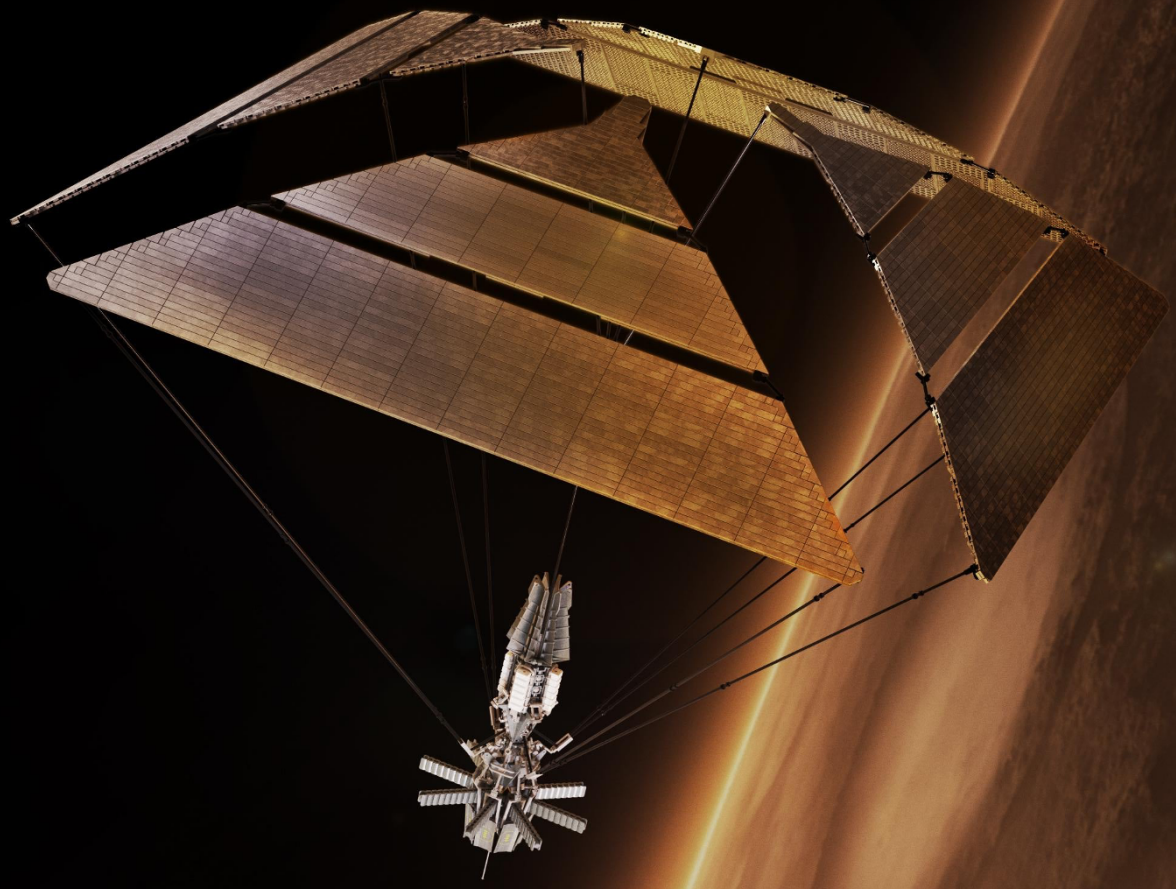
Differential Dynamic Programming applied to Interplanetary Solar-Sail Trajectory Optimization

Master Thesis

Roos Martens



Delft University of Technology

TU Delft

Differential Dynamic Programming applied to Interplanetary Solar-Sail Trajectory Optimization

by

Roos Martens

Dissertation for the degree of

Master of Science

at the Delft University of Technology,
Astrodynamics and Space Missions section,
to be defended publicly on Friday, February 17, 2023

Student number:	4359682	
Thesis committee:	Dr. ir. M. J. Heiligers	Supervisor
	Dr. ir. W. van der Wal	Committee chair
	Dr. B.V.S. Jyoti	External examinator

An electronic version of this thesis work is available at https://www.repository.tudelft.nl/

TU Delft

Preface

This work marks the end of my student days, which certainly is a big milestone for me. Having an interest in spaceflight since I was a kid, and after finishing the BSc Applied Physics, the only choice for me was to start my Masters' degree in Aerospace Engineering: specifically the Space track. Here I was first introduced to the concept of solar sailing, which for me at first was like something that is only possible in scifi movies and books. However, during my studies I learnt that solar sailing is certainly a feasible mission idea and this drove me to focus my Masters' thesis on it. Apart from this, I wanted to develop an algorithm from scratch and create a tool that would be helpful for future research. I was very lucky that I could combine these two interests in this thesis work and develop a differential dynamic programming algorithm that could be applied to different solar-sail problems. Hence, I very much hope that my work contributes to the knowledge already out there, and that piece by piece an optimization tool can be created that can handle a large variation of problems easily.

I would like to thank some people for helping me during my years of study and especially during the thesis. First and foremost, I would like to thank Jeannette Heiligers, my daily supervisor. Without her guiding, helpful comments, and eye-opening insights, I would not have been able to complete the thesis with the quality that I hope it has. Additionally, I would like to thank all previous authors on the concepts of solar sailing and differential dynamic programming. In particular, I would like to thank Gijs Leemans for always being open to sparring with me about the complex parts of the algorithm. Last but not least, I would like to thank my friends and family for always encouraging me. My roommates and boyfriend were always open to me blurting out a monologue and thinking out loud, which helped me a lot to define a good structure for the research. All in all, I hope you enjoy reading the work that has kept me busy the last year.

Roos Martens
Delft, February 2023

Executive summary

Solar sailing is a propulsion method where photons impose an acceleration on a thin, highly-reflective surface. The continuous acceleration that can be generated unlocks a large scope of high-energy mission possibilities, an extension of the mission lifespan, and a reduced mission cost in contrast to conventional propulsion methods, only limited by the lifetime of the sail itself. The benefits of solar sails can especially pay off for interplanetary or interstellar missions, since these are generally high-energy and long-duration missions. Nonetheless, due to the small acceleration a solar-sail can currently generate, there is a large need to minimize the time of flight to increase the feasibility of solar sailing in comparison to conventional propulsion methods. A robust, efficient, and accurate method is required for the optimization of high-dimensional low-thrust long-duration trajectories, e.g., interplanetary solar-sail trajectories. For these complex missions, where constraints are involved or multiple objectives need to be optimized, analytical solutions in the form of locally optimal steering laws do not suffice. Optimal control algorithms have the potential to find these optimal low-thrust trajectories, which are mostly nonlinear, constrained optimal control problems. Mainly, these problems are solved through the use of direct nonlinear programming techniques, often employed with direct collocation or shooting methods, or the use of evolutionary algorithms. However, especially in high-dimensional problems these techniques may struggle to find the global optimum and the required computational effort may increase largely. *Differential dynamic programming* (DDP) is an optimal control method that has the benefit that its computational effort scales only linearly with the size of the problem, in contrast to the exponential effort increase that most numerical optimization methods exhibit. Additionally, DDP is known to handle high-dimensional problems with multiple constraints quite well. DDP divides the to-be-optimized trajectory into stages, where for each stage a local cost function is evaluated. A quadratic approximation of the cost function for each stage is solved recursively, by taking a second-order Taylor series expansion around the state, control and Lagrange multipliers. When the cost is minimized, the optimal control is found. The entire process is iterative and thus repeated until convergence. This thesis work aims - for the first time - to investigate the performance of DDP for the optimization of interplanetary solar-sail trajectories.

The dynamical framework of this work assumes a two-body problem and an ideal solar-sail force model. The dynamics are propagated using modified equinoctial elements (MEE), which is advantageous for the numerical performance since there is only one fast-changing element. Additionally, a Sundman transformation is applied, which changes the independent variable from time to true anomaly. To investigate the performance of DDP for the optimization of interplanetary solar-sail trajectories, first of all, the results generated with DDP were compared to the results of locally optimal steering laws for the maximization of different orbital elements. Secondly, a sensitivity analysis was performed to showcase the robustness of DDP to different settings and initial guesses on a test case in the interplanetary regime. Finally, DDP was applied to a popular interplanetary rendezvous mission, namely an Earth-Mars orbit transfer.

The implementation of DDP was validated against locally optimal steering laws for orbit raising and maximizing the eccentricity. Two objectives were optimized with DDP to compare against the locally optimal steering law for orbit raising: both the semi-major axis and the specific orbital energy were maximized for three orbital revolutions, starting from an ecliptic, eccentric orbit at one astronomical unit (AU) from the Sun. Both optimizations found valid trajectories and control laws similar to the locally optimal steering law, only the maximization of the semi-major axis itself resulted in a slightly larger final semi-major axis, where the maximization of the specific orbital energy resulted in a slightly larger specific orbital energy. In addition, the maximization of the eccentricity found a larger eccentricity than the locally optimal steering law. Therefore, DDP was considered valid for use with different objectives in the interplanetary regime.

Furthermore, a sensitivity analysis was performed for the case where the semi-major axis was maximized for one orbital revolution. Firstly, the robustness of DDP to different initial guesses was investigated by feeding the algorithm different sets of constant initial guesses for the solar-sail attitude angles, the cone and clock angles. All initial guesses resulted in valid, (near)-optimal trajectories. Initial

guesses close to the optimum reduced the run time by 99% with respect to the initial guess resulting in the longest run time. The initial guess closest to the optimum had a comparable run time as when the locally optimal steering law itself was used as an initial guess, however the first found a slightly less-optimal value. Secondly, the algorithm settings were varied and analyzed. The optimality tolerance had the largest effect on the optimality of the solution as well as the run time. The smaller the optimality tolerance, the more optimal the solution, yet the longer the run time. The reduction ratio tolerance, which defines the tolerance on the desired reliability of the model, must not be set too large, since the algorithm will converge to a suboptimal solution, yet on the other hand must not be set too small, since the algorithm will converge too fast, also resulting in a suboptimal solution. Other factors, such as the trust-region scaling parameters, were concluded to have a minimal effect on both the optimality of the solution and the run time.

Finally, an Earth-Mars transfer test case was investigated by assuming circular, co-planar orbits for Earth and Mars. The transfer was modeled with a near-term lightness number of 0.01 and a futuristic lightness number of 0.1. A time-optimal transfer was sought that best met two weighted constraints, namely a constraint on the final eccentricity and the final semi-major axis. Since the current implementation does not support a dynamic trajectory discretization, an iterative process (e.g., decreasing the fixed number of orbital revolutions for each run) was used to find the time-optimal transfer that best fit the constraints. The optimality of the optimization proved to be sensitive to the values of the weights on the constraints. Larger weight values resulted in more compliance with the corresponding constraint, where for smaller weight values the emphasis was more on the objective itself. It was required to fine-tune the weights in order to achieve the optimal trajectories for both lightness numbers. Eventually, for a lightness number of 0.01, a final transfer time was found of approximately 4484 days, where the final eccentricity error was 0.0016% and the error on the semi-major axis was 0.0000%. The transfer time for this lightness number is unfeasible when compared to transfer times of conventionally propelled spacecraft. For a lightness number of 0.1, a final transfer time was found of approximately 506 days, where the final eccentricity error was 0.6896% and the error on the semi-major axis was 0.0000%. This transfer time is in agreement with transfer times in literature found with other optimization methods. It can be concluded that DDP successfully finds a time-optimal solution for an Earth-Mars transfer, simultaneously complying with two terminal constraints.

All in all, DDP finds similar or more optimal solutions than locally optimal steering laws for the optimization of different orbital elements in the interplanetary regime: for the maximization of both the semi-major axis and the eccentricity, DDP finds larger final values. In addition, DDP is robust to different initial guesses and algorithmic settings. Furthermore, the algorithm is able to comply with multiple terminal constraints for the optimization of realistic interplanetary solar-sail trajectories, such as an Earth-Mars orbit transfer. This work displays promising results for the application of DDP to interplanetary solar-sail trajectory optimization. To extend on this research and ensure a more extensive analysis, it is recommended to enhance the implementation of the algorithm and apply DDP to more solar-sail problems, possibly with more realistic dynamical models, in the interplanetary, or even interstellar, flight regime.

Contents

Nomenclature

Abbreviations

AB(M)	=	Adams-Bashforth(-Moulton)
ACS3	=	Advanced Composite Solar Sail System
AIAA	=	American Institute of Aeronautics and Astronautics
AU	=	Astronomical Unit
BS	=	Bulirsch-Stoer
DDP	=	Differential Dynamic Programming
DOPRI8	=	Dormand-Prince Runge-Kutta 8(7)
IKAROS	=	Interplanetary Kite-craft Accelerated by Radiation Of the Sun
JAXA	=	Japan Aerospace Exploration Agency
LEO	=	Low Earth Orbit
MEE	=	Modified Equinoctial Elements
NASA	=	the National Aeronautics and Space Administration
NEA	=	Near-Earth Asteroid
NLP	=	Nonlinear Programming
RK	=	Runge-Kutta
SDC	=	Static/Dynamic Control
SEP	=	Solar Electric Propulsion
STM	=	State Transition Matrix
Tudat	=	TU Delft Astrodynamics Toolbox

Symbols

a	=	Semi-major axis
e	=	Eccentricity
$\mathcal{H}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$	=	Inertial heliocentric reference frame
i	=	Inclination
v	=	Velocity
α	=	Solar-sail cone angle
β	=	Solar-sail lightness number
δ	=	Solar-sail clock angle
ν	=	True anomaly
ω	=	Argument of periapsis [rad]
Ω	=	Argument of ascending node [rad]
$\earth$	=	Earth

List of Figures

List of Tables

1 Introduction

This chapter introduces the background and novelty of the thesis: it generally summarizes the previously performed literature study, which was required to define the research questions of this work. Firstly, a general overview of solar sailing and differential dynamic programming is given in Section 1.1 and Section 1.2, respectively. Subsequently, the research objective is formulated in Section 1.4. Furthermore, the research questions are given in Section 1.5. Finally, Section 1.6 provides the structure of this thesis report.

1.1. Solar sailing

This section gives an introduction in the concept of solar sailing and demonstrates its relevance. Firstly, the background and history of solar-sail technology, as well as solar-sail applications, are given in Section 1.1.1, whereafter solar-sail trajectory optimization is discussed in Section 1.1.2

1.1.1. Background

Solar sailing is an ingenious form of low-thrust propulsion that has become increasingly popular over the last years. A large, lightweight, highly reflective sail is deployed to reflect the photons emitted by the Sun or another radiating body, where the transfer of momentum from the photons to the sail propels the spacecraft forward [1]. An example of a solar-sail spacecraft with a square sail can be seen in Figure 1.1. Solar sails are not limited by a finite amount of propellant; the continuous acceleration that can be generated unlocks a larger scope of high-energy mission possibilities, an extension of the mission lifespan, and a reduced mission cost in contrast to conventional propulsion methods [2]. These advantages are only limited by the lifetime of the sail itself [3]. Possible applications of solar sailing include missions involving close, high-inclination orbits around the Sun, cycler missions to Mercury, Venus, or Mars, small-body rendezvous missions, exotic non-Keplerian orbits, and fast solar-system escape missions [1]. The benefits of solar sails can especially pay off for interplanetary or interstellar missions, since these are generally high-energy and long-duration missions.

As early as the 1920s, the concepts of 'using tremendous mirrors of very thin sheets' and 'using the pressure of sunlight to attain cosmic velocities' for space mission applications were already discussed by Tsiolkovsky and Tsander [1]. In the 1950s these ideas became more realistic when proposed and published in popular literature by Carl Wiley [4]. However, it was not until 2010 that the first solar-sail mission in interplanetary space was deployed successfully: the Japanese space agency's (JAXA) IKAROS (Interplanetary Kite-craft Accelerated by Radiation Of the Sun) [5]. A 20 m-span sail was used to perform a flyby of Venus. Key concepts of solar sailing were demonstrated successfully, including the deployment of a large sail, the measurement of the acceleration experienced by the sail due to solar radiation pressure, and the general guidance and navigation of a solar-sail mission through interplanetary space. After IKAROS, more solar-sail missions followed. NASA successfully launched its own first solar sail in 2011, NanoSail-D2, where the deployment and de-orbit of a solar sail in low Earth orbit (LEO) was demonstrated successfully [6]. In 2019, the Planetary Society launched LightSail-2 [7] to demonstrate controlled solar sailing with a CubeSat and orbit raising from LEO, and, more recently, NASA's NEA Scout was launched (2022) [8]. The latter will serve as a robotic reconnaissance mission

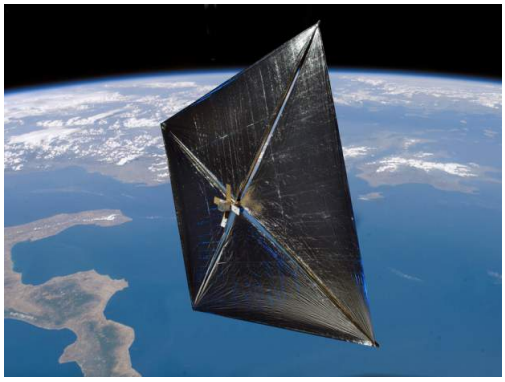


Figure 1.1: Artist's impression of NASA's NanoSail-D2 solar-sail mission[1]

to near-Earth asteroids. Many more solar-sail missions are still to come. For instance, NASA's ACS3, which will showcase advances in solar-sail technology, is planned for 2023 [9]. The combination of using new materials and deployment capabilities will help gain insight for future large-scale solar-sail missions. In 2016, the well-funded project Breakthrough Starshot was founded: a proof-of-concept for the development of a fleet of solar-sail CubeSats that are able to reach the Alpha Centauri star system [10]. Since boom-deployment technology for solar sails is advancing, the propulsion method will gain more and more support [1]. Especially with upcoming mission applications, improved boom technology will ensure that solar sailing can compete with and even outperform other forms of propulsion, such as solar-electric propulsion (SEP) [11].

1.1.2. Solar-sail trajectory optimization

Although thrust is generated continuously by a solar sail, with current sail technology the instantaneous amount of thrust is very small [12]. It can take a long time to achieve the required speed or attain the desired orbit, which reduces the applicability of solar sailing when compared to conventional propulsion methods. Hence, it is important to overcome this limitation by finding the optimal sail attitude that results in the most efficient solar-sail trajectory possible. Analytical solutions exist in the form of locally optimal steering laws that maximize the instantaneous rate of change of a specific orbital element [1]. These laws maximize the projection of the solar-radiation pressure force along a vector of functions of the solar-sail orbital elements. However, once the problem at hand becomes more complex, in the case of constrained solar-sail trajectory optimization, these analytical laws do not suffice.

Optimal control algorithms have the potential to find these, more complicated to solve, optimal trajectories. The optimization of a low-thrust propulsion trajectory, such as a solar-sail trajectory, is a nonlinear, constrained optimal control problem which can be solved indirectly or directly. With an indirect approach, the problem is converted to a boundary-value problem, where differential equations need to be solved with certain boundary conditions [13]. On the other hand, with a direct approach an infinite-dimensional problem gets transcribed to a finite-dimensional problem: the problem is discretized. In contrast to direct methods, indirect methods require explicit derivation of all the necessary conditions [14]. With indirect methods a more refined solution can be found. However, as a consequence of the practical difficulties arising with indirect methods, direct approaches are more widely used. In addition, direct methods are more robust and easy to implement. The optimization of low-thrust heliocentric trajectories is mainly achieved through the use of direct nonlinear programming (NLP) techniques, which are often employed through direct collocation or shooting methods [15, 16]. These direct transcription methods are easy to implement, yet may result in suboptimal solutions due to the approximations made in the discretization process [17]. The techniques may struggle to find the global optimum, especially in high-dimensional problems. Furthermore, evolutionary algorithms are often used to approach global optima [18, 19]. However, drawbacks in these methods are that the number of variables must be small and the required computational effort may increase largely for more complex, high-dimensional problems [16].

[1] NASA, https://www.nasa.gov/mission_pages/tdm/solarsail/index.html, retrieved January 2023

While various methods have been used in previous studies to achieve optimal low-thrust (solar-sail) trajectories, it is still required to research new optimization methods that may perform even better in high-dimensional problems, while limiting the computational effort. As Section 1.1.1 concluded, many applications of solar sailing include long-duration, high-dimensional interplanetary trajectories, and a robust, efficient, and accurate method is required for the optimization of these trajectories.

1.2. Differential dynamic programming

This section introduces differential dynamic programming and demonstrates its relevance, firstly by giving an overview of the method's main principles in Section 1.2.1 and secondly by discussing previous research in the method Section 1.2.2.

1.2.1. Overview

In general, an optimal control algorithm finds the optimal solution for the controls of a dynamical system over time by minimizing (or maximizing) a certain cost function [20]. An efficient and largely known optimization method used in many optimal control algorithms, is *dynamic programming*. This method makes use of Bellman's Principle of Optimality, where a problem is simplified by dividing it into a sequence of sub-problems that are solved recursively [21]. As a result, conventional dynamic programming requires a tremendous amount of storage space, especially when applied to problems with large amounts of segments, as is the case with the optimization of long-duration low-thrust interplanetary trajectories [20]. *Differential* dynamic programming (DDP), on the other hand, makes sacrifices in terms of global optimality of the solution by restricting the search for solutions in a quadratic trust region around a nominal solution [20]. DDP is an optimal control method, which has the benefit that its computational effort scales only linearly with the size of the problem [22]. In contrast, most optimization methods exhibit an exponential effort increase. Mainly due to this quality, DDP can handle large problems with many control variables quite well. In addition, DDP is very competent in handling complex trajectory constraints, which characterize a solar-sail problem.

The DDP algorithm generally functions as follows: firstly, in the "forward sweep" of the algorithm, an initial guess for the control law is used to forward integrate the initial state. The algorithm relies on Bellman's Principle of Optimality, which essentially argues that within an optimal trajectory, regardless of the starting point, the remaining trajectory must be optimal as well [21]. In the "backward sweep", which is based on this principle, DDP solves the problem recursively by successive backward quadratic expansions of the cost function in the neighbourhood of a nominal trajectory, which is the resulting trajectory from the forward sweep [22]. A new control law is found and used to forward integrate the initial state in the forward sweep once again. The entire procedure is repeated for multiple iterations until the algorithm is converged. The convergence of algorithm relies on multiple performance-influencing settings.

1.2.2. Previous research and applications

The DDP method was first introduced for trajectory optimization in Reference [20] and extended on and improved in many modern applications. An example is the Static/Dynamic Optimal Control (SDC) algorithm, with which DDP first earned its place in low-thrust trajectory design [23]. Reference [24] used an algorithm based on DDP to successfully find optimal low-thrust trajectories to asteroids. Still, both implementations had drawbacks in the form of slow convergence or increasing problem non-linearity. The more recent *hybrid* differential dynamic programming algorithm improves on classical DDP by using nonlinear mathematical programming techniques and parallel-computed state transition matrices (STMs) to decouple the dynamics from the optimization [22, 25]. This method has proven to be successful with high-dimensional low-thrust trajectory optimization [15, 26, 27, 28, 29, 30, 31]. Consequently, Reference [32] was the first to use DDP successfully for the optimization of a many-revolution Earth-centered solar-sail trajectory. Here, DDP found more or equally optimal solutions than state-of-the-art locally optimal steering laws. From the fact that most solar-sail applications are proposed in the interplanetary flight regime and the success of DDP in previous work, it makes sense to investigate the applicability of DDP to interplanetary solar-sail trajectories.

1.3. Solar sailing to Mars

One currently significant subject within interplanetary trajectories is the journey to Mars. To give a general understanding of the applicability of DDP to interplanetary solar-sail trajectories in this work, a test case was selected involving the transfer to Mars. This section demonstrates the relevance of this test case and previous research into the subject.

Currently, solar sailing is not a feasible propulsion method in contrast to conventional propulsion methods for a Mars mission; however, the mass-delivery potential of solar-sail cyclers can compete with other propulsion methods if the technology advancement proceeds as expected [33]. Multiple reasons highlight the relevance of a Mars mission:

- Search for life
  There are indicators that Mars could have once harboured life [34]. Learning more about life on Mars could give mankind a better general understanding of how life can exist on Earth or other planets.
- Understanding of the planet and its evolution
  Studying the evolution of Mars could give a better comprehension of the evolution and future of Earth, the solar system, and the beginning of the universe.
- Preparation for future space exploration
  Mars could become a base for space exploration of the outer Solar System or beyond. Moreover, Mars is the only planet in the Solar System where humans have the potential to survive [35, 36], especially since some believe Mars would be a viable second option after humankind exhausts resources on Earth [37].

Previous research found time-optimal solar-sail cyclers between the Earth-Mars libration points for far-term sail technology, which makes solar sailing a closer competitor to conventional propulsion methods [38, 39, 40]. In other papers, Earth-Mars orbit transfers (trajectories from the orbit of Earth to the orbit of Mars) using solar sailing were found in a reasonable amount of time, assuming heliocentric coplanar circular orbits for both the planets [41, 42, 43]. The Earth-Mars case requires the use of multiple (terminal) constraints, which makes the optimization problem much more complex. While minimizing the transfer time, an optimal control algorithm must comply with these constraints to target the final position of Mars. DDP has, in previous research, not yet shown its capabilities for such high-dimensional problems for solar-sail trajectory optimization. Therefore, the Earth-Mars case is an excellent test case to investigate the performance of DDP in an interplanetary solar-sail trajectory optimization, where previous research with other optimization methods can be used as a validation.

