Multi-Objective Hybrid Optimal Control with Application to Space Systems

Lorenzo Angelo Ricciardi

Submitted in fulfilment of the requirements

for the degree of Doctor of Philosophy

Aerospace Centre of Excellence
Mechanical and Aerospace Engineering
University of Strathclyde, Glasgow

May 30, 2019

Abstract

This dissertation presents a global method to solve multi-objective hybrid optimal control problems. The method is applicable to general problems but the emphasis here is on space systems and missions. This holistic framework combines three main building blocks: a memetic multi-objective optimisation algorithm, a transcription method to solve general optimal control problems, and the treatment of mixed integer problems. The framework is able to automatically produce a well spread set of Pareto optimal solutions, each of which can consist of an optimal open loop guidance law and system design parameters, which can include the set and order of targets or operations that compose the structure of a mission. The framework was employed to perform the multi-objective trajectory and system design of reusable launch vehicles, including the sizing the engines, structural masses, fuel tanks and wings, and to design a multiple target debris removal space mission in which the trajectory of the spacecraft and the sequence in which the targets are visited had to be simultaneously optimised.

Acknowledgements

I'm profoundly grateful to my supervisor, Prof. Massimiliano Vasile, for trusting me with this unique opportunity, providing invaluable guidance and sharing his exceptional expertise.

I'd like to thank Dr. Edmondo Minisci and Dr. Christie Maddock, for the interesting opportunities to collaborate and test my work on several interesting cases.

My gratitude goes also to the European Space Agency and Airbus Defence and Space, whose sponsorship funded part of my PhD, and in particular to Celia Yabar, Technical Officier and my point of contact within ESA.

A huge hug goes to my family: my mum Elda, my dad Nicola, my brother Marcello, my auntie Daniela, my cousins Roberta and Elena, my beloved granny Aleide, my cousin in law Luca, Francesca, and the new arrived Emma. Thanks for being so close to me, for your support, trust and persistent encouragement.

To the friends of a lifetime in Italy: Andrea D., Andrea M., Arianna, Daniele, Fabrizio, Gabriele, Luca, Matteo, Roberta and Viviana.

To the colleagues and friends I made in Glasgow: Alessandro, Andrew, Annalisa, Carlos, Christian, Dario, Eleonora, Francesco, Gianluca, Giulio, Juan, Marilena, Nicole, Pyiush and Romain.

To all the members of the Strath++ team, with whom I participated in that crazy adventure of the GTOC.

And to Laure, whose smile is a bright warm sun even in the rainiest Scottish day.

Contents

List of Figures **vi**

List of Tables **x**

List of Algorithms **xiii**

I Theory and algorithms **1**

1 Introduction **2**

2 Direct transcription of Optimal Control problems **25**

2.1 The optimal control problem . . . 26

2.2 Direct Transcription with Finite Elements in Time . . . 27

2.2.1 Problem Transcription with DFET . . . 28

2.2.2 Choice of Basis Functions . . . 30

2.2.3 Bernstein Basis . . . 34

2.3 Benchmark Cases . . . 40

2.3.1 Problem Instance 1 - Minimum Ascent Time with no Terminal Constraint on the Horizontal Velocity . . . 41

2.3.2 Problem Instance 2 - Minimum Ascent Time with Constrained Vertical Velocity . . . 50

2.3.3 Effect of the NLP algorithm . . . 58

2.3.4 Sparsity and NLP solvers . . . 61

2.4 Chapter summary . . . 62

3 Multi-objective optimisation **65**

3.1 The multi-objective optimisation problem . . . 66

3.1.1 Tchebycheff Scalarisation . . . 67

3.2 Multi Agent Collaborative Search . . . 68

3.2.1 Algorithm Description . . . 68

3.2.2 A New Archiving Strategy . . . 74

3.2.3 Generation of the weight vectors . . . 80

3.3 Benchmark Cases . . . 82

3.3.1 CEC 2009 UF functions . . . 82

3.3.2 ZDT4 and 3 impulse problem . . . 90

3.4 Chapter summary . . . 93

4 Multi-objective Optimal Control **94**

4.1 Solution of the Transcribed Problem . . . 97

4.1.1 Bi-level Global Optimisation Problem . . . 100

4.1.2 Single Level Local Search . . . 102

4.1.3 Generation of the Initial Feasible Population . . . 103

4.1.4 Definition of the descent directions and target points . . . 104

4.2 Benchmark Cases . . . 105

4.2.1 Minimum transfers to rectilinear path . . . 105

4.2.2 Maximum Energy Orbit Rise . . . 109

4.3 Chapter summary . . . 116

5 Multi Objective Hybrid optimal control **117**

5.1 The mixed integer optimal control problem . . . 118

5.2 Solution of the mixed integer optimal control problem . . . 119

5.2.1 Modification of the Heuristics of MACS . . . 120

5.2.2 Relaxation and Integrality constraints . . . 122

5.3 Benchmark Cases . . . 122

5.3.1 A time dependent motorised Travelling Salesman Problem . . . 132

5.4 Chapter summary . . . 137

II Application to Space Systems 138

6 Transatmospheric Vehicle and Mission Design 139

6.1 Physical models 140

6.1.1 Dynamical model 140

6.1.2 Atmospheric and aerodynamic models 140

6.1.3 Propulsion model 141

6.2 Optimal Descent Trajectory 142

6.2.1 Results 145

6.3 Three-objective Ascent Problem 148

6.3.1 Structural mass models 149

6.3.2 Objectives 150

6.3.3 Numerical settings 151

6.4 Optimal Ascent and Abort Scenarios 156

6.5 Chapter summary 165

7 Multiple Debris Removal Mission 166

7.1 Problem formulation 167

7.2 Results 170

7.3 Chapter summary 174

8 Conclusion 176

A Necessary conditions for optimality 184

A.1 Optimal control of time-continuous systems 185

A.2 Pascoletti-Serafini Scalarised MOCP 188

A.2.1 Convergence of the Transcribed Problem 191

B Computational complexity of the Energy Based Archiving Algorithm195

C Aerodynamic model for Chapter 6 198

Bibliography 198

List of Figures

2.1 A Bernstein basis and a Bezier curve. Circles: Bezier nodes. Dashed line: polygonal chain. Shaded area: convex hull. 37

2.2 Problem Instance 1, Test case 1: Time-history of the controls. 44

2.3 Problem Instance 1, Test case 2: Time-history of the controls. 47

2.4 Problem Instance 1, Test case 3: Time-history of the controls. 49

2.5 Problem Instance 2, Test case 1: Time-history of the controls. 53

2.6 Problem Instance 2, Test case 1: Time-history of v_y. 54

2.7 Problem Instance 2, Test case 1: Time-history of the controls. 56

2.8 Problem Instance 2, Test case 2: Time-history of v_y. 57

3.1 Inertia move with reduction of α. Black dots and black arrows indicate initial positions and $\mathbf{V}$, blue dot indicate final positions, blue arrow indicate modified $\mathbf{V}$ 71

3.2 Diagram for the Energy based archiving strategy 78

3.3 Outcomes of different archiving strategies from the same initial archive. The fronts have been shifted to enhance the comparison 80

3.4 Distribution of 106 Weight vectors 81

3.5 Distribution of 465 Weight vectors 82

3.6 Contribution of the different algorithms to the global Pareto front . . . 92

4.1 Schematic representation of the bilevel approach acting on a single solution. 101

4.2 Pareto Front, trajectories, controls and vertical component of velocities for a single run of the minimum transfers to rectilinear path problem . . 108
4.3 Comparison between solutions of the multi-objective problem vs the single objective problem 111
4.4 Comparison between the controls obtained with the present approach vs the analytic solution 112
4.5 Pareto front, trajectories, controls and radial velocities for one single run of the orbit rise problem 113
4.6 Comparison between controls of the multi-objective problem vs the single objective problem 114
4.7 Comparison between solutions of the multi-objective problem vs the single objective problem 115

5.1 Graphic representation of constraints 5.6 126
5.2 Feasible region of constraints 5.12 and 5.13 128
5.3 Pareto front and Trajectories for the dynamic Travelling Salesman Problem 129
5.4 Time histories of velocity and acceleration for the Travelling Salesman Problem. + marks when a target is visited 130
5.5 Comparison of the time histories between the reference solution and the minimum time solution computed by MACS. + marks when a target is visited 131
5.6 Trajectories of the targets for the time dependent motorised Travelling Salesmen Problem. Hollow circles indicate the position of the targets at $t = 0$ 133
5.7 Pareto front and Trajectories for the time dependent motorised Travelling Salesman Problem 133
5.8 Different snapshots of the trajectories for the time dependent motorised Travelling Salesman Problem 135
5.9 Time histories of velocity and acceleration for the time dependent Travelling Salesman Problem. + marks when a target is visited 136

6.1 Optimal descent trajectory: Pareto optimal solutions stored in the archive $\mathcal{A}$ at the end of the optimisation process and the two published single objective solutions from Betts [69]. 144
6.2 Optimal descent trajectory: time-history of a) the altitude and b) the velocity for each of the 10 solutions in the Pareto front in Fig. 6.1 plus the two published single objective solutions from Betts [69]. 144
6.3 Optimal descent trajectory: time-history of a) the lift to drag ratio and b) the heat flux for each of the 10 solutions in the Pareto front in Fig. 6.1.145
6.4 Optimal descent trajectory: time-history of a) the angle of attack and b) the bank angle for each of the 10 solutions in the Pareto front in Fig. 6.1 plus the two published single objective solutions from Betts [69]. . . 145
6.5 Optimal descent trajectory: time-history of a) the rate of change of the flight path angle $\dot{\gamma}$ and b) the heading angle $\dot{\chi}$ for each of the 10 solutions in the Pareto front in Fig. 6.1. 146
6.6 Three-Objective Ascent: set of Pareto-optimal solutions, colorbar indicates gross mass of the vehicle. The 3D Pareto front is in the middle, with orthographic projections shown on each coordinate plane. 152
6.7 Three-Objective Ascent: time-history of a) the altitude and b) the velocity for the three extreme solutions of the Pareto front in Fig. 6.6. The + indicates the stage separation point. 155
6.8 Three-Objective Ascent: time-history of a) the flight path angle and b) the throttle for the three extreme solutions of the Pareto front in Fig. 6.6. The + indicates the stage separation point. 155
6.9 Data points, black dots, and non-linear fit surfaces of the aerodynamic coefficients: a) C_L as a function of M and α and b) C_D as a function of M and α. 158
6.10 Abort at 0 seconds: a) Pareto front and b) wing loading vs downrange. 162
6.11 Abort at 0 seconds: a) altitude vs time and b) altitude vs downrange for ascent (solid lines) and descent (dashed lines). 162

6.12 Abort at 0 seconds: time histories for a) total acceleration and b) engine throttling for ascent (solid lines) and descent (dashed lines). 163

6.13 Abort at 0 seconds: time histories of a) flight path angle and b) heat flux for ascent (solid lines) and descent (dashed lines). 163

6.14 Comparison of Pareto optimal solutions for different abort times: a) downrange versus initial mass, b) wing loading for each solution ID. . . 164

7.1 Positions of the targets and initial position of the spacecraft. Starting position lays inside the circle. 169

7.2 Pareto front and Trajectories for the multiple debris removal mission . . 171

7.3 Time histories of velocity and acceleration for the time dependent Travelling Salesman Problem 172

7.4 Pareto front and Trajectories for the multiple debris removal mission, refined mesh 173

7.5 Time history of the mass with refined mesh 173

B.1 Statistical evaluation of the number of iterations n_{it} as a function of the number of candidates q 197

List of Tables

2.1 Results for Problem Instance 1, Test case 1 45
2.2 Results for Problem Instance 1, Test case 2 46
2.3 Results for Problem Instance 1, Test case 3 48
2.4 Results for Problem Instance 2, Test case 1 52
2.5 Results for Problem Instance 2, Test case 2 58
2.6 Performance comparison of the NLP algorithms for Problem Instance 1, Test case 1 59
2.7 Performance comparison of the NLP algorithms for Problem Instance 1, Test case 2 59
2.8 Performance comparison of the NLP algorithms for Problem Instance 1, Test case 3 59
2.9 Performance comparison of the NLP algorithms for Problem Instance 2, Test case 1 60
2.10 Performance comparison of the NLP algorithms for Problem Instance 2, Test case 2 60

3.1 IGD of the different archiving strategies 79
3.2 Settings for MACS, CEC problems 83
3.3 Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 and MOEA/D, CEC2009 problems. Also reported the unsigned Wilcoxon test results 83

3.4 Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 with EBA archiving vs MACS2.1 without EBA archiving on the CEC2009 problems. Also reported the Wilcoxon test result 85
3.5 Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MOEA/D with EBA archiving and standard MOEA/D, CEC2009 problems. Also reported the Wilcoxon test result 85
3.6 Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 vs. MACS2 on the CEC2009 problems. Also reported the Wilcoxon test result 88
3.7 Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 with EBA archiving and dynamic setting of maximum number of coordinates for pattern search vs MACS2.1 with EBA archiving and static setting of maximum number of coordinates for pattern search on the CEC 2009 problems. Also reported the Wilcoxon test result 88
3.8 Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 vs MACS2.1 with old social moves on the CEC2009 problems. Also reported the Wilcoxon test result 89
3.9 Success rates for the IGD of the CEC2009 functions for all the tested algorithms and their variants 89
3.10 Settings of MACS2.1 on the 3 impulse and ZDT4 problems (in brackets) 91
3.11 Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 vs MACS2 on zdt4 and triple impulse problems. Also reported the Wilcoxon test result 91
3.12 ZDT4 and triple impulse success rates for MACS2 and MACS2.1 and the various metrics 92

4.1 Lower and upper bounds for the state, control and final time variables of the Minimum transfers to rectilinear path 106
4.2 MACSoc settings 107

4.3 *fmincon* settings 107
4.4 Convergence and spreading statistics for the two problems 108
4.5 Lower and upper bounds for the state, control and final time variables of the Maximum Energy Orbit Rise problem 110

5.1 Lower and upper bounds for the state, control and final time variables of the motorised Travelling Salesman Problem 129
5.2 Comparison of intermediate and final times between the reference solution and solution 10 on the Pareto front 131
5.3 Comparison of minimum time and minimum energy solution features between fixed and time dependent motorised Travelling Salesman Problem 134

6.1 Lower bounds, upper bounds and boundary conditions for the Optimal Descent Trajectory case 142
6.2 Lower bounds, upper bounds and boundary conditions for the Three Objective Ascent case 149
6.3 Design parameters for the three extreme cases of the Three-Objective Ascent case 153
6.4 Mass breakdown for the three extreme cases of the Three-Objective Ascent case 153
6.5 Lower bounds, upper bounds and boundary conditions for the Optimal Ascent and Abort Scenarios case 157

7.1 Lower and upper bounds for the state and control variables of the multi debris removal mission 170

C.1 Coefficients for the aerodynamic model 198

List of Algorithms

1 MACS2.1 69
2 Individualistic actions 73
3 Social actions 74
4 Energy Based Archiving 77
5 MACS optimal control (MACSoc) framework 98

Part I

Theory and algorithms

Chapter 1

Introduction

Aut inveniet viam, aut faciet
I shall either find a way or make one

Seneca

The demand for more reliable, efficient, economical and higher performance products or services is constantly growing. In order to achieve these goals it is necessary to optimise the design of the product, or how it is manufactured and distributed, or how it is operated. Postponing a more rigorous definition, the term optimisation can be intuitively explained as the process of finding the set of parameters or decisions that lead to the best possible outcome for a given problem or situation. The field of mathematical optimisation has grown leaps and bounds, both from the theoretical point of view since the work of Euler and Lagrange in the 1750s, and from the applied point of view since the early World War II approaches of Operational Research [1].

The exponential growth of the computational capabilities of computers, as shown by Moore's law [2], and the advent of parallel computing have further contributed to the growth of the field of applied optimisation and of simulations. Unsurprisingly, in parallel with the development of the design tools, also the approach to the design process itself evolved.

Historically, design was approached as a sequential and compartimentalised process that started from an initial point dictated by experience and requirements. Each discipline expert received input from the previous expert and produced output for the next one until the full product was designed. In case that some requirements were not satisfied or the performances were deemed insufficient, the whole process was iterated, with the hope of eventually converging to a satisfactory design. An example of this is the so called waterfall method [3, 4], one of the very first approaches to software design.

For very complex projects, composed of many interconnected functions and subsystems, it was realised that the system's performance as a whole might not meet the specifications despite all the components meeting their individual requirements and operating correctly. As a consequence a new interdisciplinary and integrated approach called Systems Engineering emerged around the 1940s and 1950s [5]. Instead of considering the system as a sum of parts, this approach treats it as a holistic entity and attempts to design it as such, at least in the preliminary design phases. The convergence of Systems Engineering and applied optimisation gave birth in the 1970s to an optimisation based approach to the design called Multidisciplinary Design Optimisation [6].

In the spirit of including all subsystems, if the system has some kind of at least partially controllable dynamics, it should be very useful to consider in the design process the way the system is controlled and steered. Since its dynamical evolution depends both on how the system is operated and how it is designed, the formulation of the relevant optimisation problem should include all the design parameters that indirectly affect the dynamics of the system, and all the mutual constraints that the dynamics imposes on the subsystem (and vice versa). An optimal solution of that problem would thus consist of the design choices describing the various subsystems and a control policy dictating how the system should be steered or operated.

Mathematically, the problem of how to steer a general dynamical system from one initial state to a desired final state in some optimal sense can be approached by formulating and solving an optimal control problem. Very interesting historical surveys on the birth of this discipline are given by Bryson [7], Pesch et al. [8] and Sussmann and Willems [9]. The optimal control problem formulation can also include the design parameters that indirectly affect the dynamics of the system, thus the solution of the problem will return both time laws for the states of the systems and its control parameters, and all the other so called static parameters such as the size of the wings of an airplane. However, as explained in the historical surveys, unless the problem is so simple to have only an academic relevance, the only possibility of solving optimal control problems governed by nonlinear dynamics is by employing numerical techniques. As explained by Bryson [7], the most commonly employed methods to solve these problems are gradient based, requiring the differentiability of all the functions involved in the optimisation. These optimisation methods are very powerful, converge quickly to a locally optimal solution (provided that a good initial guess was given) and also very importantly have a proof of convergence towards the local minimum [10].

However, if the dynamics, the constraints or the objectives of this optimisation problem are non-convex, there may exist multiple locally optimal solutions. In some cases it might be important to look for the globally optimal solution, or at least the best solution possible in a given time. In order to achieve this it is necessary to employ global optimisation techniques, that perform an exploration of the whole search space in order not to be trapped by the first seemingly good solution. Another reason to employ these methods is that sometimes it may be difficult even for an expert to generate a good initial guess. This is especially true if the problem is so new that no expert can actually provide good initial guesses. Global methods can then be used to algorithmically find good initial guess, potentially saving a lot of man hours and sometimes finding solutions no expert thought of [11]. Finally, many if not most of the global optimisation methods are derivative free, thus do not require any of the involved functions to be differentiable. On the flip side, global methods usually converge slower than local methods [12] and many of those that have proofs of convergence [13] have no natural stopping criterion.

Another important aspect to consider is that most optimisation problems, and normally all optimal control problems, are formulated as having only one objective, meaning that there is only one performance criterion that needs to be optimised [7, 14]. This is in stark contrast with real world design needs, where there may be multiple and often conflicting performance or cost criteria that should be optimised, the most common one being the "economic cost vs performance" trade-off. The simplest method to deal with this problem is to embed all performance metrics into a single synthetic performance criterion and optimise that. This in turn requires to assign a weight to each performance criteria, indicating the designer's opinion about the relative importance of the various goals. Albeit this approach is simple and intuitive, more powerful methods have been developed in the field of multi-objective optimisation [15].

