flight-path angle versus time for pull-up phase optimized with case CN3.6.

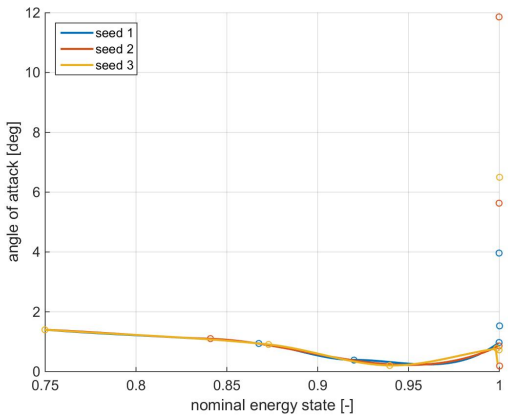


Figure 7.40: Control history of angle of attack versus the normalized energy state for pull-up phase optimized with case CN3.6.

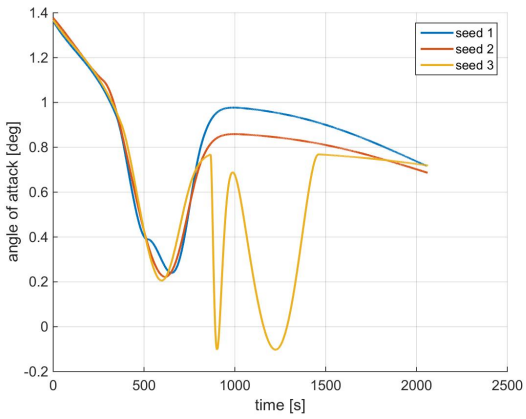


Figure 7.41: Angle of attack versus time for pull-up phase optimized with case CN3.6.

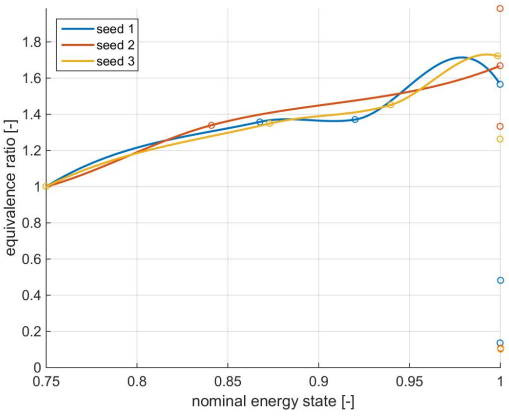


Figure 7.42: Control history of equivalence ratio versus the normalized energy state for pull-up phase optimized with case CN3.6.

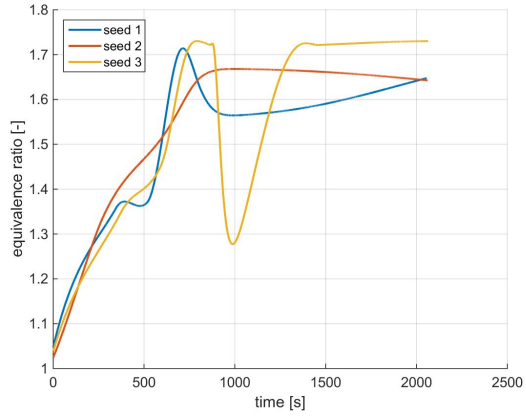


Figure 7.43: Equivalence ratio versus time for pull-up phase optimized with case CN3.6.

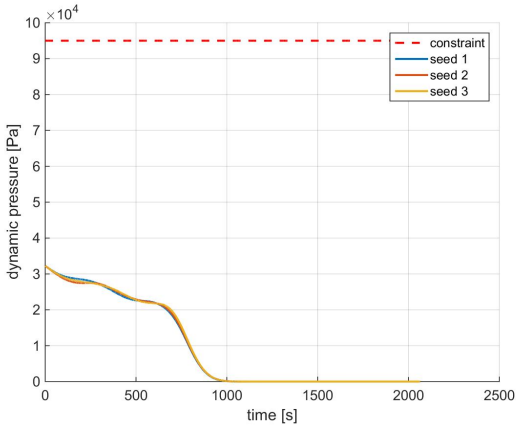


Figure 7.44: Dynamic pressure versus time for pull-up phase optimized with case CN3.6.

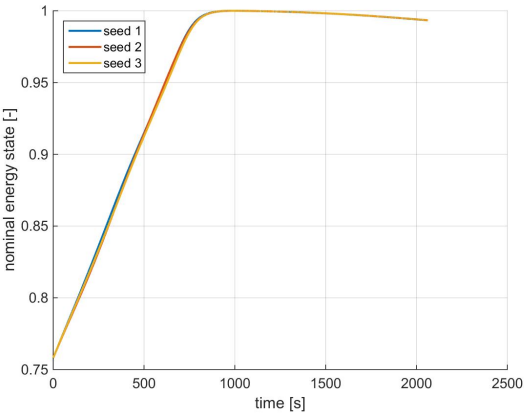


Figure 7.45: Energy state versus time for pull-up phase optimized with case CN3.6.

pull-up velocity is currently limited by the objective energy state, as the total energy can never exceed this value throughout the flight. If this limit is lifted, the pull-up manoeuvre will be easier.

Three reasons are identified to explain why higher velocities have a positive effect on the pull up. To begin with, the higher velocity results in a higher dynamic pressure during pull-up. More lift is therefore generated, meaning that the flight-path angle increases faster. On top of that, the vehicle now has more kinetic energy, which can be exchanged for potential energy during the coasting phase. The last reason is based on the local circular velocity. Figure 7.38 shows that the relative velocity is around 8,025 m/s at the start of the pull-up. This is almost 500 m/s more than the relative local circular velocity. The vehicle's orbit at that instant in time can be defined by an elliptical orbit, with the vehicle located close to perigee and on route to apogee. Following this orbital path will automatically result in an increasing flight-path angle, until apogee is reached. These effects combined make sure that an angle of attack of less than 1 degree is required for the pull-up to 408 km, as is shown in Figure 7.41. The verification, on the other hand, required an angle of attack of 10 degrees to achieve pull-up to just 120 km altitude.

The lower objective velocity results in more stringent requirements for the pull-up phase, which complicates the optimization problem. For the current definition of the pull-up phase, the optimizer was also not able to find a trajectory to 120 km with corresponding local circular velocity. However, the verification model showed that this is possible. To find a solution, the pull-up phase should be better defined. The final state of the acceleration phase can be moved to the actual pull-up state. This allows the user to specify the pull-up conditions. The domain of the optimization parameters can then be narrowed down for the pull-up phase. This should reduce the search space and with that, the complexity of the optimization problem. Unfortunately, this has to be left for future work due to a lack of time.

Although the lower objective velocity has complicated the optimization process, it will have a positive effect on the fuel consumption, and with that, the payload capacity. Because the new objective velocity is approximately 500 m/s lower, the pull-up could potentially be initiated earlier. During the acceleration from 7,600 m/s to 8,025 m/s, the vehicle uses 5,000 kg of fuel. If the pull-up can be performed at 7,600 m/s these savings in fuel can be used for payload. It is noted that the pull-up altitude will now be lower, while the mass is 5,000 kg higher. Both effects will cause the vehicle to decelerate more during the coasting phase. The final required circularization boost might therefore be larger, resulting in a higher fuel consumption. The gained payload capacity of 5,000 kg is therefore thought to be an upper limit.

7.7.4. Complete Mission

When all three flight phases are combined the final complete trajectory shown in Figure 7.46 is found. It is again noted that for the current solutions the initial conditions of one phase do not perfectly match the final conditions of the preceding phase. As a result, some discrete steps in the state and control parameters are present at the flight phase transitions. However, it is expected that this will only have a small effect on the found fuel-mass consumption.

Adding the fuel-mass consumption of the most efficient run from each flight phase together, a total fuel consumption of 76,710 kg is found. Given the initial mass of 136,079 kg of the vehicle, the mass at the final state is 59,369 kg. As the dry mass of the vehicle is 58,968 kg this means that a total of 401 kg is available for payload. The trajectory is thus feasible in terms of fuel consumption, although it is noted again that for this complete solution the axial-acceleration constraint was violated by about 1% of the objective value in the atmospheric acceleration phase.

The resulting 401 kg of payload mass available for this trajectory is a lot less than the 9146 kg found by Mooij (1998) after optimizing the reference trajectory. However, the reference trajectory only reaches a final altitude of 120 km. This accounts for a small part of the difference. When optimizing to 120 km a payload mass of 2,329 kg was found. For this run, the inertial circular velocity instead of the relative circular velocity, was also set as the objective velocity. The previous section showed that the discrepancies between these velocities could lead to a maximum additional payload capacity of 5,000 kg. Finally it is noted that the verification already found a lower payload mass than the reference trajectory, due to small differences between the simulation models. The lower payload mass can therefore not only be attributed to the performance of the optimizer.

Finally, it should be noted that the found trajectory is a pseudo optimal solution. Some oscillations can still be seen predominantly in the atmospheric acceleration phase. To improve the results a local

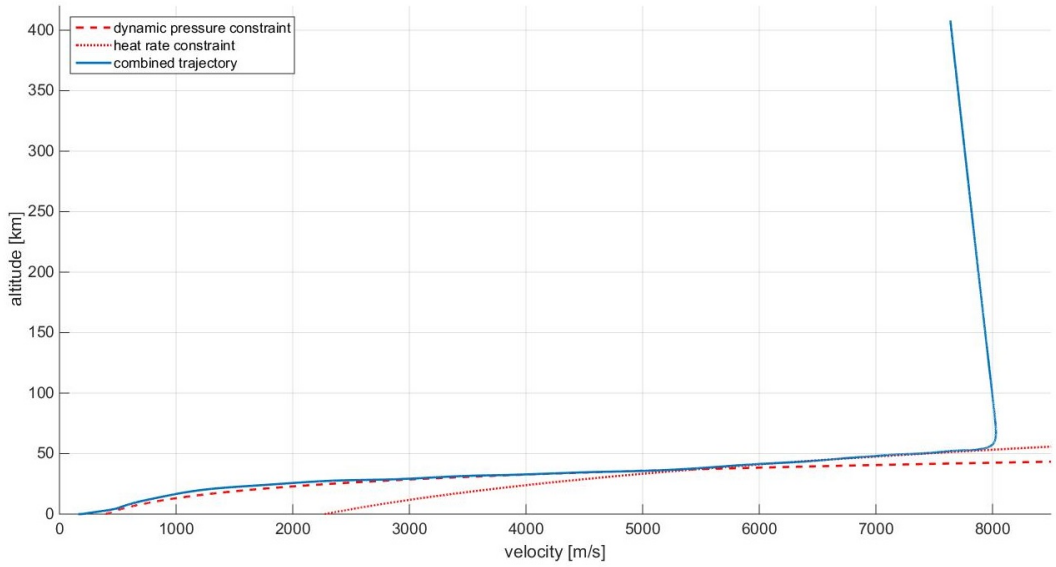


Figure 7.46: Final trajectories including take-off, atmospheric acceleration and pull-up.

optimization of the found pseudo optimal trajectory should be performed. This will be done as part of the sensitivity analysis in the next section.

7.8. Sensitivity

Throughout this research the sensitivity of the ascent trajectory optimization process to different parameters has been studied. In this section local optimization will be performed to assess the sensitivity of the resulting trajectory to the control history. Furthermore, the sensitivity to the the initial conditions evaluated. The sensitivity analysis is performed with respect to the solutions of the atmospheric acceleration for case CN3.6 with seed 1.

7.8.1. Control history

The sensitivity to the control history is analyzed separately for the angle of attack and the equivalence ratio. A Monte Carlo analysis is performed for both, allowing a maximum variation of the parameters at each control node of 0.1% of the respective domains. The resulting performance of 500 individuals with respect to the energy-state, constraint and fuel-mass objectives is shown in Figures 7.47 and 7.48 for the angle of attack and equivalence ratio, respectively.