1.4. Research aim

Section 1.1 and Section 1.2 highlighted the potential of solar-sail propulsion in comparison to conventional propulsion methods and the promising performance of the optimal control algorithm DDP in high-dimensional low-thrust trajectory optimization. The sections concluded that since most applications of solar sailing are proposed in the interplanetary flight regime, it makes sense to investigate the applicability of DDP to interplanetary solar-sail trajectories. One currently significant subject within interplanetary trajectories is the journey to Mars (see Section 1.3). The significance of the Earth-Mars transfer originates from the search for life on Mars, understanding of the origin and transformation of the planet and the solar system, and preparation for future space exploration and immigration [34, 35, 36, 37]. The relevance of the Earth-Mars mission and the fact that the capabilities of DDP, not yet shown for solar-sail trajectory optimization, can be demonstrated, make an Earth-Mars mission an interesting problem to investigate. As a result, this thesis work will focus on achieving the following research aim:

> *To evaluate the performance of DDP for interplanetary solar-sail trajectory optimization by comparing found solutions with known solutions from locally optimal steering laws in the interplanetary regime and investigating its efficiency for the optimization of an interplanetary solar-sail test case, specifically an Earth-Mars transfer.*

1.5. Research questions

From the findings of the previous sections and with regards to the research aim stated in Section 1.4, the main research question of the thesis is formulated as

"What is the performance of DDP for the optimization of interplanetary solar-sail trajectories?"

In order to answer the main question, a set of four subquestions are defined. First of all, since the true optimum is not known for many high-dimensional interplanetary problems, the optimality of the solutions found by DDP can be measured by validating the results against locally optimal steering laws.

1. How does DDP perform in terms of optimality of the solution when compared to locally optimal steering laws in the interplanetary regime?

Following up, the DDP algorithm can be tweaked and analyzed for different settings. To fully understand the performance of DDP in the interplanetary flight regime, a sensitivity analysis is required for different initial guesses and algorithm settings. Therefore, the following two research questions are introduced:

2. What is the effect on the optimality of the solution and the run time when using different initial guesses for the optimization?

3. What is the effect on the optimality of the solution and the run time when using different settings for the DDP algorithm?

Finally, after an extensive analysis on the robustness of the DDP algorithm in the handling of varying optimization problems and to fully showcase the method's performance, DDP is applied to a realistic interplanetary test case where multiple terminal constraints are involved:

4. How does DDP perform in terms of optimality of the solution and meeting the constraints for an Earth-Mars orbit transfer with respect to solutions known from literature?

The answers to these questions give an idea of how the performance of DDP will change in cases with different problem definitions or settings. Additionally, they give an overview of the method's general robustness and efficiency, and ensure an extensive analysis.

1.6. Report outline

The research questions specified in Section 1.5 will be addressed in Chapter 2. This chapter is written in the form of a journal article, specifically in the format of the American Institute of Aeronautics and Astronautics (AIAA)[2]. The journal article will start with a short abstract and concise version of this introduction, whereafter the used dynamical models will be discussed. Next, the theoretical background of DDP will be given, followed by the validation of the implemented algorithm. Subsequently, the solar-sail Earth-Mars test case will be introduced and the results and conclusions of the research are discussed. In Chapter 3, the conclusions to the thesis itself will be given in the form of answers to the research questions, in addition to recommendations for future research. In Appendix A, the verification and validation procedures are stated for the dynamical models and numerical methods used in this thesis work.

[2]Retrieved from https://www.aiaa.org/Tech-Presenter-Resources/, September 2022

2

Journal article

Differential dynamic programming applied to interplanetary solar-sail trajectory optimization

R.A. Martens *
Delft University of Technology, 2628 CD, Delft, The Netherlands

Recent studies have shown the feasibility of *differential dynamic programming* (DDP) in optimizing Earth-centered solar-sail trajectories. In order to further demonstrate the ability of DDP in the optimization of solar-sail trajectories, this work investigates the performance of DDP for optimizing interplanetary solar-sail trajectories. The selected dynamical framework is based on the two-body problem, augmented with an ideal solar-sail force model. A superior numerical performance is obtained for the optimization algorithm by propagating the state in modified equinoctial elements and applying a Sundman transformation to change the independent variable from time to the true anomaly. The developed algorithm finds similar or more optimal solutions than locally optimal steering laws for the maximization of different orbital elements. In addition, constrained time-optimal Earth-Mars orbital transfers are investigated for different sail performance levels. The DDP algorithm is proven to be efficient and robust for different optimization settings and initial guesses for solar-sail trajectory optimization in the interplanetary regime.

Nomenclature

Abbreviations

ACS3	=	Advanced Composite Solar Sail System
AU	=	Astronomical Unit
DDP	=	Differential Dynamic Programming
DOPRI8	=	Dormand-Prince Runge-Kutta 8(7)
ER	=	Expected Reduction
IKAROS	=	Interplanetary Kitecraft Accelerated By Radiation Of The Sun
JAXA	=	Japanese Space Agency
MEE	=	Modified Equinoctial Elements

u	=	Control parameter
$\boldsymbol{u}$	=	Control vector
u^L	=	Lower control bound
u^U	=	Upper control bound
$\boldsymbol{V}$	=	Dummy vector
$\boldsymbol{W}$	=	Dummy vector
$\boldsymbol{x}$	=	State vector
$\boldsymbol{x_0}$	=	Initial state vector
$\boldsymbol{X}$	=	Augmented state vector

Greek Symbols

*Graduate Student, Department of Astrodynamics and Space Missions, Faculty of Aerospace Engineering, roos_martens@hotmail.com

NEA	=	Near-Earth Asteroid
STM	=	State Transition Matrix
STT	=	State Transition Tensor
TRQP	=	Trust-Region Quadratic Subproblem

Latin Symbols

a	=	Semi-major axis [km]
$\boldsymbol{A}$	=	Feedforward control matrix
$\tilde{\boldsymbol{A}}$	=	Perturbation-mapping matrix
$\boldsymbol{b}$	=	Two-body motion vector
$\boldsymbol{B}$	=	Feedback control matrix
$\boldsymbol{D}$	=	Feedback control matrix
e	=	Eccentricity
$f, g, h, \tilde{k}$	=	Modified equinoctial elements
F	=	Dynamical system
$\boldsymbol{F}$	=	Dynamics matrix
$\boldsymbol{G}$	=	Scaling matrix
$\tilde{h}$	=	Logistic growth rate
H	=	Heaviside function
$\mathcal{H}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$	=	Inertial heliocentric reference frame
i	=	Inclination [rad]
I	=	Tuning parameter for MEE
J	=	Cost function
k	=	Stage
L	=	True longitude [rad]
$\hat{\boldsymbol{n}}$	=	Sail normal unit vector
p	=	Semi-latus rectum [km]
q, ζ, s	=	Auxiliary modified equinoctial elements
r	=	Distance of spacecraft to the Sun [km]
$\boldsymbol{r}$	=	Position vector [km]
$\mathcal{S}(\hat{\boldsymbol{r}}, \hat{\boldsymbol{s}}, \hat{\boldsymbol{q}})$	=	Spacecraft-centered reference frame
t	=	Time [s]
α	=	Sail cone angle [rad]
β	=	Sail lightness number
δ	=	Sail clock angle [rad]
$\boldsymbol{\delta}$	=	Perturbations vector (in $\mathcal{H}$)
Δ	=	Trust-region radius
$\boldsymbol{\Delta}$	=	Perturbations vector (in $\mathcal{S}$)
ϵ_1	=	Reduction-ratio tolerance
ϵ_2	=	Optimality tolerance
ε	=	Relative difference
η	=	Sundman transformation variable
κ	=	Trust-region scaling parameter
λ	=	Lagrange multiplier
$\boldsymbol{\lambda}$	=	Vector of Lagrange multipliers
$\boldsymbol{\nabla}$	=	Sundman-transformed dynamics matrix
μ	=	Gravitational parameter [$\mathrm{km^3 s^{-2}}$]
ν	=	True anomaly [rad]
ρ	=	Reduction ratio
Σ	=	Penalty matrix
ϕ	=	Objective function
$\boldsymbol{\Phi}$	=	State transition matrix
$\boldsymbol{\psi}$	=	Terminal constraints
ω	=	Argument of periapsis [rad]
Ω	=	Argument of ascending node [rad]

Additional subscripts

E	=	Specific orbital energy
$grav$	=	Gravitational acceleration
N	=	Final stage of trajectory
SRP	=	Solar radiation pressure
$\oplus$	=	Earth
$\mars$	=	Mars
$\odot$	=	Sun

I. Introduction

Solar sailing is a propulsion method where photons impose an acceleration on a thin, highly-reflective surface. This transfer of momentum propagates a spacecraft with such a surface, or sail, forward. The concept of solar sailing was already discussed in the 1920s by Tsiolkovsky and Tsander [1]. Nonetheless, it was not until 2010 that the first solar-sail mission in interplanetary space was deployed successfully: the Japanese Space Agency (JAXA) launched IKAROS (Interplanetary Kite-craft Accelerated by Radiation Of the Sun), which successfully executed a fly-by of Venus [2]. More missions followed that successfully demonstrated solar-sail technology, such as NASA's NanoSail-D2 (2011) [3], the Planetary Society's LightSail-2 (2019) [4], and NASA's Near-Earth Asteroid (NEA) Scout (2022) [5]; others are being planned, including NASA's Advanced Composite Solar Sail System (ACS3) (2023) [6]. Contrary to conventional propulsion methods, solar sails are not limited by a finite amount of propellant [1]. The continuous acceleration that can be generated unlocks a larger scope of mission possibilities, an extension of the mission lifespan, and reduced mission cost [7] - only limited by the lifetime of the sail itself [8]. Solar sailing particularly thrives in high-energy and long-duration missions in the interplanetary regime, such as highly non-Keplerian orbits [9, 10], solar-system rendezvous missions [11–13], high-latitude polar-observation missions for the Sun or another celestial body [14, 15], space-weather warning missions [16], or solar-photonic assist missions to the outer solar system or beyond [17, 18].

Nonetheless, the instantaneous thrust generated by a solar sail is still relatively small with current sail technology. Flight times of solar-sail missions need to be improved on to increase the feasibility of solar sailing in comparison to conventional propulsion methods [12]. A robust, efficient, and accurate method is required for the optimization of high-dimensional low-thrust long-duration trajectories, e.g., interplanetary solar-sail trajectories. Analytical solutions exist for interplanetary solar-sail trajectories in the form of locally optimal steering laws that maximize the instantaneous rate of change of a specific orbital element [1]. However, analytical control laws do not suffice when the complexity of the problem at hand increases, as with the involvement of constraints or the optimization of multiple orbital elements at once. Optimal control algorithms have the potential to find these high-dimensional low-thrust trajectories, which are mostly nonlinear, constrained optimal control problems. These problems are mainly solved through the use of direct nonlinear programming (NLP) techniques, which are often employed through direct collocation or shooting methods [19, 20]. These direct transcription methods are easy to implement, yet may result in suboptimal solutions due to the approximations made in the discretization process [21]. Especially in high-dimensional problems the techniques may struggle to find the global optimum. Furthermore, evolutionary algorithms are often used to approach global optima [22, 23]. However, drawbacks in these methods are that the number of variables must be small and the required computational effort may increase largely for more complex, high-dimensional problems [20].

Differential dynamic programming (DDP) is an optimal control method that has the benefit that its computational effort scales only linearly with the size of the problem, in contrast to the exponential effort increase that most methods exhibit [24]. Mainly due to this quality, DDP can successfully handle large problems with many control variables.

Furthermore, DDP is competent in handling complex trajectory constraints, which characterize a solar-sail problem. However, the DDP algorithm does make sacrifices in terms of global optimality of the solution by restricting the search for solutions in a quadratic trust region around a nominal solution [25]. The algorithm was first introduced in Reference [25], and extended on and improved in modern applications in References [26–30]. The more recent *hybrid* differential dynamic programming algorithm improves on classical DDP by using nonlinear mathematical programming techniques and parallel-computed state transition matrices (STMs) to decouple the dynamics from the optimization [24, 31]. This method has proven to be successful for high-dimensional low-thrust trajectory optimization [19, 32–34]. Consequently, the algorithm was used for the first time for *solar-sailing* in a work that proved the robustness and ability of DDP to produce optimal many-revolution Earth-centered solar-sail trajectories [35].

All in all, previous research indicates the need for globally-optimal solutions in complex high-dimensional solar-sail problems [12]. Reference [35] displayed the potential of DDP for solar-sail trajectory optimization for the application of Earth-bound solar-sail missions, while most solar-sail applications are proposed in the interplanetary regime [1]. DDP is found to be a robust, efficient trajectory-optimization method when applied to similar problems. Thus, the purpose of this work is to extend the body of knowledge on solar-sail interplanetary trajectory optimization by investigating the potential of DDP. The algorithm will be validated by comparing its results to locally optimal steering laws for the maximization of specific orbital elements in the interplanetary regime. Furthermore, its robustness to different initial guesses will be explored, as well as its sensitivity to different constraints, objectives, and dynamical or algorithmic parameters. Finally, the performance of the DDP algorithm will be tested on an interplanetary orbital rendezvous trajectory: an Earth-Mars transfer. In particular, the choice for the test case of an Earth-Mars transfer originates from its significance for the search for life on Mars, the understanding of the origin and transformation of the planet and the solar system, and preparation for future space exploration and colonization [36–39], as well as from the fact that the capabilities of DDP, not yet shown for solar-sail trajectory optimization, can be demonstrated in this case.

The paper starts with an explanation of the dynamical framework used in this work, in Section II. Subsequently, the theoretical background and implementation of the DDP algorithm is discussed in Section III. In Section IV, the results of DDP for the maximization of different orbital elements are compared to locally optimal steering laws from Reference [1], whereafter a sensitivity analysis is performed to showcase the robustness and efficiency of the DDP algorithm. The Earth-Mars orbital rendezvous test case is treated in Section V. Finally, the conclusions of this work are given in Section VI.

II. Dynamics

This section elaborates on the definition of the dynamics used to describe the propagation of the interplanetary solar-sail trajectories designed in this work. In Subsection II.A, the representation of the spacecraft's state is discussed. Considering the main focus of the research, which is the applicability of DDP to interplanetary solar-sail trajectory

optimization, a relatively simple dynamical framework is adopted, namely that of the two-body problem augmented with the perturbing acceleration induced by an ideal solar sail. In particular, this gravitational-acceleration model and solar-sail model are described in Subsection II.B and Subsection II.C, respectively. Furthermore, it is advantageous to integrate the equations of motion in a singularity-free, mainly slowly-changing coordinate set [40, 41]. In previous work, the rapid dynamics associated with the use of Cartesian coordinates required a dense discretization [19]. In this work, the numerical performance of the DDP algorithm was enhanced by using modified equinoctial elements (MEE). Hence, in Subsection II.D MEE are introduced. The section concludes with an overview of the Sundman transformation (Subsection II.E).

A. Spacecraft State

The dynamics of the system are defined in an inertial heliocentric reference frame, $\mathcal{H}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$. This reference frame consists of three right-handed orthonormal vectors $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$ with its origin at the center of the Sun. Specifically, $\hat{\boldsymbol{x}}$ points to the Vernal Equinox, $\hat{\boldsymbol{z}}$ is pointed perpendicular to the ecliptic plane, and $\hat{\boldsymbol{y}}$ is pointed perpendicular to the plane formed by the other two unit vectors. In order to define the spacecraft's state during optimization, DDP uses an augmented version of the state vector, where a control vector is appended to the state vector:

$$\boldsymbol{X}_{xyz} = \begin{bmatrix} \boldsymbol{x}_{xyz} & t & \boldsymbol{u} \end{bmatrix}^T = \begin{bmatrix} \boldsymbol{r} & \dot{\boldsymbol{r}} & t & \boldsymbol{u} \end{bmatrix}^T = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & t & \alpha & \delta \end{bmatrix}^T \tag{1}$$

In Eq. 1, the augmented state vector, $\boldsymbol{X}_{xyz}$, consists of the state vector, $\boldsymbol{x}_{xyz}$, and the control vector, $\boldsymbol{u}$. In its turn, the state vector involves the spacecraft's position vector, $\boldsymbol{r}$, and velocity vector, $\dot{\boldsymbol{r}}$. The position and velocity vector are described in Cartesian coordinates (x, y, z). The overhead dot denotes the derivative with respect to time. The current time, t, is also included in the augmented state vector for use with the Sundman transformation (see Subsection II.E). Additionally, the control vector is composed of the cone and clock control angles of the solar sail, α and δ, respectively. These control angles are defined in Subsection II.C. Subsequently, the derivative of the augmented state with respect to time can be written as

$$\dot{\boldsymbol{X}}_{xyz} = \begin{bmatrix} \dot{\boldsymbol{r}} & \ddot{\boldsymbol{r}}_{grav} + \ddot{\boldsymbol{r}}_{SRP} & 1 & \dot{\alpha} & \dot{\delta} \end{bmatrix}^T \tag{2}$$

where $\ddot{\boldsymbol{r}}_{grav}$ is the gravitational acceleration and $\ddot{\boldsymbol{r}}_{SRP}$ is the perturbing solar-sail acceleration.

B. Two-body Dynamics Model

The selected framework for the gravitational acceleration of the spacecraft is based on the two-body problem. Here, the Sun is the central body and assumed to be a point mass. The gravitational perturbations due to other celestial objects are neglected. Thus, the gravitational acceleration experienced by the spacecraft in reference frame $\mathcal{H}$ is given as

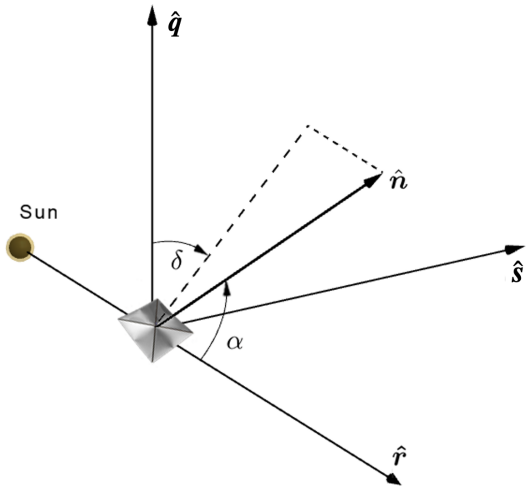


Fig. 1 The attitude angles of the solar sail in the spacecraft-centered reference frame.

$$\ddot{\boldsymbol{r}}_{grav} = -\frac{\mu_\odot}{||\boldsymbol{r}||^3}\boldsymbol{r} \tag{3}$$

where $\mu_\odot$ is the gravitational parameter of the Sun and $||\boldsymbol{r}||$ is the distance of the spacecraft to the center of the Sun. The values for $\mu_\odot$ and other typical parameters used in this paper are given in Table 1.

Table 1 Dynamical parameters [42].

Variable	Value
Gravitational parameter Sun ($\mu_\odot$) [km^{3}s^{-2}]	$1.3271244004193929 \times 10^{17}$
Astronomical unit (AU) [km]	149597871
Julian year [s]	31557600

C. Ideal Solar-sail Model

The acceleration generated by the solar radiation pressure is modeled by assuming a flat, ideal solar sail. This implies that all incident photons on the sail are specularly reflected; absorption and emissivity of the sail film are neglected [1]. Of importance is the unit vector defining the direction of the solar-sail induced acceleration, $\hat{\boldsymbol{n}}$, directed normal to the solar-sail surface, which is the result of the assumption of a flat, ideal solar sail. This vector $\hat{\boldsymbol{n}}$ is depicted in Fig. 1 and is commonly given in a spacecraft-centered reference frame $\mathcal{S}(\hat{\boldsymbol{r}}, \hat{\boldsymbol{s}}, \hat{\boldsymbol{q}})$. Specifically, $\hat{\boldsymbol{r}}$ is pointed, as before, from the center of the Sun to the spacecraft, $\hat{\boldsymbol{q}}$ is directed normal to the orbit plane, and $\hat{\boldsymbol{s}}$ completes the orthonormal set of unit vectors. Typically, $\hat{\boldsymbol{n}}$ is described in reference frame $\mathcal{S}$ as follows [1]:

$$\hat{\boldsymbol{n}}|_{\mathcal{S}} = \begin{bmatrix} \cos(\alpha) \\ \sin(\alpha)\sin(\delta) \\ \sin(\alpha)\cos(\delta) \end{bmatrix} \tag{4}$$

The cone angle, α, is defined as the angle between the Sun line, which lies in the direction of $\hat{\boldsymbol{r}}$, and the normal to the sail surface, $\hat{\boldsymbol{n}}|_{\mathcal{S}}$. The clock angle, δ, on the other hand, is defined as the angle between the projection of $\hat{\boldsymbol{n}}|_{\mathcal{S}}$ on the plane normal to the Sun line and $\hat{\boldsymbol{q}}$. A solar sail cannot generate a force component in the direction of the Sun [1]. Therefore, to model this, the cone angle is constrained to $\alpha \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. Additionally, the clock angle is constrained to $\delta \in [0, 2\pi]$. Subsequently, in order to obtain the direction of the solar radiation pressure acceleration in reference frame $\mathcal{H}$, the sail normal vector can be transformed with a rotation matrix $\boldsymbol{Q}$ through

$$\boldsymbol{Q} = \begin{bmatrix} \hat{\boldsymbol{r}} & \hat{\boldsymbol{s}} & \hat{\boldsymbol{q}} \end{bmatrix} = \begin{bmatrix} \frac{\boldsymbol{r}}{||\boldsymbol{r}||} & \frac{(\boldsymbol{r}\times\dot{\boldsymbol{r}})\times\boldsymbol{r}}{||(\boldsymbol{r}\times\dot{\boldsymbol{r}})\times\boldsymbol{r}||} & \frac{\boldsymbol{r}\times\dot{\boldsymbol{r}}}{||\boldsymbol{r}\times\dot{\boldsymbol{r}}||} \end{bmatrix} \tag{5}$$

$$\hat{\boldsymbol{n}} = \hat{\boldsymbol{n}}|_{\mathcal{H}} = \boldsymbol{Q}\ \hat{\boldsymbol{n}}|_{S} \tag{6}$$

Using Eq. 6, the acceleration generated by the solar radiation pressure imposed on the sail can be found [1]:

$$\ddot{\boldsymbol{r}}_{SRP} = \beta \frac{\mu_{\odot}}{||\boldsymbol{r}||^2} (\hat{\boldsymbol{r}} \cdot \hat{\boldsymbol{n}})^2\, \hat{\boldsymbol{n}} \tag{7}$$

In Eq. 7, β is the solar-sail lightness number, which is defined as the dimensionless ratio of the solar radiation pressure acceleration to the solar gravitational acceleration. Typical (predicted) lightness numbers for previous and upcoming missions are given in Table 2. For realistic near-term solar-sail technology, lightness numbers reach up to approximately $\beta = 0.01$, whereas far-term estimations reach up to $\beta = 0.1$ [43]. For the purpose of this research both lightness numbers will be considered.

Table 2 Typical mission parameters [35, 44].