When performing multi-objective optimisation, the interest is in finding not one optimal solution, but a set of best-compromise trade-off solutions. This approach has the advantage that by returning multiple solutions, each with a different balance of the objectives spanning all the trade-off space, the decision maker has a much better idea of the mutual influence of the various objectives and can thus make a more informed decision. Multi-objective approaches can also be employed when it seems difficult to satisfy some constraint: in those cases, it is possible to relax the constraint and turn it into an additional objective [16]. The resulting trade-off curve will tell which level of performances are compatible for each level of constraint satisfaction, without having to specify the constraint level a priori.

A final important issue considered in this work is whether the parameters to be optimised are continuous or integer. The field of optimisation is extremely vast and several approaches are possible for problems that are described only by continuous or only by discrete variables. However, problems with the simultaneous presence of integer and continuous variables, also known as mixed-integer or hybrid problems, are abundant in practice. Mixed-integer problems are significantly more difficult to treat than purely continuous or purely discrete problems, and specific techniques are required in order to approach them [17, 18].

Motivation and objectives

The considerations just exposed suggest the need for a general methodology with the following characteristics:

- general, i.e. not restricted to a particular industrial sector or dynamic model
- holistic, i.e. consider all the relevant subsystems and their interactions
- automatic, i.e. not require constant supervision or input from the designers
- able to perform a global exploration of the design space
- able to generate several different designs in one run, each striking a different balance between the high level goals
- able to return optimised designs, possibly giving some kind of guarantee that no further improvement is possible in the neighbourhood of the current solution

The aim of this work is to create an organic framework answering to all the aforementioned points. A few attempts have been proposed in the past to deal with some combination of the above mentioned aspects, like approaches to perform multi-objective optimal control, mixed integer nonlinear optimisation, hybrid optimal control or even multi-objective hybrid optimal control. These attempts, their limitations and differences with the one proposed here will be highlighted in the next subsections.

Instead of selecting a handful of existing algorithms and wrapping them together in some (often problem specific) fashion, the problem will be first analysed in its fundamental aspects: optimal control, global multi-objective optimisation and treatment of mixed integer problem. Each of these aspects, and their interactions, will be analysed from a theoretical point of view and some new theoretical results will be presented. Algorithmic approaches will be proposed for each of those aspects, with particular emphasis on how they will interface to each other. Each algorithm will be tested on known benchmark problems to ensure it is behaving as expected. Finally, the overall method will be demonstrated on some more realistic test cases.

The method developed in this thesis is applicable not only to any engineering sector, but also to the financial sector and basically any field for which mathematical models are available. However, the space and aerospace engineering sectors deserve a special mention here. In aerospace vehicles and missions the interactions between the various subsystems and components can be particularly complex. For instance, the choice of a thermal protection system influences the trajectory that a re-entry vehicle can follow both directly, excluding the ones that produce unsustainable heat loads, and indirectly by changing the mass of the vehicle. The minimisation of the cost or mass of the Thermal Protection System might require trajectories that are impossible to follow given the aerodynamic configuration of the vehicle, or, in better circumstances, difficult to ensure given the limited effectiveness of the control surfaces at high altitudes. On the opposite end of the spectrum, a high performance Thermal Protection System could be very expensive or very heavy, thus limiting the economical viability of the project or influencing the steering capabilities of the vehicle. The final choice obviously requires a careful trade-off study that considers both the dynamics and control of the vehicle and the mass, type and cost of the thermal protection system. This trade off study can be conducted by solving a Multi-Objective Hybrid Optimal Control problem. Several of the applications and examples presented in this dissertation will thus be for space and aerospace problems.

Optimal Control

Optimal control problems appear almost everywhere in modern engineering: not only in aeronautical and space vehicle trajectory optimisation, but also in chemical reactions engineering, drug dosage delivery and robotics, just to name a few. Even if the theory of optimal control is very mature, obtaining a solution for these problems is still challenging, especially for highly non-linear problems where the only possibility is to employ numerical techniques.

It is customary to classify numerical methods for the solution of open loop optimal control problems in indirect and direct methods. Indirect methods analytically derive the necessary conditions for local optimality [19] and produce a Two Points Boundary Value (TPBV) problem, a coupled set of Differential-Algebraic Equations (DAE) for the states and co-states of the system. The conditions for optimality have to be derived through analytic manipulations, and every time the objective function, the boundary conditions or the dynamics of the system change, this operation has to be repeated from scratch.

The main advantage of indirect methods are their high accuracy and the fact that, for simple enough problems, it is possible to derive closed form analytic solutions. More often, numerical methods have to be employed and an initial guess must be provided. A good initial guess for the states is usually not particularly difficult to provide, but the same does not hold for the co-states, whose physical interpretation might not be obvious. Moreover the switching structure must be provided as part of the guess, and sometimes this information is the most difficult to provide. This also means that the imposition of inequality path constraints can be challenging, since it is necessary to explicitly guess where the constraints are active or not. Finally, the optimality conditions are only first order local conditions. For these reasons, indirect methods were not considered for this work.

Direct methods for the solution of optimal control problems transcribe the original infinite dimensional problem into a finite dimensional problem, and directly solve the resulting Non Linear Programming (NLP) problem. These methods do not need any co-state and thus do not need to supply any guess for them. The switching structure of the solution emerges automatically from the solution itself. Furthermore, the transcription process is automatic, thus a change in the objective function, boundary conditions or dynamic models simply requires the user to re-run the transcription algorithm without any manual intervention. Finally the imposition of path constraints is straightforward, and several high quality open-source and commercial NLP solvers are available.

Among the many different direct methods, the most famous and commonly employed are the single or multiple shooting method and the orthogonal or pseudospectral collocation method. The transcription process generally makes use of an appropriate set of polynomials to represent states, controls or both.

Shooting methods [14] usually only approximate the controls, and integrate them through an explicit integrator like the various Runge-Kutta methods. For this reason, they tend to have a reduced number of variables and result in small but dense problems. The accuracy of the integration depends on the order of the integration method and on the size of the time step. Since, once chosen, the same integrator is employed for all shooting segments, a highly refined solution is usually produced by using very small time steps or with the careful use of adaptive step size methods instead of changing the integrator to a higher order one.

These methods can have some difficulties if there are equality path constraints, or in the equivalent case where the dynamics are described as DAEs, since the values of the intermediate stages between the initial and final condition of each shooting segment are not explicitly represented and are thus invisible to the NLP solver.

Collocation methods [20] can approximate both the states and the controls as high order polynomials. This results in a comparably larger number of variables than shooting methods, but the resulting problem has higher sparsity. Since all the nodal values for the states (and eventually the controls) are explicitly represented, the NLP solver can directly satisfy the path constraints without any need to reformulate the original problem. If the solution is smooth, pseudospectral methods have the very important property of converging exponentially fast to it, making them extremely efficient in this sense. However, if the solution contains very sharp changes, for instance due to bang-bang controls, this property no longer works because collocation methods rely on polynomial interpolation, which is known to produce spurious oscillations in those cases (Gibbs phenomenon). To obtain a better solution when bang-bang controls are present it is necessary to reformulate the problem as a multi-phase problem and optimise the switching times between one phase and the next, using the previous solution as a first guess.

In this thesis, the Direct Finite Elements in Time (DFET) transcription method will be employed, which will be described in detail in Chapter 2. The basic idea is to discretise the time domain in several elements, and represent on each element both the states and controls as polynomials. However, the differential equations are not directly satisfied on the nodes like in the various collocation methods. Instead, they are first recast in weak form. This important step ensures that the resulting formulation is mathematically correct even in the presence of discontinuities for the dynamics within an element.

DFET can enjoy the high accuracy due to high order polynomials but, as will be shown, it is also very flexible and allows to independently choose the order for each state and control polynomial on each element. Moreover, and more importantly, it will be shown that the choice of a polynomial basis different from the usual Lagrange interpolation on Gauss-Legendre or Gauss-Lobatto nodes will allow to have a smooth and monotonic approximation for the controls even in case of a bang-bang control. This fact culminates in a a Theorem, which is one of the main contributions of this thesis, dictating that with the DFET method using the proposed basis and under mild hypotheses on the inequality constraints, the trajectory will satisfy the inequality constraints at every instant of time even if it is computed only on a discrete set of points.

It is important to remark that the overall framework developed in this work does not need the DFET transcription for optimal control: in principle, any direct method should be applicable without any major alteration to the rest of the framework. However, the robustness and flexibility of the DFET transcription are very useful characteristics in the context of this work.

Multi-objective Optimisation

The need to evaluate several different solutions and compare their relative performances leads to the formulation of multi-objective optimisation problems. Along with all the complexities associated with single objective optimisation, like the presence of non-linear objective functions and constraints, points of non differentiability and existence of multiple local solutions, multi-objective optimisation problems are characterised by the fact that their solution is a set, potentially composed by an infinite number of elements. Since only a finite number of design alternatives can be evaluated, methods to solve multi-objective optimisation problems try to find a good finite approximation of the Pareto set.

There are fundamentally three different approaches to solve multi-objective optimisation problems: scalarisation based approaches, dominance based approaches and indicator based approaches. The first are conceptually very simple: they reduce the initial multi-objective problem into a series of single objective problems through a scalarisation function and a set of weights, which express the relative importance given to each objective. The second try to solve directly the multi-objective optimisation problem using the dominance criterion, which will be described in Chapter 3. The third try to maximise some metric of the approximated Pareto front.

Scalarisation based approaches only return one solution for each particular choice of the weight vector. A limitation of this approach is that it is difficult to know, a priori, how to change the weight vectors to obtain a good spreading of the solutions on the Pareto front: a uniform spread of the weights might not result in a uniform spread of solutions, and some particular weight vectors might point towards areas where no solution is to be found. A lot of research is currently being performed on how to automatically adapt the weight vectors to overcome this limitation [21, 22, 23].

One advantage of scalarisation based approaches is that they enforce the search along a particular direction, and are thus more efficient than dominance based methods for many objective problems. Another advantage of these methods is that, for smooth enough problems, the standard NLP algorithms can be employed and local optimality can be guaranteed. This is the approach followed, for example, by the Normal Boundary Intersection method [24] and by the the Normalized Normal Constraints Method [25].

Dominance based methods like NSGA-II [26], MOPSO [27] and multiMADS [28] have the advantage that their performance is not negatively influenced by a poor choice of weight vectors, since they do not use any. Since the dominance relation is not differentiable and requires the knowledge of all the other points composing the Pareto front, dominance based methods are usually not employed in association with an NLP solver.

Indicator based approaches have an intermediate behaviour between the scalarisation and the dominance based methods, since they don't require any definition of weight vectors but they internally try to optimise a scalar quantity: the chosen metric of the Pareto front. However, most metrics require a reference Pareto set, which is unknown for non academic problems. The only Pareto compliant indicator that does not require knowledge of the Pareto front is the hypervolume indicator.

While very interesting methods have been proposed using the hypervolume, like SMS-EMOEA [29] and HypE [30], it was later found that the metric directly influences the distribution of points found by those methods, favouring the central or convex regions[31]. The other drawback of the hypervolume indicator is that it is quite expensive to compute, and in addition its computation is subject to the curse of dimensionality, which means that the cost to compute it grows very quickly as the number of objectives grow. There is significant research focused on methods to approximate the hypervolume or speed up its computation [32].

This thesis will employ Multi Agent Collaborative Search (MACS), a population based memetic multi-objective optimisation algorithm which will be thoroughly described in Chapter 3. This algorithm has already been succesfully employed for the solution of complex space related multi-objective optimisation problems [33, 34, 35, 36].

One important aspect of this algorithm is that MACS employs both the dominance criterion and a scalarisation method. In addition, it is a memetic, or hybrid algorithm: new heuristics can be introduced in the algorithm without altering its overall structure. These two aspects will be synergistically exploited in Chapter 4 to introduce a gradient based refinement phase that occurs at a given frequency during the optimisation phase. Thanks to this gradient based refinement it is possible to guarantee the local optimality of the solutions. Finally, thanks to a newly developed archiving strategy which will be described in Chapter 3, MACS returns a very well spread set of solutions on the Pareto front.

Multi-objective Optimal Control

The presence of trade-offs in the design of a system does not depend only on the choice of the subsystems and the static parameters that describe them, like their mass, but also on how the vehicle is steered. The same system can operate in different ways, one for example promoting lower energy consumption and another taking less time to perform the same task.

The simplest approach to deal with multiple objectives in optimal control is to perform a weighted sum of all of them. The Bolza problem, which is the classic optimal control problem that sums a function depending on the boundary states with the integral of another function along the trajectory, is a clear example of weighted sum approach.

In multi-objective optimisation, however, it is well known that this kind of approach has several limitations, the most important of which is that weighted sum approaches cannot converge to non convex areas of the Pareto front [37]. Moreover, it is not a priori clear how a slight change of the weights influences the position of the solution along the Pareto front, especially if the objectives have significantly different scales. For these reasons, in this work the optimal control problems will be formulated as true multi-objective problems.

While conditions of optimality and related theoretical aspects of multi-objective optimal control problems have been studied in a number of papers, see [38, 39, 40, 41, 42] and references therein, less research has been dedicated to the solution of multi-objective optimal control problems.

Ober-Blöbaum et al. [43] coupled a direct transcription approach with an approach that scalarised the multi-objective vector along directions pointing at predefined unreachable points in the criteria space. Each scalar problem was then solved with a standard NLP solver. This approach was employed to solve an interplanetary trajectory optimisation problem. Kaya and Maurer [37] proposed a similar approach but used the Pascoletti-Serafini scalarisation [44] to transform the multi-objective optimisation problem in a set of single-objective optimisation problems, and employed the resulting approach to solve chemical reaction engineering and drug dosage problems. Similarly, Logist et al. [45] combined direct transcription methods with other scalarisation techniques, like the Normalized Normal Constraint method and the Normal Boundary Intersection method.

Pagano and Mooij [46] optimised the mass of the payload for a launch vehicle and minimised the violation of the constraints as a second objective, Bairstow et al. [47] performed a multi-objective optimisation of a two stage launcher minimising cost and maximising the payload and Roshanian et al. [48] performed robust design optimisation of a two stage launch vehicle by means of multi-objective optimisation, minimising both the mean and the variance of the gross take-off mass when several design and operative parameters where subject to uncertainty. In these last three works the control laws had a simple parametric shape. The parameters describing those shapes were optimisation variables, and stochastic multi-objective optimisation algorithms were employed to find the set of Pareto optimal solutions.

Coverstone-Carroll et al. [49] combined Genetic Algorithms and optimal control theory in a dual loop algorithm. In the outer loop, a Multi-Objective Genetic Algorithm (MOGA) was generating vectors of co-states and times of flight. For each set, the inner loop was solving a single objective optimal control problem with given time of flight, minimising the propellant consumption. Englander et al. [50] proposed a dual loop algorithm in which the outer loop solved a multi-objective problem handling a set of categorical variables through a multi-objective genetic algorithm and the inner loop solved a set of single objective constrained optimal control problems using Monotonic Basin Hopping [51].

The method proposed in this work is based on the DFET transcription and the solution of the resulting Multi-Objective Nonlinear Programming (MONLP) problem with a modified version of MACS called MACSoc.

The MONLP problem is reformulated as two different non-linear problems: a bi-level and a single level formulation. The bi-level formulation is used to globally explore the search space and generate a well spread set of non-dominated decision vectors while the single level formulation is used to locally converge to Pareto efficient solutions. Within the bi-level formulation, the outer level selects trial decision vectors that satisfy an improvement condition based on the Tchebycheff weighted norm, while the inner level restores the feasibility of the trial vectors generated by the outer level. The single level refinement implements a Pascoletti-Serafini scalarisation of the MONLP problem to optimise the objectives while satisfying the constraints.

The proposed method differentiates from Ober-Blöbaum et al. [43], Kaya and Maurer [37] and Logist et al. [45] in that it combines global exploration and local convergence with a smooth transition between Chebyshev and Pascoletti-Serafini scalarisation [44] and incorporates an automatic and unsupervised procedure to generate feasible first guesses. It also differentiates from [49, 47, 46, 48, 50] in that it does not use a generic MOGA but proposes a more efficient memetic approach and implements a more general direct transcription method.

Mixed integer non-linear optimisation and optimal control

The design of a system may require several discrete choices, like the type of engine for a space vehicle or which beam profile shape to use for a metallic structure. In such cases, the resulting optimisation problem becomes a mixed integer problem, which is significantly harder to solve than a purely discrete or a purely continuous problems. Even more complex problems arise if the discrete parameters decide the number of phases of an optimal control problem, because it results in a variable size problem. This is typical of multi-gravity-assist interplanetary transfers, where the number of phases depends directly on the number of gravity-assist manoeuvres. Although very important in space mission design, variable size problems will be excluded from the present work and left as a future development.

There is a vast amount of literature on the solution of mixed integer problems[17, 18], but essentially there are two main approaches to solve them. The first class, called relaxation methods, treat discrete variables as if they were continuous and then employ some technique to restore the integer nature of the variables that were relaxed. The other class instead does not relax the discrete variables, thus usually resulting in a separate treatment of the continuous and discrete variables.

Among the deterministic approaches, the most common method of the first class is the Branch and Bound algorithm, introduced by Land et al. in [52], while for the second kind there are the Outer Approximation method [53, 54], the Generalised Benders Decomposition [55], or the use of deterministic discrete optimisation algorithms like the Mesh Adaptive Descent Method (MADS) [56, 57]. Stochastic approaches can either be natively developed for discrete variables or be adapted to treat them by modifying the heuristics they employ, thus do not need any relaxation.

For mixed integer optimal control with general nonlinear dynamics and objetives, in the past have been proposed approaches coupling the Branch and Bound method with standard NLP solvers, like in [58, 59], or coupling stochastic algorithms and standard NLP solvers like in [50, 60].

In the Branch and Bound based methods, the problem is initially relaxed and solved as purely continuous. This solution is used as the lower bound for the optimal solution of the problem, while a feasible solution for the non relaxed problem is used as an upper bound. In case all the relaxed variables assumed integer values, the optimal solution would have been found. However, this will not happen in most of the cases, so one discrete variable will be branched, meaning that two different problems will be solved by imposing its value to be either 0 or 1, and leaving the other variables relaxed. Two new NLPs will be solved, which will update the lower bound and the upper bound for the solution. In case the lower bound of one branch is higher than the upper bound for the other, that branch cannot contain the optimal solution and is thus not further investigated.

This way, a finite, albeit potentially very large number of steps is produced that converges to the optimal solution. However, as shown in [58], it must be highlighted that for non convex problems it can happen that for the same values of the discrete variables there exist multiple locally optimal solutions with different values for the objective function. This can introduce the possibility that the branching procedure erroneously discards promising areas of the search space.

In the already described method proposed by Englander [50] the discrete variables remained discrete and were handled at an outer level by NSGA-II, an evolutionary based multi-objective optimiser. The discrete variables were passed as constants to the inner level NLP solver, which tried to produce feasible and locally optimal solutions.

The resulting optimal control problem was optimising only one objective (the mass of the vehicle), thus the local optimality with respect to the other objectives was not guaranteed. In addition, while a global search was performed for the discrete parameters, the continuous parameters were treated only locally by the NLP solver. As for the Branch and Bound based approaches, this can result in multiple local solutions for a given choice of the discrete parameters. This problem was partially addressed by using a Monotonic Basin Hopping method (MBH) in order to overcome this limitation.