The effect of both parameters is similar, although the resulting trajectory seems a bit more sensitive to the angle of attack than the equivalence ratio. The constraint objective is affected the most when changes are applied to the control history. A small change in the control history can already lead to a strong phugoid motion. This is illustrated by the oscillations visible in Figures 7.49 and 7.50, which show the trajectory and flight-path angle versus time, for the optimal trajectory and 5 random Monte Carlo individuals with varying angle of attack. It is noted that the frequency of the phugoid motion is the same for all solutions, with some solution having a phase shift of 180 degrees. This indicates that the results follow a characteristic motion of the trajectory.

The constant exchange of kinetic and potential energy has a very strong effect on the acceleration, dynamic pressure and heat rate. Specifically for the dynamic pressure and heat rate, constraint violation occur, as the amplitude of the oscillation is largest when the vehicle is close to these constraint values. Figures 7.51 and 7.52 show that larger oscillations will cause the dynamic-pressure constraint to be violated, whereas the heat-rate constraint is already violated for mild oscillations.

The fuel-mass objective seems to be affected less and for some individuals with higher constraint objective values the fuel-mass objective has even improved. This can be explained by the fact that the air-breathing engine performs better at a higher dynamic pressure. However, most trajectories with larger constraint violations do not reach the final state. This can be seen in Figure 7.49, where the stronger oscillating trajectories end up with a final velocity lower than the objective 7,000 m/s.

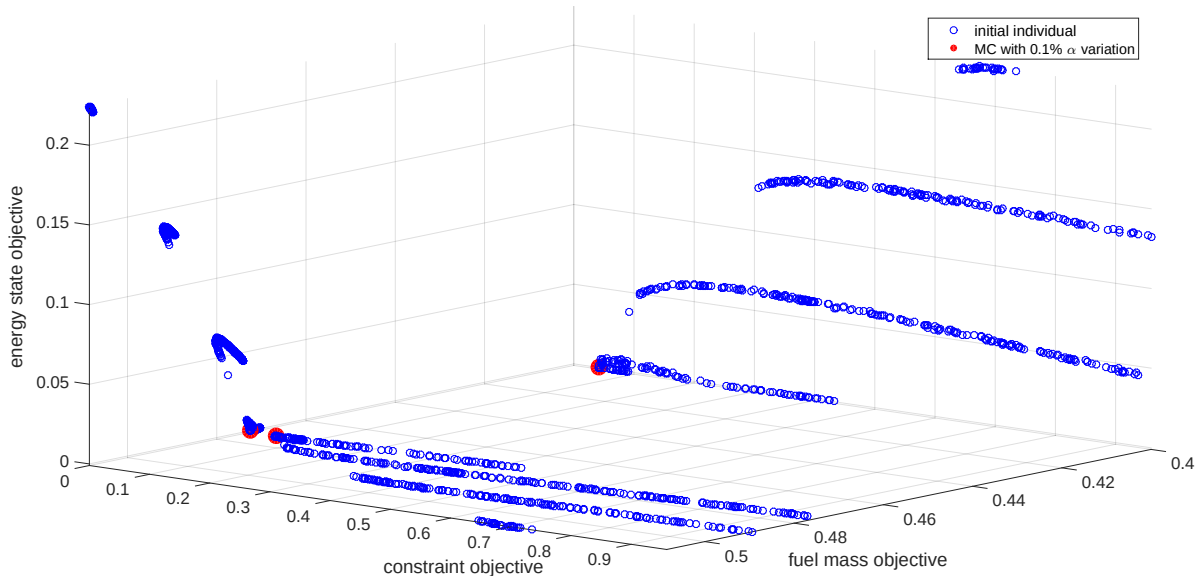


Figure 7.47: Constraints vs energy state vs fuel-mass objective for Monte Carlo analysis with maximum α variation of 0.1%.

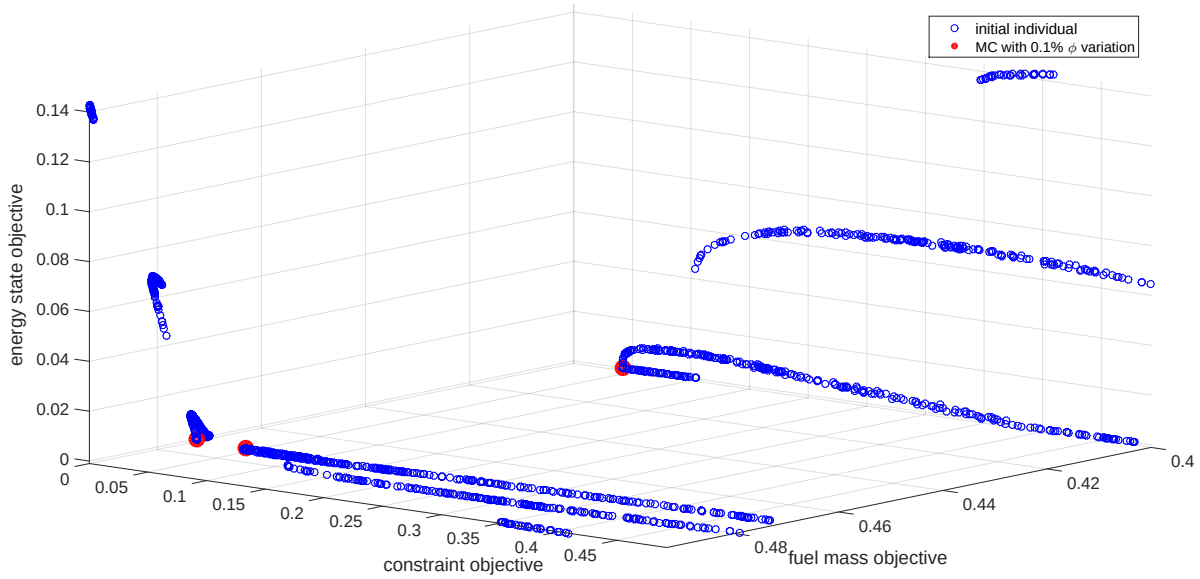


Figure 7.48: Constraints vs energy state vs fuel-mass objective for Monte Carlo analysis with maximum ϕ variation of 0.1%.

For those trajectories a virtual final boost is given by the less efficient rocket engine, resulting in a fuel-mass penalty and a higher final objective value. The fact that the trajectories do not reach the final state also explains the higher energy-state objective values.

The variation of ϕ led to almost identical trajectory results as shown in Figures 7.49 to 7.52. The same conclusion are drawn and the results are therefore not discussed separately.

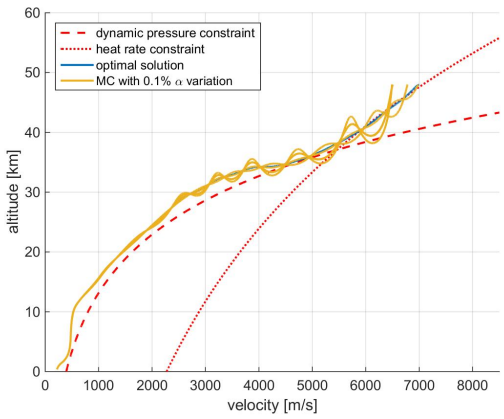


Figure 7.49: Final trajectory for atmospheric acceleration phase compared with 5 Monte Carlo results including a maximum α offset of 0.1%.

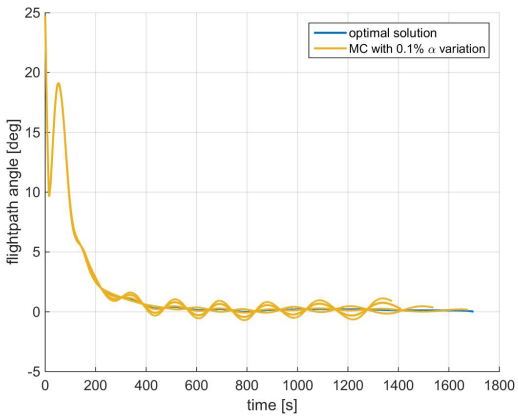


Figure 7.50: Flight-path angle versus time for atmospheric acceleration phase compared with 5 Monte Carlo results including a maximum α offset of 0.1%.

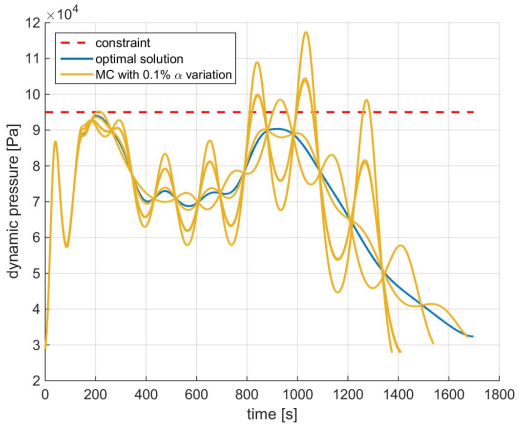


Figure 7.51: Dynamic pressure versus time for acceleration phase compared with 5 Monte Carlo results including a maximum α offset of 0.1%.

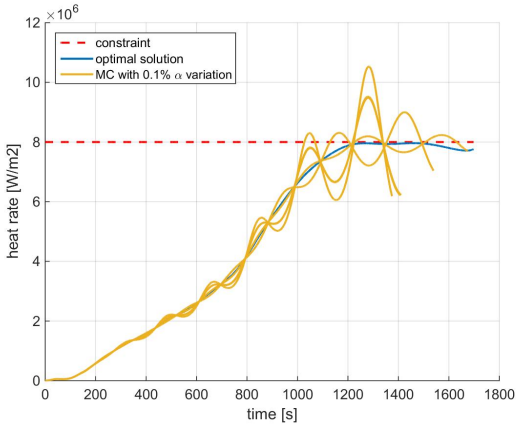


Figure 7.52: Heat rate versus time for acceleration phase compared with 5 Monte Carlo results including a maximum α offset of 0.1%.

7.8.2. Initial Conditions

To test the sensitivity of the trajectory to the initial state, the initial altitude, velocity and mass of the vehicle are varied for the optimal solution. A comparison of the final trajectory results is shown in Figures 7.53 to 7.55. Results of a number of flight parameters are also included for the mass sensitivity in Figures 7.56 to 7.62. These will be discussed first.

Figures 7.57 and 7.58 show that the change in the initial mass causes a mild oscillation of the angle of attack and the equivalence ratio versus time. The elevon deflection angle required for trim depends on both these parameters and shows a similar oscillation, as is visible in Figure 7.59. The small fluctuations in the control history result in a much stronger oscillation of the flight-path angle, shown in Figure 7.56. As a result, a strong phugoid motion is present in the trajectories. Figure 7.55 shows that this motion cause the trajectory to cross the dynamic-pressure and heat-rate constraint on multiple occasions. Figures 7.60 to 7.62 show the time histories for the flight-path constraints. Significant violations are found for all constraints.

Figures 7.53 to 7.55 show that the trajectory is not very sensitive to the initial altitude, while a strong sensitivity with respect to the initial velocity and mass is found. It is observed that for all three parameters, a negative disturbance results in a 180° phase shift of the response, with respect to a positive disturbance. This causes a significant difference in the final velocity that is reached.

The results show that the ascent trajectory is very susceptible to its characteristic motion. Although

the current guidance can largely remove the phugoid motion from the trajectory. Small changes to the control history or the initial state will cause the trajectory to fall back to its characteristic motion. To resolve this sensitivity the guidance should be improved. Instead of using the angle of attack as guidance parameter, the flight-path angle rate could be used as an active guidance parameter. Defining the flight-path angle rate for the whole trajectory would prevent oscillations in the flight-path angle which in turn removes the phugoid motion from the trajectory.