Mission	IKAROS (2010)	NanoSail D2 (2011)	LightSail 2 (2019)	NEA Scout (2022)	ACS3
Lightness number (β)	9.8×10^{-4}	3.8×10^{-3}	9.8×10^{-3}	1.1×10^{-2}	8.4×10^{-3}

D. Modified Equinoctial Elements

The modified equinoctial elements are a set of orbital elements valid for circular, elliptic, and hyperbolic orbits [45]. This element set is advantageous for integrating the equations of motion, since there is only one fast-changing element [19, 34, 45]. This characteristic is similar to the Keplerian element set, however, unlike Keplerian elements, the MEE are non-singular save from an inclination of $i = \pi$. This singularity can be resolved by defining the orbit as posigrade or

retrograde [46]. The MEE are given as

$$\begin{cases} p = a(1 - e^2) \\ f = e\cos(\omega + I\Omega) \\ g = e\sin(\omega + I\Omega) \\ h = \tan(\frac{i}{2})^I \cos(\Omega) \\ \tilde{k} = \tan(\frac{i}{2})^I \sin(\Omega) \\ L = \omega + I\Omega + \nu \end{cases} \tag{8}$$

The element set in Eq. 8 consists of five slowly-changing variables, the semi-latus rectum, p, and f, g, h, and $\tilde{k}$. The last variable to complete the set is the true longitude, L, which is the only fast-changing variable. The MEE makes use of the classical orbital elements: the semi-major axis, a, the eccentricity, e, the inclination, i, the argument of ascending node, Ω, the argument of periapsis, ω, and the true anomaly, ν. The variable I is introduced, in h and $\tilde{k}$ as an exponent and in L as a multiple, to indicate a posigrade or retrograde orbit by setting it to $+1$ or -1, respectively. Equation 8 defines the state in modified equinoctial elements, $\boldsymbol{x}$, and the augmented state vector from Eq. 1 can be expressed as

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x} & t & \boldsymbol{u} \end{bmatrix} = \begin{bmatrix} p & f & g & h & \tilde{k} & L & t & \alpha & \delta \end{bmatrix}^T \tag{9}$$

Moreover, it is necessary to obtain the derivative of the augmented state vector in MEE, or the modified equinoctial equations of orbital motion. For this purpose, auxiliary MEE are defined [34]:

$$\begin{cases} q = 1 + f\cos(L) + g\sin(L) \\ r = \frac{p}{q} \\ \zeta^2 = h^2 - \tilde{k}^2 \\ s^2 = 1 + h^2 + k^2 \end{cases} \tag{10}$$

The derivative of the state in modified equinoctial elements, $\dot{\boldsymbol{x}}$, is found as follows [34, 47]:

$$\dot{\boldsymbol{x}} = \tilde{\boldsymbol{A}}\boldsymbol{\Delta} + \boldsymbol{b} \tag{11}$$

where $\boldsymbol{\Delta}$ is a vector of perturbing accelerations in reference frame $\mathcal{S}$. The transformation matrix $\tilde{\boldsymbol{A}}$ and the vector $\boldsymbol{b}$ are given as

$$\tilde{\boldsymbol{A}} = \begin{bmatrix} 0 & \frac{2p}{q}\sqrt{\frac{p}{\mu_\odot}} & 0 \\ \sqrt{\frac{p}{\mu_\odot}}\sin(L) & \sqrt{\frac{p}{\mu_\odot}}\frac{1}{q}\left[(q+1)\cos(L)+f\right] & -\sqrt{\frac{p}{\mu_\odot}}\frac{g}{q}\left(h\sin(L)-k\cos(L)\right) \\ -\sqrt{\frac{p}{\mu_\odot}}\cos(L) & \sqrt{\frac{p}{\mu_\odot}}\frac{1}{q}\left[(q+1)\sin(L)+g\right] & \sqrt{\frac{p}{\mu_\odot}}\frac{f}{q}\left(h\sin(L)-k\cos(L)\right) \\ 0 & 0 & \sqrt{\frac{p}{\mu_\odot}}\frac{s^2\cos(L)}{2q} \\ 0 & 0 & \sqrt{\frac{p}{\mu_\odot}}\frac{s^2\sin(L)}{2q} \\ 0 & 0 & \sqrt{\frac{p}{\mu_\odot}}\frac{1}{q}\left(h\sin(L)-k\cos(L)\right) \end{bmatrix} \tag{12}$$

$$\boldsymbol{b} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \sqrt{\mu_\odot p}\left(\frac{q}{p}\right)^2 \end{bmatrix}^T \tag{13}$$

Here, the matrix $\tilde{\boldsymbol{A}}$ transforms the perturbing accelerations in frame $\mathcal{S}$ to a representation in MEE. The vector $\boldsymbol{b}$ represents the two-body acceleration in MEE. Note that the first five modified equinoctial elements are constant when perturbations are absent ($\boldsymbol{\Delta} = 0$). Consequently, the equations of motion will describe perfect two-body motion. To obtain the vector $\boldsymbol{\Delta}$, it is required to transform the perturbing accelerations in frame $\mathcal{H}$, included in the vector $\boldsymbol{\delta}$, to frame $\mathcal{S}$. In this work, the vector $\boldsymbol{\delta}$ only consists of the solar-sail acceleration given in Eq. 7. Consequently, the transformation to $\boldsymbol{\Delta}$ is defined by using the rotation matrix $\boldsymbol{Q}$ given in Eq. 5 as

$$\boldsymbol{\Delta} = \boldsymbol{Q}^T\boldsymbol{\delta} = \boldsymbol{Q}^T\ddot{\boldsymbol{r}}_{SRP} \tag{14}$$

The state in MEE is mapped to a Cartesian position vector, $\boldsymbol{r}$, and velocity vector, $\dot{\boldsymbol{r}}$, as follows [47]:

$$\boldsymbol{r} = \begin{bmatrix} \frac{r}{s^2}\left[(1+\zeta^2)\cos(L)+2hk\sin(L)\right] \\ \frac{r}{s^2}\left[(1-\zeta^2)\sin(L)+2hk\cos(L)\right] \\ \frac{2r}{s^2}\left(h\sin(L)-k\cos(L)\right) \end{bmatrix} \tag{15}$$

$$\dot{\boldsymbol{r}} = \begin{bmatrix} -\frac{1}{s^2}\sqrt{\frac{\mu_\odot}{p}}\left[(1+\zeta^2)(\sin(L)+g)-2hk(\cos(L)+f)\right] \\ -\frac{1}{s^2}\sqrt{\frac{\mu_\odot}{p}}\left[(\zeta^2-1)(\cos(L)+f)+2hk(\sin(L)+g)\right] \\ \frac{2}{s^2}\sqrt{\frac{\mu_\odot}{p}}\left(h(\cos(L)+f)+k(\sin(L)+g)\right) \end{bmatrix} \tag{16}$$

E. Sundman Transformation

Using a Sundman transformation, the independent variable for the integration is transformed from time, t, to true anomaly, ν [34]. This is particularly advantageous for eccentric orbits, since equal steps are used for the integration

instead of largely-varying time steps at periapsis and apoapsis [48]. This ensures comparable error sizes at each stage of the trajectory. In this work, orbits to an eccentricity of $e = 0.2$ are investigated, which makes the use of the Sundman transformation advantageous for the numerical performance of the integration. The Sundman transformation is defined as

$$dt = \frac{1}{\mu_\odot p}\left(\frac{p}{q}\right)^2 d\nu \quad \Longleftrightarrow \quad \eta = \frac{\delta t}{\delta \nu} = \frac{1}{\mu_\odot p}\left(\frac{p}{q}\right)^2 \tag{17}$$

where η is the Sundman transformation variable. Finally, the equations of motion, or the derivative of the state vector with respect to the new independent variable, ν, are written with an overhead circle as

$$\mathring{\boldsymbol{X}} = \frac{\delta \boldsymbol{X}}{\delta \nu} = \frac{\delta \boldsymbol{X}}{\delta t}\frac{\delta t}{\delta \nu} = \eta \dot{\boldsymbol{X}} = \eta \begin{bmatrix} \dot{\boldsymbol{x}} & 1 & \dot{\alpha} & \dot{\delta} \end{bmatrix} \tag{18}$$

III. Differential Dynamic Programming

This section elaborates on the definition and implementation of DDP for interplanetary solar-sail trajectory optimization in the dynamical framework given in Section II. Classical dynamic programming requires a large amount of storage space to find the globally optimal solution [49]. *Differential* dynamic programming, on the other hand, makes sacrifices in terms of global optimality by restricting the search for solutions in a quadratic trust region around a nominal solution [24]. This significantly reduces the required storage space and ensures robustness to initial guesses; however, only local optimality can be guaranteed. Therefore, in previous research [19, 35] a global optimization method, monotonic basin hopping, was applied to the solutions found by DDP. These solutions did not improve significantly, thus, for the purpose of this work no additional global search method is applied. In general, the theoretical basis for the DDP algorithm adopted in this work is based on Reference [24] and Reference [35]. The former allows the inclusion of constraints and increases the robustness of DDP compared to the classical implementation, whereas the latter is the first implementation of DDP for a *solar-sail* trajectory optimization. However, first an explanation of the notation used for the mathematical equations involved in the algorithm is given in Subsection III.A.

In general, DDP seeks the minimum of a cost function, $J(\boldsymbol{x}, \boldsymbol{u}; t)$, where $\boldsymbol{x}(t) = F(\boldsymbol{x}_0, \boldsymbol{u}, t)$ is a state trajectory subject to a dynamical system, F, with an initial state, $\boldsymbol{x}_0$, and a control law, $\boldsymbol{u}(t)$. Reference [24] introduces an augmented Lagrangian cost function for the handling of constraints, which is further discussed in Subsection III.B. DDP requires the trajectory to be discretized in stages, where k indicatesint the stage number. A basic overview of the structure of DDP is given in Fig. 2. Firstly, in the "forward sweep" of the algorithm (Subsection III.C), an initial guess for the control law is used to forward integrate the initial state. The algorithm relies on Bellman's Principle of Optimality, which essentially argues that within an optimal trajectory, regardless of the starting point, or stage, the remaining trajectory must be optimal as well [49]. In the "backward sweep", which is based on this principle, DDP

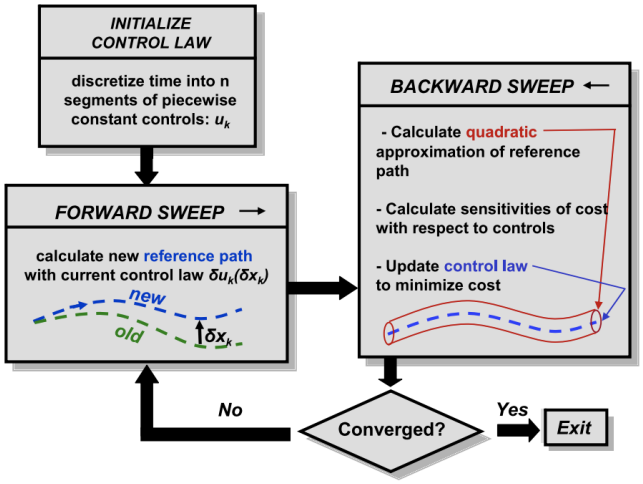


Fig. 2 Optimization flow of Differential Dynamic Programming [24]

solves the problem recursively by successive backward quadratic expansions of the cost function in the neighbourhood of a nominal trajectory, which is the resulting trajectory from the forward sweep [24]. The backward sweep is explained in more detail in Subsection III.D. A new control law is found and used to forward integrate the initial state in the forward sweep once again. The entire procedure is repeated for multiple iterations until the algorithm is converged. Subsections III.E, III.F, and III.G discuss the remaining steps required for convergence to an optimal solution.

A. Notation

The mathematical formulation of DDP consists of many terms that require clarification beforehand. A separate subsection is created for these terms, instead of including them in the nomenclature, since they require a more thorough explanation. As also used previously, bold characters indicate vectors ($\boldsymbol{x}$), an overhead hat indicates a unit vector ($\hat{\boldsymbol{x}}$), and an overhead dot indicates a time derivative ($\dot{\boldsymbol{x}}$). Subscripts are used to denote an index (J_k is a scalar quantity at stage k), or to denote a derivative with respect to the (augmented) state or control vector, e.g., the gradients required for matrices and tensors ($J_{x,k}$ is the derivative with respect to the state $\boldsymbol{x}$ at stage k). Note that $\boldsymbol{x}$ is the spacecraft's state, whereas the capital $\boldsymbol{X}$ is the augmented state.

$\lvert\boldsymbol{X}\rvert$	The cardinality or size of $\boldsymbol{X}$
$\lvert\lvert\boldsymbol{X}\rvert\rvert$	The Euclidean norm or length of $\boldsymbol{X}$
J_k	The cost-to-go at stage k
J^*	The superscript asterisk denotes the optimal value of a parameter, such as J
$J_{V,k}$	The first partial derivative of J w.r.t. the dummy vector variable $\boldsymbol{V}$, of size $n \times 1$, at stage k:

$$J_{V,k} = \nabla_V J_k = \begin{bmatrix} \frac{\delta J_k}{\delta V_1} & \dots & \frac{\delta J_k}{\delta V_n} \end{bmatrix}^T$$

$J_{VV,k}$ The second partial derivative of J w.r.t. the dummy vector variable $\boldsymbol{V}$, of size $n \times 1$, at stage k:

$$J_{VV,k} = \nabla_{VV} J_k = \begin{bmatrix} \frac{\delta J_k}{\delta V_1 \delta V_1} & \cdots & \frac{\delta J_k}{\delta V_1 \delta V_n} \\ \vdots & \vdots & \vdots \\ \frac{\delta J_k}{\delta V_n \delta V_1} & \cdots & \frac{\delta J_k}{\delta V_n \delta V_n} \end{bmatrix}$$

$J_{VW,k}$ The second partial derivative of J w.r.t. the dummy vector variable $\boldsymbol{V}$, of size $n \times 1$, and the dummy vector variable $\boldsymbol{W}$, of size $m \times 1$, at stage k:

$$J_{VW,k} = \nabla_{VW} J_k = \begin{bmatrix} \frac{\delta J_k}{\delta V_1 \delta W_1} & \cdots & \frac{\delta J_k}{\delta V_1 \delta W_m} \\ \vdots & \vdots & \vdots \\ \frac{\delta J_k}{\delta V_n \delta W_1} & \cdots & \frac{\delta J_k}{\delta V_n \delta W_m} \end{bmatrix}$$

$\boldsymbol{X}^i$ The i^{th} entry of vector $\boldsymbol{X}$

$\boldsymbol{\Phi}^{i,a}$ The $(i,a)^{th}$ entry of matrix $\boldsymbol{\Phi}$, where i indicates the row and a indicates the column number

$\boldsymbol{\Phi}^{i,ab}$ The $(i,ab)^{th}$ entry of tensor $\boldsymbol{\Phi}$, where i indicates the row number and ab indicates the column position, which is generally defined by the product of the positions a and b

$\boldsymbol{V}^{\gamma_1}\boldsymbol{V}^{\gamma_1}$ The dot product can be written using Einstein notation, with a dummy index γ_1, as:

$$\boldsymbol{V} \cdot \boldsymbol{V}^T = \boldsymbol{V}^{\gamma_1}\boldsymbol{V}^{\gamma_1} = \sum_{\gamma_1=1}^{n} \boldsymbol{V}^{\gamma_1}\boldsymbol{V}^{\gamma_1} = \boldsymbol{V}^1\boldsymbol{V}^1 + \ldots + \boldsymbol{V}^n\boldsymbol{V}^n$$

$\boldsymbol{F}^{i,\gamma_1}\boldsymbol{\Phi}^{\gamma_1,a}$ Matrix multiplication between square matrices $\boldsymbol{F}$ and $\boldsymbol{\Phi}$, with matrix size $m \times m$, can be written using Einstein notation, with a dummy index γ_1 (a double summation can be described with an extra dummy index, γ_2). The following example, which will appear in Subsection III.D, displays this:

$$\dot{\boldsymbol{\Phi}}^{i,a} = \boldsymbol{F}^{i,\gamma_1}\boldsymbol{\Phi}^{\gamma_1,a} = \sum_{\gamma_1=1}^{m} \boldsymbol{F}^{i,\gamma_1}\boldsymbol{\Phi}^{\gamma_1,a} = \boldsymbol{F}^{i,1}\boldsymbol{\Phi}^{1,a} + \ldots + \boldsymbol{F}^{i,m}\boldsymbol{\Phi}^{m,a}$$

B. Augmented Lagrangian Cost Function

To enable the inclusion of constraints, Reference [24] introduced an augmented Lagrangian method. This paper focuses on the application of DDP to singular phase trajectories with no local stage costs. Consequently, the augmented Lagrangian cost function, J, can be defined as

$$J = \phi + \boldsymbol{\lambda}^T\boldsymbol{\psi} + \boldsymbol{\psi}^T\boldsymbol{\Sigma}\boldsymbol{\psi} \tag{19}$$

where ϕ is a to-be-minimized objective, $\boldsymbol{\lambda}$ is a vector of Lagrange multipliers, $\boldsymbol{\psi}$ is a vector of terminal constraints, and $\boldsymbol{\Sigma}$ is a matrix of penalty weights to add to each constraint. The procedure for the Lagrange multipliers is discussed in Subsection III.E, whereas the other parameters are problem-defined.

C. Forward Sweep

At the start of the first iteration, the initial state is forward integrated by using a suggested initial guess for the nominal control law, $\overline{\boldsymbol{u}}$. For the following iterations the new control law is calculated by $\boldsymbol{u} = \overline{\boldsymbol{u}} + \delta\boldsymbol{u}$. The variation in control law, $\delta\boldsymbol{u}$, is calculated in the backward sweep of the same iteration. The backward sweep is described in more detail in Subsection III.D. If the current iteration is successful, the new control is set as the nominal control law for the next iteration, $\overline{\boldsymbol{u}} = \boldsymbol{u}$. The requirements for an iteration to be successful are discussed in Subsection III.G. The new nominal control law is forward integrated to find the new trajectory, which can be described as a deviation from the nominal trajectory, $\boldsymbol{x} = \overline{\boldsymbol{x}} + \delta\boldsymbol{x}$. This process is repeated until the new trajectory and the control update fulfill the convergence requirements (see Subsection III.G).

To forward integrate the trajectory in the forward sweep, a suitable integrator is required. It is necessary to use a fixed step size integrator to ensure the correct computation of STMs in the DDP algorithm [50]. Using a variable step size causes discontinuity to arise along the STMs, which reduces overall STM accuracy. Reference [51] recommends the Dormand-Prince Runge-Kutta 8(7) integrator (DOPRI8) [52] as an all-round high-accuracy fixed step-size integrator. In addition, References [19, 33–35] used DOPRI8 in combination with DDP before. As a result, in this work the dynamics are integrated with a DOPRI8 integrator as well.

D. Backward Sweep

This section defines the backward sweep, in which the optimal control update, $\delta\boldsymbol{u}_k^*$, is calculated. The algorithm is based on Bellman's Principle of Optimality: instead of calculating the cost over the entire trajectory, a cost-to-go function is evaluated at each stage that calculates the cost of the trajectory from the current to the final stage [49, 53]. The optimal control update is the solution to the recursive minimization of the cost-to-go from stage $k = N - 1$ to stage $k = 0$, with N the total amount of stages [25]. The optimal cost-to-go is written as

$$J_k^*(\boldsymbol{x}_k) = \min_{\boldsymbol{u}_k} \{J_k(\boldsymbol{x}_k)\} \tag{20}$$

where J_k denotes the cost-to-go from the current stage, k, to the last stage, $k = N$, and the superscript asterisk indicates the *optimal* value (see Subsection III.A). When Eq. 20 is solved recursively for each stage until the first stage, $k = 0$, the optimal control update, $\delta\boldsymbol{u}_k^*$, is found. Since the search space is restricted to reduce storage space, the (smooth) cost function can be approximated by a second-order Taylor series expansion around the states, controls, and Lagrange multipliers [24]. The updates for these three elements can be found by minimizing the local quadratic expansion with

respect to the controls; thus taking the derivative and setting it to zero. This results in the following unconstrained feedback control law, where the coefficients of the Taylor-series expansion must be found by recursively traversing the stages:

$$\begin{cases} \delta \boldsymbol{u}_k^* = \boldsymbol{A}_k + \boldsymbol{B}_k \delta \boldsymbol{x}_k + \boldsymbol{D}_k \delta \boldsymbol{\lambda}_k \\ \boldsymbol{A}_k = -J_{uu,k}^{-1} J_{u,k} \\ \boldsymbol{B}_k = -J_{uu,k}^{-1} J_{ux,k} \\ \boldsymbol{D}_k = -J_{uu,k}^{-1} J_{u\lambda,k} \end{cases} \tag{21}$$

The matrix $\boldsymbol{A}$ is a feedforward control matrix, whereas $\boldsymbol{B}$ and $\boldsymbol{D}$ are feedback control matrices: these matrices are stored for use in the forward sweep. Moreover, the variable $\delta \boldsymbol{x}_k$ represents the change in state due to the control update and the variable $\delta \boldsymbol{\lambda}_k$ is the update in Lagrange multiplier. In this control law, subscripts u, x, λ denote the partial derivatives with respect to the corresponding variable, see again Subsection III.A. Note that the derivatives with respect to $\boldsymbol{x}$ and $\boldsymbol{u}$ can both be found from the derivative with respect to the total augmented state, $\boldsymbol{X}$. In particular, the matrices J_{uu} and J_{ux} are submatrices of the second-order derivative with respect to the total augmented state, J_{XX}. The remaining matrices of Eq. 21 are found with the derivatives of the cost-to-go with respect to the Lagrange multipliers. Also, note that $\delta \boldsymbol{u}_k^*$ cannot be calculated directly, since it depends on $\delta \boldsymbol{x}_k$. The two terms have to be calculated simultaneously, using the stored feedforward and feedback matrices, in the forward sweep.

Furthermore, a trust-region quadratic subproblem (TRQP) ensures that $\delta \boldsymbol{x}_k$ remains within the valid region of the quadratic expansion by restricting the variations in Lagrange multipliers and the feedforward matrix $\boldsymbol{A}_k$ [54]. The TRQP also guarantees $J_{uu,k}$ to be positive definite, which verifies the descent direction for $\delta \boldsymbol{u}_k^*$. The TRQP will be discussed in more detail in Subsection III.F.

The required matrices for Eq. 21 can be calculated with the cost-to-go derivatives with respect to the augmented state and the Lagrange multipliers:

$$J_{X,k} = \boldsymbol{\Phi}^T J_{X,k+1}^* \tag{22a}$$

$$J_{\lambda,k} = J_{\lambda,k+1}^* \tag{22b}$$

$$J_{XX,k}^{i,a} = \sum_{\gamma_1=1}^{m} \sum_{\gamma_2=1}^{m} J_{XX,k+1}^{*\gamma_1,\gamma_2} \boldsymbol{\Phi}^{\gamma_1,i} \boldsymbol{\Phi}^{\gamma_2,a} + J_{X,k+1}^{*,\gamma_1} \boldsymbol{\Phi}^{\gamma_1,ia} \tag{22c}$$

$$J_{\lambda\lambda,k} = J_{\lambda\lambda,k+1}^* \tag{22d}$$

$$J_{X\lambda,k} = \boldsymbol{\Phi}^T J_{X\lambda,k+1}^* \tag{22e}$$

The subequations in Eq. 22 require some additional explanation. Firstly, in Eqs. 22a, 22c, and 22d, $\mathbf{\Phi}^{i,a}$ is a first-order state transition matrix and $\mathbf{\Phi}^{i,ab}$ is a second-order state transition tensor (STT); both will be referred to as STMs. STMs are useful because they map the derivatives with respect to $\boldsymbol{x}$ at one stage to the next in terms of perturbations in the state variables: this ensures no integration is needed in the backward sweep of the algorithm [24]. Additionally, Eq. 22c uses dummy indices, γ_1 and γ_2, in a double summation (where $\gamma_{1,2} \in [1, |\boldsymbol{X}|]$) to form the Hessian, $J_{XX,k}$ (see again Subsection III.A for the notation used). Secondly, in Eq. 22b and Eq. 22d, no STM is involved: the optimal cost-to-go derivatives at stage $k+1$ are equal to the cost-to-go derivatives at stage k. Note that these cost-to-go derivatives are not required for the calculation of the local quadratic-expansion coefficients in Eq. 21. $J_{\lambda,k}$ and $J_{\lambda\lambda,k}$ are necessary for the calculation of the Lagrange multiplier update, $\delta\boldsymbol{\lambda}$: see Subsection III.E.

Computing the STMs creates the largest computational load: to reduce the effort, the computations are executed in parallel with the open-source C++ library OpenMP [55]. The calculation of STMs requires the first and second order derivatives of the dynamics with respect to the augmented state vector:

$$\boldsymbol{F}^{i,a} = \frac{\delta \dot{\boldsymbol{X}}^i}{\delta \boldsymbol{X}^a} \tag{23a}$$

$$\boldsymbol{F}^{i,ab} = \frac{\delta^2 \dot{\boldsymbol{X}}^i}{\delta \boldsymbol{X}^a \delta \boldsymbol{X}^b} \tag{23b}$$

The matrix $\boldsymbol{F}^{i,a}$ is formed by taking the derivative of $\dot{\boldsymbol{X}}^i$ (the i^{th} element of $\dot{\boldsymbol{X}}$) with respect to $\boldsymbol{X}^a$. The tensor $\boldsymbol{F}^{i,ab}$ is the derivative of the matrix, $\boldsymbol{F}^{i,a}$, with respect to $\boldsymbol{X}^b$. The values for the derivatives of the dynamics with respect to the augmented state vector can be found analytically*. It is necessary to transform the matrices of Eq. 23 in order for the Sundman transformation (explained in Subsection II.E) to be valid and to reflect the change of independent variable. Of key importance here is Eq. 18, where the time derivative of the augmented state vector, $\dot{\boldsymbol{X}}$, is multiplied by η (see Eq. 17). The matrix $\boldsymbol{F}^{i,a}$ and tensor $\boldsymbol{F}^{i,ab}$ need to be transformed in a similar manner, using the first and second order derivatives of η with respect to the augmented state,

$$\eta^i_X = \frac{\delta \eta}{\delta \boldsymbol{X}^i} \tag{24a}$$

$$\eta^{i,a}_{XX} = \frac{\delta^2 \eta}{\delta \boldsymbol{X}^i \delta \boldsymbol{X}^a} \tag{24b}$$

With Eq. 24, the matrix $\boldsymbol{F}^{i,a}$ and tensor $\boldsymbol{F}^{i,ab}$ are transformed into

$$\mathbf{\Lambda}^{i,a} = \boldsymbol{F}^{i,a}\eta + \dot{\boldsymbol{X}}^i \eta^a_X \tag{25a}$$

*The derivatives of the dynamics were found analytically using Maple worksheets: `https://github.com/roosmartens/thesis_derivatives`

$$\boldsymbol{\Lambda}^{i,ab} = \boldsymbol{F}^{i,ab}\eta + \boldsymbol{F}^{i,a}\eta_x^b + \boldsymbol{F}^{i,b}\eta_X^a + \dot{\boldsymbol{X}}^i\eta_{XX}^{a,b} \tag{25b}$$

Here, $\boldsymbol{\Lambda}^{i,a}$ and $\boldsymbol{\Lambda}^{i,ab}$ are the matrix and tensor resulting from the Sundman transformation. Similar to what is required for the derivative of the augmented state, the matrices are evaluated in Cartesian coordinates before transforming them to MEE, since perturbing accelerations (e.g., the solar-sail acceleration) are generally straightforward in Cartesian coordinates. The method for transforming these matrices to MEE is given in the Appendix of Reference [19]. Finally, the STMs can be calculated with the help of the following differential equations:

$$\dot{\boldsymbol{\Phi}}^{i,a} = \boldsymbol{\Lambda}^{i,\gamma_1}\boldsymbol{\Phi}^{\gamma_1,a} \tag{26a}$$

$$\dot{\boldsymbol{\Phi}}^{i,ab} = \boldsymbol{\Lambda}^{i,\gamma_1}\boldsymbol{\Phi}^{\gamma_1,ab} + \boldsymbol{\Lambda}^{i,\gamma_1\gamma_2}\boldsymbol{\Phi}^{\gamma_1,a}\boldsymbol{\Phi}^{\gamma_2,b} \tag{26b}$$

Equation 26 is a set of differential equations, which are subject to the initial conditions $\boldsymbol{\Phi}_k^{\gamma_i,ab} = \boldsymbol{0}$ and $\boldsymbol{\Phi}_k^{\gamma_i,a} = \boldsymbol{I}$. Here, $\boldsymbol{0}$ is a zeroes matrix, whereas $\boldsymbol{I}$ is the identity matrix - these boundary conditions hold at the start of each stage. The STM at the end of each stage is then found by simultaneous integration with the dynamics of the system during the forward sweep.