The method proposed by Schlueter [60] instead was based on MIDACO, a single objective Ant Colony Optimisation algorithm, coupled with an NLP solver. The MIDACO solver was treating both discrete and continuous variables without any relaxation, and the constraints were treated with a new penalisation scheme. However, the satisfaction of the constraints was not required to be strict in this global exploration phase. The MIDACO solver was allowed to run for a fixed maximum runtime, and the best solution found was then refined by an NLP algorithm. Although the approach produced good results in the tests, there is no guarantee that the best weakly feasible solution found by MIDACO will not be significantly worsened by the NLP solver, which has to strictly enforce the constraints. Moreover, since the NLP can only work on continuous variables, the discrete variables were kept fixed during the solution of the NLP. Thus, since there is no guarantee that for a given choice of the discrete variables a fully feasible solution exists, the NLP solver might be doomed to fail regardless of the choice of the continuous variables.

To deal with mixed integer variables with the approach proposed in this work, the heuristics implemented in MACS will be modified in order to treat discrete variables. Thus, at the outer level, discrete variables will remain discrete. Similarly to the approach proposed by Schlueter, all variables will be handled simultaneously by the outer level global optimiser. However, the inner level will try to strictly satisfy the constraints. In order to allow the maximum flexibility to the inner level NLP, the discrete variables will be relaxed within the inner level. This gives the NLP the possibility of modifying even the discrete variables. This is particularly helpful, since it could be impossible for the NLP to satisfy some constraints for a fixed choice of the discrete variables, but this impossibility might require a large number of function evaluations before the NLP solver surrenders. Once the NLP converges to a relaxed feasible solution, a special set of constraints is added to enforce the relaxed variables to assume only discrete values, and the inner level NLP is solved again starting from the relaxed solution. This approach should allow to strictly satisfy complex mixed integer constraints and return fully feasible solutions at the outer level. When the single level gradient based refinement is invoked, it will start from a fully feasible solution and thus should have a good chance of improving the guess it received.

Contributions

The key contribution of this thesis are:

- for MACS
  - a modification of the heuristics, leading to improved performance on most benchmark problems tested [61]
  - the introduction of a new physics inspired archiving strategy, leading to an improved spreading of the solutions in the archive [61]
  - the introduction of a new way to generate the set of weight vectors, ensuring it is uniformly spread
- for the DFET transcription
  - the use of Bernstein bases, which guarantees non oscillating and converging control profiles even for bang-bang solutions [62]
  - the proof of a theorem guaranteeing that with Bernstein basis and the DFET transcription, if the feasible set of the inequality path constraints is convex, the inequality path constraints are satisfied for all times [62]
  - a demonstration that the use of Bernstein basis is beneficial also from the numerical point of view [62]

- for Multi objective optimal control
  - the introduction of a new approach to solve multi objective optimal control problems employing a general transcription method and seamlessly adopting two complementary formulations [63]
  - the reformulation of the problem as a bi-level problem, which allows for global exploration of the search space, tight satisfaction of the constraints and the generation of well spread set of points [64, 65]
  - the employment of a gradient based refinement strategy exploiting the Pascoletti-Serafini scalarisation, guaranteeing local optimality of the solutions and valid for multi objective problems [66, 63]
  - the introduction of an automatic and robust generation of the initial guesses [63]
- for Multi Objective Hybrid optimal control
  - the introduction of a solution approach based on an extension of the heuristics of MACS and the use of a relaxation approach to tackle the NLPs [67]
  - the introduction of a systematic approach to ensure that the relaxed variables assume integer values at the end of the relaxation process [67]
  - the introduction of a new set of constraints to handle combinatorial problems [67]
  - a proof of the usefulness of a unified treatment of discrete and continuous variables coupled with a global search and a local refinement [67]
- the testing of all methods against problems with known solution [64, 66, 68, 61, 63, 62, 67]
- the application of the proposed framework to tackle new and challenging space vehicle and mission design problems [63, 62, 67]

Part of the contents of this dissertations were published in different journal papers, book chapters, or presented at conferences. The list of those publications is listed hereafter.

Journal publications

1. **L. A. Ricciardi**, C. Maddock and M. Vasile. Direct Solution of Multi-Objective Optimal Control Problems Applied to Spaceplane Mission Design. Journal of Guidance, Control, and Dynamics, Vol. 42, No. 1 (2019), pp. 30-46.

2. **L. A. Ricciardi** and M. Vasile. Direct Transcriprion of Optimal Control Problems with Finite Elements on Bernstein Basis. Journal of Guidance, Control, and Dynamics, Vol. 42, No. 2 (2019), pp. 229-243.

3. C. Ortega Absil, **L. A. Ricciardi**, M. Di Carlo, C. Greco, R. Serra, M. Polnik, A. Vroom, A. Riccardi, E. Minisci, and M. Vasile. GTOC 9: Results from University of Strathclyde. Acta Futura, 9 January 2018, Vol. 11, p. 57-70.

Book chapters

M. Vasile and **L. A. Ricciardi**. Multi Agent Collaborative Search. NEO 2015: Results of the Numerical and Evolutionary Optimization Workshop NEO 2015 held at September 23-25 2015 in Tijuana, Mexico. Springer, 2017.

Peer-reviewed conference papers and presentations

1. M. Vasile and **L. A. Ricciardi**. A direct memetic approach to the solution of multi-objective optimal control problems. 2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016. 13 Feb 2017.

2. **L. A. Ricciardi**, M. Vasile and C. A. Maddock, Global solution of multi-objective optimal control problems with multi agent collaborative search and direct finite elements transcription, 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, 2016, pp. 869-876.

3. **L. A. Ricciardi**, M. Vasile, Improved Archiving and Search Strategies for Multi Agent Collaborative Search. Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences. Springer, 2019.

Conference papers and presentations

1. **L. A. Ricciardi**, M. Vasile, A Relaxation Approach for Hybrid Multi-Objective Optimal Control: Application to Multiple Debris Removal Missions (AAS 19-406). 29th AAS/AIAA Space Flight Mechanics Meeting, Ka'anapali, Hawaii, United States, 11-17 Jan 2019.

2. **L. A. Ricciardi**, M. Vasile, MODHOC - Multi Objective Direct Hybrid Optimal Control. 7th International Conference on Astrodynamics Tools and Techniques (ICATT). Oberpfaffenhofen (DLR), Germany. 6-9 Nov 2018

3. **L. A. Ricciardi**, C. A. Maddock and M. Vasile. Multi-objective optimal control of re-entry and abort scenarios. AIAA SciTech 2018 - Gaylord Palms, Kissimmee, United States. 8-12 Jan 2018.

4. C. Ortega Absil, **L. A. Ricciardi**, M. Di Carlo, C. Greco, R. Serra, M. Polnik, A. Vroom, A. Riccardi, E. Minisci and M. Vasile. GTOC9: Methods and results from the University of Strathclyde. On the generation and evolution of multiple debris removal missions. 26th International Symposium on Space Flight Dynamics (ISSFD), Matsuyama, Japan, 3-9 June 2017.

5. **L. A. Ricciardi**, M. Vasile, F. Toso, C. A. Maddock, Multi-Objective Optimal Control of the Ascent Trajectories of Launch Vehicles (AIAA 2016-5669). , AIAA/AAS Astrodynamics Specialist Conference, AIAA SPACE Forum,

6. M. Di Carlo, **L. A. Ricciardi**, M. Vasile, Multi-objective optimisation of constellation deployment using low-thrust propulsion. AIAA/AAS Astrodynamics Specialist Conference 2016, Long Beach, California, 13-16 Sept 2016.

7. S. McIntyre, T. Fawcett, T. Dickinson, M. West, C. A. Maddock, A. Mogavero, **L. A. Ricciardi**, F. Toso, K. Kontis, K. Hing Lo, S. Rengarajan, D. Evans, A. Milne, S. Feast. A commercially driven design approach to UK future small payload launch systems. 14th Reinventing Space Conference - Royal Society, London, United Kingdom, 24-27 Oct 2016.

8. S. McIntyre, T. Fawcett, T. Dickinson, C. A. Maddock, A. Mogavero, **L. A. Ricciardi**, F. Toso, M. West, K. Kontis, K. Hing Lo, S. Rengarajan, D. Evans, A. Milne, S. Feast. How to launch small payloads? Evaluation of current and future small payload launch systems. 14th Reinventing Space Conference - Royal Society, London, United Kingdom, 24-27 Oct 2016.

Structure of the thesis

This thesis is divided in two parts. Part I focuses on the theoretical and methodological development. Algorithms are described and tested against problems with known analytic solutions, or against problems for which only a numerical solution is available but is well documented in the literature. Part II presents advanced applications on problems involving the design and optimisation of space systems and missions.

In part I, Chapter 2 presents DFET, a numerical method to solve general optimal control problems and introduces the use of a new basis constructed on the Bernstein polynomials. With this addition, the method gains interesting capabilities in terms of suppressing spurious oscillations in case the controls change abruptly. In addition a new theorem is proved, showing that if the inequality path constraints have a convex feasible set, the DFET method with Bernstein basis returns a solution that satisfies those path constraints for all times.

Chapter 3 presents the general multi-objective optimisation problem and MACS, a memetic algorithm to solve that class of problems. The chapter also shows some modifications of the heuristics that proved to be beneficial in the test cases. Finally, a new archiving strategy is described, based on the phyisical concept of energy. It is shown that this archiving strategy significantly improves the spreading of the solutions obtained not only by MACS, but also by another multi-objective optimisation algorithm.

Chapter 4 extends the standard optimal control problem to its multi-objective version, and provides an algorithm to solve nonlinear multi-objective optimal control problems. Two complementary formulations are employed: a bi-level formulation, which allows for the global exploration of the search space, and a single level formulation, which allows to guarantee the local optimality of the solution. In addition, a strategy to automatically generate initial guesses is described.

Chapter 5 extends the multi-objective optimal control problem to its mixed integer version, and presents an algorithm to handle those kinds of problems. The algorithm is a further evolution of the method described in Chapter 4, where the heuristics of MACS have been modified in order to deal with integer variables. In addition it is shown how to handle the discrete variables in both levels of the bi-level approach.

Each chapter of Part II can be seen as the application of the theory and methods described in Part I to space vehicle and mission design problems. In particular, Chapter 6 presents a three objective design and optimal control problem for a new launch vehicle, and the multi-objective design and optimal control for the launch, reentry and abort trajectory of a spaceplane. Chapter 7 presents the solution of a multi-objective time dependant motorised travelling salesmen problem, and the multi-objective optimal control of a multi-target space mission, where the trajectory of the spacecraft is to be optimised together with the choice of the sequence of targets to visit. Finally, Chapter 8 closes with the conclusions.

Chapter 2

Direct transcription of Optimal Control problems[1]

Felix qui potuit rerum cognoscere causas
Fortunate who was able to know the causes of things

Virgil

This chapter introduces the general optimal control problem, and describes the Direct Finite Elements in Time transcription, a method to convert the initial infinite dimensional problem into a finite dimensional problem. The properties of the transcription method are analysed, especially in terms of the flexibility it provides. This flexibility is then exploited by changing the basis for the functions representing the states and the controls of the problem. The properties of the DFET transcription with the new basis are analysed, first theoretically, and then numerically. A simple test case, with analytic solution, is used to compare the newly proposed basis with the basis typically employed with DFET. The comparison involves the qualitative and quantitative behaviour of the resulting solution, the convergence rate to the known

[1]The contents of this chapter were published in: L. A. Ricciardi and M. Vasile. Direct Transcriprion of Optimal Control Problems with Finite Elements on Bernstein Basis. Journal of Guidance, Control, and Dynamics, Vol. 42, No. 2 (2019), pp. 229-243.

optimal solution and the number of iterations required by the NLP solvers to achieve a feasible and an optimal solution. A further analysis is performed by changing the NLP algorithm, showing some interesting synergies between the choice of the basis functions and the NLP algorithm.

2.1 The optimal control problem

A generic, single-objective, optimal control problem can be formulated as [19, 69, 70]:

$$\begin{aligned}
&\min_{\mathbf{u}\in U} J = \min_{\mathbf{u}\in U} \phi\left(\mathbf{x}(t_0), \mathbf{x}(t_f), t_0, t_f\right) + \int_{t_0}^{t_f} L\left(\mathbf{x}(t), \mathbf{u}(t), t\right) dt \\
&s.t. \\
&\dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \\
&\mathbf{g}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\
&\boldsymbol{\psi}\left(\mathbf{x}(t_0), \mathbf{x}(t_f), \mathbf{u}(t_0), \mathbf{u}(t_f), t_0, t_f\right) \leq 0 \\
&t \in [t_0, t_f]
\end{aligned} \tag{2.1}$$

where the scalar t is time, the functions $\mathbf{x}(t) : [t_0, t_f] \to \mathbb{R}^{n_x}$ are the state vector, the functions $\mathbf{u}(t) : [t_0, t_f] \to \mathbb{R}^{n_u}$ are the control vector and J is the sum of a boundary state cost function $\phi : \mathbb{R}^{2n_x+2} \to \mathbb{R}$ and of the integral of a running cost function $L : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times [t_0, t_f] \to \mathbb{R}$. The functions $\mathbf{x}(t)$ belong to the Sobolev space $W^{1,\infty}$ while the functions $\mathbf{u}(t)$ belong to L^{∞}. The state and control vectors are subject to a set of dynamic constraints $\dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x}(t), \mathbf{u}(t), t\right)$, algebraic constraints $\mathbf{g}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0$ and boundary conditions $\boldsymbol{\psi}\left(\mathbf{x}(t_0), \mathbf{x}(t_f), \mathbf{u}(t_0), \mathbf{u}(t_f), t_0, t_f\right) \leq 0$, where $\mathbf{F}\left(\mathbf{x}(t), \mathbf{u}(t), t\right)$, $\mathbf{g}\left(\mathbf{x}(t), \mathbf{u}(t), t\right)$ and $\boldsymbol{\psi}\left(\mathbf{x}(t_0), \mathbf{x}(t_f), \mathbf{u}(t_0), \mathbf{u}(t_f), t_0, t_f\right)$ are vector fields.

After introducing the Hamiltonian function

$$\mathcal{H}(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}, \boldsymbol{\mu}, t) = L\left(\mathbf{x}(t), \mathbf{u}(t), t\right) + \boldsymbol{\lambda}^T(t)\mathbf{F}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) + \boldsymbol{\mu}^T(t)\mathbf{g}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \tag{2.2}$$

which includes the Lagrange multipliers $\boldsymbol{\lambda}(t)$ (also known as co-states) and $\boldsymbol{\mu}(t)$, the conditions for optimality of this problem can be derived by applying Pontryagin's maximum principle, resulting in:

$$\begin{aligned}
&\dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x}(t), \mathbf{u}(t), t\right)\\
&\dot{\boldsymbol{\lambda}}^T = -\nabla_x \mathcal{H}\left(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\lambda}(t), \boldsymbol{\mu}(t), t\right)\\
&\mathbf{u} = \arg\min_{\mathbf{u}\in U} \mathcal{H}\left(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\lambda}(t), \boldsymbol{\mu}(t), t\right)\\
&\mathbf{g}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0\\
&\boldsymbol{\psi}\left(\mathbf{x}(t_0), \mathbf{x}(t_f), \mathbf{u}(t_0), \mathbf{u}(t_f), t_0, t_f\right) \leq 0\\
&\boldsymbol{\lambda}^T(t_f) = \left[\nabla_x \phi + \boldsymbol{\nu}^T \nabla_x \boldsymbol{\psi}\right]_{t=t_f}\\
&\left[\phi_t + \boldsymbol{\nu}^T \boldsymbol{\psi}_t + \mathcal{H}\right]_{t=t_f} = 0
\end{aligned} \tag{2.3}$$

where $\boldsymbol{\nu}$ are additional Lagrange multipliers associated to the boundary conditions $\boldsymbol{\psi} = 0$, and the last equation is needed only if the final time t_f is free.

More details about this derivation are given in Appendix A.

2.2 Direct Transcription with Finite Elements in Time

Direct Finite Elements in Time (DFET) is a direct transcription method to solve optimal control problems and was initially proposed by Vasile and Finzi [71] in 2000. Finite Elements in Time (FET) for the indirect solution of optimal control problems were initially proposed by Hodges and Bless [72], and during the late 1990s evolved to the discontinuous version. As pointed out by Borri and Bottasso [73], several possible formulations of FET are possible, the most promising ones being those deriving from the semi-discontinuous and bi-discontinuous formulation of the mixed field equations, which result in an unconditionally stable symplectic integration scheme for the bi-discontinuous version, or in an unconditionally stable scheme with the asimptotic annihilation property, resulting in very useful schemes for the integration of stiff systems. Moreover, Bottasso and Ragazzi [74] showed that FET for the forward integration of ordinary differential equations are equivalent to some classes of implicit Runge-Kutta integration schemes. Finally, FET can be extended to arbitrary high-order and Vasile [70] showed how a direct transcription method based on FET is very robust and allows for a solution refinement technique called h-p adaptivity, which will be shortly described later on.

In the past decade, direct transcription with FET on spectral bases has been successfully used to solve a range of difficult problems: from the design of low-thrust multi-gravity assist trajectories to Mercury [75] and the Sun [76], to the design of weak stability boundary transfers to the Moon, low-thrust transfers in the restricted three body problem and optimal landing trajectories to the Moon [71]. More recently they have been used to perform multi-objective optimal control of spacecraft [64, 66], ascent trajectories of launchers [65], or abort trajectories of reusable launch vehicles [68].

2.2.1 Problem Transcription with DFET

In this section we briefly recall how the transcription method based on Finite Elements in Time works. Following the general approach to DFET transcription proposed by Vasile and Finzi [71], the differential constraints can be recast in weak form and integrated by parts leading to,

$$\int_{t_0}^{t_f} \left(\dot{\mathbf{w}}(t)^T\mathbf{x}(t) + \mathbf{w}(t)^T\mathbf{F}\left(\mathbf{x}(t), \mathbf{u}(t), t\right)\right) dt - \mathbf{w}_f^T\mathbf{x}_f^b + \mathbf{w}_0^T\mathbf{x}_0^b = 0 \tag{2.4}$$

where $\mathbf{w}(t)$ are the generalised weight functions that must assume a value of 0 on either bound and $\mathbf{x}^b$ are the boundary values of the states, that may be either imposed or free. Note that, in this bi-discontinuous formulation, the value of $\mathbf{x}(t)$ at the boundaries does not coincide with either with $\mathbf{x}_f^b$ or $\mathbf{x}_0^b$. As remarked by Bottasso in [77], the weak form of the ODEs expressed in (2.4) is a mathematically correct formulation even when $\mathbf{F}\left(\mathbf{x}(t), \mathbf{u}(t), t\right)$ is not continuous or differentiable.

Let the time domain D be decomposed into N finite elements such that

$$D = \bigcup_{j=1}^{N} D_j(t_{j-1}, t_j) \tag{2.5}$$

and parametrise, over each D_j, the states, controls and weight functions as

$$\mathbf{x}_j(t) = \sum_{s=0}^{l_x} f_{sj}(t)\,\mathbf{x}_{sj} \tag{2.6a}$$

$$\mathbf{u}_j(t) = \sum_{s=0}^{l_u} g_{sj}(t)\,\mathbf{u}_{sj} \tag{2.6b}$$

$$\mathbf{w}_j(t) = \sum_{s=0}^{l_x+1} h_{sj}(t)\,\mathbf{w}_{sj} \tag{2.6c}$$

where the functions $\mathbf{x}_j, \mathbf{u}_j, \mathbf{w}_j$ are defined over each finite element D_j, the functions $f_{sj}(t)$, $g_{sj}(t)$ and $h_{sj}(t)$ are chosen among the space of polynomials of degree l_x, l_u and $l_x + 1$ respectively, and the vectors $\mathbf{x}_{sj}, \mathbf{u}_{sj}, \mathbf{w}_{sj}$ are weights given to each polynomial in each element. The polynomials h_{sj} have to assume a value of 0 on either the left or the right bound of the element.