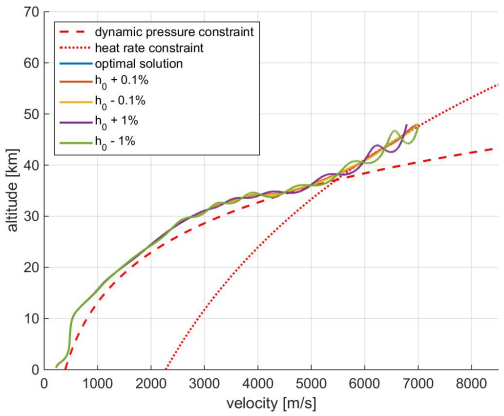


Figure 7.53: Final trajectory for atmospheric acceleration phase compared with variations of initial altitude.

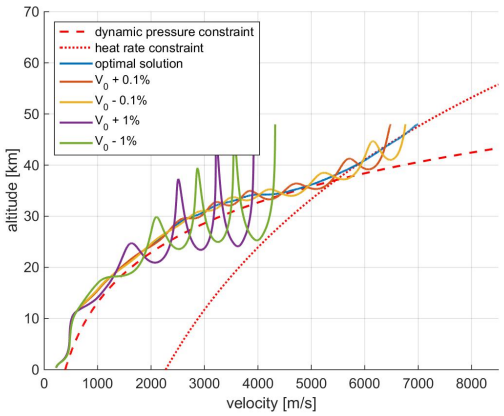


Figure 7.54: Final trajectory for atmospheric acceleration phase compared with variations of initial velocity.

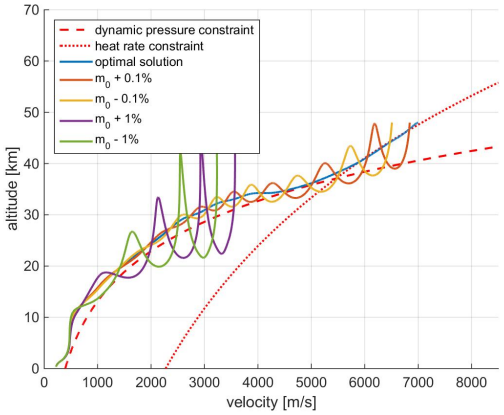


Figure 7.55: Final trajectory for atmospheric acceleration phase compared with variations of initial mass.

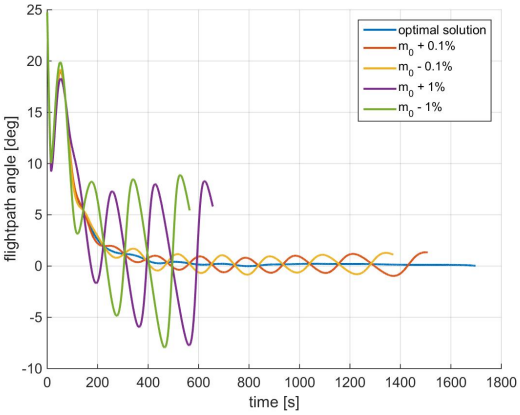


Figure 7.56: Flight-path angle versus time for atmospheric acceleration phase compared with variations of initial mass.

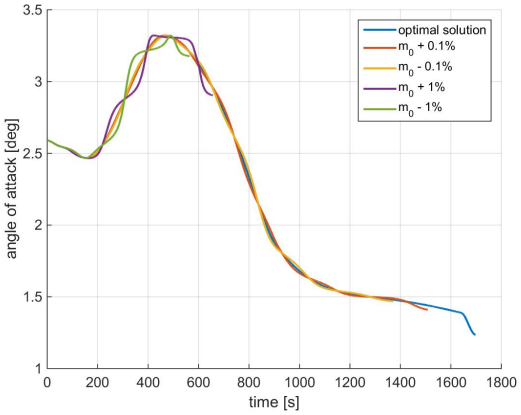


Figure 7.57: Angle of attack versus time for acceleration phase compared with variations of initial mass.

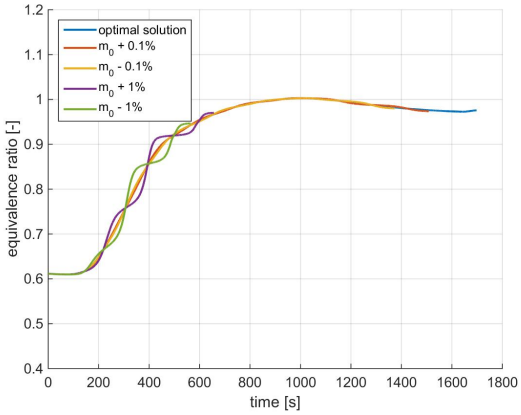


Figure 7.58: Equivalence ratio versus time for acceleration phase compared compared with variations of initial mass.

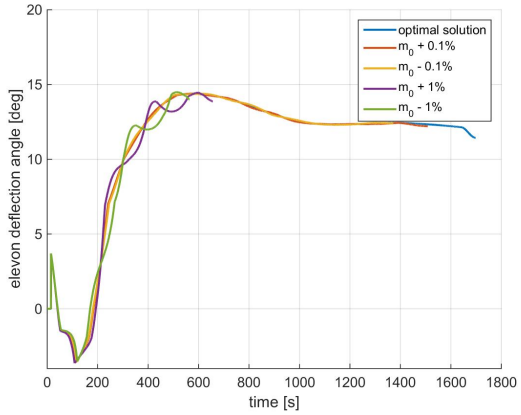


Figure 7.59: Elevon deflection versus time for acceleration phase compared with variations of initial mass.

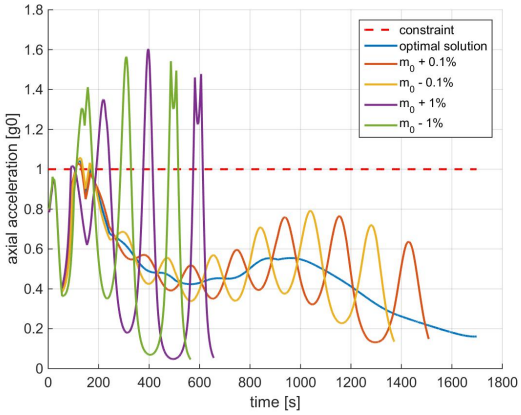


Figure 7.60: Axial acceleration versus time for acceleration phase compared with variations of initial mass.

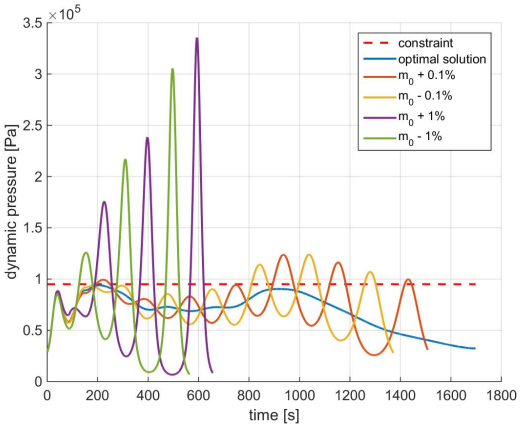


Figure 7.61: Dynamic pressure versus time for acceleration phase compared with variations of initial mass.

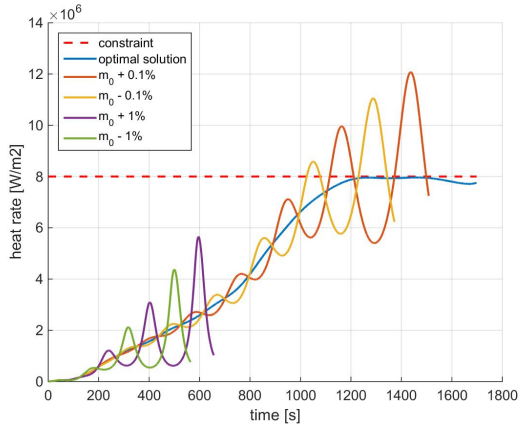


Figure 7.62: Heat rate versus time for acceleration phase compared compared with variations of initial mass.

8

Conclusions and Recommendations

In this chapter the conclusions and recommendations will be discussed. Based on the conclusions answers to the main question and its sub-questions will be formed. The main question was formulated as follows:

> *How can an ascent trajectory of an HTHL SSTO launch vehicle to ISS be found that maximizes the payload capacity for a given set of flight-path and control constraints?*

MO global optimizers were identified as a promising method to find an optimal solution to the ascent trajectory optimization problem. Many global optimization methods exist and both the ascent trajectory problem and the optimization approach can be defined in numerous ways. The objective of this research was to find a general good method and approach to optimize the ascent trajectory. Based on this objective the following set of sub-questions was defined for this research.

SQ-1 What is the impact of different control parameters on the optimization process of the ascent trajectory?
SQ-2 Which MO global optimization method is most effective for the ascent trajectory optimization problem?
SQ-3 What is the best approach, in terms of handling constraints, objectives and different flight phases, to optimize the ascent trajectory of the WCC from take-off to circularized orbit?

The final conclusions are given in Section 8.1 and are categorized based on the order of the sub-questions. The final conclusion will give a general answer to the main question. Section 8.2 will finally give some recommendations for future research derived from the conclusions.

8.1. Conclusions

The complete set of conclusions is divided into 5 parts of which the first 4 find answers to the sub-questions. The last part answers the main question.

Control Parameters

The total dimension of the optimization problem depends on the number of control parameters that is optimized and the number of control nodes at which each control parameter is defined. If more optimization parameters are used, the number of control nodes should be limited to maintain the same problem dimension. When fewer optimization parameters are used, the number of control nodes can be increased.

There exists a limit for the problem dimension at which an optimizer can still effectively find an optimal solution. Above this limit, good solutions can still be found, but the optimization process is more likely to get stuck in local optima. The exact limit of the problem dimension is hard to define and problem specific. It depends strongly on the boundary conditions set for each optimization parameter.

A large domain of an optimization parameter extends the search space for the optimizer, while limiting the domain of the optimization parameters also limits the search space. Setting more confined boundary conditions for the optimization parameters therefore allows for a larger problem dimension. The problem dimension is also linked with the population size, but this was not further investigated.

The use of different control parameters for the thrust did not strongly affect the standard trajectory optimization. Using the equivalence ratio, throttle factor or the throttle law all produced similar results. However, thrust is not part of the optimization when the throttle law is used thereby reducing the number of control parameters. The reduction in the resulting problem dimension allowed for a larger number of control nodes, or for the inclusion of TVC as an extra optimization parameter, without oversizing the problem dimension.

TVC was found to have a positive effect on the optimization process. Feasible trajectories were found more easily when TVC was included resulting in faster convergence to good solutions. Specifically the violations of flight-path constraints were less severe.

Optimization Methods

The ascent trajectory problem was optimized with four methods: MOEA/D, NSGA-II, NSGA-II-tabu and NSGA-II-tabu-reintro, where default settings were used for all methods. Some initial tests were also performed with a parallel implementation of MOEA/D and NSGA-II-tabu. Of the optimization methods compared in this research, MOEA/D performed best in terms of consistency of the results and convergence speed.

The NSGA-II method showed slower convergence than MOEA/D and the populations were sensitive to getting trapped in local optima. Implementation of the tabu-list resulted in better diversity of the populations which should prevent the optimization from getting stuck in local optima. However, the convergence speed of both methods including a tabu-list was so slow, that neither method was found competitive for the currently available computational power. Furthermore, including a tabu-list will increase the chance of converging to no-flight solutions, because initial good solutions are separated from the evolving population and not able to reproduce.

Initial tests with a parallel set-up of MOEA/D and NSGA-II-tabu showed a convergence speed comparable to MOEA/D while maintaining the diversity that was found with the NSGA-II-tabu method. Although the method was not properly tested, the results were promising for future research.

Objectives

The ascent trajectory is optimized with respect to a set of feasibility and optimality objectives. The feasibility objectives aim at reaching the objective state without violating the flight-path constraints, while the optimality objectives optimize the payload by reducing the fuel mass and the heat load. The different objectives are conflicting for the ascent trajectory leading to a number of complications. The way that each objective is defined is therefore very important.