Through Eqs. 23 - 26 all terms necessary to calculate Eq. 22 are obtained, assuming that the optimal cost-to-go derivatives of the previous stage, $k+1$, are also known. The next step is to compute the optimal cost-to-go derivatives at the current stage, k, using

$$ER_k = ER_{k+1} + J_{u,k}^T\boldsymbol{A}_k + \frac{1}{2}\boldsymbol{A}_k^T J_{uu,k}\boldsymbol{A}_k \tag{27a}$$

$$J_{x,k}^{*T} = J_{x,k}^T + J_{u,k}^T\boldsymbol{B}_k + \boldsymbol{A}_k^T J_{uu,k}\boldsymbol{B}_k + \boldsymbol{A}_k^T J_{ux,k} \tag{27b}$$

$$J_{\lambda,k}^{*T} = J_{\lambda,k}^T + J_{u,k}^T\boldsymbol{D}_k + \boldsymbol{A}_k^T J_{uu,k}\boldsymbol{D}_k + \boldsymbol{A}_k^T J_{u\lambda,k} \tag{27c}$$

$$J_{xx,k}^{*T} = J_{xx,k}^T + \boldsymbol{B}_k J_{uu,k}^T\boldsymbol{B}_k + \boldsymbol{B}_k^T J_{ux,k} + J_{ux,k}\boldsymbol{B}_k \tag{27d}$$

$$J_{\lambda\lambda,k}^{*T} = J_{\lambda\lambda,k}^T + \boldsymbol{D}_k J_{uu,k}^T\boldsymbol{D}_k + \boldsymbol{D}_k^T J_{u\lambda,k} + J_{u\lambda,k}\boldsymbol{D}_k \tag{27e}$$

$$J_{x\lambda,k}^{*T} = J_{x\lambda,k}^T + \boldsymbol{B}_k J_{uu,k}^T\boldsymbol{D}_k + \boldsymbol{B}_k^T J_{u\lambda,k} + J_{ux,k}\boldsymbol{D}_k \tag{27f}$$

Equations 27b - 27f give the optimal cost-to-go derivatives at the current stage, k, which now can be used for the next stage, $k-1$. With Eq. 27a, the expected reduction of the cost at stage k, ER_k, is calculated. The latter is a measure to define the success of the DDP iteration and is not used in the calculation of the optimal control update in Eq. 21 (see Subsection III.G). At the final stage, the expected reduction is set to zero.

In summary, at the start of the backward sweep the derivatives of the optimal cost-to-go at the final stage, $k = N$, are

calculated. At the final stage, the optimal cost is equal to the nominal cost, as $J_N^* = J$. Note that the derivatives of the cost-to-go are simply the derivatives of the augmented Lagrangian here, which are found analytically[†]. Additionally, at the start of the backward sweep the STMs for the entire trajectory are calculated (in parallel). Subsequently, the current stage's cost-to-go derivatives are calculated (Eq. 22), the feedforward and feedback matrices are calculated (Eq. 21), and the optimal cost-to-go derivatives for the current stage are calculated (Eq. 27). When all preceding elements are computed for the current stage, the algorithm can move on to the next stage, $k-1$, until it arrives at the first stage, $k=0$.

E. Lagrange Multiplier Update

One term remains to be solved for in the control update in Eq. 21 - the Lagrange multiplier update, $\delta\boldsymbol{\lambda}$. The Lagrange multipliers are initialized at zero at the beginning of the optimization and only updated once per iteration, at the end of the backward sweep. The Lagrange multiplier update can be found by using the optimal cost-to-go derivatives from Eqs. 27c and 27e at the first stage, $k=0$:

$$\delta\boldsymbol{\lambda} = -J_{\lambda\lambda,0}^{*-1} J_{\lambda,0}^* \tag{28}$$

Additionally, the expected reduction in cost, at stage $k=0$, has to be updated to account for the Lagrange multiplier update:

$$ER_0 = ER_0 + J_\lambda^T \delta\boldsymbol{\lambda} + \frac{1}{2}\delta\boldsymbol{\lambda}^T J_{\lambda\lambda}\delta\boldsymbol{\lambda} \tag{29}$$

F. Trust Region Quadratic Subproblem

As shortly touched upon in Subsection III.D, it is necessary to limit the variations in state, $\delta\boldsymbol{x}_k$, and control, $\delta\boldsymbol{u}_k$, such that the second-order Taylor-series expansions still remain reliable. Another issue is that convergence of the algorithm is only guaranteed if $J_{uu,k}$ is positive definite and $J_{\lambda\lambda}$ is negative definite. Reference [24] overcomes these challenges by introducing a trust region method that solves a trust-region quadratic subproblem (TRQP) at each stage. The TRQP is mathematically written as

$$\min_{\delta u_k} \left[J_{u,k}\delta\boldsymbol{u}_k + \frac{1}{2}\delta\boldsymbol{u}_k^T J_{uu,k}\delta\boldsymbol{u}_k \right] \text{ such that } \|\boldsymbol{G}\delta\boldsymbol{u}_k\| \leq \Delta \tag{30}$$

where $\boldsymbol{G}$ is some positive-definite scaling matrix and Δ is the trust-region radius, in which, for instance, $\delta\boldsymbol{u}_k$ is required to lie. The TRQP for $J_{\lambda\lambda}$ is defined by replacing $J_{uu,k}$ and $J_{u,k}$ in Eq. 30 by $-J_{\lambda\lambda}$ and $-J_\lambda$, respectively. The two TRQP's are then posed as TRQP (J_u, J_{uu}, Δ) and TRQP $(-J_\lambda, -J_{\lambda\lambda}, \Delta)$. On the whole, the scaling matrix $\boldsymbol{G}$ becomes significant when small variations in a few parameters affect the variations in the objective function more than others

[†]Analytically found using Maple worksheets: `https://github.com/roosmartens/thesis_derivatives`.

[24]. In this work, $\boldsymbol{G}$ is set to the identity matrix. Reference [54] illustrates many different methods to solve the TRQP, where, in particular, one of these methods (the "classical" TRQP) has been used to solve the TRQP in DDP in previous research successfully [19, 24]. Reference [24] introduced a simpler trust region method using arbitrary Hessian shifting: changing the diagonal values of the Hessians $J_{uu,k}$ and $J_{\lambda\lambda}$ at each stage to ensure that the control update lies inside the trust region. Reference [35] compared the "classical" TRQP with the "simple" TRQP and found that the "simple" TRQP produced more optimal solutions and made the algorithm converge faster. Therefore, in this work this trust region method is used as well.

G. End Of Iteration

At the end of the backward sweep, a reduction ratio is calculated as

$$\rho = \frac{J_{new} - \overline{J}}{ER_0} \tag{31}$$

Here, $\overline{J}$ is the previous nominal augmented Lagrangian and J_{new} is the new augmented Lagrangian at the end of the iteration. The reduction ratio, ρ, defines the success of the current iteration together with a user-defined reduction-ratio tolerance, ϵ_1. If $\rho \approx 1$, where ϵ_1 defines the deviation from one, the cost reduction of the algorithm is as expected and the quadratic model is satisfactory. Consequently, the iteration is successful. When ρ is further from one, the quadratic model is poor and the iteration is rejected. Furthermore, if the iteration is rejected, the trust-region radius is reduced to decrease the region in which the quadratic model is reliable. Poor quadratic approximations could result in convergence to a local optimum. In the case of rejection, the STMs are not calculated again and the algorithm proceeds from the backward sweep once again. On the other hand, if the iteration is accepted, the trust-region radius is enlarged. The trust region is part of the trust-region quadratic subproblem (TRQP), as was discussed in Subsection III.F.

Using the reduction ratio calculated with Eq. 31, the trust-region radius for the next iteration is given as

$$\Delta_{\text{next}} = \begin{cases} \min\left((1+\kappa)\Delta_{\text{previous}}, \Delta_{\text{max}}\right), & \text{if } \rho \in (1-\epsilon_1, 1+\epsilon_1) \\ \max\left((1-\kappa)\Delta_{\text{previous}}, \Delta_{\text{min}}\right), & \text{otherwise} \end{cases} \tag{32}$$

where κ is the reduction or increase of the trust region at each iteration ($0 < \kappa < 1$), and Δ_{max} and Δ_{min} are the maximum and minimum trust region radii, respectively. In addition, an optimality tolerance, ϵ_2, is defined. Finally, when $ER_0 < \epsilon_2$ and when all the Hessians $J_{uu,k}$ are positive definite and $J_{\lambda\lambda}$ is negative definite, the algorithm is said to be converged.

H. Control bounds

Control parameters can be constrained by defining control bounds, $u^L \leq u_k \leq u^U$, where u^L is the lower and u^U the upper control bound. As stated in Subsection II.C, bounds are defined for the control angles, $\alpha \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ and $\delta \in [0, 2\pi]$. The DDP algorithm in Reference [31] used a null-space method to account for the control bounds at the end of an iteration. In this method, the trust region is updated before accounting for the control bounds, which could result in underestimation of the control update. To avoid this shortcoming, Reference [35] presented a Heaviside-based approach, where the control bounds are acknowledged throughout the entire iteration by intertwining them with the dynamics, e.g., the STMs. The Heaviside function is defined as

$$H\{j\} = \begin{cases} 1, & j > 0 \\ 0, & j < 0 \end{cases} \tag{33}$$

In order to account for the DDP requirement that the dynamics be twice differentiable, the Heaviside function in Reference [35] is approximated by a smooth logistic function, as

$$H\{j\} \approx \frac{1}{2} + \frac{1}{2}\tanh(\tilde{h}j) \tag{34}$$

where $\tilde{h}$ defines the steepness of the curve. If $\tilde{h}$ is set too large, the approximated function will approach the "normal" Heaviside function, derivatives will exceedingly increase at the bounds, and the algorithm will become unstable. If an update to a control parameter, u_k, causes it to violate the control bounds, it will be corrected as follows:

$$u_{k,new} = u_k + (u^L - u_k) \times H\{-\tilde{h}(u_k - u^L)\} + (u^U - u_k) \times H\{\tilde{h}(u_k - u^U)\} \tag{35}$$

Note that when a control parameter is within the control bounds, Eq. 35 reduces to $u_{k,new} = u_k$; when the control parameter is outside the specified control bounds, it will be corrected to the value of its closest bound.

IV. Validation

This section focuses on the validation of the implemented DDP algorithm. Firstly, in Subsection IV.A the results of the optimization are compared to known analytical solutions from locally optimal steering laws in Reference [1]. Furthermore, a sensitivity analysis is performed in Subsection IV.B to study the effect of the complexity of the dynamics, the algorithmic settings, and the robustness of DDP to different initial guesses. The DDP implementation was programmed in C++ and compiled on a Quad-Core Intel Core i5 @1.4GHz processor. STM computations were distributed in parallel along three cores with OpenMP [55].

A. Comparison to Locally Optimal Steering Laws

In order to validate the implemented algorithm, the optimization results of the DDP algorithm are compared against locally optimal steering laws (explained in detail in Reference [1]). These control laws maximize the instantaneous rate of change of a specific orbital element and provide the optimal sail attitude as an analytical solution. In particular, the projection of the solar-radiation pressure force along a vector of functions of the solar-sail orbital elements is maximized. This ensures that the rate of change of the chosen orbital element is maximized at all stages of the trajectory. As of now, the DDP implementation only supports the maximization of the magnitude of the orbital element. These differences in objective must be kept in mind for the upcoming validation. Subsection IV.A.1 displays the performance of DDP for the maximization of the orbital radius starting from a heliocentric orbit. There are two different manners to define orbit raising, namely the maximization of the final semi-major axis or the maximization of the final specific orbital energy. The difference in performance for these two different definitions of orbit raising is compared. Secondly, Subsection IV.A.2 illustrates the performance of DDP for the maximization of another orbital element, namely the eccentricity. All in all, the versatile nature of DDP in the optimization of different objectives in the interplanetary flight regime is showcased.

1. Orbit raising

The first objective investigated is orbit raising, which is expressed through either maximizing the semi-major axis or the specific orbital energy [1]. In this case, no constraints are included. The objective function ϕ_E corresponds to the maximization of the final specific orbital energy, at stage N. The total Lagrangian cost function, J_E, is then written as

$$J_E = \phi_E = -\left(\frac{||\dot{\boldsymbol{r}}_{\boldsymbol{N}}||^2}{2} - \frac{\mu_\odot}{||\boldsymbol{r}_N||}\right) \tag{36}$$

where $\boldsymbol{r}$ and $\dot{\boldsymbol{r}}$ are obtained in MEE from Eq. 15 and Eq. 16, respectively. The objective function ϕ_a corresponds to the maximization of the final semi-major axis, such that the Lagrangian cost function, J_a, is given as

$$J_a = \phi_a = -\frac{p_N}{1 - f_N^2 - g_N^2} \tag{37}$$

Note that Eqs. 36 and 37 indicate the objective values at the final stage of the trajectory. A minus sign is included in the cost functions to account for the minimization by DDP. The algorithm settings used for the optimization are given in Table 4.

Table 4 Algorithm settings used for optimization.

Algorithm setting	ϵ_1	ϵ_2	κ	Δ_0 [rad]	Δ_{min} [rad]	Δ_{max} [rad]
Value	0.1	10^{-12}	0.05	π	0	π

The initial guess for the cone angle was set to a constant cone angle of $\alpha = 0^\circ$ and for the a constant clock angle of $\delta = 0^\circ$. The problem is modeled using a realistic lightness number of $\beta = 0.01$ and an initial orbit as given in Table 5.

Table 5 Initial state in Keplerian elements

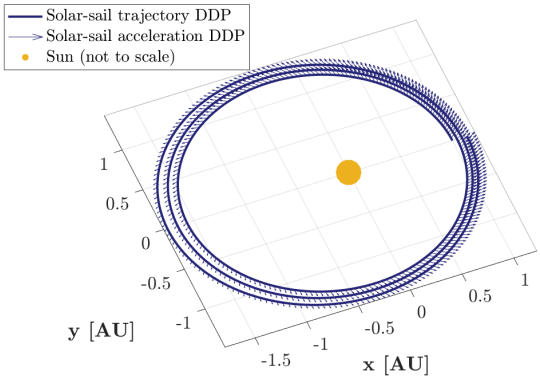
Kepler element	a [AU]	e [-]	i [rad]	ω [rad]	Ω [rad]	ν [rad]
Value	1.25	0.2	0	0	0	0

It is well known that the efficiency of an optimal control algorithm can be increased by scaling the problem [56]. Thus, in the implementation distances are scaled by an astronomical unit (AU) and the time, which is expressed in seconds, is scaled by the number of seconds in a year (see Table 1). This proved advantageous for the optimization. The results for both optimizations after three orbital revolutions are given in Table 6. Here, the final semi-major axis and final specific orbital energy are displayed for the locally optimal steering law, as well as for both the optimizations. In addition, the relative difference of the DDP optimization with the locally optimal steering law, ε, is given for both parameters.

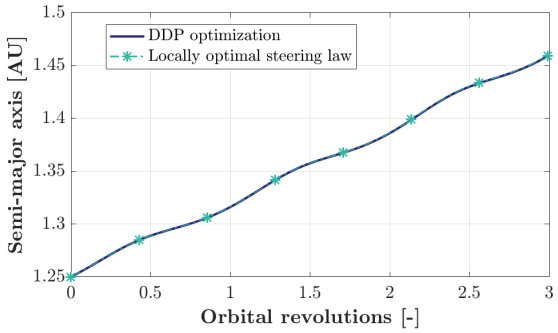
Table 6 Comparison of the final values for the semi-major axis and specific orbital energy found from DDP optimization of two Lagrangian cost functions, J_a and J_E, with the results from the locally optimal steering law.

	Final semi-major axis [km]	Final specific orbital energy [J]
Locally optimal steering law	2.184410×10^8	-3.037718×10^8
J_E optimization	2.183946×10^8	-3.038364×10^8
Relative difference (ε)	-0.0213%	0.0213%
J_a optimization	2.184421×10^8	-3.037703×10^8
Relative difference (ε)	0.0005%	-0.0005%

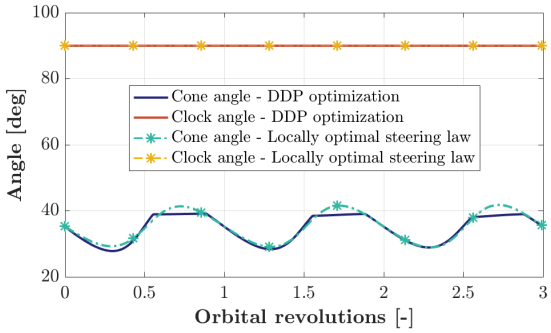
The results in Table 6 show that the optimization of the semi-major axis finds a larger final semi-major axis than the locally optimal steering law, namely a relative difference of $\varepsilon = 0.0005\%$, or approximately 1017 km. The final corresponding specific orbital energy is less negative, with a relative difference of $\varepsilon = -0.0005\%$ with the locally optimal steering law. For the optimization of the final specific orbital energy, J_E, a smaller final semi-major axis is found in comparison to the locally optimal steering law. This is a relative difference of $\varepsilon = -0.0213\%$, or approximately -64661 km. In addition, for the optimization of J_E, a larger final specific orbital energy is found compared to the locally optimal steering law. Nonetheless, the percentage differences are small and could be the result of small numerical optimization errors. Note that DDP finds even more optimal solutions when comparing the results for a specific optimized parameter with the results of the same parameter from the locally optimal steering law. For example, the optimization of J_a results in a more optimal solution for the semi-major axis when compared to the locally optimal steering law than the optimization of J_E. On the other hand, the latter finds a more optimal solution for the specific orbital energy. Even though the semi-major axis and specific orbital energy go hand in hand [46], small differences



(a) The optimal solar-sail trajectory.



(b) The semi-major axis.



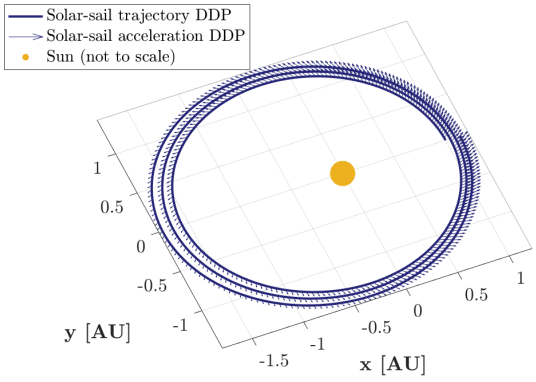
(c) The optimal control law.

Fig. 3 Results of the optimization of the specific orbital energy (solid lines) for three orbital revolutions, in comparison to a locally optimal steering law (dashed lines) [1].

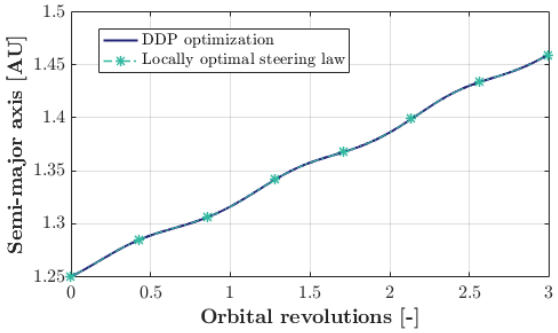
occur when one or the other objective is maximized. Consequently, these results imply that the problem definition used for the optimization with DDP influences the outcome slightly.

Furthermore, in Fig. 3 the final control law found by DDP for J_E is displayed with the resulting trajectory and semi-major axis. Figure 4 shows similar plots for the optimization of J_a. The figures confirm that the semi-major axis found by optimization of both cost functions is comparable to the semi-major axis found by the locally optimal steering law. Furthermore, Figs. 3c and 4c show that the resulting control law is more similar to the locally optimal steering law's control law for optimization of J_a, than J_E. Since DDP finds similar solutions to the locally optimal steering law for the optimization of the cost functions, J_E and J_a, it can be concluded that the solutions are valid.

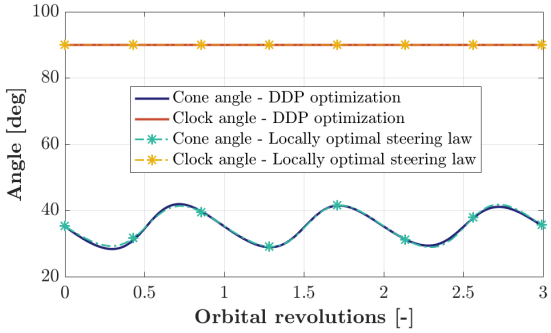
Figure 5 displays the development of the optimized control law for the cone angle over time, or the amount of iterations needed until convergence, for the two Lagrangian cost functions. Figure 5a displays larger "steps" and 22/296 iterations were successful, whereas Fig. 5b displays a "smoother flow" across the iterations, with less difference in the control law between iterations, and 69/291 iterations were successful. In both optimizations, the last few iterations do not show significant changes in the control law, so that an earlier truncation of the algorithm (i.e., after fewer iterations) may have also resulted in a satisfactory solution: for the optimization of J_E (Fig. 5a), approximately 120/296 iterations



(a) The optimal solar-sail trajectory.

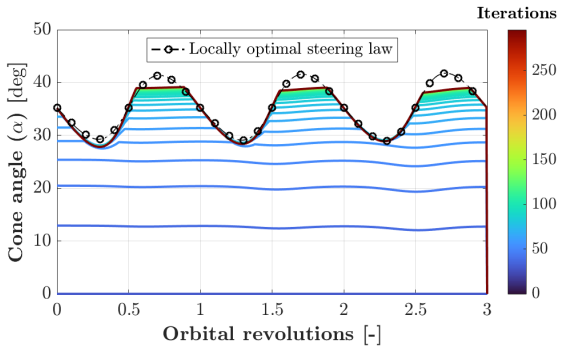


(b) The semi-major axis.

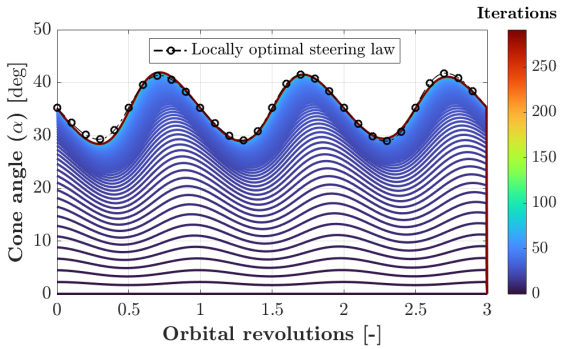


(c) The optimal control law.

Fig. 4 Results of the optimization of the semi-major axis (solid lines) for three orbital revolutions, in comparison to a locally optimal steering law (dashed lines) [1].



(a) Maximization of the specific orbital energy, J_E.



(b) Maximization of the semi-major axis, J_a.

Fig. 5 The development of the control law for the cone angle (solid colored lines) across the iterations and the locally optimal steering law (dashed black line).

are sufficient, whereas for the optimization of J_a (Fig. 5b), approximately 70/291 iterations are sufficient. This indicates that the strict optimality tolerance of $\epsilon_2 = 10^{-12}$ can be loosened, which is in favor of the run time. A sensitivity analysis on this parameter and other algorithmic settings is treated in Subsection IV.B.2.

Additionally, the optimization of J_a converges quicker to an optimal solution (approximately 70 iterations required) than the optimization of J_E (approximately 120 iterations). Since the search for the optimal control law behaves differently over time, it can be concluded that DDP is indeed considerably sensitive to the problem definition. Note that J_a is a simple equation, involving only three MEE, whereas J_E is a more complex, lengthy equation involving all six MEE. Furthermore, after scaling, the Lagrangian cost function J_E is still more than ten times as large as J_a after scaling, which could influence the optimization. In solving an optimal control problem, it is desirable to have all variables of a similar order of magnitude [56], thus it is likely that using a different scaling method could improve the results.

In summary, it can be concluded that DDP is slightly sensitive to different formulations of the optimization problem. For the maximization of the semi-major axis, the optimization of J_a outperforms the optimization of J_E. In addition, the difference in the formulation of the objective for the locally optimal steering law and the optimization, as well as small numerical errors in the optimization can also explain the differences between the results of the locally optimal steering laws and the optimization. Nonetheless, the results in this section conclude the validity of DDP and are a good start for illustrating the robust performance of DDP.

2. Eccentricity raising

The implementation of DDP is further validated by considering a different objective function; the maximization of the eccentricity. The Lagrangian cost function, without constraints, equals the final eccentricity:

$$J_e = \phi_e = -\sqrt{f_N^2 + g_N^2} \tag{38}$$

Similar algorithmic settings (see Table 4) and dynamical models are used as in Subsection IV.A.1, as well as a similar initial orbit (see Table 5). In addition, the same scaling method is used as in Subsection IV.A.1. The results for the optimization after three orbital revolutions are given in Table 7. Again, in Table 7 the final eccentricity found by DDP is compared to a locally optimal steering law from Reference [1] that maximizes the instantaneous rate of change of the eccentricity.

Table 7 Comparison of the final value for the eccentricity found from DDP optimization of the Lagrangian cost function J_e, with the result from the locally optimal steering law.