It is practical to define each D_j over the normalised interval $[-1,\, 1]$ through the transformation,

$$\tau = 2\frac{t - \frac{t_j - t_{j-1}}{2}}{t_j - t_{j-1}} \tag{2.7}$$

This way the domain of the basis function is constant and irrespective of the size of the element and also overlaps with the interval of the Gauss nodes that will be employed for the integration of the dynamics.

Substituting the definitions of the polynomials into the objective functions and integrating with Gauss quadrature formulas with q nodes leads to

$$\tilde{J} = \phi(\mathbf{x}_0^b, \mathbf{x}_f^b, t_0, t_f) + \sum_{j=1}^{N}\sum_{k=1}^{q} \sigma_k L(\mathbf{x}_j(\tau_k), \mathbf{u}_j(\tau_k), \tau_k)\frac{\Delta t}{2} \tag{2.8}$$

and for the variational constraints leads for every element j to the system

$$\sum_{k=1}^{q} \sigma_k \left[\dot{\mathbf{w}}_j(\tau_k)^T \mathbf{x}_j(\tau_k) + \mathbf{w}_j(\tau_k)^T \mathbf{F}_j(\tau_k)\frac{\Delta t}{2}\right] - \mathbf{w}_j(1)^T \mathbf{x}_j^b + \mathbf{w}_j(-1)^T \mathbf{x}_{j-1}^b = 0 \tag{2.9}$$

where τ_k and σ_k are the Gauss nodes and weights, $\mathbf{F}_j(\tau_k)$ is the shorthand notation for $\mathbf{F}\left(\mathbf{x}_j(\tau_k), \mathbf{u}_j(\tau_k), \tau_k\right)$, and $\mathbf{x}_j^b$ and $\mathbf{x}_{j-1}^b$ denote the boundary values of element j at $t = t_j$ and $t = t_{j-1}$ respectively. Algebraic constraints are usually evaluated at the Gauss integration nodes, but as it will be shown this is not the only choice.

It was shown in Vasile [70] that in the limit where the number of integration points σ_k tends to infinity, the DFET transcription schemes satisfies a set of necessary optimality conditions that converges to the necessary optimality conditions (2.3). For completeness, that derivation is reported in Appendix A.

The DFET transcription is thus not only equipped with a convergence theory, but it is also very flexible, since it allows to parameterise both states and controls choosing any basis, and these bases could also be different for every variable and element. Similarly it is possible to employ several choices for the type of quadrature nodes or their number q. This flexibility was not exploited in any previous work. In the following a new scheme with unique characteristics will be generated by adopting a new set of basis functions. Although the derived scheme is different than the one proposed by [70], the convergence results demonstrated for the DFET transcription are still applicable because the proof does not require any particular choice for the basis functions.

2.2.2 Choice of Basis Functions

In the DFET literature, the basis is typically generated through a Lagrange interpolation on the quadrature nodes, usually either Gauss-Legendre or Gauss-Lobatto:

$$f_{sj}(\tau) = \prod_{k=0,k\neq s}^{l_x} \frac{\tau - \tau_k}{\tau_s - \tau_k}, \qquad g_{sj}(\tau) = \prod_{k=0,k\neq s}^{l_u} \frac{\tau - \tau_k}{\tau_s - \tau_k}, \qquad h_{sj}(\tau) = \prod_{k=0,k\neq s}^{l_x+1} \frac{\tau - \tau_k}{\tau_s - \tau_k} \tag{2.10}$$

Since the test functions w must assume a value of 0 on either bound, the nodes for the construction of test functions with Lagrange interpolation must be of Lobatto type.

Since the basis has to be evaluated only at the quadrature nodes, constructing the bases as Lagrange interpolations ensures that $f_{sj}(\tau_k) \neq 0$ only if $s = k$. This should generally result in a good sparsity pattern for the Jacobian of the constraints. However a proper implementation of DFET already ensures a highly sparse block diagonal Jacobian of the constraints regardless of the choice of the basis. Thus the further improvement due to this particular choice is expected to be modest because it can only act on the individual block. Moreover, even if a sparse Jacobian generally means that the NLP solver will have an easier time to converge to the final solution, this does not mean that a slightly sparser Jacobian will result in a quicker convergence.

In other words, this basis seems to provide marginal benefits overall. Other polynomial bases could instead provide different and maybe more significant benefits, but to the authors' knowledge the only polynomial bases used for DFET are Lagrange interpolation on either the Gauss-Legendre or the Gauss-Lobatto quadrature nodes.

Although DFET was proven to be a valid technique to solve difficult optimal control problems, when the solution presents very sharp variations in states and controls, like bang-bang control profiles, the polynomial representation of states and controls can display undesired oscillations exceeding the bounds, even if the nodal solution is correct. This is known in the literature as Gibbs phenomenon. Vasile [70] proposed h-p adaptivity strategies to improve the accuracy of the solutions found with the DFET transcription. Although those strategies can be used to mitigate the problem, oscillations can remain regardless of the choice of the position of the nodes. This is undesirable in a number of cases:

- Even if the polynomial representation of states and controls is generally evaluated at the collocation points only, the associated polynomial cannot be practically used as interpolant of the nodal solution at any other instant of time. Thus it cannot be used as guidance law.
- If different polynomial orders or collocation points are used for states and controls, one could end up evaluating the polynomials at points affected by the Gibbs

phenomenon.

- In a post-processing phase, one might want to evaluate some derived quantities at arbitrary points along the solution. Values coming from those spurious oscillations that are outside the boundaries of states and controls could lead to numerical exceptions.
- A regular behaviour in between two collocation nodes allows for a better re-integrability of the control law.
- H-p adaptive techniques are iterative procedures which take a given solution, re-evaluate it on a more refined grid and re-optimise it on this new grid. The re-evaluation on the more refined grid requires the evaluation of the solution at time instants that are different from the nodal values. Thus, the resulting initial solution defined over the new grid could be out of bounds, with resulting numerical difficulties or need for ad hoc procedures. All these problems could be solved with a linear interpolation of the nodal solution for the controls, but a linear interpolation would be inconsistent if a higher order polynomial representation was used to obtain the solution.

This section proposes the use of Bernstein polynomials, for the DFET transcription method, instead of the commonly used Lagrange interpolation on spectral bases. As will be proved, the use of Bernstein polynomials guarantees that the representation of states and controls remains within the feasible set over the whole time domain and not only at the collocation nodes and avoids the Gibbs phenomenon at points of jump discontinuity [78] or sharp slope variations.

In addition, a quantification of the computational cost for comparable accuracy of both Lagrange and Bernstein bases will be provided . Two metrics will be used: the sparsity pattern of the Jacobian of the constraints of the resulting NLP problem and the number of iterations required to converge to the desired feasibility and optimality of the NLP solution. The computational cost will be measured for an increasing order of the polynomials (p-refinement), at constant number of finite elements, and for an increasing number of elements (h-refinement), at constant order of the polynomials. Two cases will be considered: one with no path constraints but a sharp change in the control profile and one with a path constraint on one state variable.

Linear combinations of Bernstein polynomials generate the so called Bezier curves. The use of Bezier curves to solve optimal control problems can be found in the work of Rogalsky [79] who solved optimal control problems with a linear dynamics, using Differential Evolution and Bezier curves to represent the control profile. He noted that the resulting curves always lie in the convex hull of the control points and never present undesirable oscillations away from its defining control points. Thus, Bézier curves could be used to parameterize smooth, non-oscillatory functions, with minimal epistasis, using only a few parameters.

In [80, 81], Gomanjani et al. used Bezier curves in the analytical solution of some optimal control problems, with linear dynamics, because they led to a simpler symbolic manipulations. Darehmiraki et al. [82] proposed the use of Bernstein polynomials with a weighted residual methods to solve distributed optimal control problems. Similarly Mirkov and Rasuo in [83] proposed the use of Bernstein polynomials to solve Elliptic Boundary value problems in a collocation method, while Bhatti and Bracken [84] proposed the use of Bernstein polynomials in a Galerkin method to solve ODEs. Mirkov and Rasuo showed that on linear problems with continuous solutions, collocation methods based on Bernstein polynomials had an exponential convergence rate though slower than pseudo-spectral and Chebyshev collocation schemes. Finally Farouki and Rajan [85] showed that the numerical conditioning of polynomials using Bernstein form is particularly stable under finite precision arithmetic.

It is interesting to note that only a few researchers exploited the convex hull and variation diminishing properties of Bezier curves for optimal control problems. These properties are instead the main reason behind the choice of the use of Bernstein polynomials: here we show that by choosing Bernstein polynomials as a basis for the parameterisation of states and controls, one can generate solutions that display the aforementioned properties. Moreover, differently from previous works, Bernstein polynomials are here used to solve general nonlinear optimal control problems.

This chapter also includes a theorem that provides some necessary conditions for the satisfaction of general path constraints, provided that the feasible region is convex and states and controls are described by Bezier curves. Finally, the main contribution of this chapter is to show the benefit of using Bernstein basis, within the DFET transcription framework, when the solution has sharp variations in the control law or simple path constraints.

2.2.3 Bernstein Basis

A Bernstein basis of order n is defined as

$$B_{\nu,n}(t) = \binom{n}{\nu} t^{\nu}(1-t)^{n-\nu} \qquad 0 \le \nu \le n, 0 \le t \le 1 \tag{2.11}$$

As shown in Fig. 2.1a Bernstein polynomials either assume a value of 0 on both boundaries or a value of 1 on one boundary and of 0 on the other, so can be used as test functions w. Since the resulting curves will be integrated through Gauss quadrature, whose nodes are defined on $[-1, 1]$, Bernstein polynomials must also be redefined on the interval $[-1, 1]$:

$$\tilde{B}_{\nu,n}(\tau) = B_{\nu,n}(t) \qquad \tau = 2t - 1 \tag{2.12}$$

In the following, for clarity of notation, the subscript n to denote the order of the Bernstein polynomials will be dropped. That degree will equal l, m or $l+1$ depending on whether the polynomials will be used to describe respectively the states, controls or test functions. Substituting (2.12) into (2.6) we obtain

$$\mathbf{x}_j(t) = \sum_{s=0}^{l_x} \tilde{B}_{sj}(\tau)\, \mathbf{x}_{sj} = \boldsymbol{\mathcal{B}}_j(\tau) \tag{2.13a}$$

$$\mathbf{u}_j(t) = \sum_{s=0}^{l_u} \tilde{B}_{sj}(\tau)\, \mathbf{u}_{sj} = \boldsymbol{\mathcal{C}}_j(\tau) \tag{2.13b}$$

$$\mathbf{w}_j(t) = \sum_{s=0}^{l_x+1} \tilde{B}_{sj}(\tau)\, \mathbf{w}_{sj} = \boldsymbol{\mathcal{D}}_j(\tau) \tag{2.13c}$$

Thus, if the basis is made of Bernstein polynomials the resulting state curves $\mathcal{B}_j(\tau)$, control curves $\mathcal{C}_j(\tau)$ and test curves $\mathcal{D}_j(\tau)$ are by definition Bezier curves. Bezier curves are a class of curves commonly employed in computer graphics and computer aided design because they enjoy several interesting properties, among which we employ the following:

1. a Bezier curve of degree n can be equivalently described by a vector of $n+1$ nodes or weights, equally spaced in time as shown in Fig. 2.1b. In this work, the vector of nodes is $\left(\frac{2s}{n} - 1, \mathbf{x}_{sj}\right), 0 \leq s \leq l_x$ for the states and $\left(\frac{2s}{m} - 1, \mathbf{u}_{sj}\right), 0 \leq s \leq l_u$ for the controls. It is important to note that these nodes are equispaced in time, and thus never completely coincide with the Gauss integration nodes τ_k. Moreover, weights $\mathbf{x}_{sj}$ and $\mathbf{u}_{sj}$, in the case of Legendre interpolation can coincide with the integration nodes τ_k, thus the weights can be seen as the nodal solution on τ_k because, by construction, $f_{sj}(\tau_k) = 1$ if $k = s$. If the Bernstein basis is used, instead, the weights $\mathbf{x}_{sj}$ and $\mathbf{u}_{sj}$ cannot be interpreted as the nodal solution on either the nodes τ_s or τ_k. However, they can be interpreted as the vertices of the polygonal chain that the Bezier curve will approximately follow.

2. Bezier curves are completely contained in the convex hull of the polygonal chain connecting the nodes that define them, as shown again in Fig. 2.1b. This is a property that can be often found mentioned in the literature [79, 80, 81, 82]. Here we propose a formal demonstration of this property.

Lemma 1 *Bezier curves are contained in the convex hull defined by their nodes.*

Proof 1 *A generic Bezier curve $\mathcal{Q}(t)$ is defined as*

$$\mathcal{Q}(t) = \sum_{\nu=0}^{n} B_{\nu,n}(t) x_\nu \tag{2.14}$$

where $B_{\nu,n}(t)$ are Bernstein polynomials of degree n and x_ν are called weights or nodes. By definition, Bernstein polynomials satisfy the positivity condition

$$B_{\nu,n}(t) = \binom{n}{\nu} t^\nu (1-t)^{n-\nu} \geq 0, \qquad 0 \leq t \leq 1, \quad 0 \leq \nu \leq n \in \mathbb{N} \tag{2.15}$$

and the partition of unity condition:

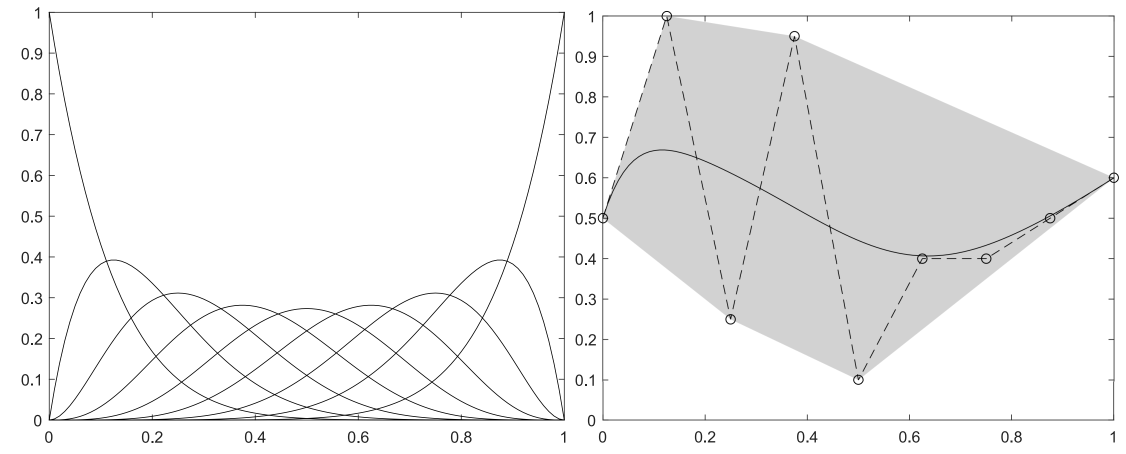
$$\sum_{\nu=0}^{n} B_{\nu,n}(t) = (1-t+t)^n = 1 \qquad 0 \leq t \leq 1, \quad 0 \leq \nu \leq n \in \mathbb{N} \tag{2.16}$$

where the central equality derives from the binomial theorem. The vertices x_ν can be enclosed in a convex hull, which is defined as :

$$Conv(x_\nu) := \left\{ \sum_{\nu=0}^{n} \lambda_\nu x_\nu \middle| (\forall \nu : \lambda_\nu \geq 0) \wedge \sum_{\nu=0}^{n} \lambda_\nu = 1 \right\} \tag{2.17}$$

Since each $B_{\nu,n}(t)$ satisfies the requirements to be a λ_ν for every t, it follows that Bezier curves are contained in the convex hull enclosing the nodes x_ν which define them.■

3. Bezier curves have the variation diminishing property (for the proof see Ait-Haddou et al. [86]): the number of times a straight line intersects the curve is always less or equal to the number of intersection the same line has with the polygonal chain. Intuitively, this means that the Bezier curve oscillates less than the polygonal chain defining it, see again Fig. 2.1b. This property guarantees that if the polygonal chain connecting the nodal values of the controls is monotonic, the resulting Bezier curve will be monotonic too. Thus, when an optimal control problem has a bang-bang solution and the discretisation is not



(a) Bernstein basis of order 8 (b) A Bezier curve of order 8

Figure 2.1: A Bernstein basis and a Bezier curve. Circles: Bezier nodes. Dashed line: polygonal chain. Shaded area: convex hull.

perfectly capturing the discontinuity, the resulting representation with Bezier curves will be a smooth and monotonic curve completely within the prescribed bounds.

When Bezier curves are used to parameterise states and controls we can prove the following theorem:

Theorem 1 *If the feasible region for the path constraints is convex, the states and controls are represented as Bezier curves and the nodes of the Bezier curves satisfy the path constraints, then the Bezier curves for the states and controls are inside the feasible region for all $t \in [t_0, t_f]$.*

Informally, the proof directly descends from Lemma 1, the hypothesis that the feasible region for path constraints is convex and the hypothesis that the nodes of the Bezier curve are feasible. Since the Bezier curves for states and controls are included in the convex hull (which includes some or all the nodes of the polygonal chain) and the convex hull is included in the feasible region, it follows that the Bezier curves are included in the feasible region, and thus are feasible for all $t \in [t_0, t_f]$. More formally we can prove that the whole Bezier curves that approximate the time history of states and controls are contained in the feasible set.