At the start of each optimization, random individuals are created to fill the population. Each individual represents a guidance, but most of these initial guidance laws cause the vehicle to crash within a few seconds. These no-flight solutions are undesirable as they do not reach the objective state. However, not flying at all results in a trajectory without constraint violations, with no fuel consumption, and a low heat load. The initial population therefore tends to converge towards no-flight solutions. Including a no-flight penalty to the constraint objective will stimulate solutions with better energy states to reproduce. This penalty function has increased the convergence speed to feasible solutions, while also maintaining more diversity in the population.

Due to the conflicting objectives, optimizing the feasibility and optimality objectives in consecutive stages is not effective, as both sets of objectives push the results in a different direction. The final stage of the optimization should therefore include all objectives that need to be optimized. An initial optimization stage can be implemented to optimize for just the feasibility objectives first, but this was not found to be beneficial for the current implementation. Defining a separate function for each objective was found to be the most robust as the performance of objective functions that combine multiple objectives are very dependent on the weights that are applied to each objective.

Flight Phases and Final Trajectory

Optimizing the complete ascent trajectory from take-off to an altitude of the ISS at orbital velocity was found to be impossible within a reasonable amount of time for the current implementation. The trajectory was therefore split in three phases: take-off, atmospheric acceleration and pull-up. Separating

the different flight phases allowed for more confined domains of the optimization parameters for each phase. This resulted in a considerably reduced search space for each of the problems.

For each of the phases different objectives were used. The take-off requires a steep ascent. If the flight-path angle is not high enough at the end of the take-off, the optimizer will not be able to find a solution for the atmospheric acceleration phase. A flight-path angle objective should therefore be included for the take-off phase.

For the pull-up phase the energy-state objective is replaced by an altitude objective. The vehicle will accelerate to velocities higher than the final objective velocity. The velocity can therefore not be included as a stop condition in the simulation. Without this stop condition the energy-state objective will push the vehicle to accelerate more to reach its final energy state rather than ascend.

The MOEA/D optimizer has found solutions for the first two flight phases. The take-off complied to all constraints. For the atmospheric acceleration phase the optimizer struggled to find a trajectory without constraint violations. The best results were found when the throttle law was used and 50% of the trim was managed by TVC. The optimized trajectory consumed 3,000 kg less fuel up to the end of the atmospheric acceleration than the verification run, which followed the reference guidance. This shows that the optimizer has improved the trajectory. For this setup only the axial-acceleration constraint was violated by just 1% of its objective value, but it is expected that this violation can be eliminated with some improvements to the throttle law.

For the pull-up phase the inertial circular velocity of 7,668 m/s was set for the objective state, where the relative velocity of 7,175 m/s should have been used. A higher velocity was generally found to facilitate an easier pull-up manoeuvre. As a result, the optimizer did find a feasible solution when the inertial velocity was included as the final objective. However, when the objective velocity was reduced to the correct relative local circular velocity, the optimizer was no longer able to find a solution. The optimizer also did not find a feasible trajectory to an altitude of 120 km with the current implementation of the pull-up phase, even though the verification run indicated that a trajectory to this altitude can be found with the current simulation model. It is expected that a better definition of the pull-up conditions will help the optimizer to find a solution. As not enough time was currently available to further investigate this problem, it will be included in the recommendation for future work discussed in the next section.

When the solutions of the separate phases are combined, the complete trajectory is found. Based on the best currently found pseudo optimal solution, the WCC should be able to bring a payload mass of 401 kg to an altitude of 408 km with a velocity of 7,668 m/s. This payload capacity is much lower than the approximately 9,000 kg found by Powell et al. (1991) and Mooij (1998), who optimized to a final velocity of 7,358 m/s at 120 km. Although the optimizer is currently still not able to find a solution to this objective state, it is estimated that a maximum gain of 7,000 kg could be obtained as a result of the lower final velocity and altitude. It is also noted that the verification of the current simulation model showed a higher fuel consumption to fly the reference trajectory, than was found by Mooij (1998) for his simulation. A difference of 3,000 kg was found up to the moment of pull-up. The difference in the resulting payload mass is therefore partly attributed to discrepancies between the simulation models. Although it cannot be concluded from the results that global optimization will find better trajectories than those found with local optimization in the literature, combining all the effects, results in a theoretically similar possible result as found by Powell et al. (1991) and Mooij (1998). To confirm this, optimization of the pull-up phase should be further improved.

Finally it is noted that the resulting trajectories are very sensitive to small changes in the initial state or the control history. Small disturbances will immediately cause the vehicle to follow its characteristic motion. As the optimal trajectory closely follows the flight-path constraints, this characteristic phugoid motion will result in severe constraint violations.

Final Research Conclusion

With all the previously stated conclusions an answer to the main research question can be formulated:

The ascent trajectory of an HTHL SSTO launch vehicle to ISS can be optimized with the MO global optimization method MOEA/D. Including objectives for the final state, the flight-path constraints and the fuel-mass will lead the optimization to a feasible pseudo optimal trajectory that maximizes the payload capacity. Compliance to the control constraints is achieved by setting proper boundary conditions to the optimization parameters.

Although the global optimizer has found a solution to the ascent trajectory problem when objective

velocity was set approximately 500 m/s higher than the local circular velocity, no solutions were found when the actual circular velocity was used for the final state. It is also noted that the resulting solution is not necessarily the optimal solution. The MOEA/D method is based on heuristic reasoning and can therefore not be proven to find the actual absolute optimum. Additional local optimization should therefore be performed to improve the final result.

Using a global optimization method based on heuristics was initially thought to be advantageous because these methods can handle problems as a black box. However, the results have shown that with the current implementation and computational power, the ascent trajectory has to be split in phases to be able to optimize the trajectory effectively. The error made for the set objective velocity of the pull-up phase also indicated that the results are very sensitive to the definition of the flight phases. This requires a lot of insight in the ascent trajectory problem neutralizing the main advantage of the global optimization methods.

Although the currently implemented global optimization of the HTHL SSTO ascent trajectory is not as effective as hoped for, it does show promise for the future. An improved guidance based on controlling the flight-path angle rate could help eliminate the phugoid motion. This would allow the optimization to find feasible trajectories without flight-path constraint violations more easily. Furthermore the parallel implementation of multiple optimization methods showed a promising performance in terms of both the diversity of the population and convergence speed. Combined with the ever more increasing computational power it is expected that an MO global optimization method should be able to find a good pseudo optimal solution for the complete ascent trajectory problem effectively in the near future.

8.2. Recommendations

A number of recommendations for future research have already been mentioned in the previous section. This section will discuss these recommendations more extensively. Each of the recommendations given, is aimed at improving the performance of the optimizer for the ascent trajectory optimization problem studied in this research.

Optimization Methods

The currently investigated methods have been compared for default settings. Each method defines its own settings that influence the performance in terms of diversity of the population and convergence speed. As MOEA/D was found to be the most promising method studied in this research, it is recommended to further investigate different combinations of settings for this method in order to further improve its performance.

Initial tests of a parallel implementation of MOEA/D and NSGA-II-tabu showed good results in terms of both the diversity of the population and convergence speed. A parallel implementation can benefit from the individual strengths of each method. Further investigation of different combinations of methods that combine good diversity of the population and fast convergence could therefore lead to methods that are more effective for the ascent trajectory problem.

Objectives and Optimization Approach

The objectives defined for the current ascent trajectory optimization problem are conflicting. Because each objective pushes the population in a different direction the convergence is slow. Furthermore, it limits the flexibility of the optimization approach as all objectives have to be optimized at the same time in order to keep the population within a feasible area for all objectives. It is therefore interesting to see if penalty functions can be applied to the objective functions in such a way that mutual conflicts of interest are eliminated.

Defining the objective function in such a way could increase the performance of the optimization as it is currently implemented. On top of that it opens the door to a number of different optimization approaches. Hänninen (2009) proposed a distributed approach that optimizes the feasibility and optimality objectives separately in two populations that are optimized in parallel. This worked well for the re-entry trajectory problem, but not for the ascent problem as both sets of objectives are too conflicting. If this conflict of interest in the objectives can be eliminated, this would be an interesting approach to further investigate.

Guidance Parameters
The currently found trajectories are very sensitive to disturbances in both the initial state and the control history. Small changes will result in a strong phugoid motion. Although this motion is thought to be the characteristic motion of the trajectory, the computed eigen frequency does not match the frequency found in the results. Further research on the vehicle and its response should be performed to find the actual cause of the strong oscillatory motion.

It is clear that the current guidance implementation has a hard time to find trajectories without the previously mentioned phugoid motion. The optimization relies on the idea the individuals with similar parameters have a similar performance. The fact that small changes in the control history can lead to strong oscillations and very different trajectory results, therefore has a negative influence on the performance of the optimization.

Defining the aerodynamic guidance with the flight-path angle rate instead of the angle of attack should eliminate the oscillation from the flight-path angle. It is expected that the resulting guidance laws will find smoother trajectories more easily. Furthermore, the resulting trajectories are thought to be less sensitive to disturbances in the control parameters, which should positively influence the performance of the optimization.

Previous research has found that the implementation of TVC reduces the fuel consumption. In this research the implementation of TVC was found to have a positive influence on the optimization performance. However, in the current research TVC was only given limited freedom as a guidance parameter. As TVC might be more effective on certain parts of the trajectory than others, it would be interesting to further study the implementation of TVC as a free guidance parameter.

Flight Phase Boundaries
The complete ascent trajectory was split into three flight phases. With the current implementation both the state and the control histories do not seamlessly transition from one flight phase to the next. To achieve this, a number of improvements has to be made to the existing simulation.

A smooth transition of the state can only be obtained when the successive flight phases are optimized consecutively. To obtain a smooth transition of the control history, each guidance parameter should be defined at both the initial and final control node. Furthermore, the derivative should be defined for each guidance parameter at the flight phase boundaries. Although a fixed final node can already be included in the current simulation, the fixed derivative values should still be implemented.

The location of the flight-phase boundaries are currently extracted from the reference trajectory. However, splitting the trajectory in phases, forces the vehicle to exactly pass the boundary conditions of each state. It is therefore reasonable to assume that the location of these flight-phase boundaries has a significant impact on the final result. This effect should be further investigated.

Pull-up Phase
The optimizer is currently not able to find solutions for the pull-up phase when the relative local circular velocity is set as the objective velocity. Further research in the optimization of this phase is therefore required. As noted before, the location of the flight-phase boundaries will have an impact on the results. The final state of the acceleration phase can be moved to the actual pull-up state. This allows the user to specify the pull-up conditions. The domain of the optimization parameters can then be narrowed down for the pull-up phase. This should reduce the search space and with that, the complexity of the optimization problem.

Extended Mission Complexity
In this research a simplified ascent mission was optimized. Although the current implementation already struggled with this mission, with the recommendations given in this section the performance of the optimizer could be improved significantly in the near future. When an improved performance is found, the complexity of the mission can be extended to study the effects of:

- Including the bank angle as a control parameter, resulting in an ascent trajectory not limited to a vertical plane.
- Including different initial inclinations and heading angles.
- Including different inclinations to the final state.

A

Eigenmotion of the Reference Trajectory

The open-loop behavior of the vehicle gives some insight in the sensitivity of the trajectory to disturbances in the current state. To find the open-loop behavior, the equations of motion are written in state-space form, which results in the following matrix notation:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \tag{A.1}$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ represent state and control coefficient matrices and $\boldsymbol{x}$ and $\boldsymbol{u}$ represent the state and control vector, with:

$$\boldsymbol{x} = \begin{pmatrix} \Delta V \\ \Delta\gamma \\ \Delta r \end{pmatrix} \quad \text{and} \quad \boldsymbol{u} = \begin{pmatrix} \Delta\alpha \\ \Delta\phi \end{pmatrix}$$

The open-loop behavior of the ascending vehicle can be described by the eigenvalues and corresponding eigenvectors of the A-matrix. To find the eigenvalues, the non-linear set of equations of motion will have to be linearized. The EOM in spherical components and the rotational frame are derived by Mooij (1994). For the mission defined in Section 2.4 some simplifications can be made. The vehicle will fly along the Equator, moving only in the equatorial plane. For this 2-dimensional motion the following variables will stay constant during the trajectory:

$$\chi = 90^\circ \quad \text{and} \quad \delta = 0^\circ$$

Furthermore, to stay in this vertical plane along the Equator, the vehicle should not bank or go into a side-slip, because that would result in forces outside the vertical plane. This means that:

$$\beta = 0^\circ, \qquad \sigma = 0^\circ \quad \text{and} \quad \psi_T = 0^\circ$$

To be able to linearize the EOM some additional simplification are made:

- The Earth is non-rotating ($\omega_{cb} = 0$).
- Thrust vector control is not used (ϵ_T).