	Final eccentricity
Locally optimal steering law	0.319188
J_e optimization	0.319569
Relative difference (ε)	0.1195%

The results show that the final eccentricity found by DDP is larger, with a relative difference of $\varepsilon = 0.1195\%$, than the eccentricity found by the locally optimal steering law. Moreover, Fig. 6 displays the final optimal trajectory, together with the eccentricity and optimal control law over three orbital revolutions. Note that the control laws in Fig. 6c appear different but are actually the same when considering the ambiguity in defining the cone and clock angles, i.e., a cone angle of $\alpha = -90^\circ$ and a clock angle of $\delta = 90^\circ$ result in the same orientation of the sail normal vector as a cone angle of $\alpha = 90^\circ$ and a clock angle of $\delta = 270^\circ$. Figure 7 shows the development of the control law across the iterations. The figure shows a very smooth progression of the control law, where 3234 iterations (of which 2002 successful) are required to reach the optimality tolerance of $\epsilon_2 = 10^{-12}$. The many successful iterations indicate a reliable quadratic model approximated in the optimization. This rhymes with the fact that J_e has an even simpler formulation than the more complex formulation of the objectives from Subsection IV.A.1, which displayed less successful iterations and therefore less reliable quadratic approximations. In contrast to the results in Subsection IV.A.1, the eccentricity optimization requires almost all iterations to reach an equally optimal solution as the locally optimal steering law, which results in a long run time. This can be explained when inspecting Fig. 7: as more successful iterations occur, the trust region shrinks more and more, which limits the allowed change in control law. However, to construct the final sharp peaks in the control law a large change in control is required: consequently, more iterations are needed.

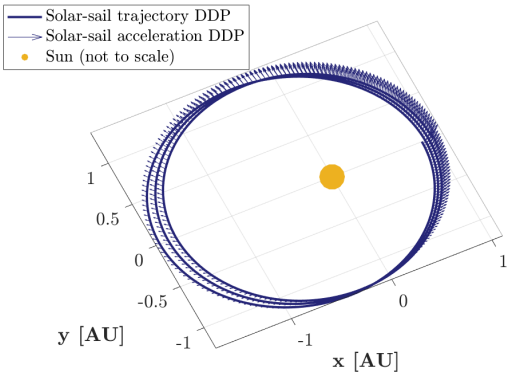
In general, from these results and the results in Subsection IV.A.1, the performance of DDP for optimizing different objectives is validated. Additionally, DDP finds similar or more optimal solutions than the locally optimal steering laws from Reference [1]. The more optimal solution goes hand in hand with a larger computational effort. However, DDP is expected to thrive when solving more complex problems with more complex objective functions and multiple constraints, which the locally optimal steering law cannot solve.

B. Sensitivity Analysis

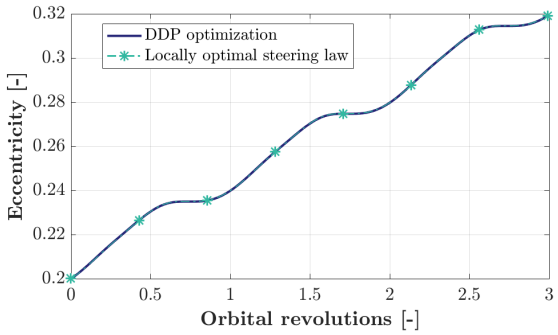
For further validation purposes of the DDP algorithm itself and the combination of DDP with the dynamical framework, a sensitivity analysis is performed. Firstly, Subsection IV.B.1 validates the robustness of DDP to initial guesses. Furthermore, in Subsection IV.B.2 specific algorithmic parameters are varied, such as the reduction-ratio tolerance, the optimality tolerance, and the trust region parameters. Subsections IV.B.1 and IV.B.2 combined give insight in the robustness of the algorithm and in the most optimal settings for interplanetary solar-sail trajectory optimization. For the sensitivity analysis in this section the optimization of J_a is considered, since Section IV showed a slightly better performance for the use of J_a compared to J_E.

1. Initial guess

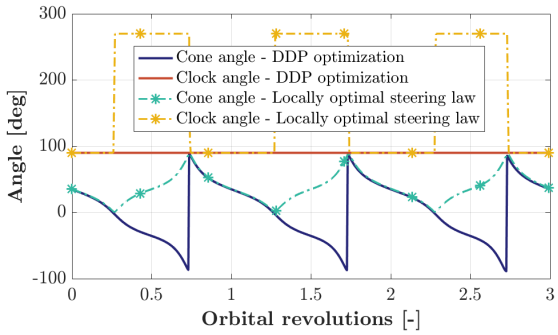
In order to illustrate its robustness, DDP is initialized with different initial guesses for the optimization of the semi-major axis, J_a, for one orbital revolution. Again, the optimization is performed with a lightness number of $\beta = 0.01$



(a) The optimal solar-sail trajectory.



(b) The eccentricity.



(c) The optimal control law.

Fig. 6 Results of the optimization of the eccentricity for three orbital revolutions, in comparison to a locally optimal steering law [1].

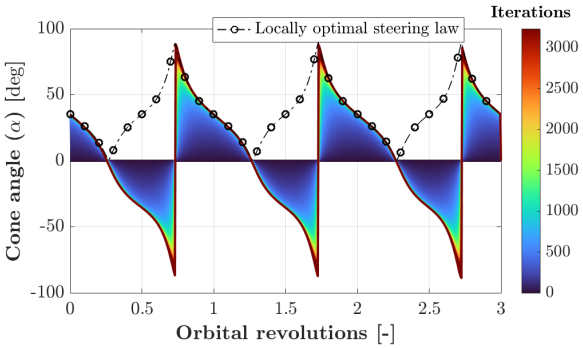


Fig. 7 The development of the control law for the cone angle (solid colored lines) across the iterations and the locally optimal steering law (dashed black line).

and similar algorithmic settings (see Table 4) and scaling methods as in Subsection IV.A.1 and IV.A.2. All initial guesses assume a constant cone and clock angle throughout the trajectory, where the constant value changes across the different initial guesses. Along with feeding constant initial guesses for the controls into the algorithm, the locally optimal steering law itself was also used as an initial guess. The results generated by DDP for the cost function J_a, for the different initial guesses, can be found in Table 8. The final value of the semi-major axis is displayed after one orbital revolution for each initial guess and the relative difference with respect to the locally optimal steering law in Reference [1], ε, is given. In addition, the total number of iterations and successful iterations for the algorithm to converge is given, as well as the run time. As stated in Subsection III.G, a larger share of successful iterations indicates that the quadratic truncations of the changes in the cost are reliable, although sacrificing on run time, since a successful iteration ensures recalculation of the most time-consuming part of the algorithm; the STM calculation. Section IV already displayed that the optimization had a more reliable quadratic model for a simple objective function, however, in this section, the effect of the initial guess on this reliability, as well as the run time of the algorithm, is shown. To explain the results, a "good" initial guess is defined as a constant cone and clock angle close to the optimal solution (e.g., the best initial guesses describe in-plane motion), whereas the poorer an initial guess is, the further it is from the optimum. For instance, Fig. 4c displays the optimal control law, such that a good initial guess would consist of a clock angle equal to $\delta = 90°$ and a cone angle value in the range $\alpha = 30° - 40°$. The further the values are from this optimum, the poorer the initial guess is.

Observing Table 8, it becomes clear that good initial guesses result in the shortest run time but also the smallest semi-major axis values. On the other hand, initial guesses further away from the optimal angles lead to longer run times but larger semi-major axis values. The clock angle plays a larger role in the run time or iterations needed for convergence, as seen from an increase in run time from bottom to top in Table 8, which is ordered by clock angle value. The cone angle also impacts the run time and iterations needed for convergence, with closer-to-optimal cone angles resulting in less difficulty for the algorithm to find the solution. This difficulty is expressed in the total amount of iterations, and its share of successful iterations, required. A larger amount of iterations indicates more difficulty for the optimization, where the share of successful iterations indicate the reliability of the approximations of the gradient of the cost function. The latter contribute the most to the run time (see Subsection III.G). The run time can be reduced by approximately 99.4% with respect to the initial guess resulting in the longest run time (similar to the reduction when using the locally optimal steering law as an initial guess), by using a "good" initial guess such as $[\alpha, \delta] = [45°, 90°]$, and can be reduced by 53.6% with an "average" initial guess such as $[\alpha, \delta] = [45°, 45°]$.

All in all, the run time can be significantly reduced by choosing a suitable initial guess. However, the small amount of successful iterations corresponding to "good" initial guesses result in a (small) reduction in optimality of the solution. In this case the controls are already close to the optimal value, which implies that the quadratic approximations of the changes in cost are less reliable when DDP keeps searching for the optimum: this results in an unsuccessful iteration (see Subsection III.G). If the approximations are less reliable, the algorithm could faster converge to a local optimum. The

Table 8 Results of the optimization of the semi-major axis for one orbital revolution for different initial control law guesses.

Initial guess α [deg]	Initial guess δ [deg]	Semi-major axis [km]	Relative difference (ε)	$\frac{\#\text{Successful}}{\#\text{Total}}$ iterations	Run time [s]
90°	0°	1.969521×10^8	0.0005%	1213 / 1311	897
0°	0°	1.969521×10^8	0.0005%	1142 / 1142	661
45°	0°	1.969521×10^8	0.0005%	964 / 1036	464
90°	45°	1.969521×10^8	0.0005%	902 / 902	441
0°	45°	1.969521×10^8	0.0005%	881 / 881	423
45°	45°	1.969521×10^8	0.0005%	869 / 869	416
90°	90°	1.969523×10^8	0.0006%	179 / 399	96
0°	90°	1.969479×10^8	−0.0016%	62 / 278	34
45°	90°	1.968812×10^8	−0.0355%	6 / 238	5
Locally optimal steering law		1.969525×10^8	0.0007%	1 / 204	3

initial guess from Table 8 of $[\alpha, \delta] = [90°, 90°]$ provides for a smaller share of successful iterations than most, yet still enough for a sufficient model reliability, which results in the largest semi-major axis ($\varepsilon = 0.0006\%$) within a sufficient amount of time (96 seconds) compared to using the locally optimal steering law as an initial guess ($\varepsilon = 0.0007\%$). Using the locally optimal steering law as an initial guess requires three seconds to converge and provides for only one successful iteration. However, because the initial guess is equal to the optimal solution, DDP is still able to make the step to a slightly more optimal solution, instead of converging to a local optimum. Ultimately, a larger share of successful iterations ensures a trustworthy convergence of the algorithm, although at the cost of longer run times. Most initial guesses converge to a similar semi-major axis with a difference of $\varepsilon = 0.0005\%$ with respect to the locally optimal steering law. On the whole, it is concluded that the implemented DDP algorithm is highly robust to different initial guesses.

2. Algorithm settings

The influence of different algorithmic settings that were explained in Section III is investigated for a more extensive robustness analysis and to research the most optimal settings for the remaining Earth-Mars orbital rendezvouz. Firstly, the variation of the optimality tolerance, ϵ_2, is investigated, followed by the variation of the reduction-ratio tolerance, ϵ_1. Lastly, the influence of the trust region parameters, such as the trust region scaling factor, κ, the maximum trust-region radius, Δ_{max}, and the initial trust-region radius, Δ_0, is analyzed. Note that the minimum trust-region radius is held at a constant $\Delta_{min} = 0$ to allow the trust-region radius to shrink when required: similar to the findings of Reference [35], suboptimal trajectories are found for any larger value. Table 9 provides an overview of the different cases and settings analyzed in this section, where each case is given a case number for referral.

Table 9 Algorithm settings used for the optimization of different cases.

Case of variation	Initial guess $[\alpha, \delta]$	Algorithm settings during optimization					
		ϵ_1	ϵ_2	κ	Δ_0 [rad]	Δ_{min} [rad]	Δ_{max} [rad]
1. ϵ_2	$[45°, 45°]$	0.1	-	0.05	π	0	π
2. ϵ_1	$[45°, 45°]$	-	10^{-8}	0.05	π	0	π
3. Δ_{max} & Δ_0	$[90°, 90°]$	0.1	10^{-8}	0.05	-	0	-
4. κ	$[90°, 90°]$	0.1	10^{-8}	-	π	0	π

For the analysis of case 1, the variation in optimality tolerance, ϵ_2, and case 2, the variation in reduction-ratio tolerance, ϵ_1, an "average" initial guess with many successful iterations is used to give the best impression of the overall influence of these settings. "Average" denotes an initial guess from the analysis in Table 8 that was not the closest or furthest from the optimum and performed sufficiently. Hence, an initial guess of $[\alpha, \delta] = [45°, 45°]$ is used. Furthermore, to encapsulate the influence of these parameters in a broader sense, two Lagrangian cost functions, J_a and J_e (see Subsection IV.A), are both optimized using the settings in Table 9 for one orbital revolution. The effect of the optimality tolerance on the relative difference with the corresponding locally optimal steering law (for maximization of the semi-major axis or the eccentricity) and the run time of the algorithm is given in Table 10. Note that the smaller the optimality tolerance, the more optimal the solution for the semi-major axis or eccentricity is, but also the larger the run time is. For optimality tolerances larger than $\epsilon_2 = 10^{-8}$ the final maximized parameter does not increase significantly, yet the run time does. Thus, this optimality tolerance is used in the remaining analyses. Note that larger optimality tolerances can be used in preliminary design to quickly find a sufficient solution, whereafter the solution can be finetuned with a small optimality tolerance.

Table 10 The relative difference with the solution of a locally optimal steering law (ε) and the run time when varying the optimality tolerance for the optimization of the cost functions J_a and J_e.

		Optimality tolerance (ϵ_2)							
		10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}
J_a	Relative difference (ε)	-0.0283%	-0.0024%	0.0002%	0.0005%	0.0005%	0.0005%	0.0005%	0.0005%
	Run time [s]	86	131	186	246	288	343	422	497
J_e	Relative difference (ε)	-0.0993%	-0.0105%	0.0023%	0.0044%	0.0047%	0.0047%	0.0048%	0.0048%
	Run time [s]	110	142	146	216	322	494	755	1282

The reduction-ratio tolerance, ϵ_1, specifies a tolerance for which an iteration is accepted or not. If this tolerance is set to a large value, iterations are accepted too easily and the algorithm converges to a less-optimal solution. Thus, the reduction-ratio tolerance was varied to a maximum value of $\epsilon_1 = 0.15$. Table 11 displays the optimization results for a variation of reduction-ratio tolerances (case 2), again with the settings stated in Table 9. For maximizing the eccentricity, these results show an equal relative difference with the locally optimal steering law ($\varepsilon = 0.0044\%$) and a similar run time for each reduction-ratio tolerance. The optimization of the semi-major axis exhibits fast convergence to

a less optimal solution for reduction-ratio tolerances smaller than $\epsilon_1 = 0.05$. Since larger values for ϵ_1 do not affect the performance of the algorithm and to ensure convergence to an optimal solution, the reduction-ratio tolerance was set to $\epsilon_1 = 0.1$ for the upcoming test cases.

Table 11 The relative difference with the solution of a locally optimal steering law (ε) and the run time when varying the reduction-ratio tolerance for the optimization of the cost functions J_a and J_e.

		Reduction-ratio tolerance (ϵ_1)						
		0.01	0.025	0.05	0.075	0.1	0.125	0.15
J_a	Relative difference (ε)	-1.6895%	-1.6895%	0.0005%	0.0005%	0.0005%	0.0005%	0.0005%
	Run time [s]	2	2	254	244	254	259	247
J_e	Relative difference (ε)	0.0044%	0.0044%	0.0044%	0.0044%	0.0044%	0.0044%	0.0044%
	Run time [s]	218	218	218	219	219	223	222

Finally, the trust region parameters are investigated in case 3 and 4. The trust region only shrinks, therefore limiting the size of the control update, if iterations are unsuccessful, yet enlarges if iterations are successful (see Eq. 32). Hence, to best test the influence of the trust region parameters, a test case with an initial guess is chosen that displays multiple unsuccessful iterations, but still arrives at an optimal solution; the optimization of J_a with an initial guess of $[\alpha, \delta] = [90°, 90°]$, with the settings stated in Table 9. The difference in results of the optimization by varying the maximum trust-region radius, Δ_{max}, and the initial trust-region radius, Δ_0, appeared to be minimal (case 3). All parameter combinations resulted in the same relative difference of the semi-major axis with the locally optimal steering law and similar run times. Hence, these two parameters are held at an average $\Delta_0 = \pi$ and $\Delta_{max} = \pi$ for the upcoming test case. The results of case 4, varying the trust region scaling factor, is displayed in Table 12.

Table 12 The effect of varying the trust region scaling factor, κ, on the relative difference in the final semi-major axis with the locally optimal steering law, ε, and the run time.

κ	0.01	0.1	0.2	0.3	0.4
Relative difference (ε)	0.00062313%	0.00062313%	0.00062305%	0.00062298%	0.00062290%
Run time [s]	96	94	96	110	94

From Table 12 can be seen that the optimality of the solution reduces faintly from a trust region scaling factor of $\kappa = 0.2$ and on. Because the variations result in minimal change of the final semi-major axis or run time, a trust region scaling factor of $\kappa = 0.1$ is selected for the upcoming test cases.

In summary, the optimality tolerance and the reduction-ratio tolerance have the largest effect on the optimization results. The trust region parameters have minimal impact. However, note that for every different problem definition, different algorithmic settings can have slightly different effects. For a more extensive analysis, this sensitivity analysis should be extended to more problem definitions. All in all, the settings selected from the findings of this section are stated in Table 13. These parameters are used for the upcoming test cases.

Table 13 Selected algorithm settings.

Algorithm setting	ϵ_1	ϵ_2	κ	Δ_0 [rad]	Δ_{min} [rad]	Δ_{max} [rad]
Value	0.1	10^{-8}	0.1	π	0	π

V. Earth-Mars transfer

The significance of the Earth-Mars transfer originates from the search for life on Mars, understanding of the origin and transformation of the planet and the solar system, and preparation for future space exploration and immigration [36–39]. Secondly, solar-sail propelled spacecrafts become close competitors with conventionally-propelled spacecrafts in an Earth-Mars transfer if the time of flight is minimal [57]. Therefore, to demonstrate the performance of the DDP algorithm implemented in this work in an interplanetary test case, the trajectory optimization of a solar sail from an Earth orbit to the orbit of Mars is simulated. The problem definition and implementation are discussed in Subsections V.A and V.B, respectively, whereafter the results are given in Subsection V.C.

A. Problem definition

Firstly, the conditions defined for the optimization problem at hand are discussed. The minimum transfer time is sought as

$$\phi = t_N - t_0 \tag{39}$$

where t_N denotes the time at the end and t_0 the time at the beginning of the propagation, in this case set to $t_0 = 0$. In this test case, two circular, co-planar orbits are assumed for the orbits of Mars and Earth. Mars is assumed to have a semi-major axis of $a_{\mathrm{♂}} = 1.524$ AU and Earth of $a_{\mathrm{♁}} = 1$ AU [42]. The terminal constraints of the problem are therefore formulated by constraining the final eccentricity of the orbit to zero and the final semi-major axis of the orbit to Mars' semi-major axis. The differences between the final values of the two orbital parameters and those of Mars need to be minimized, thus the terminal constraints are formulated as

$$\boldsymbol{\psi} = \begin{bmatrix} \frac{p_N}{-f_N^2 - g_N^2 + 1} - a_{\mathrm{♂}} \\ f_N^2 + g_N^2 \end{bmatrix} \tag{40}$$

This test case displays the ability of DDP to minimize multiple constraints simultaneously. The total Lagrangian cost function to-be-optimized by DDP can then be written as

$$J = t_N + \boldsymbol{\lambda}^T \begin{bmatrix} \frac{p_N}{-f_N^2 - g_N^2 + 1} - a_{\mathrm{♂}} \\ f_N^2 + g_N^2 \end{bmatrix} + \begin{bmatrix} \left(\frac{p_N}{-f_N^2 - g_N^2 + 1} - a_{\mathrm{♂}}\right)^2 & \left(f_N^2 + g_N^2\right)^2 \end{bmatrix} \boldsymbol{\Sigma} \tag{41}$$

The total Lagrangian cost function is updated during the optimization until the objective function and constraints violation is minimal. The matrix $\mathbf{\Sigma}$ contains weights that can be assigned to each constraint separately. If large weights are used for both constraints, DDP will seek a solution that has an eccentricity closer to zero and a semi-major axis closer to Mars semi-major axis, but requires a longer transfer time to obtain this. Alternatively, different weights can be used for each constraint. If a larger weight is used for the eccentricity constraint, DDP will seek a final eccentricity closer to zero, but the final semi-major axis will be further from Mars semi-major axis. As also used before, distances are scaled by an AU and the time is scaled by a Julian year for optimal algorithmic performance (see Table 1).

B. Problem implementation

The current DDP implementation does not support a dynamic number of stages, which requires the definition of a fixed number of orbital revolutions for the transfer at the start of the optimization. On top of the terminal constraints defined in Eq. 40, it could be favorable to track the precise position, or true longitude, of Mars, in order to arrive at the exact position of Mars instead of an arbitrary position in the orbit of Mars. Because of the fixed number of orbital revolutions, yet varying time of flight each iteration, the target true longitude of Mars will vary for each iteration. Furthermore, when multiple orbital revolutions are required, angle wrapping occurs, where the true longitude of Mars and the solar sail jump to zero after completing one revolution. These topics make it numerically difficult for DDP to track the precise true longitude of Mars: the handling of this terminal constraint will be highly non-linear. Consequently, in this work the constraint on the final true longitude of Mars is not treated. To still be able to model a realistic test transfer, Subsection V.C will contain a sample departure and arrival date calculated from the first opposition between Earth and Mars.

In addition, by fixing the number of orbital revolutions, DDP has limited flexibility to minimize the time of flight. This fact ensures that DDP does not have much freedom in searching a minimum-time solution. Thus, an iterative process is used to find the minimum amount of orbital revolutions, and therefore the minimum time, required to reach the target orbit. The fixed number of orbital revolutions is decreased for multiple runs until no more sufficient solutions are found by DDP. One of the runs is selected as the optimal solution by a trade-off between the shortest time of flight as well as sufficient compliance with the terminal constraints. The results of this process are displayed in Subsection V.C, whereafter the optimization results are presented using the found optimal control law.

C. Results

The problem specified in Subsections V.A and V.B was implemented and tested. Firstly, a lightness number of $\beta = 0.01$ is used for consistency in this paper. Subsequently, the analysis is performed for a futuristic lightness number of $\beta = 0.1$, which is closer to the sail technology used in similar optimization cases from literature [57, 58]. The algorithmic settings used are stated in Table 13 and an initial guess is used of $[\alpha, \delta] = [45°, 90°]$. The weights of the

constraints are initially set to $\mathbf{\Sigma} = [100, 100]$ and will be varied for more optimal results. As explained in Subsection V.B, an iterative process is used to find the fixed amount of orbital revolutions resulting in the most optimal trajectory. The number of orbital revolutions is decreased from 11 to 7.1 by 0.05 from one iteration to the next. Fig. 8 displays the relation between the fixed number of orbital revolutions and time of flight: note that this is not linear. The results of the iterative process for a lightness number of $\beta = 0.01$ can be seen in Fig. 9, which shows how the two constraints in Eq. 40 are met as a function of the number of orbital revolutions. Note that the relative difference of the final eccentricity is the difference with the eccentricity of Mars, which is assumed zero ($e_{\male} = 0$), therefore, this relative difference is alternatively calculated as $\varepsilon = 100\frac{|e_{\male} - e|}{1+e_{\male}}$, throughout this paper.

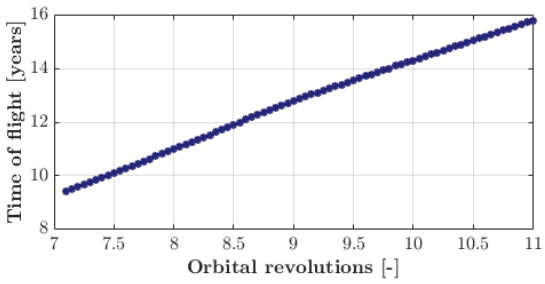
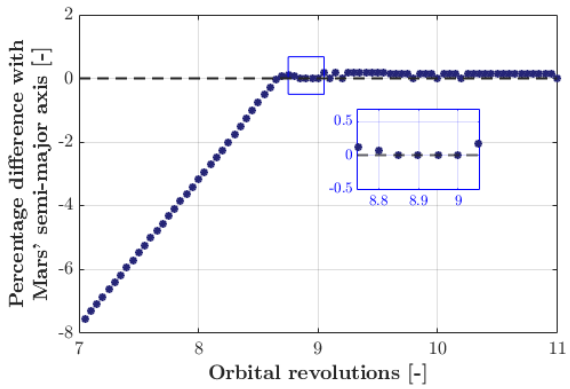
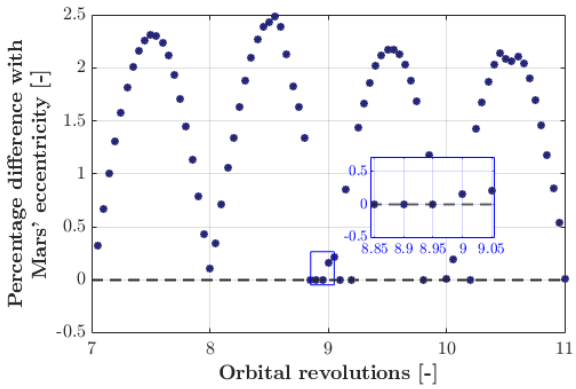


Fig. 8 The relation between the fixed amount of orbital revolutions and the time of flight.



(a) Relative difference of the final semi-major axis with Mars' semi-major axis.