Proof 2 *The feasible set* $\mathbb{F}$ *of the path constraints* $\mathbf{g}(\mathbf{x}(t), \mathbf{u}(t), t) \leq 0$ *is defined as*

$$\mathbb{F} := \left\{ \mathbf{l}(t) \middle| \mathbf{g}(\mathbf{l}(t)) \leq 0 \right\} \tag{2.18}$$

with $\mathbf{l}(t) = (\mathbf{x}(t), \mathbf{u}(t), t)$. *If* $\mathbb{F}$ *is convex, then any linear convex combination of a generic number* L *of its elements* $\mathbf{l}_s(t)$ *also belongs to the set, thus satisfies*

$$\mathbf{g}\left(\sum_{s=1}^{L} \lambda_s \mathbf{l}_s(t)\right) \leq 0, \qquad \forall s : 0 \leq \lambda_s \leq 1, \quad \sum_{s=1}^{L} \lambda_s = 1 \tag{2.19}$$

If $\mathbf{l}_s(t)$ *are the corners of the convex hull:*

$$\mathcal{H} = Conv\left(\mathbf{l}_s(t)\right) := \left\{ \sum_{s=1}^{L} \lambda_s \mathbf{l}_s(t) \middle| (\forall s : \lambda_s \geq 0) \wedge \sum_{s=1}^{L} \lambda_s = 1 \right\} \tag{2.20}$$

then all points belonging to the convex hull $\mathcal{H}$ *are also feasible:*

$$\mathcal{H} \subseteq \mathbb{F} \tag{2.21}$$

Now let the time domain $[t_0, t_f]$ *be partitioned into finite elements and states* $\mathbf{x}(t)$ *and controls* $\mathbf{u}(t)$, *within every finite element* $D_j = [t_j, t_{j+1}]$, *be represented as Bezier curves with nodes* $\mathbf{l}_s(t)$. *From Lemma 1 it follows that the state-control vector* $\mathbf{l}(t)$ *is contained in the convex hull* $\mathcal{H}$ *for every time element.* ■

Remark 1 *If the conditions of Theorem 1 are applicable then path constraints are satisfied for all* $t \in [t_0, t_f]$. *This is a unique feature of the direct transcription method proposed in this work. From an algorithmic point of view, this translates into imposing constraint conditions on the weights* $\mathbf{x}_{s,j}$ *and* $\mathbf{u}_{s,j}$ *such that* $\mathbf{g}(\mathbf{x}_{s,j}, \mathbf{u}_{s,j}, t_j) \leq 0$ *is satisfied because, see property 1 here above, these weights are equivalent to a set of nodes in the space of the states and controls, equispaced in time. Thus if* $\mathbf{g}$ *satisfies convexity condition (2.19) then* $\mathbf{g}(\mathbf{x}_{s,j}, \mathbf{u}_{s,j}, t_j) \leq 0$ *satisfies the conditions of the theorem.*

Remark 2 *Since the convex hull is a subset of the feasible set, some regions of the feasible set might be left out. This means that if the optimal solution lies in the left* *out region, this method can only return the fully feasible solution closest to the optimal one. However, if the number of corners of the convex hull is increased, for example by increasing the number of DFET elements or the order of Bezier curves, those left out regions will asymptotically vanish, and the method can converge to the true optimal solution. A practical example of this behaviour will be shown in the following section.*

Remark 3 *As reported in the literature, Bernstein polynomials for function approximation display a lower convergence rate than Chebychev polynomials, or Lagrange interpolation schemes in a number of cases [78, 87, 88]. However, in [83] it was shown that the convergence rate of collocation schemes, based on Bernstein polynomials, for the solution of elliptic boundary value problems was exponential albeit worse than pseudo-spectral or Chebychev collocation schemes. A thorough theoretical assessment of the convergence rate of the DFET transcription with Bernstein polynomials will require a dedicated study and is left for future work. In the following examples, however, we will show that the value of the objective function, computed with Bernstein basis, converges slower than the one computed with Lagrange polynomials on spectral basis.*

Remark 4 *A result on the guaranteed feasibility of the optimal control solution for all* $t \in [t_0, t_f]$ *can be found also in Loock et al. [89] for differentially flat systems parameterised with B-splines, and more recently in Cichella et al. [90] for differentially flat systems parameterised with Bernstein polynomials. Differentially flat systems are such that a change of variables exists that allows writing states and controls as explicit functions of the new variables without integration. However, as stated by the authors, there is no systematic way to assess if a nonlinear problem is differentially flat and to find an associated variable trasformation. The approach proposed in this work, instead, does not require any change of variables and simply requires to verify the convexity of the feasible set defined by the path constraints. To be noted that for non-convex constraints one can still apply the theory and method proposed in this work after using the existing convexification techniques [91, 92, 93].*

2.3 Benchmark Cases

The following examples demonstrate the effect of these properties on the solution of a simple optimal control problem. As a simple but representative test case, we consider the minimum time transfer to rectilinear path problem, initially proposed in [19]. This problem deals with the ascent of a point mass, subject only to a constant gravitational acceleration in an inertial frame and to a control acceleration with constant magnitude. The direction of the thrust vector depends on the control angle u, and the dynamics of the point mass is defined by the following set of differential equations:

$$\begin{cases} \dot{x} &= v_x \\ \dot{v}_x &= a\cos u \\ \dot{y} &= v_y \\ \dot{v}_y &= -g + a\sin u \end{cases} \tag{2.22}$$

where x and y are the horizontal and vertical position coordinates of the point mass, v_x and v_y are its horizontal and vertical velocity components, g is the magnitude of gravitational acceleration and a is the modulus of the control acceleration. The point mass is initially at rest, and it has to reach a specified altitude with zero vertical velocity, in minimum time. The terminal conditions are:

$$\begin{cases} y(t_f) = h \\ v_y(t_f) = 0 \end{cases} \tag{2.23}$$

This problem was previously solved in [70] with DFET to demonstrate an h-p adaptivity strategy, and a multi-objective version of the same problem was solved in [64, 66] with DFET and multi-objective memetic algorithms. For consistency with the aforementioned references we will use the following values $g = 1.6 \cdot 10^{-3}$, $a = 4 \cdot 10^{-3}$, $h = 10$ and $u \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. Two different instances of the problem will be solved: 1) minimum ascent time to a given altitude and no terminal constraint on the horizontal velocity and 2) minimum ascent time to a given altitude with no terminal constraint on the horizontal velocity but a path constraint on the vertical velocity.

All the cases were initialised in the same way, and a feasible solution was first sought using the Interior Point method of the NLP solver *fmincon* from the MATLAB® Optimization Toolbox. After a fully feasible solution was found, it was used as starting point to be optimised. Solutions of all test cases converged below the relative tolerance of 10^{-6} both in feasibility and optimality.

2.3.1 Problem Instance 1 - Minimum Ascent Time with no Terminal Constraint on the Horizontal Velocity

In the first instance of the problem the horizontal velocity has no prescribed terminal value. The associated optimal control law is:

$$u(t) = \begin{cases} \frac{\pi}{2} & 0 \leq t \leq \sqrt{\frac{h(a+g)}{a(a-g)}} \\ -\frac{\pi}{2} & \sqrt{\frac{h(a+g)}{a(a-g)}} < t \leq \sqrt{\frac{h(a+g)}{a(a-g)}} + \sqrt{\frac{h(a-g)}{a(a+g)}} \end{cases} \tag{2.24}$$

which is the asymptotic limit of the bilinear tangent law when the final horizontal velocity is zero. For the values of a, h and g used, this corresponds to a switching time $t_s = 76.3763$ and a final time of $t_f = 109.1089$, which is also the optimal value of the objective function.

Solution (2.24) can be derived by observing that x and v_x have no final state constraints, no path constraints and the objective does not depend explicitly on them, thus the time evolution of the x coordinate is only dependent on the initial conditions and one can concentrate only on the dynamics along the y axis. This means that u has to be either $\frac{\pi}{2}$ or $-\frac{\pi}{2}$ for $0 \leq t \leq t_f$. Moreover, since thrust has to contrast gravity at t_0, u must initially points upwards and switch to point downwards at a time t_s in order to satisfy the terminal constraints $y(t_f) = h$ and $v_t(t_f) = 0$. Thus we have

$$u(t) = \begin{cases} \frac{\pi}{2} & 0 \leq t \leq t_s \\ -\frac{\pi}{2} & t_s \leq t \leq t_f \end{cases} \tag{2.25}$$

which corresponds to the following time law for the vertical velocity

$$v_y(t) = \begin{cases} (g-a)t & 0 \le t \le t_s \\ (g-a)t_s - (g+a)(t-t_s) & t_s \le t \le t_f \end{cases} \tag{2.26}$$

and the following time law for the vertical position

$$y(t) = \begin{cases} \frac{1}{2}(g-a)t^2 & 0 \le t \le t_s \\ \frac{1}{2}(g-a)t_s^2 + (g-a)t_s t - \frac{1}{2}(g+a)(t-t_s)^2 & t_s \le t \le t_f \end{cases} \tag{2.27}$$

Imposing the boundary conditions for t_f, we get

$$\begin{cases} (g-a)t_s - (g+a)(t_f-t_s) = 0 \\ \frac{1}{2}(g-a)t_s^2 + (g-a)t_s t_f - \frac{1}{2}(g+a)(t_f-t_s)^2 = h \end{cases} \tag{2.28}$$

which is a system with one linear equation and one second order equation. This system can be solved analytically to get the aforementioned values for t_s and t_f.

For this first instance, three different test cases are presented. In the first case, Problem 2.22 was solved with an increasing order of the polynomials for both states and controls, in the second case only the order of the polynomials of the controls was increased, while in the third the order of the polynomials was fixed but number of elements was increased.

Problem Instance 1, Test case 1

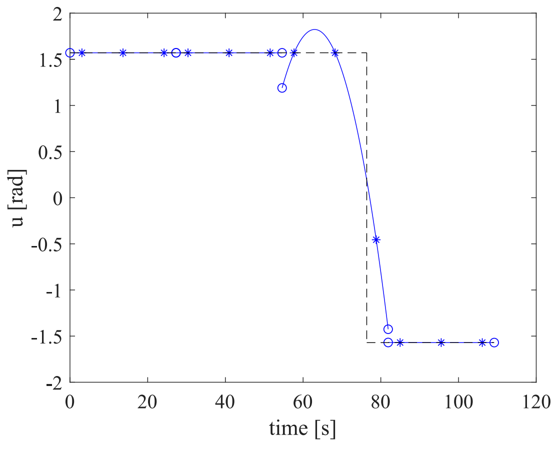
In the first test case, the order of the polynomials of both states and controls were varied simultaneously from 2 to 14 and were represented using either the Lagrange interpolation basis constructed on Legendre nodes, or the Bernstein basis. Integration was performed using Gauss-Legendre quadrature rules with $q = l_x + 1$. This resulted, for the Lagrange interpolation basis, in a matching of the integration and collocation nodes, which is the most common choice for both DFET and pseudo-spectral methods.

Figure 2.2 shows, for test case 1, the control profiles for the two bases (continuous line) using different orders and their comparison with the analytical solution (dashed line). It can be seen that Lagrange bases display significant oscillations near the discontinuity even if the collocation nodes (asterisk marker) correctly follow the analytical bang-bang profile. The circles correspond, in control space, to the boundary values of each finite element. The use of solid lines for the polynomial interpolation, dashed line for analytical solution, asterisk marker for nodal solution and circle for boundary values will be consistent for all plots of this work.

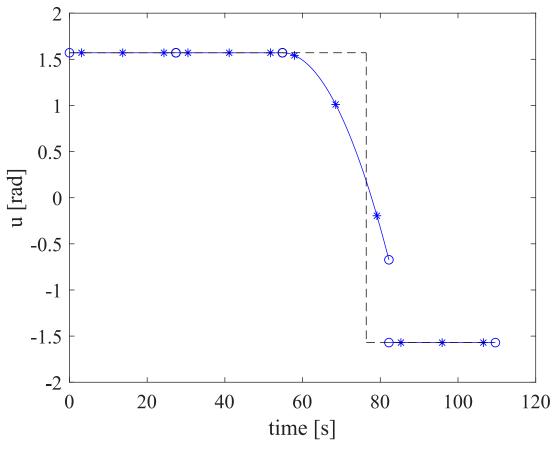
With progressively higher order the nodal solutions and the polynomial interpolations become sharper, but the oscillating behaviour does not disappear or decrease significantly. On the other hand, the solutions computed using Bernstein polynomials display no oscillation and the control solutions are completely within the feasible set. As the order increases, the nodal values converge more slowly to the analytical solution but remain feasible without oscillations. This also means that if one propagates Bernstein solution with a generic forward marching integrator, the resulting trajectory is expected to be close to the one computed with DFET.

Table 2.1 shows the number of iterations required by *fmincon* to reach feasibility, the sparsity (number of non-zero elements) of the Jacobian of the constraints of the NLP problem, the number of iterations to reach optimality and the final objective value. From Table 2.1 one can see that the sparsity of the Jacobian is identical for both Lagrange and Bernstein bases. The choice of the basis does not significantly influence the number of iterations needed to reach a feasible solution.

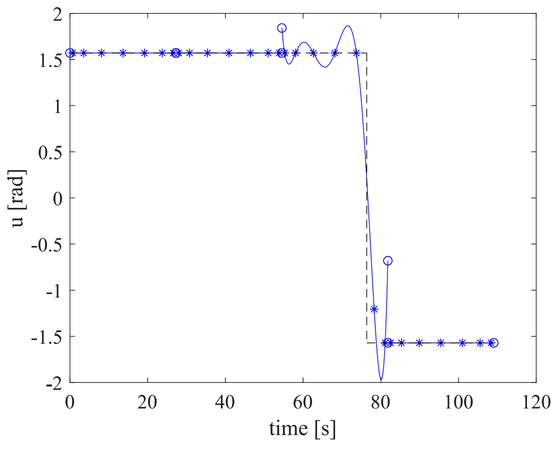
On the other hand the number of iterations required to converge to optimality is lower for Bernstein bases. For this problem the transcription with Bernstein basis takes on average 25% less iterations than the transcription with Lagrange basis. The convergence to the exact optimal cost function is instead slightly slower, with a relative difference which is always less than 0.4%. To be noted that although Lagrange basis always returns slightly lower objective values, these values oscillate around the



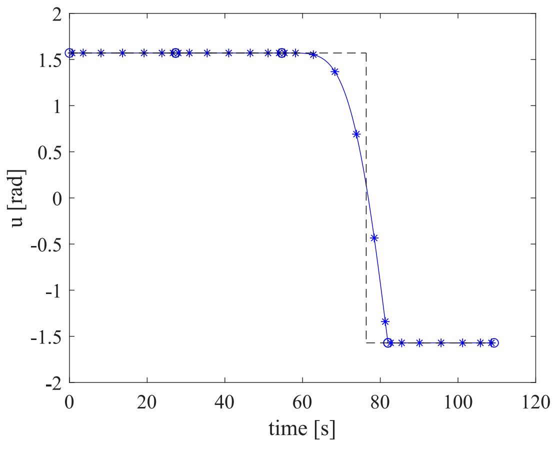
(a) Lagrange basis: order 2



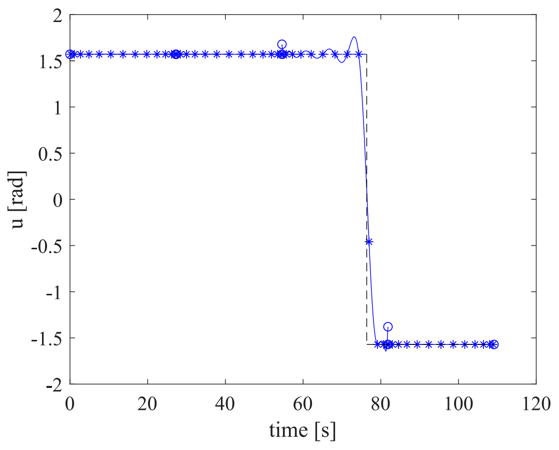
(b) Bernstein basis: order 2



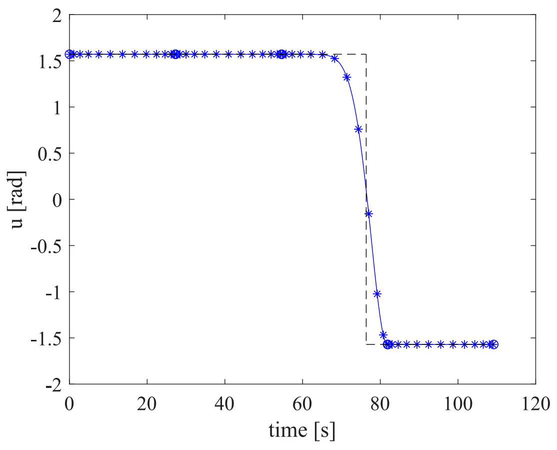
(c) Lagrange basis: order 6



(d) Bernstein basis: order 6



(e) Lagrange basis: order 12



(f) Bernstein basis: order 12

Figure 2.2: Problem Instance 1, Test case 1: Time-history of the controls.

Table 2.1: Results for Problem Instance 1, Test case 1

Order		Jacobian	Non zero elements of Jacobian		Feas iter		Opt iter		Objective value	
States	Control	size	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]
2	2	63x52	438 (13.4%)	438 (13.4%)	17	19	40	27	109.19	109.62
4	4	103x84	1046 (12.1%)	1046 (12.1%)	17	14	37	29	109.25	109.35
6	6	143x116	1910 (11.5%)	1910 (11.5%)	11	10	36	28	109.08	109.30
8	8	183x148	3030 (11.2%)	3030 (11.2%)	11	10	35	32	109.15	109.23
10	10	223x180	4406 (11.0%)	4406 (11.0%)	11	11	35	21	109.12	109.22
12	12	263x212	6038 (10.8%)	6038 (10.8%)	11	11	42	26	109.12	109.20
14	14	303x244	7926 (10.7%)	7926 (10.7%)	11	12	37	28	109.12	109.18

[a] Lagrange basis. [b] Bernstein basis.

exact one, while Bernstein basis monotonically converges to the exact solution. This monotonic improvement of the final objective value with Bernstein basis is a practical example of what was stated in Remark 2: the convex hull of the nodes for the controls is leaving out small portions of the real feasible set. As the order of the Bernstein basis increases, the number of nodes of the associated Bezier curves also increases and the convex hull of the nodes also captures greater regions of the feasible set.

Problem Instance 1, Test case 2

DFET is a flexible scheme that allows one to choose different polynomial orders for states and controls. Thus the second test case explores the behaviour of DFET in the case in which the order of the polynomials representing the controls is different (lower or higher) than the one of the polynomials representing the states.

In this test case the number of elements was kept fixed to 4 and the order of the states was kept constant to 6, but the order of the controls was varied from 1 to 24. The quadrature order was chosen in such a way that it could always integrate the highest order polynomials, thus $q = 7$ when the order of the polynomials for the control is lower than that of the states, and $q = l_u + 1$ for the other cases. For the Lagrange basis, this Test case also induces a mismatch between the nodes on which the polynomials of the states or of the controls are constructed, and the quadrature nodes. In particular, the quadrature nodes match with the nodes on which the states are defined until control order reaches 6, and match with the nodes on which the controls are defined from control order 6 onwards.

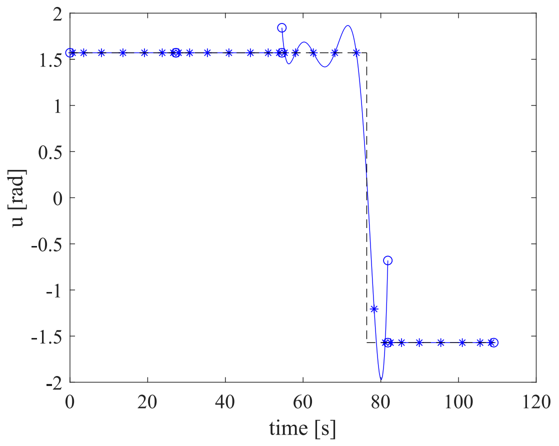
Table 2.2: Results for Problem Instance 1, Test case 2

Order		Jacobian	Non zero elements of Jacobian		Feas iter		Opt iter		Objective value	
States	Control	size	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]
6	1	123x116	1590 (11.1%)	1590 (11.1%)	11	10	33	22	109.93	110.18
6	2	127x116	1654 (11.2%)	1654 (11.2%)	11	10	33	31	109.35	109.65
6	3	131x116	1718 (11.3%)	1718 (11.3%)	11	11	33	28	109.25	109.41
6	6	143x116	1910 (11.5%)	1910 (11.5%)	11	10	36	28	109.08	109.30
6	8	151x116	2038 (11.5%)	2038 (11.6%)	11	10	37	27	109.15	109.23
6	10	159x116	2166 (11.7%)	2166 (11.7%)	10	13	31	26	109.12	109.22
6	12	167x116	2294 (11.8%)	2294 (11.8%)	10	13	35	28	109.12	109.20
6	16	183x116	2550 (12.0%)	2550 (12.0%)	12	12	37	29	109.11	109.18
6	20	199x116	2806 (12.1%)	2806 (12.1%)	11	21	34	25	109.11	109.16
6	24	215x116	3062 (12.2%)	3062 (12.2%)	12	12	38	32	109.11	109.15

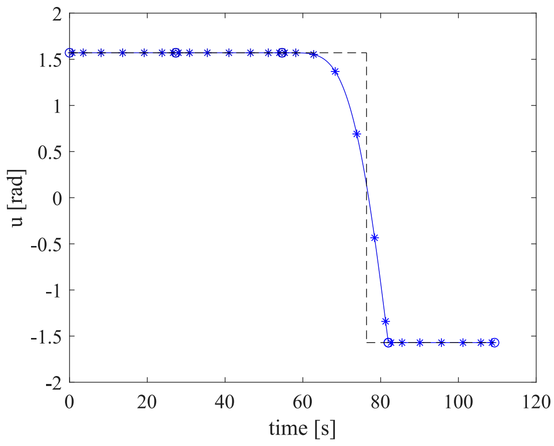
[a] Lagrange basis. [b] Bernstein basis.