Including these simplification in the equations of motion results in the following set of equations:

$$\dot{V} = \frac{-D + T\cos\alpha}{m} + g\sin\gamma \tag{A.2}$$

$$\dot{\gamma} = \frac{L + T\sin\alpha}{mV} + \left(\frac{V}{r} - \frac{g}{V}\right)\cos\gamma \tag{A.3}$$

$$\dot{r} = V\sin\gamma \tag{A.4}$$

The equations of motion have been linearized using the method described in Mooij (1998, Chapter 6), with the addition of thrust. This results in the following set of equations:

$$\Delta\dot{V} = -g_0\cos\gamma_0\Delta\gamma + 2\frac{g_0}{r_0}\sin\gamma_0\Delta r - \frac{1}{m_0}\Delta D + \frac{\cos\alpha_0}{m_0}\Delta T + \frac{D - T\cos\alpha}{m_0^2}\Delta m \tag{A.5}$$

$$\begin{aligned}\Delta\dot{\gamma} = &\left(-\dot{\gamma} + \frac{2V_0}{r_0}\cos\gamma_0\right)\frac{\Delta V}{V_0} + \left(\frac{2g_0}{r_0} - \frac{V_0^2}{r_0^2}\right)\frac{\cos\gamma_0}{V_0}\Delta r - \left(\frac{V_0}{r_0} - \frac{g_0}{V_0}\right)\sin\gamma_0\Delta\gamma + \\ &\frac{1}{m_0V_0}\Delta L + \frac{\sin\alpha_0}{m_0}\Delta T - \frac{L + T\sin\alpha}{m_0^2V}\Delta m\end{aligned} \tag{A.6}$$

$$\Delta\dot{r} = \sin\gamma_0\Delta V + V_0\cos\gamma_0\Delta\gamma \tag{A.7}$$

where $\Delta m = 0$, as it only varies with time and not as a function of the state variables alone. ΔD, and ΔL can be found following the method described by (Mooij, 1998) resulting in :

$$\Delta D = \frac{\partial D}{\partial M}\Delta M + \frac{\partial D}{\partial \alpha}\Delta\alpha = \left(\frac{M_0}{V_0}\frac{\partial C_D}{\partial M} + \frac{2C_D}{V_0}\right)\bar{q}_0 S_{ref}\Delta V + \frac{\partial C_D}{\partial \alpha}\bar{q}_0 S_{ref}\Delta\alpha \tag{A.8}$$

$$\Delta L = \frac{\partial L}{\partial M}\Delta M + \frac{\partial L}{\partial \alpha}\Delta\alpha = \left(\frac{M_0}{V_0}\frac{\partial C_L}{\partial M} + \frac{2C_L}{V_0}\right)\bar{q}_0 S_{ref}\Delta V + \frac{\partial C_L}{\partial \alpha}\bar{q}_0 S_{ref}\Delta\alpha \tag{A.9}$$

where ΔM was substituted by $\frac{M_0}{V_0}\Delta V$, and the values for $\frac{\partial D}{\partial M}$, $\frac{\partial D}{\partial \alpha}$, $\frac{\partial L}{\partial M}$ and $\frac{\partial L}{\partial \alpha}$ can be extracted from the graphs given by Shaughnessy et al. (1990). In a similar way ΔT can be rewritten to:

$$\Delta T = \frac{\partial T}{\partial M}\Delta M + \frac{\partial T}{\partial \bar{q}}\Delta\bar{q} + \frac{\partial T}{\partial \phi}\Delta\phi = \frac{\partial T}{\partial M}\frac{M_0}{V_0}\Delta V + \frac{\partial T}{\partial \bar{q}}\frac{1}{2}\rho_0\Delta V + \frac{\partial T}{\partial \phi}\Delta\phi \tag{A.10}$$

where the same expression is used for ΔM as before. For the majority of the trajectory the vehicle accelerates at a very small flight-path angle. The change in air density ρ is therefore negligible with respect to the change in velocity. As a result $\Delta\bar{q}$ can be substituted by $\frac{1}{2}\rho_0\Delta V$. The values for $\frac{\partial T}{\partial M}$, $\frac{\partial T}{\partial \bar{q}}$ and $\frac{\partial T}{\partial \phi}$ can be extracted from the graphs given by Shaughnessy et al. (1990).

For the open-loop behavior only the state is of interest. The controls will therefore be neglected for this analysis. Writing Equations A.5 to A.7 in matrix form including only the state results in:

$$\begin{bmatrix}\Delta\dot{V}\\ \Delta\dot{\gamma}\\ \Delta\dot{r}\end{bmatrix} = \begin{bmatrix}a_{VV} & a_{V\gamma} & a_{Vr}\\ a_{\gamma V} & a_{\gamma\gamma} & a_{\gamma r}\\ a_{rV} & a_{r\gamma} & a_{rr}\end{bmatrix} = \begin{bmatrix}\Delta V\\ \Delta\gamma\\ \Delta r\end{bmatrix} \tag{A.11}$$

with:

$$a_{VV} = -\frac{1}{mV_0}\left(M_0\frac{\partial C_D}{\partial M}\bar{q}_0 S_{ref} + M_0\cos\alpha_0\left(\frac{\partial T}{\partial M} + \frac{1}{2}\rho_0\frac{\partial T}{\partial \bar{q}}\right) + 2D_0\right) \tag{A.12a}$$

$$a_{V\gamma} = -g_0 \cos\gamma_0 \tag{A.12b}$$

$$a_{Vr} = 2\frac{g_0}{r_0}\sin\gamma_0 \tag{A.12c}$$

$$a_{\gamma V} = \frac{1}{V_0}\left(-\dot{\gamma} + \frac{2V_0}{r_0}\cos\gamma_0\right) + \frac{1}{mV_0^2}\left(M_0\frac{\partial C_L}{\partial M}\bar{q}_0 S_{ref} + 2L_0\right) + \frac{\sin\alpha_0}{m_0}\left(\frac{M_0}{V_0}\frac{\partial T}{\partial M} + \frac{1}{2}\rho_0\frac{\partial T}{\partial \bar{q}}\right) \tag{A.12d}$$

$$a_{\gamma\gamma} = -\left(\frac{V_0}{r_0} - \frac{g_0}{r_0}\right)\sin\gamma_0 \tag{A.12e}$$

$$a_{\gamma r} = -\left(\frac{2g_0}{V_0} - \frac{V_0}{r_0}\right)\frac{\cos\gamma_0}{r_0} \tag{A.12f}$$

$$a_{rV} = \sin\gamma_0 \tag{A.12g}$$

$$a_{r\gamma} = V_0\cos\gamma_0 \tag{A.12h}$$

$$a_{rr} = 0 \tag{A.12i}$$

Matlab® is used to compute the eigenvalues $\boldsymbol{\lambda}$ and corresponding eigenvectors $\boldsymbol{\mu_i}$ for the reference trajectory at four points along the trajectory. The computed values give insight in the state that traces a motion and the stability of that motion. The damping ratio ζ can be determined by:

$$\zeta = -\frac{Re(\lambda)}{\sqrt{Re(\lambda)^2 + Im(\lambda)^2}} \tag{A.13}$$

where a positive damping ratio indicates a stable system and a negative value indicates in fact an amplification. The natural frequency is given by:

$$\omega_n = \sqrt{Re(\lambda)^2 + Im(\lambda)^2} \tag{A.14}$$

With the damping ratio and the natural frequency the period can be computed with:

$$P = \frac{2\pi}{\omega_n\sqrt{1-\zeta^2}} \tag{A.15}$$

To find out what state variables are affected by the motion, the modulus of each complex pair in the eigenvector is computed according to:

$$z = \sqrt{Re(\lambda)^2 + Im(\lambda)^2} \tag{A.16}$$

where a high value of the modulus shows that the corresponding state variable does indeed trace the eigenmotion. The phase of the motion can be computed by:

$$\theta = \arctan\left(\frac{Im(\lambda)}{Re(\lambda)}\right) \tag{A.17}$$

Results of the eigenmotion at four points along the reference trajectory are given in Tables A.1 to A.4. All modes describe a periodic phugoid motion where kinetic and potential energy are interchanged. The altitude will clearly be affected most. The first two modes have periods smaller than the flight-time, but are well damped and therefore will not cause any stability problems. The third mode is not damped at all and will be unstable. However, the frequency is extremely small resulting in a doubling time greater than the flight-time of the ascent trajectory. No stability problems are therefore anticipated.

	mode 1		mode 2		mode 3	
Re(λ_i)	-0.0045		-0.0045		$3.0 \cdot 10^{-4}$	
Im(λ)	0.0088 i		-0.0088 i		0.0 i	
ζ (-)	0.4561		0.4561		-1	
ω_n (rad/s)	0.0099		0.0099		$3.0 \cdot 10^{-4}$	
P (s)	713.0		713.0		∞	
$T_{1/2}$ (s)	153.5		153.5		-2290.7	
$\boldsymbol{\mu_i}$	z	$\theta(^\circ)$	z	$\theta(^\circ)$	z	$\theta(^\circ)$
ΔV	0.0099	52.8	0.0099	-52.8	$2.9 \cdot 10^{-4}$	0
$\Delta\gamma$	$1.0 \cdot 10^{-5}$	-68.1	$1.0 \cdot 10^{-5}$	68.1	$3.3 \cdot 10^{-7}$	0
Δr	1.0	0	1.0	0	1.0	0

Table A.1: Results of Eigenmotion at V = 1000 m/s.

	mode 1		mode 2		mode 3	
Re(λ_i)	-0.0011		-0.0011		$1.5 \cdot 10^{-4}$	
Im(λ)	0.0062 i		-0.0062 i		0.0 i	
ζ (-)	0.1675		0.1675		-1	
ω_n (rad/s)	0.0063		0.0063		$1.5 \cdot 10^{-4}$	
P (s)	1007.4		1007.4		∞	
$T_{1/2}$ (s)	654.0		654.0		-4506.7	
$\boldsymbol{\mu_i}$	z	$\theta(^\circ)$	z	$\theta(^\circ)$	z	$\theta(^\circ)$
ΔV	0.0049	17.9	0.0049	-17.9	$3.3 \cdot 10^{-4}$	0
$\Delta\gamma$	$3.2 \cdot 10^{-6}$	-81.0	$3.2 \cdot 10^{-6}$	81.0	$7.9 \cdot 10^{-8}$	0
Δr	1.0	0	1.0	0	1.0	0

Table A.2: Results of Eigenmotion at V = 2000 m/s.

	mode 1		mode 2		mode 3	
Re(λ_i)	-0.0006		-0.0006		$1.8 \cdot 10^{-4}$	
Im(λ)	0.004 i		-0.004 i		0.0 i	
ζ (-)	0.1334		0.1334		-1	
ω_n (rad/s)	0.004		0.004		$1.8 \cdot 10^{-4}$	
P (s)	1579.0		1579.0		∞	
$T_{1/2}$ (s)	1143.7		1143.7		-3840.1	
$\boldsymbol{\mu_i}$	z	$\theta(^\circ)$	z	$\theta(^\circ)$	z	$\theta(^\circ)$
ΔV	0.0033	14.8	0.0033	-14.8	$4.8 \cdot 10^{-4}$	0
$\Delta\gamma$	$1.3 \cdot 10^{-6}$	-81.4	$1.3 \cdot 10^{-6}$	81.4	$6.0 \cdot 10^{-8}$	0
Δr	1.0	0	1.0	0	1.0	0

Table A.3: Results of Eigenmotion at V = 3000 m/s.

	mode 1		mode 2		mode 3	
Re(λ_i)	-0.0004		-0.0004		$1.9 \cdot 10^{-4}$	
Im(λ)	0.0029 i		-0.0029 i		0.0 i	
ζ (-)	0.1334		0.1334		-1	
ω_n (rad/s)	0.0029		0.0029		$1.9 \cdot 10^{-4}$	
P (s)	2197.5		2197.5		∞	
$T_{1/2}$ (s)	1801.5		1801.5		-3687.7	
$\boldsymbol{\mu_i}$	z	$\theta(^\circ)$	z	$\theta(^\circ)$	z	$\theta(^\circ)$
ΔV	0.0024	11.6	0.0024	-11.6	$5.8 \cdot 10^{-4}$	0
$\Delta\gamma$	$7.2 \cdot 10^{-7}$	-82.5	$7.2 \cdot 10^{-7}$	82.5	$4.7 \cdot 10^{-8}$	0
Δr	1.0	0	1.0	0	1.0	0

Table A.4: Results of Eigenmotion at V = 4000 m/s.