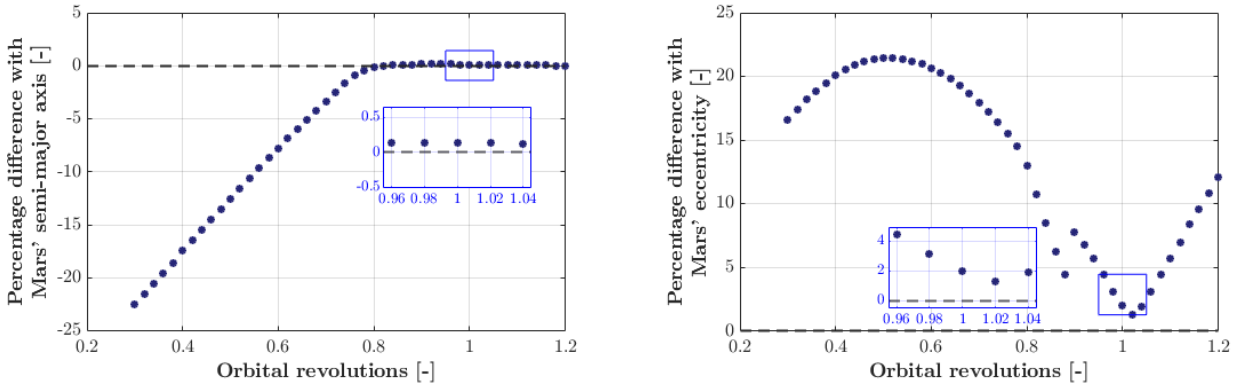
(b) Relative difference of the final eccentricity with Mars' eccentricity.

Fig. 9 The relative differences of the semi-major axis and the eccentricity between Mars' orbit and the final state resulting from the optimization of multiples of orbital revolutions for a lightness number of $\beta = 0.01$

Logically, the trajectory requires approximately more than 8.5 orbital revolutions to raise its semi-major axis to that of Mars (see Fig. 9a). The constraining of the eccentricity is basically periodic (see Fig. 9b). The optimal interplay of these constraints has to be found such that both constraints are met while the transfer time is as small as possible. For the specified settings, an optimal amount of orbital revolutions of 8.85 was found. This resulted in a final semi-major axis of 1.524000 AU (relative difference with Mars' semi-major axis of 0.000%), final eccentricity of 0.000016 (relative

difference with Mars' eccentricity of 0.002%), and transfer time of 4484.982 days or approximately 12.28 years. The final optimal trajectory and control law can be seen in Fig. 11.

To improve on the solar-sail transfer time between Earth and Mars, a similar procedure is followed for the futuristic lightness number of $\beta = 0.1$. The number of orbital revolutions is decreased from 1.2 to 0.3 by 0.02 from one run to the next. The results can be found in Fig. 10. From these figures an optimal amount of 1.02 orbital revolutions is selected, such that the optimal transfer time is 651.804 days. The optimization resulted in a final semi-major axis of 1.525923 AU (0.126% difference with Mars' semi-major axis) and a final eccentricity of 0.012971 (1.297% difference with Mars' eccentricity).



(a) Relative difference of the final semi-major axis with Mars' semi-major axis.

(b) Relative difference of the final eccentricity with Mars' eccentricity.

Fig. 10 The relative differences of the semi-major axis and the eccentricity between Mars' orbit and the final state resulting from the optimization of multiples of orbital revolutions for a lightness number of $\beta = 0.1$.

When compared to the transfer time found in the literature, e.g., 505.056 days for a lightness number of $\beta = 0.1$ [57] , the transfer time obtained with DDP is still far from optimal. However, transfer times for both lightness numbers ($\beta = 0.01$ and $\beta = 0.1$) can be improved by finetuning the weights on the constraints, $\mathbf{\Sigma}$. The same iterative procedure as illustrated in Figs. 9 and 10 is applied to the use of different sets of weights. This time, the number of orbital revolutions is decreased by 0.01 from one run to the next. Table 14 displays the results for different weights combinations. For brevity, the weight matrix $\mathbf{\Sigma} = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix}$ is from now on written as $\mathbf{\Sigma} = [\Sigma_1, \Sigma_2]$. Here, Σ_1 denotes the weight on the semi-major axis constraint and Σ_2 the weight on the eccentricity constraint. For each combination of weights, the corresponding optimal trajectory after the iterative searching process is displayed: the final semi-major axis and eccentricity values, the final percentage differences with respect to Mars' orbital parameters, the transfer time, and the run time is given. Note that all runs where $\Sigma_1 > \Sigma_2$ were infeasible, since percentage differences with Mars' semi-major axis turned out to be 20% or larger. Therefore, only runs where $\Sigma_2 \geq \Sigma_1$ were included in Table 14.

Table 14 shows that the optimal pair of weights for the test case with a lightness number of $\beta = 0.01$ is the initial set of $\mathbf{\Sigma} = [100, 100]$. The corresponding optimal control law and trajectory are displayed in Fig. 11. The optimal pair of

Table 14 The optimal results of varying the weights for the constraints for the Earth-Mars transfer optimization, for two different lightness numbers

		$\beta = 0.01$	$\beta = 0.1$
$\boldsymbol{\Sigma} = [1, 10]$	Final semi-major axis [AU]	1.133559	1.215208
	Relative difference with Mars	-25.6195%	-20.2619%
	Final eccentricity	0.000385	0.003368
	Relative difference with Mars	0.0385%	0.3368%
	Transfer time [days]	3685.678	529.511
	Run time [s]	231	16
$\boldsymbol{\Sigma} = [10, 10]$	Final semi-major axis [AU]	1.277432	1.524605
	Relative difference with Mars	-16.1790%	0.0397%
	Final eccentricity	0.003050	0.007983
	Relative difference with Mars	0.3050%	0.7983%
	Transfer time [days]	4010.872	654.691
	Run time [s]	183	162
$\boldsymbol{\Sigma} = [10, 100]$	Final semi-major axis [AU]	1.278743	1.524959
	Relative difference with Mars	-16.0930%	0.0629%
	Final eccentricity	0.000918	0.0130
	Relative difference with Mars	0.0918%	0.9280%
	Transfer time [days]	4035.837	648.305
	Run time [s]	185	162
$\boldsymbol{\Sigma} = [100, 100]$	Final semi-major axis [AU]	**1.524000**	1.525923
	Relative difference with Mars	**0.0000%**	0.1262%
	Final eccentricity	**0.000016**	0.0130
	Relative difference with Mars	**0.0016%**	1.2974%
	Transfer time [days]	**4484.982**	651.804
	Run time [s]	**1251**	17
$\boldsymbol{\Sigma} = [100, 1000]$	Final semi-major axis [AU]	1.511961	**1.523999**
	Relative difference with Mars	-0.7900%	**-0.0000%**
	Final eccentricity	0.001146	**0.006896**
	Relative difference with Mars	0.1146%	**0.6896%**
	Transfer time [days]	4623.771	**506.560**
	Run time [s]	392	**328**
$\boldsymbol{\Sigma} = [100, 10000]$	Final semi-major axis [AU]	1.511960	1.524106
	Relative difference with Mars	-0.7900%	0.0070%
	Final eccentricity	0.001146	0.002121
	Relative difference with Mars	0.1146%	0.2121%
	Transfer time [days]	4623.770	560.926
	Run time [s]	259	272

weights for the test case with a lightness number of $\beta = 0.1$ is $\boldsymbol{\Sigma} = [100, 1000]$. An accurate final semi-major axis of $a = 1.523999$ AU, an eccentricity of 0.006896, and a transfer time of 506.560 days was found. The resulting trajectory and control law can be seen in Fig. 12. Note that, for both cases, the chosen results are a trade-off between transfer time and constraint satisfaction and, for example, slightly better transfer times can be obtained if larger errors on the constraints are allowed.

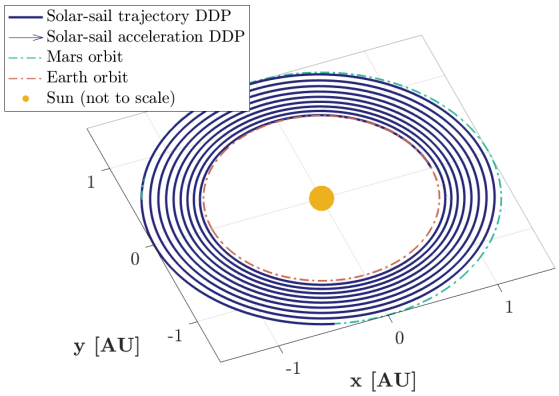
In addition, it can be seen that setting the weights too small ($\Sigma \leq 10$) results in suboptimal solutions. The algorithm then sets more importance on minimizing the objective, instead of meeting the constraints. Furthermore, whereas for $\beta = 0.01$ weights of $\boldsymbol{\Sigma} = [100, 100]$ appeared most optimal, for $\beta = 0.1$ a larger weight on the eccentricity was preferred, namely $\boldsymbol{\Sigma} = [100, 1000]$. The difference in weight on the eccentricity constraint could be explained due to the faster increasing semi-major axis for a larger lightness number. A larger weight is required to still keep the eccentricity in check with this larger increase. The control laws for the cone angle in Fig. 11b and Fig. 12b can be compared to the control law found for optimizing J_a in Fig. 4c. The first test case ($\beta = 0.01$) requires a slightly flattened control law to account for lowering the eccentricity over the longer time it needs to increase the semi-major axis. The second test case ($\beta = 0.1$), on the other hand, requires a larger change of the control law to account for dynamically increasing the semi-major axis and decreasing the final eccentricity. A realistic mission arrival and departure date[‡] for both lightness numbers is given in Table 15, assuming departure from Earth's position and arrival at Mars' position in their circular, co-planar orbits.

Lightness number	Departure date	Departure time [UTC]	Arrival date	Arrival time [UTC]
0.01	May 16th, 2025	01:40:28	August 26th, 2037	01:13:57
0.1	April 4th, 2025	19:01:02	August 24th, 2026	08:28:08

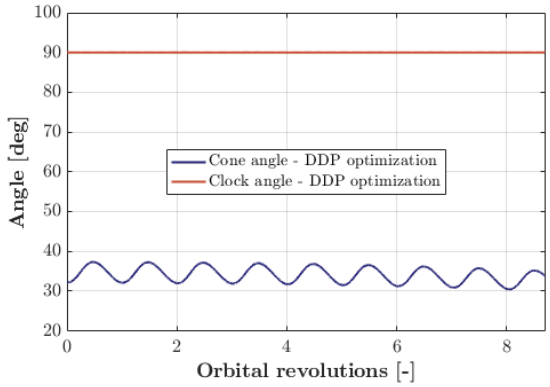
Table 15 Selected mission arrival and departure moments for a lightness number of $\beta = 0.1$ and $\beta = 0.01$.

The final transfer time of the test case with a lightness number of $\beta = 0.1$ can be compared with literature to validate its optimality. The transfer time of 506.560 days is comparable to the transfer time of 505.056 days from Reference [57], where a circular co-planar orbital rendezvous was implemented for a lightness number of $\beta = 0.1$ as well. Other papers find similar orders of magnitude for slightly differing problem definitions, e.g., for a larger lightness number [58, 59] or for an outbound transfer of a heteroclinic connection between Earth and Mars [11]. Reference [11] found that there is a nearly inversely-proportional relationship between the lightness number and the transfer time for a transfer from Earth to a pole-sitter at Mars. Starting from the result of a transfer time of 505 days for $\beta = 0.1$ this would result in a transfer time of 5050 days for $\beta = 0.01$, which is close to the 4485 days obtained by DDP. The found transfer time and orbital parameters could be improved on by further fine tuning of the weights. All in all, both found transfer times do not compare against a transfer time of approximately 260 days for a Hohmann transfer from Earth to Mars [46]. However,

[‡]Information to generate dates and times retrieved from `https://mars.nasa.gov/all-about-mars/night-sky/opposition/`

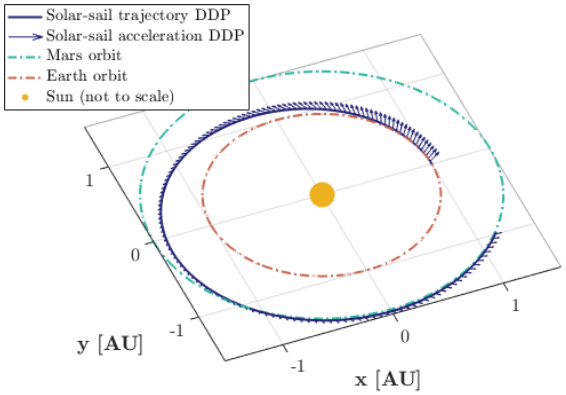


(a) The final optimal Mars transfer.

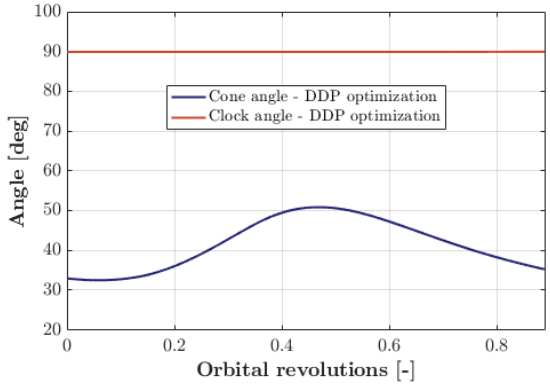


(b) The final optimal control law.

Fig. 11 The optimization results for an Earth-Mars solar-sail transfer trajectory with a lightness number of $\beta = 0.01$.



(a) The final optimal Mars transfer.



(b) The final optimal control law.

Fig. 12 The optimization results for an Earth-Mars solar-sail transfer trajectory with a lightness number of $\beta = 0.1$.

since solar-sail propulsion is not bound to a launch date or an amount of mass, futuristic lightness numbers certainly bring feasible and competitive transfer times to the table [7].

VI. Conclusion

In this paper, the performance of differential dynamic programming (DDP) in the optimization of interplanetary solar-sail trajectories has been investigated. In particular, the DDP algorithm has been compared to locally optimal steering laws, for orbit raising and eccentricity raising, in the interplanetary flight regime for a near-term lightness number after three orbital revolutions. The locally optimal steering law for orbit raising has been compared to the optimization of two objectives, the maximization of the specific orbital energy and the maximization of the semi-major axis, where for both the DDP algorithm has proven to find exceedingly similar solutions. The control-law search across

the iterations was smoother for the less complex objective (the maximization of the semi-major axis). Although both objectives relate to orbit raising, only this objective found a larger semi-major axis than the locally optimal steering law. However, the relative differences in semi-major axis for both objectives were small (+0.0005% or -0.0005%), such that apart from the formulation of the objective, this could also be explained due to small numerical optimization errors. The maximization of the eccentricity also finds a very similar final eccentricity as the locally optimal steering law (+0.1195%). Here, DDP displayed the smoothest control-law search across the iterations, which fits with the objective being the least complex of the three.

Furthermore, the DDP implementation has proven to be robust to different initial guesses and algorithmic settings. For different initial guesses, the same solutions for the final semi-major axis were obtained. However, run times could be reduced by using an initial guess close to the optimum. Concerning the algorithm settings, the optimality tolerance had the largest effect on the solution. Large optimality tolerances can be used in preliminary mission design for fast convergence to a sufficient optimal solution, whereafter the solution can be finetuned with a small optimality tolerance. Other settings had a minimal effect, however, for a more extensive analysis other problem definitions should be considered.

Finally, DDP has proven to be efficient and robust in finding a time-optimal Earth-Mars orbit transfer for a near-term and futuristic lightness number. After finetuning the weights on the constraints, which proved vital, time-optimal trajectories were found that were in accurate compliance with the terminal constraints on the final semi-major axis and eccentricity. To conclude, DDP is a trustworthy and viable method that provides optimal solutions for different constrained and unconstrained optimal controls problems in the interplanetary flight regime.

References

[1] McInnes, C. R., *Solar sailing: technology, dynamics, and mission applications*, Springer, 2004.

[2] Tsuda, Y., Mori, O., Funase, R., Sawada, H., Yamamoto, T., Saiki, T., Endo, T., Yonekura, K., Hoshino, H., and Kawaguchi, J., "Achievement of IKAROS — Japanese deep space solar sail demonstration mission," *Acta Astronautica*, Vol. 82, No. 2, 2013, pp. 183–188. https://doi.org/10.1016/J.ACTAASTRO.2012.03.032.

[3] Heaton, A. F., Faller, B. F., and Katan, C. K., "NanoSail:D Orbital and Attitude Dynamics," *Advances in Solar Sailing*, 2014, pp. 95–113. https://doi.org/10.1007/978-3-642-34907-2{_}7.

[4] Spencer, D. A., Betts, B., Bellardo, J. M., Diaz, A., Plante, B., and Mansell, J. R., "The LightSail 2 solar sailing technology demonstration," *Advances in Space Research*, Vol. 67, No. 9, 2021, pp. 2878–2889. https://doi.org/10.1016/J.ASR.2020.06.029.

[5] Pezent, J., Sood, R., and Heaton, A., "High-fidelity contingency trajectory design and analysis for NASA's near-earth asteroid (NEA) Scout solar sail Mission," *Acta Astronautica*, Vol. 159, 2019, pp. 385–396. https://doi.org/10.1016/J.ACTAASTRO.2019.03.050.

[6] Wilkie, W. K., "Overview of the NASA Advanced Composite Solar Sail System (ACS3) Technology Demonstration Project," *AIAA Scitech 2021 Forum*, 2021. https://doi.org/10.2514/6.2021-1260.

[7] Wright, J. L., *Space Sailing*, Gordon and Breach Science Publishers, 1992.

[8] Dachwald, B., Mengali, G., Quarta, A. A., and Macdonald, M., "Parametric Model and Optimal Control of Solar Sails with Optical Degradation," *Journal of Guidance, Control, and Dynamics*, Vol. 29, No. 5, 2012, pp. 1170–1178. https://doi.org/10.2514/1.20313.

[9] McKay, R. J., MacDonald, M., Biggs, J., and McInnes, C., "Survey of highly-non-Keplerian orbits with low-thrust propulsion," *Journal of Guidance, Control, and Dynamics*, Vol. 34, No. 3, 2011, pp. 645–666. https://doi.org/10.2514/1.52133.

[10] Heiligers, J., Macdonald, M., and Parker, J. S., "Extension of Earth-Moon libration point orbits with solar sail propulsion," *Astrophysics and Space Science*, Vol. 361, No. 7, 2016, pp. 1–20. https://doi.org/10.1007/S10509-016-2783-3/FIGURES/22.

[11] Vergaaij, M., and Heiligers, J., "Time-optimal solar sail heteroclinic-like connections for an Earth-Mars cycler," *Acta Astronautica*, Vol. 152, 2018, pp. 474–485. https://doi.org/10.1016/j.actaastro.2018.08.008.

[12] MacDonald, M., and McInnes, C., "Solar sail science mission applications and advancement," *Advances in Space Research*, Vol. 48, No. 11, 2011, pp. 1702–1716. https://doi.org/10.1016/J.ASR.2011.03.018.

[13] Heiligers, J., Mingotti, G., and McInnes, C. R., "Optimal solar sail transfers between Halo orbits of different Sun-planet systems," *Advances in Space Research*, Vol. 55, No. 5, 2015, pp. 1405–1421. https://doi.org/10.1016/J.ASR.2014.11.033.

[14] Heiligers, J., Parker, J. S., and Macdonald, M., "Novel Solar-Sail Mission Concepts for High-Latitude Earth and Lunar Observation," *Journal of Guidance, Control, and Dynamics*, Vol. 41, No. 1, 2017, pp. 212–230. https://doi.org/10.2514/1.G002919.

[15] Macdonald, M., Hughes, G. W., Mcinnes, C. R., Lyngvi, A., Falkner, P., and Atzei, A., "Solar Polar Orbiter: A Solar Sail Technology Reference Study," *Journal of Spacecraft and Rockets*, Vol. 43, No. 5, 2006. https://doi.org/10.2514/1.16408.

[16] West, J. L., "The Geostorm Warning Mission: enhanced opportunities based on new technology." *14th AAS/AIAA Space Flight Mechanics Conference*, Pasadena, CA: Jet Propulsion Laboratory, National Aeronautics and Space Administration, 2004.

[17] Forgan, D. H., Heller, R., and Hippke, M., "Photogravimagnetic assists of light sails: A mixed blessing for Breakthrough Starshot?" *Monthly Notices of the Royal Astronomical Society*, Vol. 474, No. 3, 2018, pp. 3212–3220. https://doi.org/10.1093/MNRAS/STX2834.

[18] Macdonald, M., McInnes, C., and Hughes, G., "Technology Requirements of Exploration Beyond Neptune by Solar Sail Propulsion," *Journal of Spacecraft and Rockets*, Vol. 47, No. 3, 2012, pp. 472–483. https://doi.org/10.2514/1.46657.

[19] Aziz, J. D., Parker, J. S., Scheeres, D. J., and Englander, J. A., "Low-Thrust Many-Revolution Trajectory Optimization via Differential Dynamic Programming and a Sundman Transformation," *The Journal of the Astronautical Sciences*, Vol. 65, 2018, pp. 205–228. https://doi.org/10.1007/s40295-017-0122-8.

[20] Pritchett, R., "Strategies for Low-Thrust Transfer Design Based on Direct Collocation Techniques," Ph.D. thesis, NASA, 2020.

[21] Topputo, F., and Zhang, C., "Survey of direct transcription for low-thrust space trajectory optimization with applications," *Abstract and Applied Analysis*, Vol. 2014, 2014. https://doi.org/10.1155/2014/851720.

[22] Yam, C. H., Izzo, D., and Lorenzo, D. D., "Low-thrust trajectory design as a constrained global optimization problem," *Proceedings of the Institution of Mechanical Engineers: Part G, Journal of Aerospace Engineering*, Vol. 225, No. 11, 2011, pp. 1243–1251. https://doi.org/10.1177/0954410011401686.

[23] Izzo, D., Sprague, C. I., and Tailor, D. V., "Machine learning and evolutionary techniques in interplanetary trajectory design," *Springer: Optimization in Aerospace Engineering*, Vol. 144, 2018, pp. 191–210. https://doi.org/10.48550/arxiv.1802.00180.

[24] Lantoine, G., and Russell, R. P., "A Hybrid Differential Dynamic Programming Algorithm for Constrained Optimal Control Problems. Part 1: Theory," *Journal of Optimization Theory and Applications*, Vol. 154, 2012, pp. 382–417. https://doi.org/10.1007/s10957-012-0039-0.

[25] Jacobson, D. H., and Mayne, D. Q., *Differential dynamic programming*, American Elsevier Pub. Co, New York, 1970.

[26] Whiffen, G. J., and Sims, A. J., "Application of a novel optimal control algorithm to low-thrust trajectory optimization." *Advances in the Astronautical Sciences*, Vol. 108, No. Pt.2, 2001, pp. 1525–1540. URL https://hdl.handle.net/2014/16157.

[27] Pellegrini, E., and Russell, R. P., "A multiple-shooting differential dynamic programming algorithm. Part 1: Theory," *Acta Astronautica*, Vol. 170, 2020, pp. 686–700. https://doi.org/10.1016/J.ACTAASTRO.2019.12.037.

[28] Pellegrini, E., and Russell, R. P., "A multiple-shooting differential dynamic programming algorithm. Part 2: Applications," *Acta Astronautica*, Vol. 173, 2020, pp. 460–472. https://doi.org/10.1016/J.ACTAASTRO.2019.12.038.

[29] Ozaki, N., Campagnola, S., Yam, H., and Funase, R., "Differential Dynamic Programming Approach for Robust-optimal Low-thrust Trajectory Design Considering Uncertainty," *International Symposium on Space Flight Dynamics*, 2015.

[30] Colombo, C., Vasile, M., and Radice, G., "Optimal low-thrust trajectories to asteroids through an algorithm based on differential dynamic programming," *Celestial Mechanics and Dynamical Astronomy*, Vol. 105, No. 1, 2009, pp. 75–112. https://doi.org/10.1007/S10569-009-9224-3.

[31] Lantoine, G., and Russell, R. P., "A Hybrid Differential Dynamic Programming Algorithm for Constrained Optimal Control Problems. Part 2: Application," *Journal of Optimization Theory and Applications*, Vol. 154, No. 2, 2012, pp. 418–442. https://doi.org/10.1007/S10957-012-0038-1.

[32] Aziz, J. D., Scheeres, D. J., and Lantoine, G., "Differential dynamic programming in the three-body problem," *Space Flight Mechanics Meeting*, American Institute of Aeronautics and Astronautics Inc, AIAA, 2018. https://doi.org/10.2514/6.2018-2223.

[33] Aziz, J. D., Scheeres, D. J., and Lantoine, G., "Hybrid Differential Dynamic Programming in the Circular Restricted Three-Body Problem," *Journal of Guidance, Control, and Dynamics*, Vol. 42, No. 5, 2019, pp. 963–975. https://doi.org/10.2514/1.G003617.

[34] Aziz, J. D., and Scheeres, D. J., "Sundman-Transformed Differential Dynamic Programming with Modified Equinoctial Elements," *The Journal of the Astronautical Sciences*, Vol. 66, 2019, pp. 419–445. https://doi.org/10.1007/s40295-019-00173-4.

[35] Leemans, G., Carzana, L., and Heiligers, J., "Many-Revolution Earth-Centred Solar-Sail Trajectory Optimisation Using Differential Dynamic Programming," *AIAA SCITECH 2022 Forum*, TU Delft, American Institute of Aeronautics and Astronautics, Reston, Virginia, 2022. https://doi.org/10.2514/6.2022-1776.