Figure 2.3 shows that, as in test case 1, Lagrange basis display oscillations that do not disappear with the increasing order of the polynomials while Bernstein basis present a smoother and gradual approach to the exact solution. At the same time the nodal values, in the case of Bernstein basis, converge slower to the exact solution.

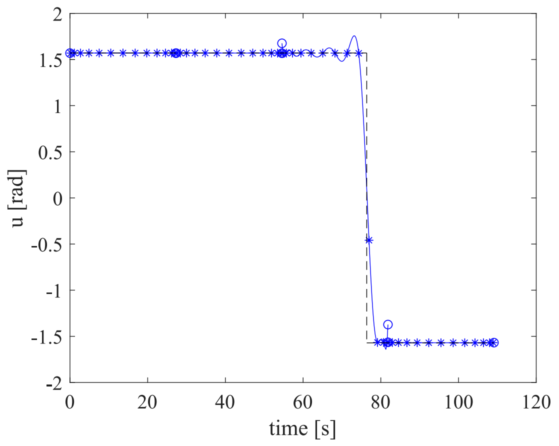
Table 2.2 shows sparsity of the Jacobian of the constraints, the number number of iterations to reach feasibility and optimality, and the final value of the cost function. The sparsity of the Jacobian of the constraints is identical for both bases, the number of iterations to converge to a feasible solution is not significantly affected by the choice of the basis, while the number of iterations required to converge to an optimal solution is. In particular, with Bernstein basis less iterations are needed to converge to the required optimality of the NLP solution and a monotonic convergence towards the analytical value as the order of the polynomials increases.



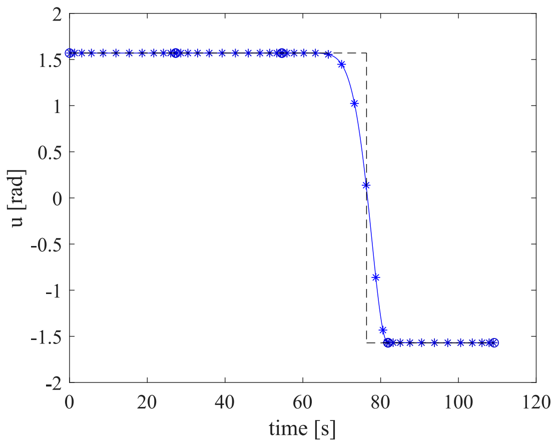
(a) Lagrange basis: states order 6, controls order 6



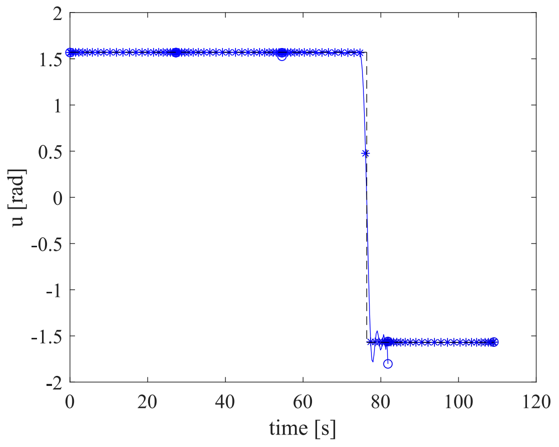
(b) Bernstein basis: states order 6, controls order 6



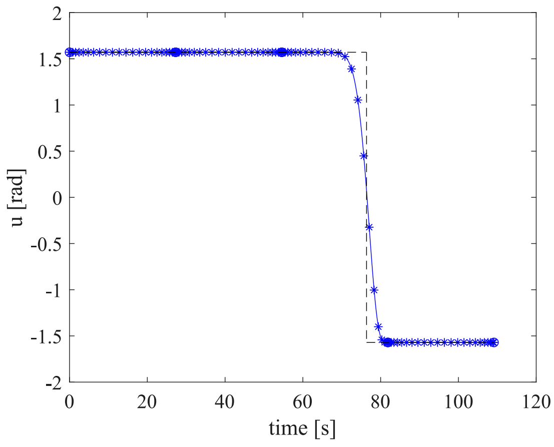
(c) Lagrange basis: states order 6, controls order 12



(d) Bernstein basis: states order 6, controls order 12



(e) Legendre basis: states order 6, controls order 24



(f) Bernstein basis: states order 6, controls order 24

Figure 2.3: Problem Instance 1, Test case 2: Time-history of the controls.

Problem Instance 1, test case 3

Test case 3 shows the effect of increasing the number of elements while keeping the orders of states and controls constant: the order of the polynomials was kept fixed to 6, while the number of elements was increased from 4 to 20. Integration was performed with Gauss-Legendre quadrature with $q = l_x + 1$.

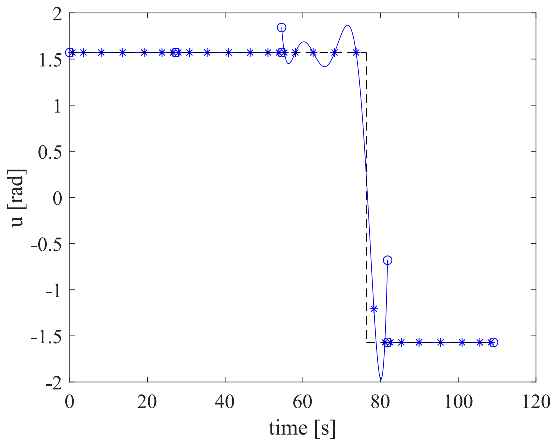
Figure 2.4 shows the controls profiles for three different refinements. In both cases, increasing the number of elements leads to capturing the discontinuity more accurately. With Lagrange basis, however, the oscillations increase at 12 elements and are still present, though moderate, at 20 elements, while Bernstein presents, once again, no oscillations.

Table 2.3 shows the sparsity of the Jacobians, the number of iterations required to converge to a feasible solution, the number of iterations required to converge to an optimal solution and the final objective value. As in previous cases, the sparsity is identical between the two bases. More interestingly, even with a fivefold increase of the number of elements, the number of iterations required to converge to a feasible solution is practically constant. Both bases converge monotonically to the correct final objective value, though Bernstein basis converges slightly slower.

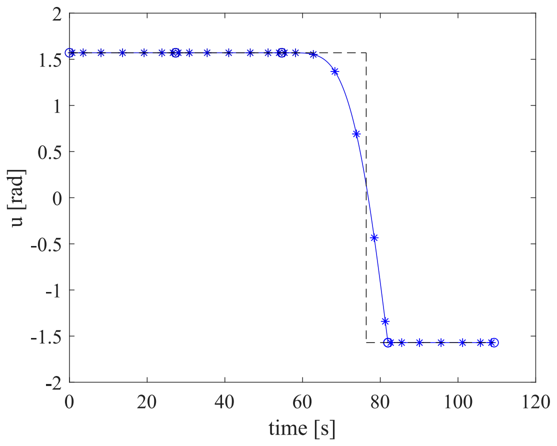
Table 2.3: Results for Problem Instance 1, Test case 3

Number of Elements	Jacobian size	Non zero elements of Jacobian L[a]	Non zero elements of Jacobian B[b]	Feas iter L[a]	Feas iter B[b]	Opt iter L[a]	Opt iter B[b]	Objective value L[a]	Objective value B[b]
4	143x116	1910 (11.5%)	1910 (11.5%)	11	10	36	28	109.08	109.30
8	283x228	3814 (5.9%)	3814 (5.9%)	13	10	33	28	109.10	109.17
12	423x340	5718 (4.0%)	5718 (4.0%)	11	11	37	35	109.10	109.14
16	563x452	7622 (3.0%)	7622 (3.0%)	14	11	46	34	109.11	109.12
20	703x564	9526 (2.4%)	9526 (2.4%)	14	11	40	45	109.11	109.11

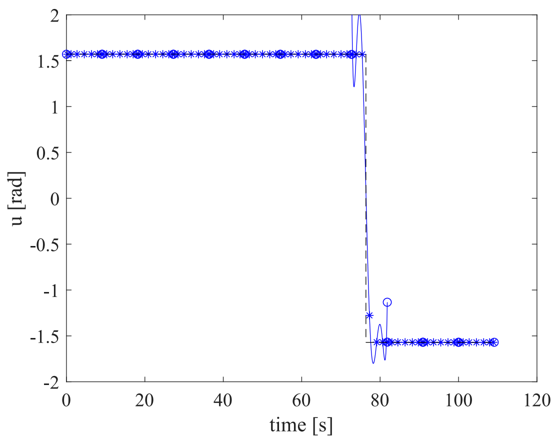
[a] Lagrange basis. [b] Bernstein basis.



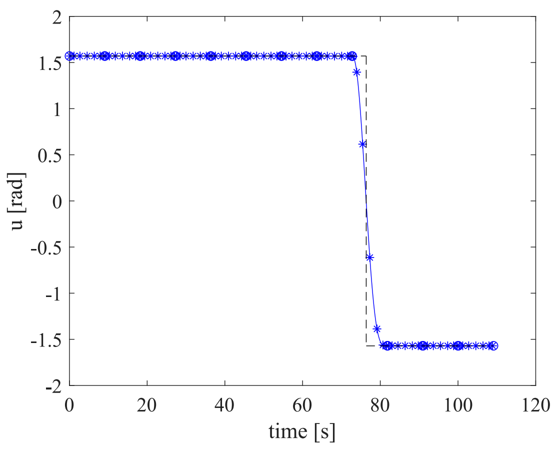
(a) Legendre 4 elements of order 6



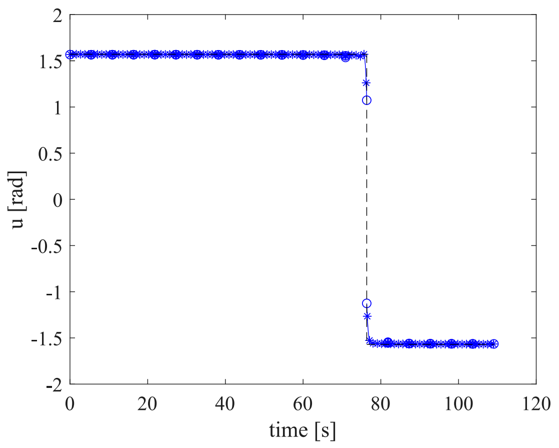
(b) Bernstein 4 elements of order 6



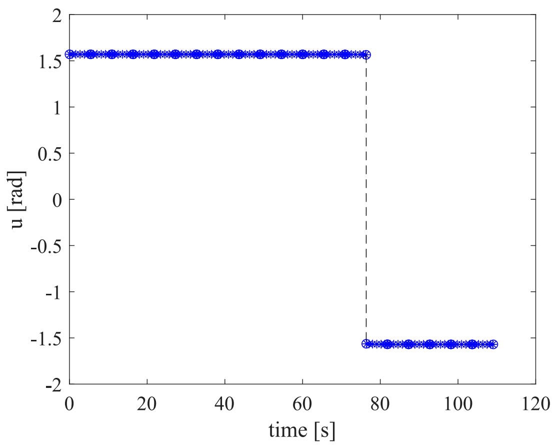
(c) Legendre 12 elements of order 6



(d) Bernstein 12 elements of order 6



(e) Legendre 20 elements of order 6



(f) Bernstein 20 elements of order 6

Figure 2.4: Problem Instance 1, Test case 3: Time-history of the controls.

2.3.2 Problem Instance 2 - Minimum Ascent Time with Constrained Vertical Velocity

This instance of the problem includes the following path constraint on the vertical velocity:

$$v_y(t) \leq 0.1 \qquad \forall 0 \leq t \leq t_f \tag{2.29}$$

The inclusion of this path constraint changes the profile of the vertical velocity from a triangular into a trapezoidal shape. Intuitively, the minimum time solution will start with a maximum vertical acceleration until switch time t_{s1} when the maximum allowed vertical velocity is reached. At that point, the controls will assume a constant value for which the vertical acceleration is zero. At time t_{s2}, a deceleration phase starts that brings the vertical velocity to 0 at time t_f. As a result of the constraint on the maximum vertical velocity, the objective function value is higher than for the previous instance of this problem. Moreover, a double discontinuity in the control profile is present.

The analytical control solution in this case is:

$$u(t) = \begin{cases} \frac{\pi}{2}, & 0 \leq t \leq \frac{V_{max}}{(a-g)} \\ \arcsin\left(\frac{g}{a}\right), & \frac{V_{max}}{(a-g)} \leq t \leq \frac{h(a+g)(a-g)-V_{max}^2 g}{V_{max}(a+g)(a-g)} \\ -\frac{\pi}{2}, & h - \frac{h(a+g)(a-g)-V_{max}^2 g}{V_{max}(a+g)(a-g)} < t \leq \frac{h(a+g)(a-g)-V_{max}^2 a}{V_{max}(a+g)(a-g)} \end{cases} \tag{2.30}$$

For the values of a, h, V_{max} and g used in this paper, the first switching time is $t_{s1} = 41.667$, the second switching time is $t_{s2} = 111.9048$ and the final time is $t_f = 129.7619$. This solution can be obtained in the same way as the analytic solution of the previous instance, by imposing continuity conditions across the switching points and solving for the switching times.

For the second instance, two tests sets are presented: in the first set the order of the polynomials of both states and controls are increased, while in the second set the order is constant but the number of elements is increased.

Problem Instance 2, test case 1

For test case 1 the time domain was discretised with 4 finite elements, and the order of the polynomials for both states and controls was varied from 2 to 14.

Table 2.4 shows that the number of iterations to reach a feasibility is very marginally lower for Bernstein, while the number of iterations to reach optimality is lower in the case of Bernstein basis except for order 14. As in the previous cases, Lagrange basis converge faster to the exact value of the cost function, while the Bernstein basis converges slower but monotonically. The sparsity of the Jacobian is not shown because identical to the previous instance of the problem.

Figure 2.5 shows the time history of u for both bases and different orders of the polynomials. While the control laws obtained with the Legendre basis are very oscillatory, especially when the path constraint is active, the variation diminishing property of Bernstein polynomials is here fundamental, because the resulting polynomial solution will oscillate less than the nodal solutions. In fact, while there are some small oscillations with Bernstein basis of order 2 and 6, which are due to a slightly oscillating nodal solution obtained by the NLP solver, the solution computed using Bernstein basis presents no oscillations at order 12.

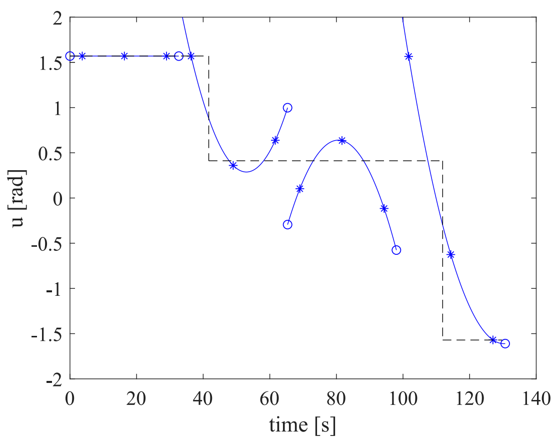
Generally, oscillations in the control profile are to be expected when path constraints are imposed on states and no higher order derivatives of the path constraint are provided to the NLP solvers. This is fundamentally due to an indeterminacy of the control profile that satisfies the path constraint. It is important to note that even if the value of the objective function converges faster for the Lagrange basis than for the Bernstein basis, as the order increases, the control law in Bernstein basis is very close to the analytic solution while the one in Lagrange basis is significantly different.

Table 2.4: Results for Problem Instance 2, Test case 1

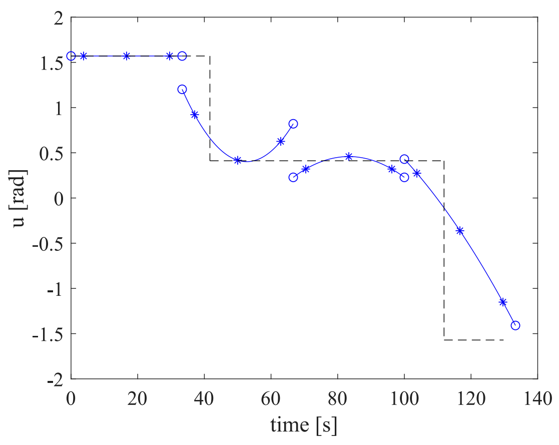
Order		Feas iter		Opt iter		Objective value	
States	Control	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]
2	2	14	13	28	22	130.68	133.33
4	4	16	13	36	24	129.87	131.43
6	6	15	12	25	24	129.96	130.98
8	8	15	12	31	27	129.85	130.67
10	10	18	12	37	28	129.83	130.53
12	12	16	12	42	27	129.79	130.40
14	14	17	12	40	88	129.78	130.33

[a] Lagrange basis. [b] Bernstein basis.

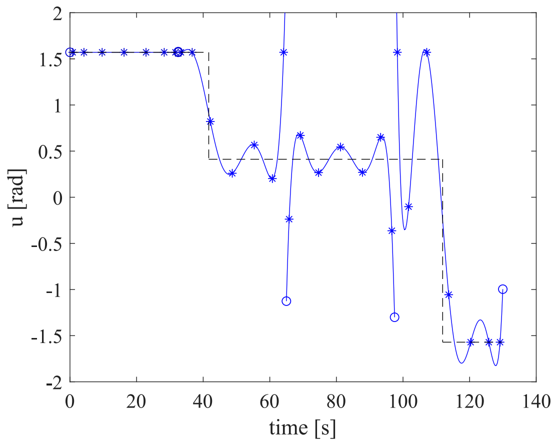
Figure 2.6 shows the time history of v_y. The solutions of order 2 present discontinuities in the states. These discontinuities are the sign that a polynomial solution of order 2 is inadequate to capture the dynamics. As the order of the polynomials increases, the profile of v_y converges to the analytical one (dashed line). With the Lagrange basis, although the nodal values converge faster than Bernstein, some infeasible oscillations in the states are present, while Bernstein basis remains feasible for all $0 \leq t \leq t_f$.