B

Integrator Test Results

This section performs a trade-off between fixed and variable step-size integrators for the integration of the trajectory simulation. Variable step-size integrators have the advantage of being able to adapt the step-size of the integrator based on the local truncation error. This allows the integrator to use much larger step-sizes on parts of the problem while still using a small step-size for the more demanding parts of the problem. However, for the trajectory simulation the maximum integration step-size is limited by the guidance step-size. The variable step-size will therefore only be advantageous if many integration steps will be used within each guidance step.

Tests have been performed to find the effect of the guidance and integration step-sizes. Tables B.1 to B.3 show the resulting final time (t_f), velocity (V_f), altitude (h_f) and mass (m_f) for the same ascent trajectory simulation with different integration settings. For each run the RK-4 integrator was used and both the guidance step-size (Δt_g) and the integration frequency (f_{int}) were varied, where f_{int} is defined as the number of integration steps per guidance update.

B.1 shows that the integration frequency has almost no effect on the result for a guidance step-size of 0.1 seconds. B.2 shows results for a changing guidance step-size while the integration step-size remains at 0.1 seconds. The guidance step-size has a stronger effect on the results than the integration step-size. Increasing the guidance step-size from 0.1 to 0.6 seconds will result in a 0.23% change in the final altitude. B.3 shows the result of changing both the guidance and integration step-size. Increasing both values from 0.05 to 0.6 seconds leads to significant changes in the final results of up to 8%. However, at 0.2 seconds the changes are all less then 0,1%

Based on the results the following can be concluded. When a larger guidance step-size is selected, the integration frequency will have to be increased to make sure that the integration step-size does not become too large. However, to guarantee accuracy a small guidance step-size should be selected, meaning that a single integration step-size per guidance update should be sufficient.

Input	**Run 1**	**Run 2**	**Run 3**	**Run 4**				
Δt_g	0.1	0.1	0.1	0.1				
f_{int}	1	2	4	6				
Output					**Δ wrt Run 1**	**Run 2**	**Run 3**	**Run 4**
t_f [s]	2,182.90	2,182.80	2,182.90	2,182.90	Δt_f [%]	-0.0046	0.0000	0.0000
V_f [m/s]	6,903.42	6,903.39	6,903.42	6,903.44	ΔV_f [%]	-0.0005	0.0001	0.0003
h_f [m]	88,391.41	88,390.59	88,392.16	88,392.42	Δh_f [%]	-0.0009	0.0008	0.0011
m_f [kg]	68,185.23	68,186.01	68,184.76	68,184.61	Δm_f [%]	0.0011	-0.0007	-0.0009

Table B.1: Results of changing integration step-size with fixed guidance step-size.

Input	**Run 1**	**Run 2**	**Run 3**	**Run 4**				
Δt_g	0.1	0.2	0.4	0.6				
f_{int}	1	2	4	6				
Output					**Δ wrt Run 1**	**Run 2**	**Run 3**	**Run 4**
t_f [s]	2,182.90	2,183.60	2,181.50	2,186.50	Δt_f [%]	0.0321	-0.0641	0.1649
V_f [m/s]	6,903.42	6,903.81	6,904.40	6,904.95	ΔV_f [%]	0.0056	0.0142	0.0222
h_f [m]	88,391.41	88,437.08	88,459.61	88,600.26	Δh_f [%]	0.0517	0.0772	0.2363
m_f [kg]	68,185.23	68,176.29	68,200.19	68,133.95	Δm_f [%]	-0.0131	0.0219	-0.0752

Table B.2: Results of changing guidance step-size with fixed integration step-size.

Input	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**
Δt_g	0.05	0.1	0.2	0.4	0.6
f_{int}	1	1	1	1	1

Output						**Δ wrt Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**
t_f [s]	2,183.45	2,182.90	2,183.40	2,181.20	2,332.20	Δt_f [%]	-0.0252	-0.0023	-0.1030	6.8126
V_f [m/s]	6,903.22	6,903.42	6,903.69	6,904.54	6,912.43	ΔV_f [%]	0.0029	0.0068	0.0191	0.1333
h_f [m]	88,382.27	88,391.41	88,433.22	88,452.67	83,848.59	Δh_f [%]	0.0103	0.0576	0.0796	-5.1296
m_f [kg]	68,176.82	68,185.23	68,179.15	68,206.49	62,700.20	Δm_f [%]	0.0123	0.0034	0.0435	-8.0330

Table B.3: Results of changing both guidance and integration step-size.

C

Optimization Results

In this appendix a number of results are shown and discussed, for which conclusions were stated in Chapter 7. Appendix C.1 first discusses the optimization of combined flight phases. Second, Appendix C.2 will show results for the optimization of the acceleration phase with case CN3.6, as defined in Section 7.4.

C.1. Combined Optimization of Flight Phases

The complete trajectory is split in three flight-phases: the take-off, the atmospheric acceleration and the pull-up. A clear definition of each of the flight-phases is given in Section 7.7. In this section the combined optimization of different flight phases is discussed. The following combinations were attempted:

- take-off + atmospheric acceleration
- atmospheric acceleration + pull-up
- take-off + atmospheric acceleration + pull-up

Figures C.1 to C.3 show the resulting objective function values of the populations after 500 generations, as projections on the planes. Just looking at the bottom plane of each of the figures already shows that none resulting solutions are feasible. The maximum fuel-mass objective value allowed is 0.5666. A larger value means that more fuel has been used than can be taken on board. Figures C.1 and C.2 include no individuals that comply with this value. Figure C.3 does show a few individuals on the bottom-right with a lower fuel-mass objective value. However, these individuals have very high

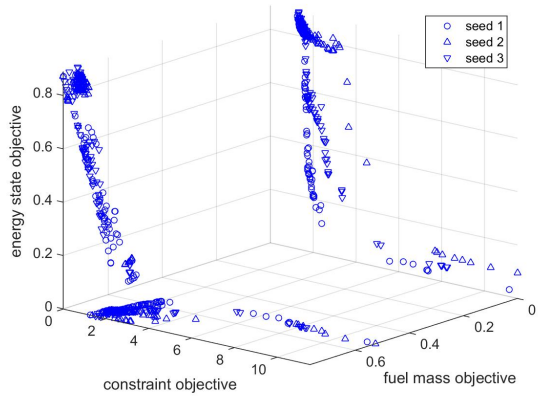


Figure C.1: Constraints vs energy state vs fuel mass objective, projected on planes, for combined optimization of take-off and atmospheric acceleration.

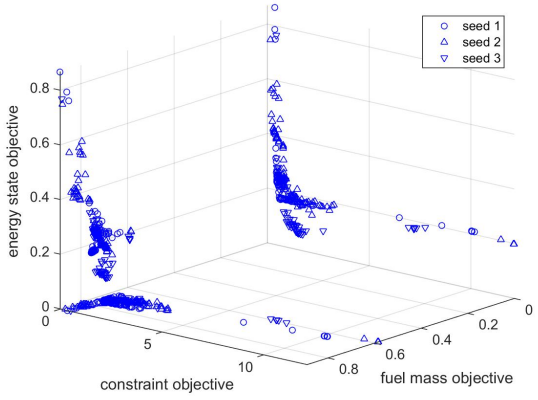


Figure C.2: Constraints vs energy state vs fuel mass objective, projected on planes, for combined optimization of atmospheric acceleration and pull-up.

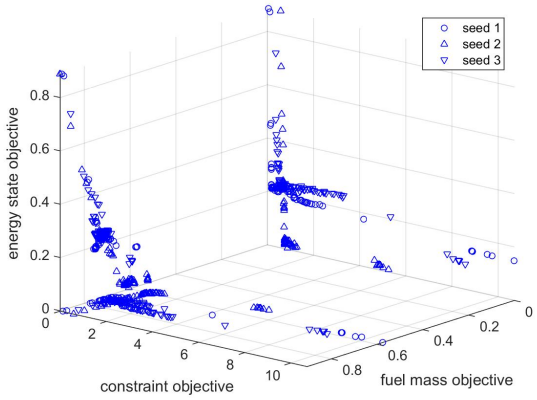


Figure C.3: Constraints vs energy state vs fuel mass objective, projected on planes, for combined optimization of take-off, atmospheric acceleration and pull-up.

constraint objective values, and are therefore also not feasible. Furthermore, the pull-up still has to be performed, which requires more fuel as well.

Based on these results, it is concluded that the current implementation is not able to find a feasible trajectory when the different flight-phases are combined.

C.2. Atmospheric Acceleration for case CN3.6

The trajectory results of the atmospheric acceleration phase, found by the optimizer for case CN3.6 defined in Section 7.4, are discussed here. The results show strong similarity with the results found for case TVC-S2. These results are discussed in Section 7.7.2.

The results also show strong similarities with the result found by Mooij (1998) for the reference trajectory shown in Figure 6.17 to Figure 6.28. Figures C.6 and C.8 show the control histories of the angle of attack and equivalence ratio. Just like the reference, the angle first drops to around 2 degrees, before it increases to a maximum value of approximately 3 degrees. As the vehicle accelerates, the angle of attack slowly reduces to around 1.5 degrees. The equivalence ratio decreases at the start of the trajectory to comply with the constraints. For the rest of the flight it remains very close to the nominal value of 1. Figure C.14 shows that the energy state almost increases linearly with time. Both the angle of attack and the equivalence ratio versus time therefore have a very similar pattern as shown in Figures C.7 and C.9.

The flight-path angle and the trajectory shown in Figures C.5 and C.4 show the same pattern and oscillation as the results found by TVC-S2. Figures Figure C.10, Figure C.12 and Figure C.13 show that the axial-acceleration, dynamic-pressure and heat-rate constraint are all violated at a peak of the oscillation. While the throttle law, used for TVC-S2, was able to limit constraint violations, the implementation of case CN3.6 is found to be less effective. With the throttle law implemented, thrust is reduced immediately, when a violation occurs, limiting the magnitude of the violation. For case CN3.6, the thrust magnitude is defined by the optimizer, resulting in less flexibility. Thrust can therefore not be used effectively to reduce the constraint violation.

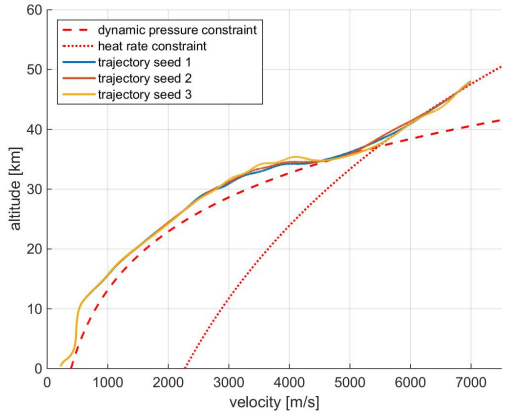


Figure C.4: Final trajectory for atmospheric acceleration phase optimized with case CN3.6.