[36] Vago, J. L., Coates, A. J., Jaumann, R., Korablev, O., Ciarletti, V., Mitrofanov, I., Josset, J.-L., Westall, F., De Sanctis, M. C., Bibring, J.-P., Rull, F., Goesmann, F., Brinckerhoff, W., Raulin, F., Sefton-Nash, E., Svedhem, H., Kminek, G., Rodionov, D., Baglioni, P., and The ExoMars Team, "Searching for Traces of Life With the ExoMars Rover," *From Habitability to Life on Mars*, 2018, pp. 309–347. https://doi.org/10.1016/B978-0-12-809935-3.00011-6.

[37] Szocik, K., Wójtowicz, T., and Baran, L., "War or peace? The possible scenarios of colonising Mars," *Space Policy*, Vol. 42, 2017, pp. 31–36. https://doi.org/10.1016/J.SPACEPOL.2017.10.002.

[38] Zurbuchen, T. H., "NASA Science Mars Exploration Program," *Presentation to the National Academies*, Vol. 28, 2017. URL http://sites.nationalacademies.org/cs/groups/ssbsite/documents/webpage/ssb_181241.pdf.

[39] Stoner, I., "Humans Should Not Colonize Mars," *Journal of the American Philosophical Association*, Vol. 3, No. 3, 2017, pp. 334–353. https://doi.org/10.1017/APA.2017.26.

[40] Morante, D., Rivo, M. S., and Soler, M., "A survey on low-thrust trajectory optimization approaches," , 3 2021. https://doi.org/10.3390/aerospace8030088.

[41] Pascarella, A., Woollands, R., Pellegrini, E., Net, M. S., Xie, H., and Hook, J. V., "Low-thrust trajectory optimization for the solar system pony express," *Acta Astronautica*, 2022. https://doi.org/10.1016/J.ACTAASTRO.2022.11.046.

[42] Wertz, J. R., *Mission Geometry: Orbit and Constellation Design and Management: Spacecraft Orbit and Attitude Systems*, Microcosm Press and Springer, 2001.

[43] Dachwald, B., "Optimal solar-sail trajectories for missions to the outer solar system," *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 6, 2005, pp. 1187–1193. https://doi.org/10.2514/1.13301.

[44] Spencer, D. A., Johnson, L., and Long, A. C., "Solar sailing technology challenges," *Aerospace Science and Technology*, Vol. 93, 2019, p. 105276. https://doi.org/10.1016/J.AST.2019.07.009.

[45] Walker, M. J., Ireland, B., and Owens, J., "A set of modified equinoctial orbit elements," *Celestial mechanics 1985 36:4*, Vol. 36, No. 4, 1985, pp. 409–419. https://doi.org/10.1007/BF01227493.

[46] Wakker, K. F., *Fundamentals of Astrodynamics*, Institutional Repository Library Delft University of Technology, 2015.

[47] Betts, J. T., "Optimal low–thrust orbit transfers with eclipsing," *Optimal Control Applications and Methods*, Vol. 36, No. 2, 2015, pp. 218–240. https://doi.org/10.1002/OCA.2111.

[48] Berry, M., Healy, L., and Antonio, S., "The generalized Sundman transformation for propagation of high-eccentricity elliptical orbits," *AAS/AIAA Space Flight Mechanics Meeting*, Vol. 112, 2002, pp. 913–923.

[49] Bellman, R., "The theory of dynamic programming," *Bulletin of the American Mathematical Society*, Vol. 60, No. 6, 1954, pp. 503–515. https://doi.org/bams/1183519147.

[50] Pellegrini, E., and Russell, R. P., "On the computation and accuracy of trajectory state transition matrices," *Journal of Guidance, Control, and Dynamics*, Vol. 39, No. 11, 2016, pp. 2485–2499. https://doi.org/10.2514/1.G001920.

[51] Montenbruck, O., and Gill, E., *Satellite Orbits: Models, Methods & Applications*, Springer Berlin Heidelberg, 2000. https://doi.org/10.1007/978-3-642-58351-3.

[52] Prince, P. J., and Dormand, J. R., "High order embedded Runge-Kutta formulae," *Journal of Computational and Applied Mathematics*, Vol. 7, No. 1, 1981, pp. 67–75. https://doi.org/10.1016/0771-050X(81)90010-3.

[53] Bellman, R., and Kalaba, R. E., *Dynamic programming and modern control theory*, Academic Press, 1965.

[54] Conn, A. R. A. R., Gould, N. I. M., and Toint, P. L. P. L., *Trust-region methods*, Society for Industrial and Applied Mathematics, 1978.

[55] Dagum, L., and Menon, R., "OpenMP: an industry standard API for shared-memory programming," *IEEE Computational Science and Engineering*, Vol. 5, No. 1, 1998, pp. 46–55. https://doi.org/10.1109/99.660313.

[56] Ross, I. M., Gong, Q., Karpenko, M., and Proulx, R. J., "Scaling and balancing for high-performance computation of optimal controls," *Journal of Guidance, Control, and Dynamics*, Vol. 41, No. 10, 2018, pp. 2086–2097. https://doi.org/10.2514/1.G003382.

[57] Guo, Y., Feng, D., Wang, X., Li, C., and Liu, Y., "The Earth-Mars Transfer Trajectory Optimization of Solar Sail Based on hp-Adaptive Pseudospectral Method," *Discrete Dynamics in Nature and Society*, Vol. 2018, 2018. https://doi.org/10.1155/2018/6916848.

[58] Zhang, M., Wang, G., and Sun, Y., "Solar sail trajectory optimization for an Earth-Mars mission," *Proceedings of the 29th Chinese Control Conference*, 2010, pp. 139–143.

[59] Selvaraj, S., "Interplanetary Trajectory Optimization of Solar Sails," *ProQuest Dissertations and Theses*, 2018, p. 63.

3

Conclusions and recommendations

This work investigated the performance of differential dynamic programming for the optimization of interplanetary solar-sail trajectories. Firstly, the DDP implementation was validated by comparing its results for different objectives in the interplanetary flight regime against known solutions using locally optimal steering laws. A sensitivity analysis was also completed for the algorithm settings and the initial guess. Subsequently, the validated implementation was used to generate time-optimal trajectories for an Earth-Mars orbit transfer. This chapter concludes the thesis work, firstly by giving answers to the posed research questions in Section 3.1, and secondly by stating recommendations for future work in Section 3.2.

3.1. Conclusions

The research questions for this work were structured according to one main research question, with four subquestions that collectively were used to answer the main research question. The first subquestion focuses on the validation of the use of DDP for interplanetary solar-sail trajectory optimization, the next two subquestions focus on the sensitivity of DDP to different settings, and the last subquestion focuses on the performance of DDP for generating time-optimal trajectories for an Earth-Mars orbit transfer. The main research question is defined as:

What is the performance of DDP for the optimization of interplanetary solar-sail trajectories?

For the purpose of this work, the dynamical framework is based on the two-body problem, with the Sun as the main gravitational force, augmented with an ideal solar-sail force model. A superior numerical performance is obtained for the DDP algorithm by propagating the state in modified equinoctial elements and applying a Sundman transformation to change the independent variable from time to the true anomaly. In order to answer the main question, the answer to each subquestion is given separately, wherafter the answer to the main question is given at the end of the section.

1. **How does DDP perform in terms of optimality of the solution when compared to locally optimal steering laws in the interplanetary flight regime?**

   In order to answer this question, different orbital elements were maximized for three orbital revolutions starting from a heliocentric ecliptic eccentric orbit ($a = 1$ AU, $e = 0.2$), for a near-term lightness number of $\beta = 0.01$. The optimization results were compared to locally optimal steering laws for orbit raising and eccentricity raising. For the optimization, an initial guess was used that assumed a constant cone and clock angle of zero throughout the trajectory.

   - *Orbit raising*
     For comparison to the first locally optimal steering law for orbit raising, two different objective formulations were used. The first formulation, which was the maximization of the specific orbital energy, found a final semi-major axis that was 0.0213% smaller than the semi-major axis found with the locally optimal steering law. However, this optimization did result in a

larger specific orbital energy. The second formulation, the maximization of the semi-major axis, found a semi-major axis that was 0.0005% larger than the semi-major axis from the locally optimal steering law. In addition, for the second formulation, DDP displayed more successful iterations and thus leads to more reliable approximations of the model and a smoother control-law flow across the iterations of the DDP algorithm. Also, fewer iterations were required such that DDP converged to a solution considerably quicker. The first formulation of maximizing the specific orbital energy is more complex and the cost function values are in a larger order of magnitude than the second formulation (maximizing the semi-major axis), which contributes to the manner DDP handles the optimization. Consequently, the optimality of the solution compared to the locally optimal steering law is influenced slightly by the formulation of the problem that is used for the optimization. Still, it can be concluded that both formulations resulted in valid solutions that are very similar to the locally optimal steering laws.

- *Eccentricity raising*
  The second optimization of eccentricity raising found a final eccentricity that was 0.1195% larger than the eccentricity found with the locally optimal steering law. This optimization displayed more successful iterations and an even smoother control-law flow across the iterations than was displayed for the orbit-raising objective, which fits with the less complex objective formulation.

All in all, DDP finds similar or slightly more optimal solutions than the considered locally optimal steering laws. However, apart from explaining this due to the different formulation of the objectives, the small differences could also be explained due to numerical optimization errors. Still, DDP is considered valid for use with different objectives in the interplanetary flight regime. Although DDP logically requires a larger computational effort to solve these problems than the locally optimal steering laws, it is expected from literature that DDP is able to solve more complex problems with multiple constraints that locally optimal steering laws cannot solve.

2. **What is the effect on the optimality of the solution and the run time when using different initial guesses for the optimization?**

   In order to answer this question, the semi-major axis was maximized for one orbital revolution starting from a heliocentric ecliptic eccentric orbit ($a = 1$ AU, $e = 0.2$) for a near-term lightness number of $\beta = 0.01$ and for different constant initial guesses for the control law. The optimization results for each initial guess were compared to a locally optimal steering law. All initial guesses assumed a constant cone and clock angle throughout the trajectory, where the constant values differed across the different initial guesses. Along with feeding constant initial guesses for the controls into the algorithm, the locally optimal steering law itself was also used as an initial guess. All initial guesses resulted in an optimal value for the final semi-major axis, where the difference with the locally optimal steering law solution ranged from -0.0362% to +0.0006% and 66% of the initial guesses resulted in a difference of +0.0005%. Using the locally optimal steering law itself as an initial guess resulted in a relative difference of the final semi-major axis with the locally optimal steering law of 0.0007%. ”Good” initial guesses (e.g., with $\delta = 90^\circ$, that resulted in in-plane motion in the initial guess) allow a large decrease of the run time: in the best case, the run time could be reduced by approximately 99.4% with respect to the initial guess resulting in the longest run time, which is similar to the decrease in run time when using the locally optimal steering law as an initial guess. Nonetheless, ”good” initial guesses also resulted in a smaller amount of successful iterations of the DDP algorithm, which leads to convergence to a slightly less optimal value for the final semi-major axis. On the other hand, poorer initial guesses (e.g., with $\delta \neq 90^\circ$, that resulted in out-of-plane motion in the initial guess) found comparable optimality as when the locally optimal steering law was used as an initial guess. ”Good” initial guesses are already close to the optimum: the smaller share of successful iterations that the optimization displays indicates poorer approximations of the model by DDP (perhaps overstepping in its search to less optimal values), therefore converging to a slightly more local optimum, although with a significantly shorter run time. All in all, DDP is proven to be robust to a variety of initial guesses. All initial guesses result in a valid solution when comparing to the locally optimal steering law. Nonetheless, the initial guesses influence the run time of the algorithm significantly.

3. **What is the effect on the optimality of the solution and the run time when using different settings for the DDP algorithm?**

   A sensitivity analysis was performed to investigate the effect of different algorithmic settings on the optimality of the solution and the algorithm's run time. The effect was analyzed for the maximization of the semi-major axis for one orbital revolution from a heliocentric ecliptic eccentric orbit ($a = 1$ AU, $e = 0.2$) for a near-term lightness number of $\beta = 0.01$. The optimality tolerance had the largest effect on the run time as well as on the optimality of the solution. Large optimality tolerances can be used in preliminary mission design to obtain a quick, valid solution, whereafter a small optimality tolerance can be used to finetune the optimality of the solution. The reduction ratio tolerance must not be set too large, since the algorithm will converge to a suboptimal solution, yet on the other hand must not be set too small, since the algorithm will converge too fast, also resulting in a suboptimal solution. The variation of the initial trust region radius, the maximum trust region radius, and the trust region scaling factor resulted in a minimal effect on the run time as well as on the optimality of the solution.

4. **How does DDP perform in terms of optimality of the solution and meeting the constraints for an Earth-Mars orbit transfer with respect to solutions known from literature?**

   An Earth-Mars transfer test case was investigated by assuming circular, co-planar orbits for Earth and Mars. The transfer from Earth orbit to Mars orbit was modeled using a realistic, near-term lightness number ($\beta = 0.01$) as well as a futuristic lightness number ($\beta = 0.1$). A time-optimal transfer was sought that best met two constraints simultaneously, namely the difference of the final semi-major axis and the final eccentricity with the orbital parameters of Mars. The current DDP implementation does not support a dynamic number of stages, such that the minimization of the time of flight is severely limited by the fixed number of orbital revolutions given at the beginning of the optimization. Consequently, an iterative process was used for the fixed amount of orbital revolutions to find the optimal solution. The fixed number of orbital revolutions was decreased for multiple runs until no more sufficient solutions were found by DDP. One of the runs was selected as the optimal solution by a trade-off between the shortest time of flight as well as sufficient compliance with the terminal constraints. In addition, a sensitivity analysis was performed for the weights that were set on the constraints. The optimality of the optimization proved to be sensitive to the values of the weights. Larger weight values resulted in more compliance with the corresponding constraint, where for smaller weight values the emphasis was more on the objective itself. The case with the futuristic lightness number ($\beta = 0.1$) required a larger weight on the eccentricity constraint than on the semi-major axis constraint, in contrast to the case with the near-term lightness number ($\beta = 0.01$), where the weights were allowed to be similar. This is linked to the fact that for the futuristic case, the semi-major axis increases more rapidly, such that more correction is required for the eccentricity. Finally, for the test case where the lightness number was $\beta = 0.1$, an optimal trajectory was found that resulted in a transfer time of approximately 506 days, a constraint compliance for the eccentricity of 0.6896%, and a constraint compliance for the semi-major axis of -0.0000%. For the test case where the lightness number was $\beta = 0.01$, an optimal trajectory was found that resulted in a transfer time of approximately 4485 days, a constraint compliance for the eccentricity of 0.0016%, and a constraint compliance for the semi-major axis of 0.0000%. The found transfer times and orbital parameters could be improved on by further fine tuning of the weights. Additionally, note that slightly better transfer times could be obtained if larger errors on the constraints are allowed. After comparing to literature, where, e.g., for a lightness number of $\beta = 0.1$ a transfer time of 505 days was found, it can be concluded that DDP successfully finds a time-optimal solution for an Earth-Mars transfer, complying with the two terminal constraints.

Ultimately, the main research question can be answered by keeping the answers to the subquestions in mind. DDP slightly outperforms or performs similarly as locally optimal steering laws for different orbital elements in the interplanetary flight regime: for the maximization of both the semi-major axis and the eccentricity, DDP finds larger final values. Although more complex Lagrangian cost functions, where multiple state parameters are involved simultaneously, result in more difficulty across the iterations for DDP to find the optimum, DDP still finds more optimal final values than the locally optimal steering laws. Furthermore, DDP is robust to different initial guesses and algorithmic settings. Finally, the algorithm is

able to comply with multiple terminal constraints for the optimization of realistic interplanetary solar-sail trajectories, such as an Earth-Mars orbit transfer, where the DDP implementation found similar results as in literature.

3.2. Recommendations

A number of recommendations are formulated for future research based on the findings and conclusions of this thesis work. Here, topics are highlighted that need further investigation in order to advance research in the application of the DDP algorithm to solar-sail trajectory optimization. Suggestions are given regarding the dynamical models, the DDP algorithm itself, and other applications. The recommendations are divided according to these topics to provide a clear overview.

Dynamical models

Improvements can be made to this work by employing higher fidelity dynamical models. First and foremost, a two-body problem is employed in this work. This framework assumes that the motion of the spacecraft is only influenced by the gravitational force of one celestial body, in this case the Sun. Multiple celestial objects are involved when in interplanetary space, which means that a framework that assumes the motion under influence of multiple bodies is more realistic. A circular restricted three-body problem, which assumes circular motion of two bodies around the barycenter, or an elliptic restricted three-body problem, which accounts for the eccentric orbits of the two bodies around the barycenter, are more realistic frameworks to model the problem. For instance, the Earth-Mars orbit transfer could then be modeled in two phases; the first a three-body problem with the Sun and the Earth and the second with the Sun and Mars [38, 44]. The dynamical framework can be also extended by adding third- or fourth-body perturbations, corresponding to the gravitational influence of the other bodies in the solar system. In addition, ephemeris data can provide more precise planet positions to ensure an even more realistic model.

Furthermore, to model the solar-sail acceleration, an ideal force model was assumed. This framework assumes that all photons impacting the sail are specularly reflected. Also taking into account the force on the solar sail due to emission, absorption, and diffuse reflection (optical force model) or the bulging of the sail (parametric force model) provides for a more realistic framework [1]. Reference [45] evaluated the differences between more realistic solar-sail force models and the ideal force model for a heliocentric Mars rendezvous: for an optical model the flight time increased with 12-13% compared to the ideal case and for a parametric model with 14-15%.

Differential dynamic programming algorithm

In this work, the DDP algorithm was implemented for application to a singular-phase trajectory with a fixed amount of stages. A strong recommendation is to extend this implementation to handle multi-phase trajectories. This will allow application to a wider range of missions and linkage of different Lagrangian cost functions to the separate phases. To ensure DDP's applicability to more interplanetary problems, it is important to include a variable amount of stages. By keeping the amount of orbital revolutions for the trajectory variable, the time of flight can be optimized freely, without the limitation of a fixed endpoint in terms of the number of orbital revolutions, and *true* time-optimal trajectories can be found. The DDP implementation can be extended with the use of problem parameters, where the time of flight could be included as a static parameter.

Additionally, the current implementation of DDP does not support the use of Cartesian coordinates in the propagation of the state trajectories. MEE ensures better accuracy for a larger step size, which reduces the computational effort [15]. Also, a Sundman transformation is applied, which is advantageous for use in optimization of especially eccentric orbits. When the orbit is under influence of large perturbations, precise position and velocity constraints need to be targeted, or when a more realistic dynamical framework is used, a Cartesian coordinate set may be useful [46]. Aside from using smaller step sizes, it is recommended to use a different scaling method to ensure valid propagation in Cartesian coordinates. A balancing technique can be used to scale the variables and increase the algorithm's efficiency in the propagation of Cartesian coordinates [47].

The simple method used in this work has proved robust and efficient in solving the trust region quadratic subproblem. However, DDP is sensitive to the shape of the scaling matrix, especially when small variations in a few parameters affect the variations in the Lagrangian cost function more than others [15]. For instance, this is the case when a terminal constraint is set on tracking the true longitude

of Mars. Reference [25] suggests scaling methods that could improve this. It is also advisable to extend the DDP algorithm implemented in this work by the implementation of path-inequality constraints, where costs can be induced locally at each stage to enforce compliance with a constraint. This could be particularly useful to ensure the rate of change of the control angles stays within bounds, which is an important factor in realistic solar-sail trajectories [48].

Other applications

In this work, the implemented DDP algorithm was applied to the optimization of different orbital elements in the interplanetary regime. Additionally, it was applied to an Earth-Mars orbit transfer. The versality of DDP can be tested through application to more solar-sail problems. Firstly, since DDP is exceptionally efficient for the optimization of high-dimensional multi-phase trajectories, it is recommended to investigate the use of DDP for finding optimal trajectories to visit multiple asteroids [25].

Another characteristic of DDP that should be exploited is its capability for solving problems with a large amount of stages: Reference [32] already successfully employed DDP in many-revolution Earth-centered solar-sail orbits, yet this could be extended to many-revolution orbits in an interplanetary or interstellar flight regime. This could translate to solar-system escape orbits, or orbit cranking to achieve a (solar) polar orbit.

Furthermore, an interesting concept to explore with DDP is hybrid thrust, where the advantages of solar-sail propulsion and solar-electric propulsion complement one another. The solar-electric propulsion component ensures the possibility of an acceleration in the direction of the Sun, which is not possible with solar-sail propulsion alone. For instance, mission applications include Earth-observation pole-sitters [49], loosely-displaced geostationary orbits [50], or many other non-Keplerian orbits.

References

[1] McInnes, C. R. (2004). *Solar sailing: technology, dynamics, and mission applications*. Springer.

[2] Wright, J. L. (1992). *Space Sailing*. Gordon; Breach Science Publishers.

[3] Dachwald, B., Mengali, G., Quarta, A. A., & Macdonald, M. (2012). Parametric Model and Optimal Control of Solar Sails with Optical Degradation. *Journal of Guidance, Control, and Dynamics*, *29*(5), 1170–1178. https://doi.org/10.2514/1.20313

[4] Wiley, C. A. (1951). Clipper Ships of Space. *Astounding Science Fiction*.

[5] Tsuda, Y., Mori, O., Funase, R., Sawada, H., Yamamoto, T., Saiki, T., Endo, T., Yonekura, K., Hoshino, H., & Kawaguchi, J. (2013). Achievement of IKAROS — Japanese deep space solar sail demonstration mission. *Acta Astronautica*, *82*(2), 183–188. https://doi.org/10.1016/J.ACTAASTRO.2012.03.032

[6] Heaton, A. F., Faller, B. F., & Katan, C. K. (2014). NanoSail:D Orbital and Attitude Dynamics. *Advances in Solar Sailing*, 95–113. https://doi.org/10.1007/978-3-642-34907-2{_}7

[7] Spencer, D. A., Betts, B., Bellardo, J. M., Diaz, A., Plante, B., & Mansell, J. R. (2021). The LightSail 2 solar sailing technology demonstration. *Advances in Space Research*, *67*(9), 2878–2889. https://doi.org/10.1016/J.ASR.2020.06.029

[8] Pezent, J., Sood, R., & Heaton, A. (2019). High-fidelity contingency trajectory design and analysis for NASA's near-earth asteroid (NEA) Scout solar sail Mission. *Acta Astronautica*, *159*, 385–396. https://doi.org/10.1016/J.ACTAASTRO.2019.03.050

[9] Wilkie, W. K. (2021). Overview of the NASA Advanced Composite Solar Sail System (ACS3) Technology Demonstration Project. *AIAA Scitech 2021 Forum*. https://doi.org/10.2514/6.2021-1260

[10] Lubin, P. (2016). A Roadmap to Interstellar Flight. *JBIS - Journal of the British Interplanetary Society*, *69*(2-3), 40–72. https://arxiv.org/abs/1604.01356v8

[11] Seefeldt, P., Grundmann, J. T., Hillebrandt, M., & Zander, M. (2021). Performance analysis and mission applications of a new solar sail concept based on crossed booms with tip-deployed membranes. *Advances in Space Research*, *67*(9), 2736–2745. https://doi.org/10.1016/J.ASR.2020.10.001

[12] MacDonald, M., & McInnes, C. (2011). Solar sail science mission applications and advancement. *Advances in Space Research*, *48*(11), 1702–1716. https://doi.org/10.1016/J.ASR.2011.03.018

[13] Rao, A. (2010). A Survey of Numerical Methods for Optimal Control. *Advances in the Astronautical Sciences*, *135*. https://www.researchgate.net/publication/268042868_A_Survey_of_Numerical_Methods_for_Optimal_Control

[14] Betts, J. T. (2010). *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming* (2nd ed.). Society for Industrial; Applied Mathematics. https://doi.org/10.1137/1.9780898718577

[15] Aziz, J. D., Parker, J. S., Scheeres, D. J., & Englander, J. A. (2018). Low-Thrust Many-Revolution Trajectory Optimization via Differential Dynamic Programming and a Sundman Transformation. *The Journal of the Astronautical Sciences*, *65*, 205–228. https://doi.org/10.1007/s40295-017-0122-8

[16] Pritchett, R. (2020). *Strategies for Low-Thrust Transfer Design Based on Direct Collocation Techniques* (Doctoral dissertation). NASA.

[17] Topputo, F., & Zhang, C. (2014). Survey of direct transcription for low-thrust space trajectory optimization with applications. *Abstract and Applied Analysis*, *2014*. https://doi.org/10.1155/2014/851720

[18] Yam, C. H., Izzo, D., & Lorenzo, D. D. (2011). Low-thrust trajectory design as a constrained global optimization problem. *Proceedings of the Institution of Mechanical Engineers: Part G, Journal of Aerospace Engineering*, *225*(11), 1243–1251. https://doi.org/10.1177/0954410011401686

[19] Izzo, D., Sprague, C. I., & Tailor, D. V. (2018). Machine learning and evolutionary techniques in interplanetary trajectory design. *Springer: Optimization in Aerospace Engineering*, *144*, 191–210. https://doi.org/10.48550/arxiv.1802.00180

[20] Jacobson, D. H., & Mayne, D. Q. (1970). *Differential dynamic programming*. American Elsevier Pub. Co.