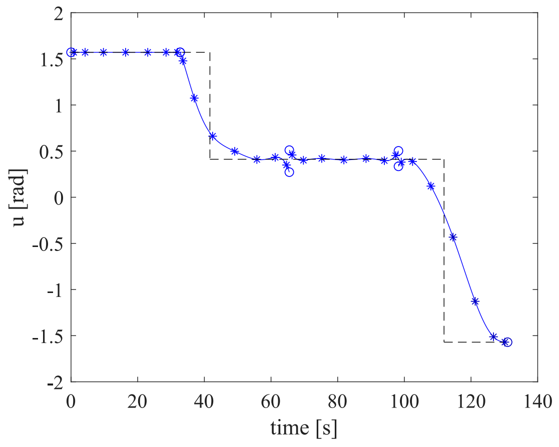
(a) Lagrange basis: 4 elements of order 2



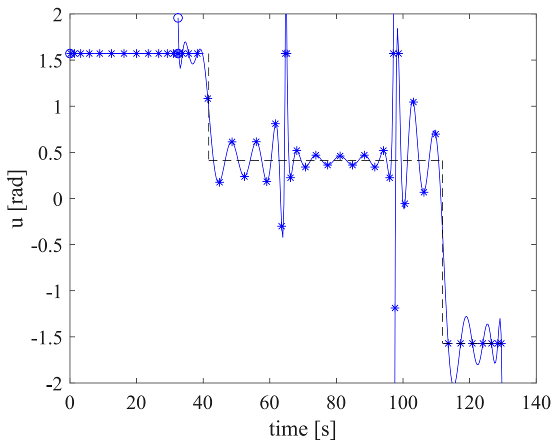
(b) Bernstein basis: 4 elements of order 2



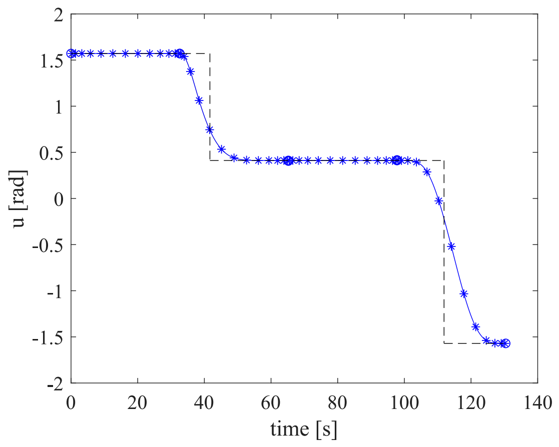
(c) Lagrange basis: 4 elements of order 6



(d) Bernstein basis: 4 elements of order 6

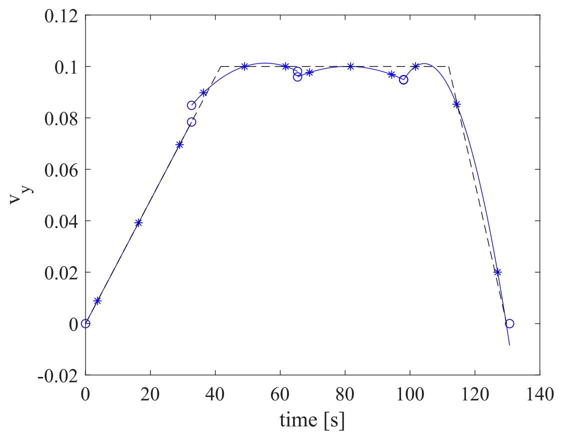


(e) Lagrange basis: 4 elements of order 12

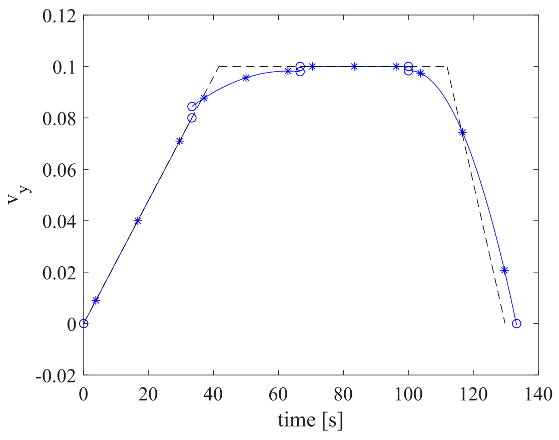


(f) Bernstein basis: 4 elements of order 12

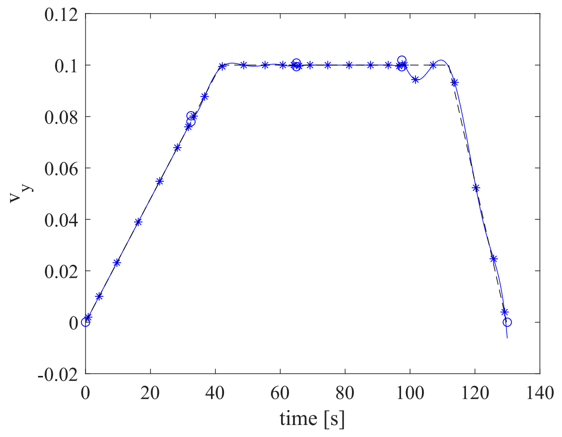
Figure 2.5: Problem Instance 2, Test case 1: Time-history of the controls.



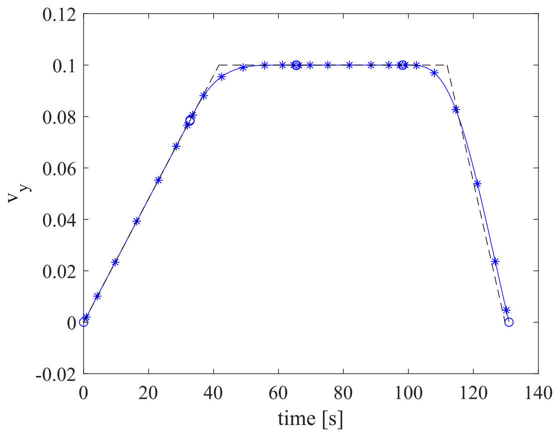
(a) Lagrange basis: 4 elements of order 2



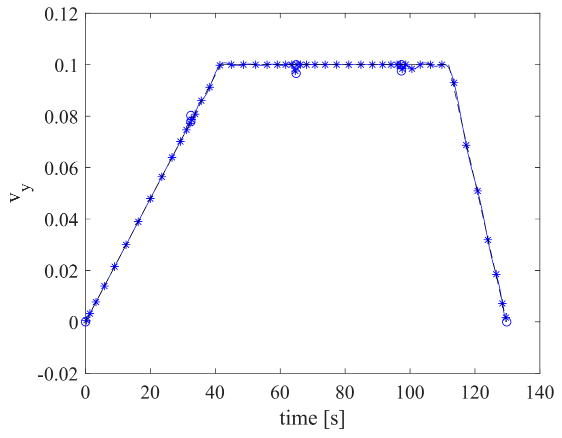
(b) Bernstein basis: 4 elements of order 2



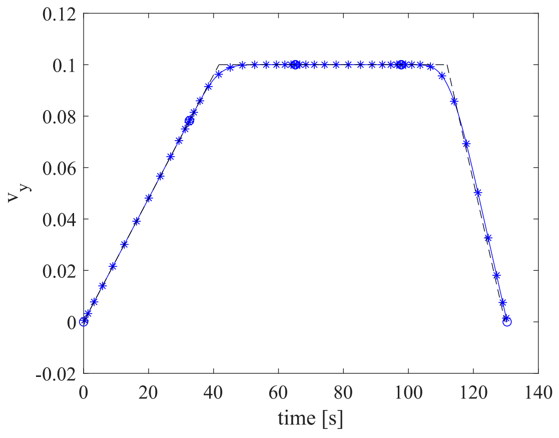
(c) Lagrange basis: 4 elements of order 6



(d) Bernstein basis: 4 elements of order 6



(e) Lagrange basis: 4 elements of order 12



(f) Bernstein basis: 4 elements of order 12

Figure 2.6: Problem Instance 2, Test case 1: Time-history of v_y.

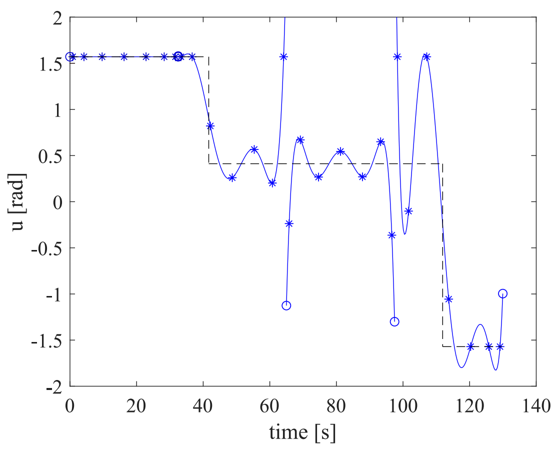
Problem Instance 2, test case 2

For test case 2, similarly to test case 3 of Instance 1, problem (2.22) with path constraint (2.29) is solved with an increasing number of finite elements but keeping their order constant.

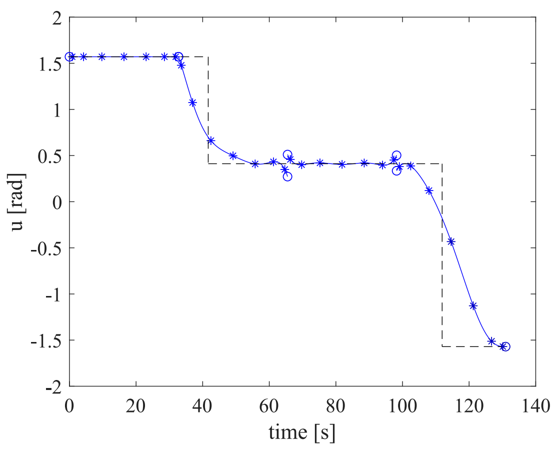
Table 2.5 shows that the choice of the basis impacts the number of iterations to achieve feasibility and optimality. In this case, Bernstein basis is cheaper in most of the cases, up to a factor of 2.5, but becomes more expensive, by a factor of almost 2, for high number of elements and converges slower to the exact value of the cost function. This will be further investigated in the next subsection.

Figure 2.7 shows the time histories of u as the number of elements is increased. For both bases, an increase in the number of elements leads to the formation of high frequency oscillations in the nodal solutions when the path constraint is active. The magnitude of these oscillations is however contained in the case of Bernstein and very high in the case of Lagrange. Similarly to the previous case, even if the value of the objective function converges faster for the Lagrange basis than for the Bernstein basis as the order is increased, the representation of the control law differs substantially: with the Bernstein basis it's very close to the analytic solution while with the Legendre case is very difficult to notice any resemblance, even if one looks only at the nodal values.

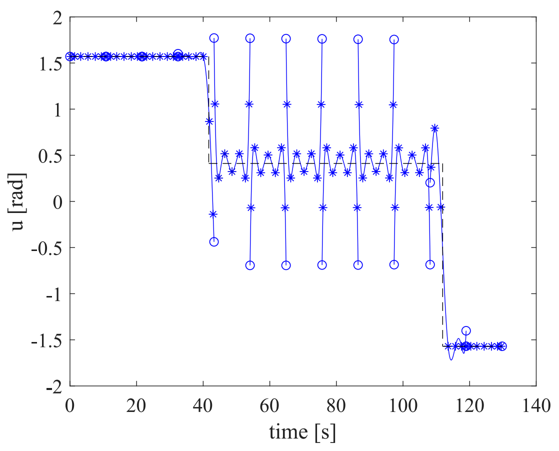
The time history of v_y is shown in figure 2.8. Both bases converge quickly to the exact solution though Lagrange presents some infeasible oscillations at low number of elements while Bernstein always remains within the feasible region for all $0 \leq t \leq t_f$.



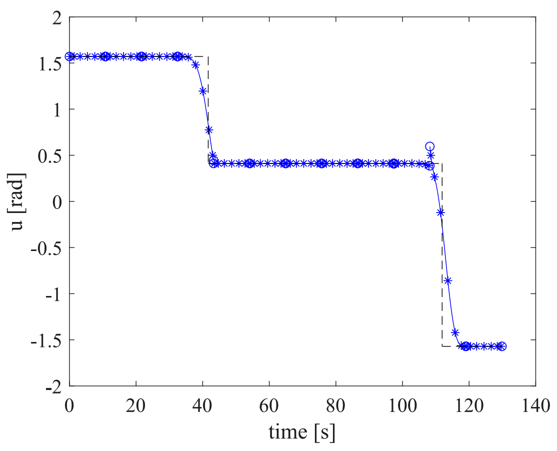
(a) Lagrange basis: 4 elements of order 6



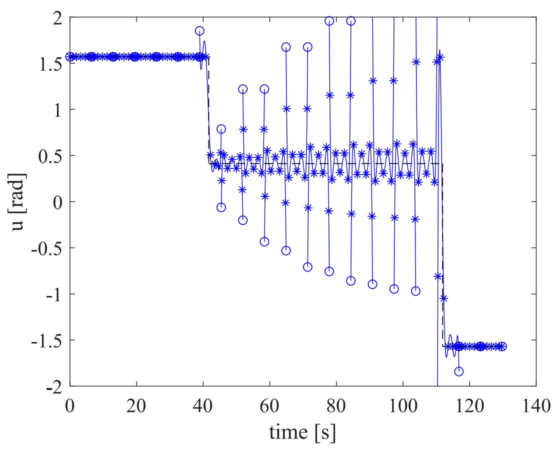
(b) Bernstein basis: 4 elements of order 6



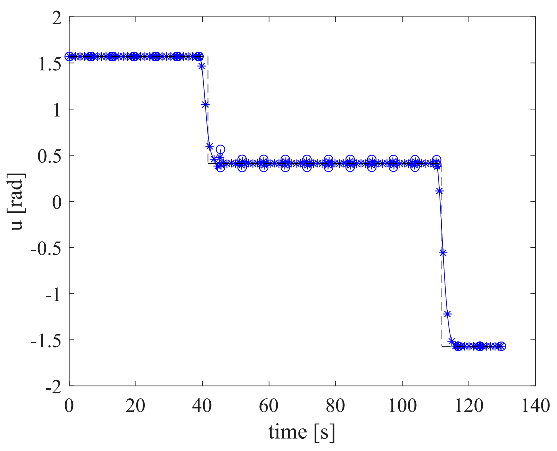
(c) Lagrange basis: 12 elements of order 6



(d) Bernstein basis: 12 elements of order 6

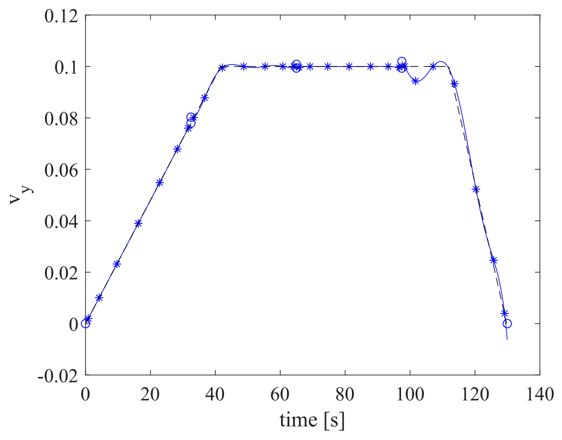


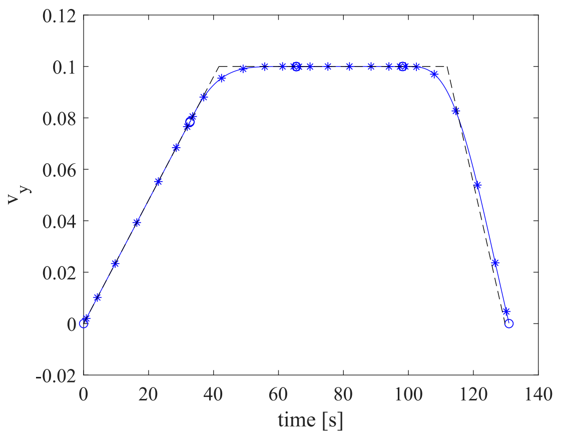
(e) Lagrange basis: 20 elements of order 6

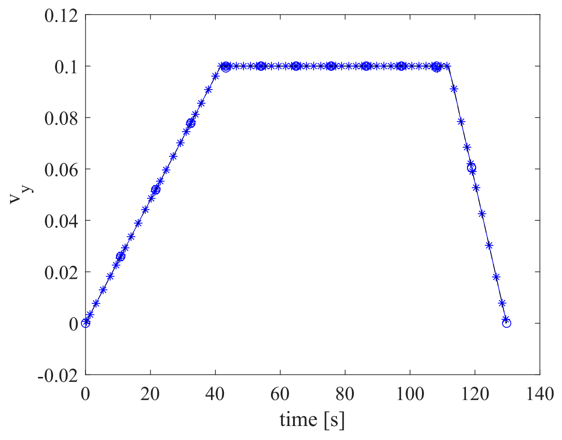


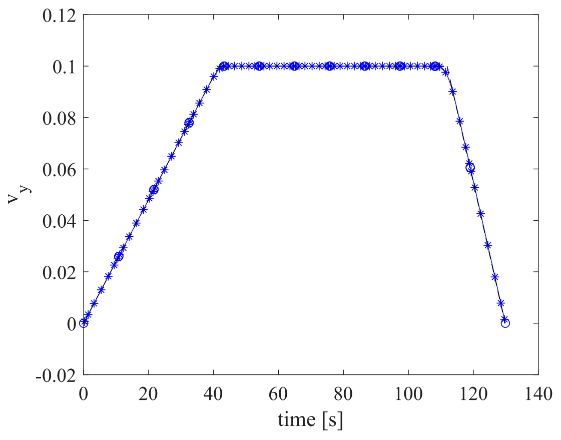
(f) Bernstein basis: 20 elements of order 6

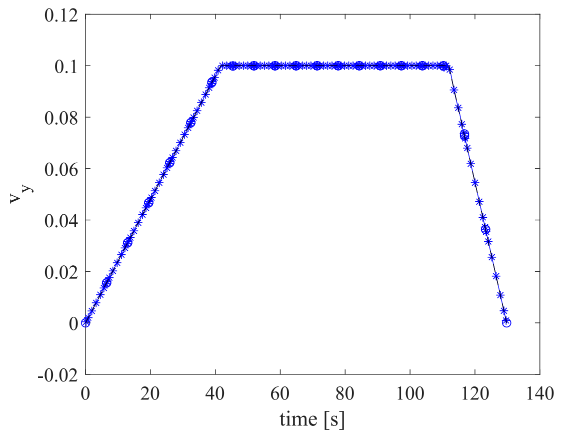
Figure 2.7: Problem Instance 2, Test case 1: Time-history of the controls.


(a) Lagrange basis: 4 elements of order 6


(b) Bernstein basis: 4 elements of order 6


(c) Lagrange basis: 12 elements of order 6


(d) Bernstein basis: 12 elements of order 6


(e) Lagrange basis: 20 elements of order 6

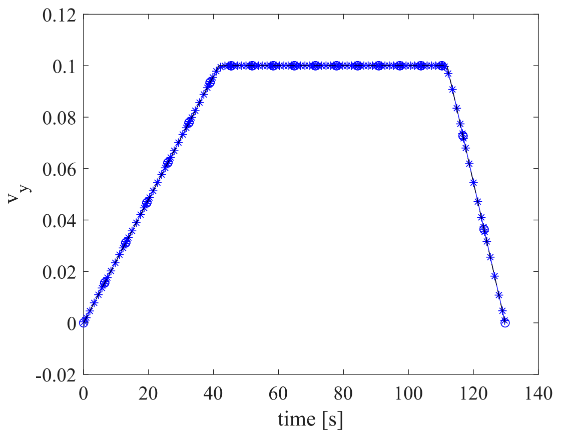

(f) Bernstein basis: 20 elements of order 6

Figure 2.8: Problem Instance 2, Test case 2: Time-history of v_y.

Table 2.5: Results for Problem Instance 2, Test case 2

Number of Elements	Feas iter		Opt iter		Objective value	
	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]
4	15	12	25	24	129.96	130.98
8	17	12	43	26	129.78	129.98
12	16	11	65	27	129.77	129.86
16	16	12	38	63	129.77	129.82
20	16	12	46	59	129.77	129.80

[a] Lagrange basis. [b] Bernstein basis.

2.3.3 Effect of the NLP algorithm

All test cases were solved again with the SQP algorithm to check if a different NLP algorithm would give different results in terms of convergence speed or value of the objective function, and to check and whether there are synergies between the use of a particular choice of basis functions and an NLP algorithm.

Tables 2.6 to 2.10 show the number of iterations to reach optimality and the final value of the objective function for both bases and NLP algorithms. In most cases the number of iterations remains fairly constant with respect to the size of the problem. For the Lagrange basis, the number of iterations required by the SQP algorithm is approximately 50% more than the Interior point algorithm, while for the Bernstein basis the SQP algorithm requires approximately 50% less iterations than the Interior point algorithm. Moreover, with a single exception using the Interior point algorithm in Test Instance 2 Case 1 and two in Test Instance 2 Case 2, the number of iterations required to converge to optimality is lower for Bernstein basis than for Lagrange basis, most of the times by a significant amount.