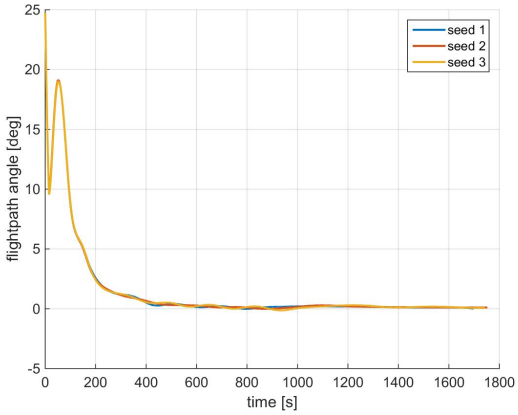


Figure C.5: Flight-path angle versus time for atmospheric acceleration phase optimized with case CN3.6.

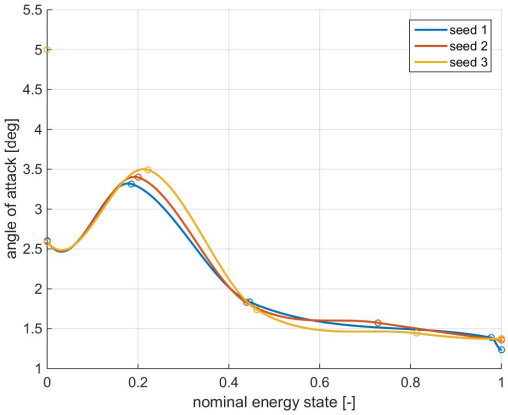


Figure C.6: Control history of angle of attack versus the normalized energy state for acceleration phase optimized with case CN3.6.

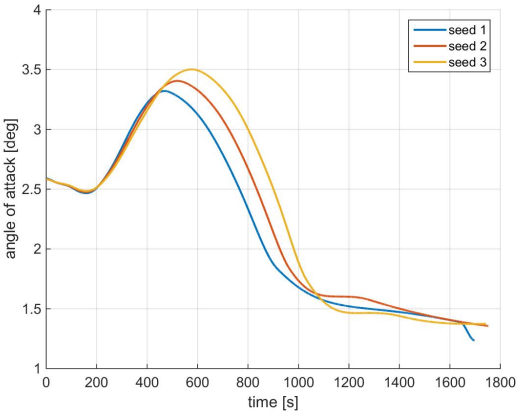


Figure C.7: Angle of attack versus time for acceleration phase optimized with case CN3.6.

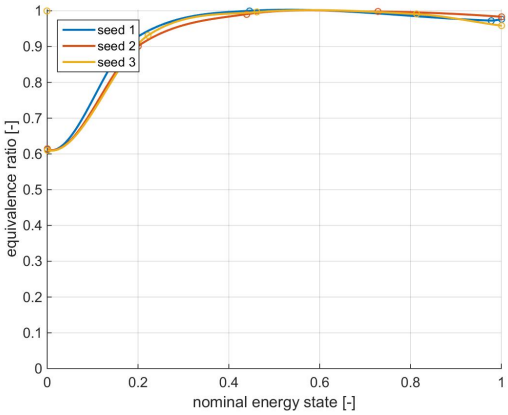


Figure C.8: Control history of equivalence ratio versus the normalized energy state for acceleration phase optimized with case CN3.6.

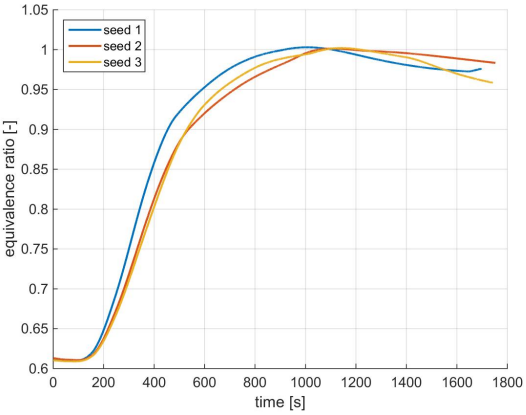


Figure C.9: Equivalence ratio versus time for acceleration phase optimized with case CN3.6.

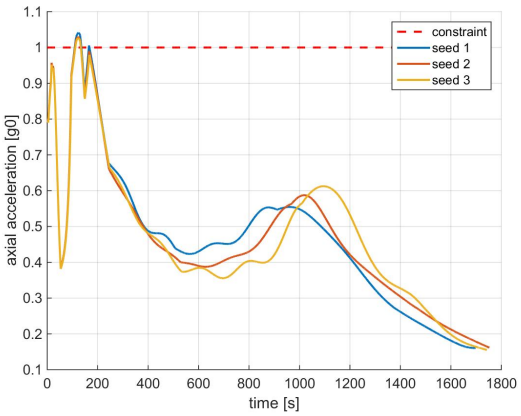


Figure C.10: Axial acceleration versus time for acceleration phase optimized with case CN3.6.

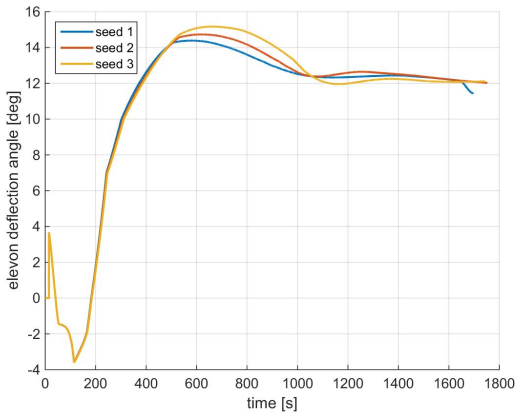


Figure C.11: Elevon deflection versus time for acceleration phase optimized with case CN3.6.

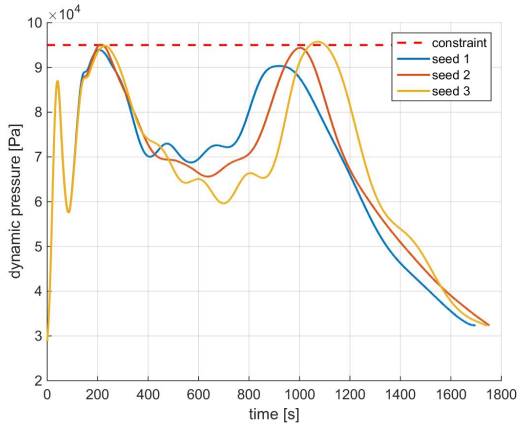


Figure C.12: Dynamic pressure versus time for acceleration phase optimized with case CN3.6.

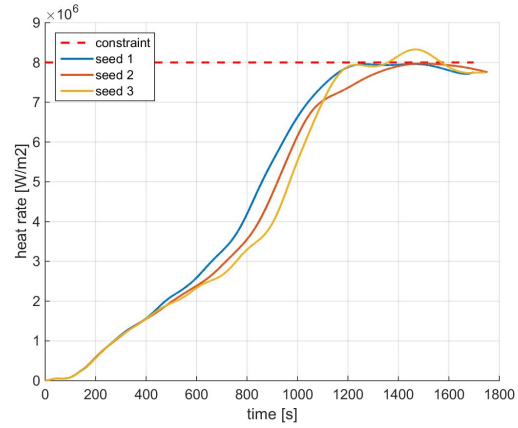


Figure C.13: Heat rate versus time for acceleration phase optimized with case CN3.6.

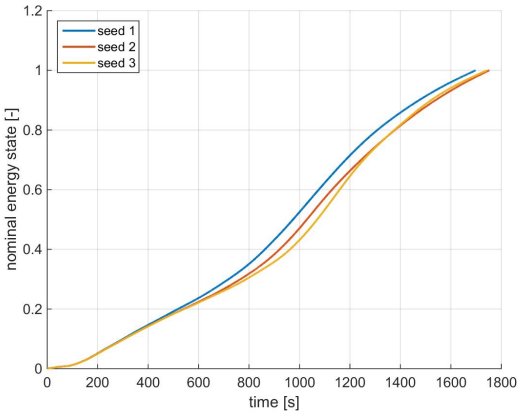


Figure C.14: Energy state versus time for acceleration phase optimized with case CN3.6.

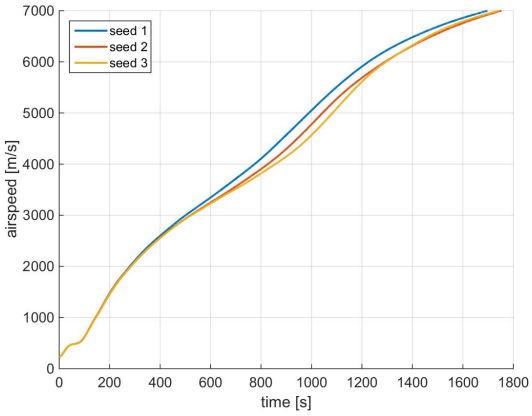


Figure C.15: Velocity versus time for acceleration phase optimized with case CN3.6.

References

Al-Garni, A. and Kassem, A. (2005). "optimizing aerospace plane ascent trajectory with thermal constraints using genetic algorithms". In *AIAA/CIRA 13th International Space Planes and Hypersonics Systems and Technologies Conference*. AIAA-2005-3293.

Al-Garni, A. and Kassem, A. H. (2007). "on the optimization of aerospace plane ascent trajectory". *Transactions of the Japan Society for Aeronautical and Space Sciences*. V. 50, No. 168, pp. 113-120.

Ashford, D. (1993). "a strategy for developing a safe spaceplane soon". In *5th International Aerospace Planes and Hypersonics Technologies Conference*. AIAA.

Ashford, D. (2002). *Spaceflight Revolution*. Imperial College Press, London.

Ashford, D. (2009). "an aviation approach to space transportation". *Aeronautical Journal*. V. 113, No. 1146, pp. 499.

Betts, J. T. (1998). "survey of numerical methods for trajectory optimization". *Journal of Guidance, Control, and Dynamics*. V. 21, No. 2, pp. 193-207.

Biscani, F., Izzo, D., and Yam, C. H. (2010). "a global optimisation toolbox for massively parallel engineering optimisation". *arXiv preprint arXiv:1004.3824*.

Bonner, E., Clever, W., and Dunn, K. (1991). "aerodynamic preliminary analysis system 2. part 1: Theory". NASA-CR-182076.

Brauer, G., Cornick, D., and Stevenson, R. (1977). "capabilities and applications of the program to optimize simulated trajectories (POST)". *NASA CR-2770*.

Burden, R. L. and Faires, J. D. (2001). *Numerical analysis. 2001*. Brooks/Cole, USA.

Cantú-Paz, E. (1999). "migration policies and takeover times in parallel genetic algorithms". *Report No. 99008*.

Castellini, F. (2008). "global optimization techniques in space missions design". Phd thesis, Politecnico di Milano.

Chase, N., Rademacher, M., Goodman, E., Averill, R., and Sidhu, R. (2009). "a benchmark study of multi-objective optimization methods". BMK-3021.

Cremaschi, F., Weikert, S., Wiegand, A., Jung, W., and Scheuerpflug, F. (2009). "sounding rocket trajectory simulation and optimization with astos". In *Proceedings of the 19th ESA Symposium on European Rocket and Balloon Programmes and Related Research*. ESA SP-671.

Cruz, C. I. and Wilhite, A. W. (1989). "prediction of high-speed aerodynamic characteristics using the aerodynamic preliminary analysis system (APAS)". In *Applied Aerodynamics Conference*. AIAA-89-2173. pp. 11.

Curran, E. (2001). *Scramjet propulsion*. AIAA. Vo. 189.