[21] Bellman, R. (1954). The theory of dynamic programming. *Bulletin of the American Mathematical Society*, *60*(6), 503–515. https://doi.org/bams/1183519147

[22] Lantoine, G., & Russell, R. P. (2012a). A Hybrid Differential Dynamic Programming Algorithm for Constrained Optimal Control Problems. Part 1: Theory. *Journal of Optimization Theory and Applications*, *154*, 382–417. https://doi.org/10.1007/s10957-012-0039-0

[23] Whiffen, G. J., & Sims, A. J. (2001). Application of a novel optimal control algorithm to low-thrust trajectory optimization. *Advances in the Astronautical Sciences*, *108*(Pt.2), 1525–1540. https://hdl.handle.net/2014/16157

[24] Colombo, C., Vasile, M., & Radice, G. (2009). Optimal low-thrust trajectories to asteroids through an algorithm based on differential dynamic programming. *Celestial Mechanics and Dynamical Astronomy*, *105*(1), 75–112. https://doi.org/10.1007/S10569-009-9224-3

[25] Lantoine, G., & Russell, R. P. (2012b). A Hybrid Differential Dynamic Programming Algorithm for Constrained Optimal Control Problems. Part 2: Application. *Journal of Optimization Theory and Applications*, *154*(2), 418–442. https://doi.org/10.1007/S10957-012-0038-1

[26] Aziz, J. D., Scheeres, D. J., & Lantoine, G. (2018). Differential dynamic programming in the three-body problem. *Space Flight Mechanics Meeting*, (210009). https://doi.org/10.2514/6.2018-2223

[27] Aziz, J. D., Scheeres, D. J., & Lantoine, G. (2019). Hybrid Differential Dynamic Programming in the Circular Restricted Three-Body Problem. *Journal of Guidance, Control, and Dynamics*, *42*(5), 963–975. https://doi.org/10.2514/1.G003617

[28] Aziz, J. D., & Scheeres, D. J. (2019). Sundman-Transformed Differential Dynamic Programming with Modified Equinoctial Elements. *The Journal of the Astronautical Sciences*, *66*, 419–445. https://doi.org/10.1007/s40295-019-00173-4

[29] Pellegrini, E., & Russell, R. P. (2020a). A multiple-shooting differential dynamic programming algorithm. Part 1: Theory. *Acta Astronautica*, *170*, 686–700. https://doi.org/10.1016/J.ACTAASTRO.2019.12.037

[30] Pellegrini, E., & Russell, R. P. (2020b). A multiple-shooting differential dynamic programming algorithm. Part 2: Applications. *Acta Astronautica*, *173*, 460–472. https://doi.org/10.1016/J.ACTAASTRO.2019.12.038

[31] Ozaki, N., Campagnola, S., Yam, H., & Funase, R. (2015). Differential Dynamic Programming Approach for Robust-optimal Low-thrust Trajectory Design Considering Uncertainty. *International Symposium on Space Flight Dynamics*.

[32] Leemans, G., Carzana, L., & Heiligers, J. (2022). Many-Revolution Earth-Centred Solar-Sail Trajectory Optimisation Using Differential Dynamic Programming. *AIAA SCITECH 2022 Forum*. https://doi.org/10.2514/6.2022-1776

[33] Percy, T., Taylor, T., & Powell, T. (2004). A Study of Possible Solar Sail Applications for Mars Missions. *NASA Technical Reports Server*. https://doi.org/10.2514/6.2004-3996

[34] Vago, J. L., Coates, A. J., Jaumann, R., Korablev, O., Ciarletti, V., Mitrofanov, I., Josset, J.-L., Westall, F., De Sanctis, M. C., Bibring, J.-P., Rull, F., Goesmann, F., Brinckerhoff, W., Raulin, F., Sefton-Nash, E., Svedhem, H., Kminek, G., Rodionov, D., Baglioni, P., & The ExoMars Team. (2018). Searching for Traces of Life With the ExoMars Rover. *From Habitability to Life on Mars*, 309–347. https://doi.org/10.1016/B978-0-12-809935-3.00011-6

[35] Szocik, K., Wójtowicz, T., & Baran, L. (2017). War or peace? The possible scenarios of colonising Mars. *Space Policy*, *42*, 31–36. https://doi.org/10.1016/J.SPACEPOL.2017.10.002

[36] Zurbuchen, T. H. (2017). NASA Science Mars Exploration Program. *Presentation to the National Academies*, *28*. http://sites.nationalacademies.org/cs/groups/ssbsite/documents/webpage/ssb_181241.pdf

[37] Stoner, I. (2017). Humans Should Not Colonize Mars. *Journal of the American Philosophical Association*, *3*(3), 334–353. https://doi.org/10.1017/APA.2017.26

[38] Vergaaij, M., & Heiligers, J. (2018). Time-optimal solar sail heteroclinic-like connections for an Earth-Mars cycler. *Acta Astronautica*, *152*, 474–485. https://doi.org/10.1016/j.actaastro.2018.08.008

[39] Russell, R. P., & Ocampo, C. A. (2012). Systematic Method for Constructing Earth-Mars Cyclers Using Free-Return Trajectories. *https://doi.org/10.2514/1.1011*, *27*(3), 321–335. https://doi.org/10.2514/1.1011

[40] Heiligers, J., Mingotti, G., & McInnes, C. R. (2015). Optimal solar sail transfers between Halo orbits of different Sun-planet systems. *Advances in Space Research*, *55*(5), 1405–1421. https://doi.org/10.1016/J.ASR.2014.11.033

[41] Otten, M., & McInnes, C. R. (2012). Near Minimum-Time Trajectories for Solar Sails. *Journal of Guidance, Control, and Dynamics*, *24*(3), 632–634. https://doi.org/10.2514/2.4758

[42] Guo, Y., Feng, D., Wang, X., Li, C., & Liu, Y. (2018). The Earth-Mars Transfer Trajectory Optimization of Solar Sail Based on hp-Adaptive Pseudospectral Method. *Discrete Dynamics in Nature and Society*, *2018*. https://doi.org/10.1155/2018/6916848

[43] Zhang, M., Wang, G., & Sun, Y. (2010). Solar sail trajectory optimization for an Earth-Mars mission. *Proceedings of the 29th Chinese Control Conference*, 139–143.

[44] Heiligers, J., Vergaaij, M., & Ceriotti, M. (2021). End-To-End Trajectory Design for a Solar-Sail-Only Pole-Sitter at Venus, Earth, and Mars. *Advances in Space Research*, *67*(9), 2995–3011. https://doi.org/10.1016/J.ASR.2020.06.011

[45] Mengali, G., & Quarta, A. A. (2012). Optimal Three-Dimensional Interplanetary Rendezvous Using Non-Ideal Solar Sail. *Journal of Guidance, Control, and Dynamics*, *28*(1), 173–177. https://doi.org/10.2514/1.8325

[46] Wakker, K. F. (2015). *Fundamentals of Astrodynamics*. Institutional Repository Library Delft University of Technology.

[47] Ross, I. M., Gong, Q., Karpenko, M., & Proulx, R. J. (2018). Scaling and balancing for high-performance computation of optimal controls. *Journal of Guidance, Control, and Dynamics*, *41*(10), 2086–2097. https://doi.org/10.2514/1.G003382

[48] Liu, J., Rong, S., Shen, F., & Cui, N. (2014). Dynamics and Control of a Flexible Solar Sail. *Mathematical Problems in Engineering*, *2014*, 1–25. https://doi.org/10.1155/2014/868419

[49] Ceriotti, M., & McInnes, C. R. (2011). Hybrid solar sail and solar electric propulsion for novel Earth observation missions. *Acta Astronautica*, *69*(9-10), 809–821. https://doi.org/10.1016/j.actaastro.2011.06.007

[50] Liu, Y., Heiligers, J., & Ceriotti, M. (2018). Loosely-displaced geostationary orbits with hybrid sail propulsion. *Aerospace Science and Technology*, *79*, 105–117. https://doi.org/10.1016/j.ast.2018.05.034

[51] Sargent, R. G. (2013). Verification and validation of simulation models. *Journal of Simulation*, *7*(1), 12–24. https://doi.org/10.1057/JOS.2012.20/FIGURES/7

[52] Prince, P. J., & Dormand, J. R. (1981). High order embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*, *7*(1), 67–75. https://doi.org/10.1016/0771-050X(81)90010-3

[53] Montenbruck, O., & Gill, E. (2000). *Satellite Orbits: Models, Methods & Applications*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-58351-3

[54] Roy, A. E. (E. (1982). *Orbital motion* (4th ed.). Institute of Physics Publishing.

[55] Walker, M. J., Ireland, B., & Owens, J. (1985). A set of modified equinoctial orbit elements. *Celestial mechanics 1985 36:4*, *36*(4), 409–419. https://doi.org/10.1007/BF01227493

A

Verification & validation

The purpose of this appendix is to give a description of the verification and validation of the implementation of the numerical methods (Section A.1) and dynamical models (Section A.2) used in this work, and to verify the implementation of DDP (Section A.3). Verification is the procedure where the accuracy of the results is investigated [51]. Here, the model's implementation and the data are tested: a comparison is made with the conceptual model. In the validation procedure, on the other hand, the results are compared to real-world results. The conceptual model is validated by comparing to the real system.

A.1. Numerical integration

In this work, a fixed-step Dormand-Prince Runge-Kutta 8(7) (DORPI8) integrator [52] was used to integrate the dynamics. The choice for this integrator arose from Reference [53], where the integrator was recommended as an all-round high-accuracy fixed step-size integrator. In addition, DOPRI8 was used in combination with DDP before [15, 27, 28, 32]. Since the DOPRI8 integrator has been implemented from scratch for the C++ program, it was required to verify and validate the implementation. Firstly, Section A.1.1 focuses on the verification of the implemented integrator by comparison to the outcome of a trajectory integration; a known example case from Tudat[1] was integrated with a verified and validated variable step-size MATLAB® integrator, `ode45`. Subsequently, the implementation of DOPRI8 is validated in Section A.1.2 by comparing its outcome to the outcome of other fixed step-size integrators, that are implemented in Tudat.

A.1.1. Verification

For integrators implemented in Tudat or MATLAB®, it is assumed that they are verified. However, for self-implemented integrators, such as the DOPRI8 integrator, it is required to verify the numerical integration. The DOPRI8 integrator is verified by comparing its results to the outcome of a trajectory integration in literature. An initial state was retrieved from an example in Tudat[2] and is displayed in Table A.1. In order to verify the self-implemented DOPRI8 fixed step-size integrator, the initial state was forward propagated with a step size of one second for 86400 seconds, assuming a two-body problem around Earth, where the gravitational parameter of the Earth is equal to $\mu_{\oplus} = 3.98600441800000 \times 10^{14}$ $\mathrm{m^3 s^{-2}}$. The initial state in Table A.1 was propagated forward using the variable step-size MATLAB® integrator `ode45`. The difference in position norm between this result and the result of the DOPRI8 propagation, after 86400 seconds, was 1.0144×10^{-5} m. The entire trajectory deviation can be seen in Figure A.1. A small, increasing, periodic difference can be seen: a similar periodic deviation is also perceived when comparing two verified and validated integrators in Tudat with one another. Additionally, this behaviour is amplified when a different, less precise value is used for the gravitational parameter.

[1] Documentation found at https://docs.tudat.space/en/stable/; Code found at https://github.com/tudat-team/tudat-bundle, February 2022.

[2] Retrieved from https://docs.tudat.space/en/latest/_src_getting_started/_src_examples/notebooks/propagation/keplerian_satellite_orbit.html, February 2022.

Table A.1: Initial state in Keplerian elements.

Kepler element	a [km]	e	i [deg]	ω [deg]	Ω [deg]	ν [deg]
Value	7500	0.1	85.3°	235.7°	23.4°	139.87°

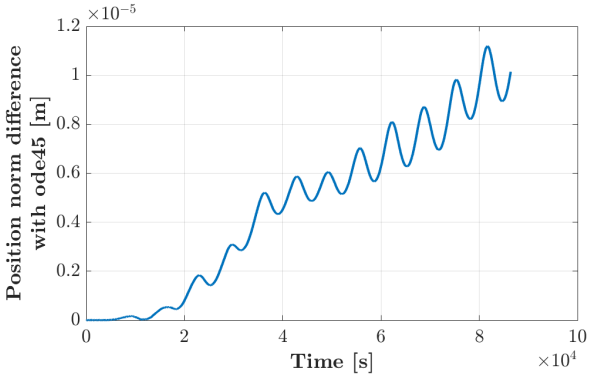


Figure A.1: The difference in position norm of DOPRI8 with MATLAB's ode45 scheme.

A.1.2. Validation

To validate the choice of the DOPRI8 integator, its results are compared to other fixed step-size integrators. The integrator can be validated by making use of the pre-programmed different integrators in Tudat and comparing their outcomes to a benchmark solution. To generate the benchmark, the Runge-Kutta 4 was used; an allround fixed step-size integrator that is verified and validated in Tudat. The deviations from the benchmark must be compared, as well as the number of iterations that are required. Different verified and validated (forced) fixed step-size integrators in Tudat were used to integrate (with a step size of one second) the same trajectory as in Section A.1.1: Adam-Bashforth 6-11, Bulirsch-Stoer 8, Bulirsch-Stoer 4, Runge-Kutta 4(5), Runge-Kutta 5(6), and Runge-Kutta 7(8). These results could be compared with the results of the DOPRI8 scheme for validation. For the comparison, an accurate benchmark solution, generated with a very small step size, was required. However, one must keep in mind that rounding errors can dominate when the step size is too small. Hence, a step size of 0.1 seconds was chosen.

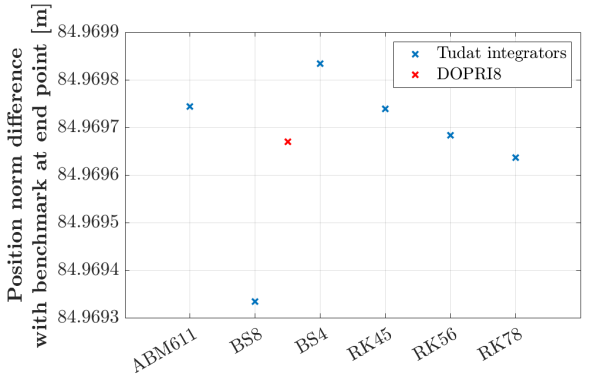


Figure A.2: The difference in position norm with the benchmark for multiple integrators after propagating the initial state in Table A.1 for 86400 seconds.

Figure A.2 displays the position norm difference with the benchmark obtained for each integrator. These differences all lie in a range of 5.0004×10^{-4} m. It can be seen that the Bulirsch-Stoer (BS) integration resulted in the smallest difference with the benchmark. However, it must be noted that BS8 requires the largest amount of function evaluations; an order of magnitude larger than the other integrators. The Runge-Kutta and Adam-Bashforth-Moulton integrators require the least amount of function evaluations, followed by DOPRI8 and finally, the BS integrators. However, when the step size

increases, e.g., to 300 seconds, differences in final position magnitude with respect to the benchmark become much larger for the Runge-Kutta and the Adam-Bashforth-Moulton integrators. The BS integrators and DOPRI8 find more accurate final position magnitudes for increasing step sizes, and because of the large number of function evaluations that BS8 needs, the self-implemented DOPRI8 integrator is proclaimed valid.

A.2. Dynamical models

To verify the correct implementation of the dynamical framework used in this work, which is based on the two-body problem augmented with an ideal solar-sail force model and presented in a heliocentric reference frame, $\mathcal{H}$, an equilibrium solution was sought in the form of a displaced, circular, Earth-synchronous one-year orbit. The implementation of the dynamical models is considered verified when after exactly one Earth year, the trajectory arrives at the same position with the same velocity. In order to verify all dimensions possible, the propagation took place outside of the two-dimensional planes of the reference frame, and an initial state was used as stated in Table A.2. Using Equations 5.10a and 5.10b from [1], a cone angle and lightness number were found that correspond to a displaced, circular, Earth-synchronous one-year orbit. These variables are stated in Table A.2, where in Figure A.3 the results of the propagation, using the implementation in this work, can be seen.

Table A.2: The initial conditions corresponding to a displaced, circular, Earth-synchronous orbit.

β	0.832309820332246
α [deg]	28.371665602288850°
x [AU]	0.5
y [AU]	0.5
z [AU]	0.5
v_x [m/s]	-14892.62721266109
v_y [m/s]	14892.62721266109
v_z [m/s]	0.0

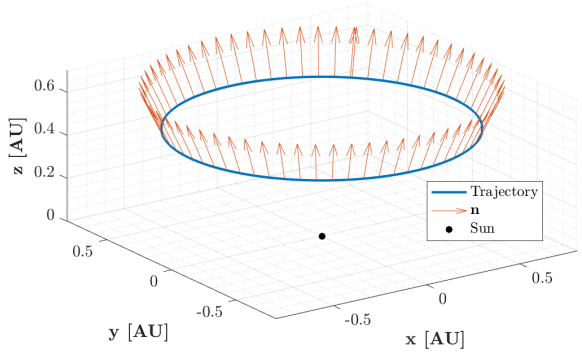


Figure A.3: The closed orbit solution when propagating the initial conditions of Table A.2.

The difference in the Cartesian state, for forward propagating in Cartesian coordinates, after exactly one Earth year, or 31557600 seconds, is displayed in Table A.3. From Table A.3 can be seen that there is still a small difference in the state after one Earth year. This small difference is accounted to the used precision of the gravitational parameter and number π. Hence, the implemented dynamical models can be declared verified. Furthermore, this work uses propagation of the state in MEE, such

Table A.3: The difference in Cartesian state between the initial state and the final state after propagating in two different coordinate sets for exactly 31557600 s.

Propagation element set	δx [m]	δy [m]	δz [m]	$\delta \mathrm{v}_x$ [m/s]	$\delta \mathrm{v}_y$ [m/s]	$\delta \mathrm{v}_z$ [m/s]
Cartesian coordinates	0.0249	-0.0148	0.0000	3.7999×10^{-9}	2.2010×10^{-9}	1.0582×10^{-9}
MEE	12.8484	-9.7567	2.3657	2.5450×10^{-6}	1.9294×10^{-6}	-8.2861×10^{-8}

that this has to be verified as well. The initial state of Table A.2 is transformed to MEE and forward propagated using the methods explained in the journal article (Chapter 2). Subsequently, the final state in MEE is transformed back to Cartesian coordinates. The difference of the final state with respect to the initial state is displayed in Table A.3 as well. When the difference in the state, which is of order $[\delta x, \delta y, \delta z] \sim 10^1$ m, is compared to the total state, which is of order $[x, y, z] \sim 10^{11}$ m, it can be stated that the propagation using the dynamical framework is verified. Note that this difference is larger than the difference found from the propagation with Cartesian coordinates. For the propagation in MEE, only 200 steps were taken, whereas for the propagation in Cartesian coordinates a step size of one second was used. Less steps are required to describe the trajectory in MEE with similar accuracy as in Cartesian coordinates [54]. For longer propagation, a smaller step size is needed to describe the trajectory in Cartesian coordinates, since there are six fast-changing elements and the state at one moment in time does not contain any information about the future or the past [46]. Because the MEE set consists of five slowly-changing parameters, in contrast to the fast-changing Cartesian coordinates, the set becomes favourable to enhance the numerical performance of the optimization algorithm [55]. Thus, for the purpose of this work the propagation of the dynamical framework in MEE is verified and validated.

A.3. Optimization

The implemented optimal control algorithm that is the focus of this work, namely DDP, needs to be verified and validated as well. This seems tricky, since the *goal* of this work is to verify and validate DDP for interplanetary solar-sail trajectories. The validation of DDP is given by the outcome of this work, e.g., the comparison to the locally optimal steering laws and the comparison of the optimization of an Earth-Mars trajectory with similar problems in literature in Chapter 2. However, before the implemented version of DDP is used accurately to generate solutions for this work, it needs to be verified. Firstly, the verification of the use of the dynamical framework for DDP can be performed by substituting a solution for the control law found by DDP in a MATLAB® integrator such as `ode45`, and comparing the final states with one another. This part of the verification is displayed in Section A.3.1. In addition, the implementation of DDP algorithm in this work can be verified by comparing its outcome to results from Reference [32], which applied DDP successfully to many-revolution Earth-centered orbits (see Section A.3.2).

A.3.1. Verification of DDP using the dynamical framework

To verify the use of the dynamical framework for the implemented DDP algorithm, the results of DDP were verified with MATLAB®. The optimization started from the initial state given in Table A.4, and the semi-major axis was maximized for three orbital revolutions. The only accelerations acting on the spacecraft are the gravitational acceleration of the Sun and the solar-sail acceleration, assuming an ideal solar-sail force model and a lightness number of $\beta = 0.01$. An initial guess was used that assumed a constant cone and clock angle of zero throughout the trajectory.

Table A.4: Initial state in Keplerian elements.

Kepler element	a [AU]	e	i [deg]	ω [deg]	Ω [deg]	ν [deg]
Value	1.25	0.2	0	0	0	0

The optimization resulted in an optimal trajectory and an optimal control law. The latter was substituted, together with the initial state, in the MATLAB® integrator `ode45`, which resulted in a new trajectory.

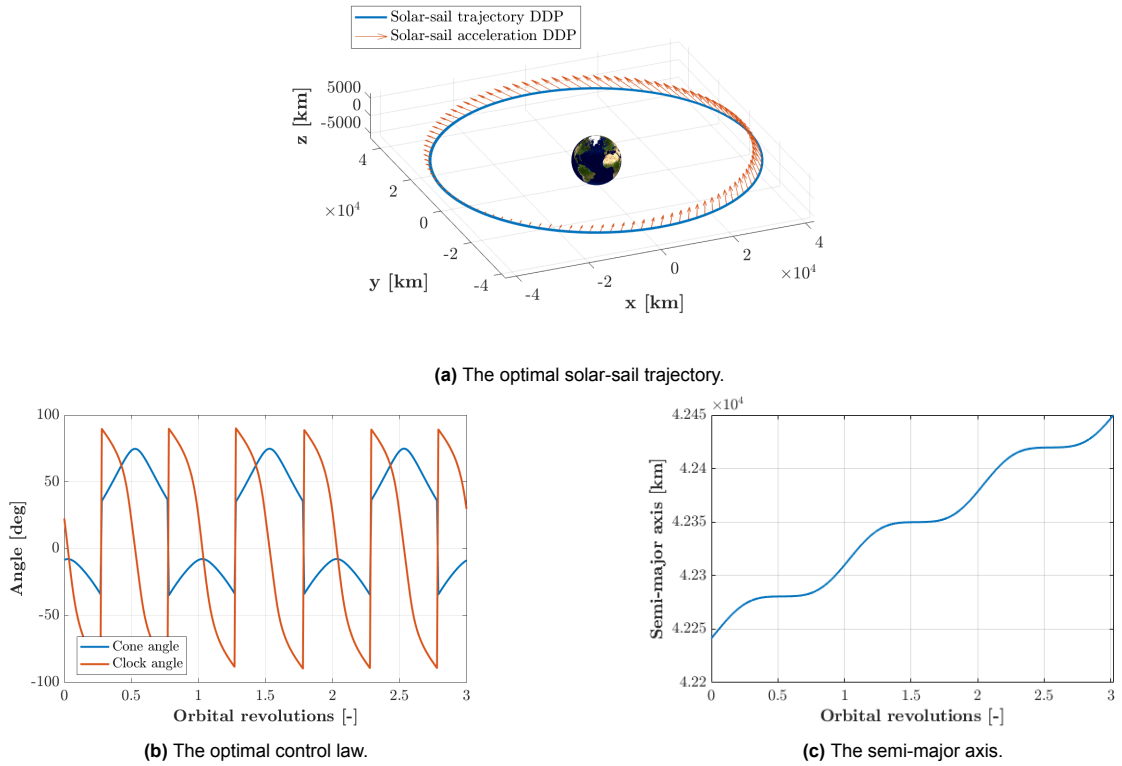
Table A.5: The absolute difference in the state between the optimization results and the MATLAB integration (using these results) after three orbital revolutions.

δx [km]	δy [km]	δz [km]	δv_x [m/s]	δv_y [m/s]	δv_z [m/s]
3.3834×10^{1}	6.4791×10^{3}	1.1106×10^{-5}	8.9409	0.9807	5.0607×10^{-8}

The differences in the final Cartesian state of this new trajectory with the final state of optimal trajectory found by DDP are given in Table A.5. From Table A.5 can be seen that the difference between the Cartesian states is in the order of hundreds of kilometers. This seems like a large difference, however, certain aspects shed more light on this. First of all, the DDP optimization is completely propagated in MEE, whereas the MATLAB integrator used Cartesian elements. Secondly, note that a large amount of time has passed (approximately 4.8 years) and errors in long-duration optimization or integration accumulate over time. As a final remark must be noted that still, the order of hundreds of kilometers is relatively small with regards to the scale of the problem. Consequently, the use and implementation of the dynamical framework in DDP is verified.

A.3.2. Verification with Earth-centered optimization

The DDP implementation is further verified by comparing its results with the results from Reference [32]. The implementation of DDP was expanded with the possibility of setting heliocentric or Earth-centered dynamics at the beginning of the optimization. The optimization from Reference [32] that was simulated, was the three-dimensional geostationary orbit for three orbital revolutions. A lightness number of $\beta = 0.011$ is used and the only accelerations assumed to act on the spacecraft are the gravitational acceleration of the Earth and the solar-sail acceleration (ideal force model). Similar as in Reference [32], constant initial guesses of zero for both control angles are used. The resulting optimal control law, trajectory and semi-major axis from DDP's maximization of the specific orbital energy are shown in Figure A.4. When comparing the results from Figure A.4 to the results in Reference [32], a



(a) The optimal solar-sail trajectory.

(b) The optimal control law.

(c) The semi-major axis.

Figure A.4: Results of DDP's optimization of the specific orbital energy (solid lines) for three orbital revolutions, in comparison to a locally optimal steering law (dashed lines) [1].

small difference in control law is observed. The precise algorithmic settings that generated the plots in Reference [32] are not known and could account for the difference, or the definition of the objective could be stated differently. All in all, the final increase in semi-major axis after three orbital revolutions found by DDP's optimization was $\delta a = 208.9674$ km, which is actually approximately 100 m more than found in Reference [32]. Since the plots and final change in semi-major axis are very similar, the implementation of DDP in this work is considered valid.