Darling, D. (2003). *The complete book of spaceflight: from Apollo 1 to zero gravity*. John Wiley & Sons.

Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. (2000). "a fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II". In *Parallel problem solving from nature PPSN VI*. Springer. pp. 849-858.

Deb, K. and Deb, D. (2014). ”analysing mutation schemes for real-parameter genetic algorithms”. *International Journal of Artificial Intelligence and Soft Computing*. Vo. 4, No. 1, pp. 1-28.

Deb, K. and Kumar, A. (1995). ”real-coded genetic algorithms with simulated binary crossover: studies on multimodal and multiobjective problems”. *Complex Systems*. Vo. 9, No. 6, pp. 431-454.

Dijkstra, M. (2012). ”trajectory optimization of Hyperion-II for the study of hypersonic aerothermodynamic phenomena”. Msc thesis, Delft University of Technology.

Dissel, A. F., Kothari, A. P., and Lewis, M. J. (2006). ”comparison of horizontally and vertically launched airbreathing and rocket vehicles”. *Journal of Spacecraft and Rockets*. Vo. 43, No. 1, pp. 161-169.

Dujarric, C. (1999). ”possible future european launchers- a process of convergence”. *ESA bulletin*. No. 97, pp. 11-97.

Fritsch, F. N. and Carlson, R. E. (1980). ”monotone piecewise cubic interpolation”. *SIAM Journal on Numerical Analysis*. Vo. 17, No. 2, pp. 238-246.

Goldberg, D. E. and Deb, K. (1991). ”a comparative analysis of selection schemes used in genetic algorithms”. *Foundations of genetic algorithms*. Morgan Kaufmann Publishers Inc., Vo. 1, pp. 69-93.

Hallion, R. (2005). ”the history of hypersonics: Or, back to the future - again and again”. *43rd AIAA Aerospace Sciences Meeting and Exhibit*, AIAA-05-329.

Hattis, P. and Malchow, H. (1992). ”evaluation of some significant issues affecting trajectory and control management for air-breathing hypersonic vehicles”. In *4th Symposium on Multidisciplinary Analysis and Optimization*. AIAA-92-5011.

Hattis, P., Malchow, H., Shaughnessy, J., and Chowdhry, R. (1991). ”integrated trajectory and control analysis for generic hypersonic vehicles”. In *3rd International Aerospace Planes Conference*. AIAA-91-5052.

Hänninen, P. G. (2009). ”multidisciplinary distributed optimization for space projects”. Phd thesis, Politecnico Di Milano, Dipartimento Di Ingegneria a Erospaziale.

Izzo, D. (2006). ”advances in global optimisation for space trajectory design”. In *Proceedings of the international symposium on space technology and science*. Vo. 25.

Jackson, K., Enright, W., and Hull, T. (1978). ”a theoretical criterion for comparing Runge-Kutta formulas”. *SIAM Journal on Numerical Analysis*. Vo. 15, No. 2, pp. 618-641.

Jategaonkar, R., Behr, R., Gockel, W., and Zorn, C. (2006). ”data analysis of Phoenix reusable launch vehicle demonstrator flight test”. *Journal of Aircraft*. Vo. 43, No. 6, pp. 1732-1737.

Jolly, C., Berend, N., and Aumasson, C. (2003). ”FLOP: ONERA’s answer to trajectory optimization for future launchers”. In *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, Guidance, Navigation, and Control and Co-located Conferences. AIAA-03-5545.

Koelle, D. and Janovsky, R. (2007). ”development and transportation costs of space launch systems”. In *Proceedings of the DGLR/CEAS European Air and Space Conference*.

Kors, D. (1990). ”design considerations for combined air breathing-rocket propulsion systems”. In *2nd International Aerospace Planes Conference*. AIAA-90-5216.

Li, H. and Zhang, Q. (2009). ”multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II”. *IEEE, Transactions on Evolutionary Computation*. Vo. 13, No. 2, pp. 284-302.

Livingston, J. (2008). ”the case against horizontal takeoff launch vehicles”. In *15th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*. AIAA-08-2640.

Longstaff, R. and Bond, A. (2011). ”the SKYLON project”. In *17th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*. AIAA.

Lu, P. (1991). "trajectory optimization and guidance for a hypersonic vehicle". In *3rd International Aerospace Planes Conference*. AIAA-91-5068.

Mooij, E. (1994). "the motion of a vehicle in a planetary atmosphere". *Delft University of Technology, Faculty of Aerospace Engineering, Report LR-768*.

Mooij, E. (1998). *Aerospace-plane flight dynamics: analysis of guidance and control concepts*. Phd thesis, Delft University of Technology.

Mooij, E. (2015). *Re-Entry Systems, Draft lecture notes*. Astrodynamics and Space Missions, Delft University of Technology. Course AE-4870B.

Murata, T., Ishibuchi, H., and Gen, M. (2001). "specification of genetic search directions in cellular multi-objective genetic algorithms". In *Evolutionary multi-criterion optimization*. Springer. pp. 82-95.

Myatt, D., Becerra, V. M., Nasuto, S. J., and Bishop, J. (2004). "advanced global optimisation for mission analysis and design". Ariadna id: 03/4101.

NOAA, NASA, and USAF (1976). *U.S. Standard atmosphere, 1976*. Washington, D.C.

Oppenheimer, M., Skujins, T., Cesnik, C., and Doman, D. (2008). "canard-elevon interactions on a hypersonic vehicle". In *AIAA Atmospheric Flight Mechanics Conference and Exhibit*. AIAA-08-6383.

Pagano, A. and Mooij, E. (2010). "global launcher trajectory optimization for lunar base settlement". In *AIAA/AAS Astrodynamics Specialist Conference*, Guidance, Navigation, and Control and Co-located Conferences. AIAA-10-8387.

Papp, Z. (2014). "mission planner for heating-optimal re-entry trajectories with extended range capability". Msc thesis, Delft University of Technology.

Parsopoulos, K. E. and Vrahatis, M. N. (2002). "particle swarm optimization method for constrained optimization problems". *Intelligent Technologies – Theory and Application: New Trends in Intelligent Technologies*. Vo. 76, pp. 214-220.

Pezzella, G., Marini, M., Roncioni, P., Kauffmann, J., and Tomatis, C. (2009). "preliminary design of vertical takeoff hopper concept of future launchers preparatory program". *Journal of Spacecraft and Rockets*. Vo. 46, No. 4, pp. 788-799.

Powell, R. W., Shaughnessy, J. D., Cruz, C. I., and Naftel, C. (1991). "ascent performance of an air-breathing horizontal-takeoff launch vehicle". *Journal of Guidance, Control, and Dynamics*. Vo. 14, No. 4, pp. 834-839.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2002). *Numerical recipes 2nd edition: The art of scientific computing*. Cambridge university press.

Prince, P. and Dormand, J. (1981). "high order embedded Runge-Kutta formulae". *Journal of Computational and Applied Mathematics*. Vo. 7, No. 1, pp. 67-75.

Qi, Y., Ma, X., Liu, F., Jiao, L., Sun, J., and Wu, J. (2014). "MOEA/D with adaptive weight adjustment". *Evolutionary computation*. Vo. 22, No. 2, pp. 231-264.

Roenneke, A., Moulin, J., and Chavagnac, C. (2003). "overview of european RLV demonstrator vehicles". In *54th International Astronautical Congress of the International Astronautical Federation, the International Academy of Astronautics, and the International Institute of Space Law*. IAC-03-V.6.03.

Sänger, E. and Bredt, J. (1944). "a rocket drive for long range bombers".

Seedhouse, E. (2015). *Virgin Galactic: The First Ten Years*. Springer.

Shaughnessy, J. D. and Gregory, I. M. (1991). "trim drag reduction concepts for horizontal takeoff single-stage-to-orbit vehicles". NASA Technical Memorandum 102687.

Shaughnessy, J. D., Pinckney, S. Z., McMinn, J. D., Cruz, C. I., and Kelley, M.-L. (1990). "hypersonic vehicle simulation model: winged-cone configuration".

Sippel, M. (2015). ”spaceliner technical progress and mission definition”. In *20th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*. AIAA-15-3582.

Sippel, M., van Foreest, A., Bauer, C., and Cremaschi, F. (2011). ”system investigations of the spaceliner concept in FAST20XX”. In *17th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*. AIAA-11-2294.

Sova, G., Divan, P., and Spacht, L. (1991). ”aerodynamic preliminary analysis system 2. part 2: User’s manual”. NASA-CR-182077.

Spies, J. (2003). ”RLV Hopper: Consolidated system concept”. *Acta Astronautica*. Vo. 53, No. 4, pp. 709-717.

Storn, R. (1996). ”on the usage of differential evolution for function optimization”. In *Fuzzy Information Processing Society, 1996. NAFIPS., 1996 Biennial Conference of the North American*. IEEE. pp. 519-523.

Storn, R. (1999). ”system design by constraint adaptation and differential evolution”. *IEEE, Transactions on Evolutionary Computation*. Vo. 3, No. 1, pp. 22-34.

Storn, R. and Price, K. (1997). ”differential evolution–a simple and efficient heuristic for global optimization over continuous spaces”. *Journal of global optimization*. Vo. 11, No. 4, pp. 341-359.

Talbot, C. (2003). ”different approaches to optimise reusable launch vehicles trajectories”. In *54th International Astronautical Congress of the International Astronautical Federation, the International Academy of Astronautics, and the International Institute of Space Law*. AIAA. IAC-03-A.7.04.

Tan, K. C., Khor, E. F., Lee, T. H., and Yang, Y. (2003). ”a tabu-based exploratory evolutionary algorithm for multiobjective optimization”. *Artificial Intelligence Review*. Vo. 19, No. 3, pp. 231-260.

Van Buren, M. and Mease, K. (1991). ”aerospace plane guidance using time-scale decomposition - a geometric approach”. In *Navigation and Control Conference*. AIAA-91-2722.

Van Veldhuizen, D. A., Zydallis, J. B., and Lamont, G. B. (2003). ”considerations in engineering parallel multiobjective evolutionary algorithms”. *IEEE, Transactions on Evolutionary Computation*. Vo. 7, No. 2, pp. 144-173.

Varvill, R., M., H., and R., L. (2014). ”SKYLON users’ manual”. Report, Reaction Engines Limited.

Vasile, M. (2003). ”a global approach to optimal space trajectory design”. *Advances in the Astronautical Sciences*. Vo. 114, pp. 629-647.

Vesterstøm, J. and Thomsen, R. (2004). ”a comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems”. In *Evolutionary Computation*. IEEE. Vo. 2, pp. 1980-1987.

Vinkó, T., Izzo, D., and Bombardelli, C. (2007). ”benchmarking different global optimisation techniques for preliminary space trajectory design”. In *58th International Astronautical Congress, International Astronautical Federation (IAF)*.

Von Stryk, O. and Bulirsch, R. (1992). ”direct and indirect methods for trajectory optimization”. *Annals of operations research*. Vo. 37, No. 1, pp. 357-373.

Wakker, K. F. (2015). *Fundamentals of Astrodynamics*. Institutional Repository Library, Delft University of Technology.

Wolpert, D. H. and Macready, W. G. (1997). ”no free lunch theorems for optimization”. *IEEE, Transactions on Evolutionary Computation*. Vo. 1, No. 1, pp. 67-82.

Yang, J. M., Chen, Y. P., Horng, J. T., and Kao, C.-Y. (1997). ”applying family competition to evolution strategies for constrained optimization”. In *Evolutionary Programming VI*. Springer. pp. 201-211.

Zhang, Q. and Li, H. (2007). ”MOEA/D: A multiobjective evolutionary algorithm based on decomposition”. *IEEE, Transactions on Evolutionary Computation*. Vo. 11, No. 6, pp. 712-731.

Zitzler, E. and Thiele, L. (1998). ”multiobjective optimization using evolutionary algorithms, a comparative case study”. In *Parallel problem solving from nature — PPSN V*. Springer. pp. 292-301.