This shows an important interplay between the basis employed to represent the polynomials and the NLP algorithm. This interplay is due to the different way the problem is represented in the numerical approximation, and the lower number of iterations required by the Bernstein basis is very likely related to their superior numerical robustness as explained in [85].

Table 2.6: Performance comparison of the NLP algorithms for Problem Instance 1, Test case 1

Order		IP iterations		IP Objective value		SQP iterations		SQP Objective value	
States	Control	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]
2	2	40	27	109.19	109.62	51	8	109.19	109.62
4	4	37	29	109.25	109.35	54	16	109.25	109.35
6	6	36	28	109.08	109.30	50	21	109.08	109.30
8	8	35	32	109.15	109.23	71	10	109.15	109.23
10	10	35	21	109.12	109.22	41	9	109.19	109.22
12	12	42	26	109.12	109.20	32	11	109.12	109.20
14	14	37	28	109.12	109.18	37	12	109.12	109.18
Average		37.28	27.57			48.00	12.43		

[a] Lagrange basis. [b] Bernstein basis.

Table 2.7: Performance comparison of the NLP algorithms for Problem Instance 1, Test case 2

Order		IP iterations		IP Objective value		SQP iterations		SQP Objective value	
States	Control	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]
6	1	33	22	109.93	110.10	13	10	113.77	110.18
6	2	33	31	109.35	109.65	21	10	111.80	109.65
6	3	33	28	109.25	109.41	31	9	109.25	109.41
6	6	36	28	109.08	109.30	50	21	109.08	109.30
6	8	37	27	109.15	109.23	108	9	109.15	109.23
6	10	31	26	109.12	109.22	52	14	109.12	109.22
6	12	35	28	109.12	109.20	46	14	109.12	109.20
6	16	37	29	109.11	109.18	50	15	109.11	109.18
6	20	34	25	109.11	109.16	37	15	109.11	109.16
6	24	38	32	109.11	109.15	55	17	109.11	109.15
Average		37.40	27.60			46.30	13.40		

[a] Lagrange basis. [b] Bernstein basis.

Table 2.8: Performance comparison of the NLP algorithms for Problem Instance 1, Test case 3

Number of Elements	IP iterations		IP Objective value		SQP iterations		SQP Objective value	
	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]
4	36	28	109.08	109.30	50	21	109.08	109.30
8	33	28	109.10	109.17	54	15	109.10	109.17
12	37	35	109.10	109.14	62	15	109.10	109.14
16	46	34	109.11	109.12	54	18	109.11	109.12
20	40	45	109.11	109.11	64	23	109.11	109.11
Average	38.40	34.00			56.80	18.40		

[a] Lagrange basis. [b] Bernstein basis.

Table 2.9: Performance comparison of the NLP algorithms for Problem Instance 2, Test case 1

Order		IP iterations		IP Objective value		SQP iterations		SQP Objective value	
States	Control	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]
2	2	28	22	130.68	133.33	19	10	130.68	133.33
4	4	36	24	129.87	131.43	12	10	130.70	131.43
6	6	25	24	129.96	130.98	66	11	129.96	130.98
8	8	31	27	129.85	130.67	62	14	129.85	130.67
10	10	37	28	129.83	130.53	40	16	129.83	130.53
12	12	42	27	129.79	130.40	66	20	129.79	130.40
14	14	40	88	129.78	130.33	154	21	130.00	130.33
Average		34.14	34.29			59.86	14.57		

[a] Lagrange basis. [b] Bernstein basis.

Table 2.10: Performance comparison of the NLP algorithms for Problem Instance 2, Test case 2

Number of Elements	IP iterations		IP Objective value		SQP iterations		SQP Objective value	
	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]	L[a]	B[b]
4	25	24	129.96	130.98	25	24	129.96	130.98
8	43	26	129.78	129.98	58	8	129.78	129.98
12	65	27	129.77	129.86	30	14	129.77	129.86
16	38	63	129.77	129.82	112	32	129.77	129.81
20	46	59	129.77	129.80	62	23	129.77	129.79
Average	46.17	39.83			62.00	19.50		

[a] Lagrange basis. [b] Bernstein basis.

2.3.4 Sparsity and NLP solvers

The sparsity of the Jacobian is an important factor affecting the computational cost required to solve an NLP problem. The reason for this is that if a Jacobian is sparse, it does not need to be stored explicitly as a full array and it is possible to employ sparse linear algebra tools. Not needing to store the full matrix means that it is possible to solve larger problems for a given memory budget. Additionally, a smaller Jacobian could allow the problem to fit in the cache memory instead of the ordinary RAM, allowing for significant speedups. Sparse linear algebra tools also allow to efficiently perform matrix operations, skipping many multiplications and sums of null terms.

The sparsity of the Jacobian however is not the only important factor, since the same sparsity level can be obtained with different patterns. The implementation of DFET here described was carefully studied in order to maintain as much as possible a block diagonal arrangement with narrow band. This is an important characteristic because, for example, the inverse of a block diagonal matrix is itself a block diagonal matrix where each new block is the inverse of an original block, thus preserving the sparsity structure of the problem and speeding up the computation of the inverse. Even if no matrix inversion is actually performed by the NLP algorithm, high quality linear algebra software, with specialised factorisation and matrix multiplication routines, exploits several of those properties.

Different NLP algorithms also exploit sparsity differently: in the documentation of *fmincon* it can be found that the interior point algorithm is considered a large scale algorithm while the SQP is considered a medium scale algorithm [94]. This classification stems from the fact that the interior point algorithm implemented in Matlab can have a user supplied Hessian, and if this Hessian is sparse, sparse linear algebra routines will be employed. If the Hessian is not supplied, a BFGS approximation is used instead. The SQP algorithm on the other hand is not classified as large scale because it does not accept a Hessian, which is instead internally computed from the gradients and Jacobians using a Quasi-Newton approximation, and it always employs dense linear algebra routines. This is a particular choice of the implementation of SQP adopted in *fmincon*: for example, the NLP solver WORHP [95, 96] is a large scale sparse implementation of the SQP algorithm.

Summarising, the same NLP algorithm can be implemented in different ways, particularly regarding the use of sparse linear algebra routines. A sparse implementation allows to solve larger problems for the same memory budget, and should also perform faster since many operations are not executed. This seems to contradict the results discussed in the previous section. The implementation of the NLP algorithm thus probably plays a major role on these interactions. It is reasonable to assume that the observed faster convergence obtained by using SQP in conjunction with the Bernstein basis will persist even when using a sparse SQP implementation like the one of WORHP, since the sparsity patterns of the two bases are identical, the only difference between them being in the actual values of the Jacobian.

2.4 Chapter summary

This chapter presented the DFET transcription for the solution of optimal control problems. It was shown that with the use of Bernstein basis, states and controls are represented by Bezier curves, from which they inherit the convex hull and variation diminishing properties.

A Theorem was proved, stating that, under some general convexity conditions about the feasible region, this transcription scheme can guarantee that path constraints will be satisfied everywhere in the time domain and not only at the quadrature nodes. These properties allow one to have an everywhere feasible representation of the controls even in the case of a bang-bang or bang-zero-bang switching structure. This fact was experimentally demonstrated with two different instances of a known optimal control problem.

It was also shown that the sparsity of the Jacobian of the constraint of the resulting NLP problem is comparable to the one of DFET transcription schemes based on Lagrange polynomials on spectral basis. For this problem the convergence of the NLP solver was faster than in the case of Lagrangian basis, however the convergence rate towards the exact value of the cost function was slower. Furthermore, a slower convergence of the nodal values to the exact solution was registered, as expected, given the slower convergence of Bernstein polynomials.

This slower convergence rate is expected to be present also in the case of continuous solutions, as demonstrated by other authors, but in the case of jump discontinuities the convergence rate is comparable because of the unwanted oscillations of Lagrange bases, in the neighbourhood of the discontinuity, for times different from the nodal values, that reduces the convergence rate to $O(1)$.

On the the other hand, when path constraints on the states were imposed, the use of Bernstein bases yielded solutions that displayed little or no oscillations of the controls along the path constraint even if no higher order derivatives were provided to the NLP solver. In this case, the nodal solutions were also closer to the analytical solution than the nodal solutions obtained with the Lagrange basis.

The smooth behaviour of Bernstein polynomials is retained when different orders are used for states and controls. This property allows for flexible h/p adaptivity schemes which independently adapt the order of states and controls in each element and avoid infeasible oscillations when the solution is interpolated on a new set of nodes.

Finally, an interesting interplay between the polynomial basis and the NLP algorithm was shown. In most cases, the use of Bernstein basis required less iterations to converge to an optimal solution than the Lagrange basis. Depending on the NLP algorithm and size of the problem, this difference was more or less significant, reaching a maximum of one order of magnitude. This improved convergence rate of the NLP solver could be related to the superior numerical robustness of Bernstein polynomials, previously demonstrated by other authors. A theoretical quantification of the convergence rate of the proposed method is left for future work, and could help explain the observed behaviour.

Chapter 3

Multi-objective optimisation [1]

Omnis ars naturae imitatio est

All art is an imitation of nature

Seneca

This chapter introduces multi-objective optimisation and presents the Multi Agent Collaborative Search algorithm (MACS), a global memetic multi-objective optimisation algorithm which will be extensively used in the rest of this thesis. Several variants and iterations of MACS exist in the literature [33, 34, 35, 36]. A new variant of MACS is here introduced, with modified heuristics and a new solution archiving strategy. The modified heuristics are here presented and tested against benchmark problems, showing some performance improvement over the previous version of MACS [35]. A new archiving strategy is also introduced, and it is shown how its adoption results in better spread Pareto fronts for both MACS and another algorithm commonly employed for multi-objective optimisation.

[1] The contents of this chapter were published in: L. A. Ricciardi, M. Vasile, Improved Archiving and Search Strategies for Multi Agent Collaborative Search. Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences. Springer, 2019.

3.1 The multi-objective optimisation problem

Mathematically, a multi-objective optimisation problem can be formulated as:

$$\min_{\mathbf{x}\in D} \mathbf{f}(\mathbf{x}) \tag{3.1}$$

where D is a hyperrectangle defined as $D = \{x_j | x_j \in [b_j^l \ b_j^u] \subseteq \mathbb{R},\ j = 1, ..., n\}$ and $\mathbf{f}$ is the vector function:

$$\mathbf{f} : D \to \mathbb{R}^m, \qquad \mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_m(\mathbf{x})]^T \tag{3.2}$$

The optimality of a particular solution is defined through the concept of dominance: with reference to problem (3.1), a vector $\mathbf{y} \in D$ is dominated by a vector $\mathbf{x} \in D$ if $f_l(\mathbf{x}) \leq f_l(\mathbf{y})$ for all $l = 1, ..., m$ and there exists k so that $f_k(\mathbf{x}) \neq f_k(\mathbf{y})$. The relation $\mathbf{x} \prec \mathbf{y}$ states that $\mathbf{x}$ dominates $\mathbf{y}$. A decision vector in D that is not dominated by any other vector in D is said to be Pareto optimal. All non-dominated decision vectors in D form the Pareto set D_P and the corresponding image in criteria space is the Pareto front.

Starting from the concept of dominance, it is possible to associate, to each solution in a finite set of solutions, the scalar dominance index:

$$I_d(\mathbf{x}_i) = |\{i^* \mid i, i^* \in N_p \wedge \mathbf{x}_{i^*} \prec \mathbf{x}_i\}| \tag{3.3}$$

where the symbol $|.|$ is used to denote the cardinality of a set and N_p is the set of the indices of all the solutions. All non-dominated and feasible solutions $\mathbf{x}_i \in D$ with $i \in N_p$ form the set:

$$X = \{\mathbf{x}_i \in D \mid I_d(\mathbf{x}_i) = 0\} \tag{3.4}$$

The set X is a subset of D_P, therefore, the solution of problem (3.1) translates into finding the elements of X. If D_P is made of a collection of compact sets of finite measure in $\mathbb{R}^n$, then once an element of X is identified it makes sense to explore its neighbourhood to look for other elements of X. On the other hand, the set of non-dominated solutions can be disconnected and its elements can form islands in D. Hence, multiple parallel exploration can increase the collection of elements of X.

3.1.1 Tchebycheff Scalarisation

Another approach for the solution of problem (3.1), known as Tchebyceff Scalarisation, is to solve a number of scalar optimization problems in the form:

$$\min_{\mathbf{x}\in D} g(\mathbf{f}(\mathbf{x}), \boldsymbol{\omega}, \mathbf{z}) = \min_{\mathbf{x}\in D} \max_{l=1,...,m} \{\omega_l |f_l(\mathbf{x}) - z_l|\} \tag{3.5}$$

where $\mathbf{z} = [z_1, ..., z_m]^T$ is the reference objective vector whose components are $z_l = \min_{\mathbf{x}\in D} f_l(\mathbf{x})$, for $l = 1, ..., m$, and ω_l is the l-th component of a weight vector $\boldsymbol{\omega}$.

By solving a number of problems (3.5), with different weight vectors, one can obtain different Pareto optimal solutions. Although the final goal is always to find the set X, using the solution of problem (3.5) or index (3.3) has substantially different consequences on the way samples are generated and selected.

In [33, 34, 35, 36] a version of MACS, called MACS2, was introduced, combining Pareto dominance and Tchebycheff scalarisation to select potential improvements towards the Pareto front. The effectiveness of this approach was tested on different benchmark and challenging real problems: since MACS2 was successfully used for the design of space missions for the removal of space debris by means of low-thrust, many revolutions orbits, and for the design of the initial, low-thrust rising phase for the technology demonstrator mission DESTINY. Both are real engineering multi-objective optimisation problems for which no previous solution was known, and involved the concurrent minimisation of fuel consumption, mission time, and, for the DESTINY mission, radiation exposition time.

3.2 Multi Agent Collaborative Search

The key idea underneath MACS is to combine local and global search in a coordinated way such that local convergence is improved while global exploration is retained [97]. This combination of local and global search is achieved by endowing a set of agents with a repertoire of actions producing either the sampling of the whole search space or the exploration of a neighbourhood of each agent. In this section, the key heuristics underneath MACS will be described in detail.

3.2.1 Algorithm Description

In MACS, a population of virtual agents is deployed at random locations in the search space. Each agent locally explores its neighbourhood performing a set of local search actions, also named individual actions. Then the population as a whole performs a set of social actions, to concurrently advance towards the front. An external archive is used to store the current best representation of the Pareto set. Individual actions, social actions and archive update are repeated until a stopping criterion is met.

With reference to Algorithm 1, MACS2.1 starts by initialising a population of n_{pop} agents at random locations within the search domain D with a Latin Hypercube Sampling. Non-dominated agents are copied in the archive A to form the first approximation of the Pareto set. The archive A has specified maximum size max_{arch}.

A set of n_ω m-dimensional unit vectors ω_k (defining m directions in criteria space) is then generated.The first m ω_k vectors form a base in $\Re^m$, so that the solution vectors that best optimise each individual objective function are always in the final approximation of the Pareto set. A user-defined fraction p_{social} of agents is specified, and each social agent is then associated to the scalarised sub-problem it solves best, and to the corresponding weight vector ω_k (see Line 9).

After initialising the velocity $\mathbf{V}_i$ of each agent (Line 10) the main loop starts (Line 12). Until a maximum number of function evaluations is reached, the agents first perform local search actions, described in the next section and Algorithm 2, to move towards the Pareto set. A local action is considered successful when it generates a dominating solution or a solution that satisfies the Tchebycheff scalarisation criterion corresponding to the particular sub-problem associated to the agent.

After all local actions have been completed, the archive A is updated with all non-dominated solutions. A total of n_{social} agents then perform the social actions described in Section 3.2.1 and Algorithm 3. The archive A is updated again with all non-dominated solutions.

Algorithm 1 MACS2.1

```
1: Set $n_{feval,max}$, $max_{arch}$, $n_{pop}$, $p_{social}$, $F$, $CR$, $\rho_{ini}$, $\rho_{contr}$, $\rho_{max,contr}$
2: Set $n_{social} = n_{pop} p_{social}$
3: Set iteration counter $h = 0$
4: Initialise population $P_h, n_{feval} = 0$
5: Initialise neighbourhood size $\rho_i = \rho_{ini}$ $\forall i \in \{1, .., n_{pop}\}$
6: Insert the non-dominated elements of $P_0$ in the archive $A$
7: Initialise $n_\omega$ vectors $\omega_k$ for $k \in \{1, .., n_\omega\}$ such that $||\omega_k|| = 1$
8: Initialise the vector of agents' velocities $\mathbf{V}_i = 0$ $\forall i \in \{1, .., n_{pop}\}$
9: while $n_{feval} < n_{feval,max}$ do
10:     h=h+1
11:     update number of directions to scan in pattern search
12:     Perform local actions through Algorithm 2
13:     Update archive $A$ with non-dominated elements through Algorithm 4
14:     Perform social actions through Algorithm 3
15:     Update archive $A$ with non-dominated elements through Algorithm 4
16: end while
```

Individualistic actions

In the following the individualistic search actions are described. These individualistic actions are specific of this version of MACS. Each agent has a repertoire of three different actions, namely: *inertia, pattern search and differential evolution.* Each agent performs each action sequentially until an improvement is registered (i.e. the algorithm generates either a dominant solution or a solution that satisfies Tchebycheff criterion, if the agent is associated to a ω_k). The pseudo-code is given in Algorithm 2.

Inertia If the previous moves defined a search direction $\mathbf{V}_i$ in parameter space, inertia generates a new sample in the same direction (lines 3-8). The trial position for the $i-th$ agent, $\mathbf{x}_{trial}$ is, defined as:

$$\mathbf{x}_{trial} = \mathbf{x}_i + \alpha \mathbf{V}_i \tag{3.6}$$

where α is a random number between 0 and 1. In MACS2.1, if $\mathbf{x}_{trial}$ is outside the admissible domain D, α is contracted with a simple backtracking procedure so that $\mathbf{x}_{trial}$ falls on the boundary of D. In the case a number of components of $\mathbf{x}_i$ lower than n is already equal to either their lower or upper limit and $\mathbf{x}_i + \alpha \mathbf{V}_i$ is outside D, then the corresponding components of $\mathbf{V}_i$ are set to zero before the backtracking procedure is applied. This correction was introduced to improve the exploration of the boundary of the search space, since without it the backtracking would in this case impose $\alpha = 0$. Figure 3.1 shows for examples of this strategy: in one case inertia is applied without modifications, in one case α is reduced in order to avoid sampling outside the allowed region, in a third case one component of $\mathbf{V}$ is removed because the initial position is already on the boundary, and in the fourth case one component of $\mathbf{V}$ is removed and α is also reduced in order not to sample outside of the bounds.

Pattern Search If inertia gave no improvement or was not performed ($\mathbf{V}_i = 0$), a simple pattern search strategy is implemented. This heuristic changes only one randomly chosen component j of $\mathbf{x}_i$ at a time (lines 14-20). The trial position $\mathbf{x}_{trial}$ is thus equal to $\mathbf{x}_i$, except for the $j-th$ component, which is:

$$x_{trial,j} = x_{ij} + \alpha \Delta_j \rho_i \tag{3.7}$$

where this time α is a random number between -1 and 1, Δ_j is the difference between the upper and lower boundaries for variable j and ρ_i defines the size of a hyper-rectangle centred in $\mathbf{x}_i$. If direction $\alpha \Delta_j \rho_i$ is not successful, the opposite direction $-sign(\alpha)\beta\Delta_j\rho_i$ is attempted with β a random number between 0 and 1. If also this move fails, a new random direction (different from the previous ones) is chosen.

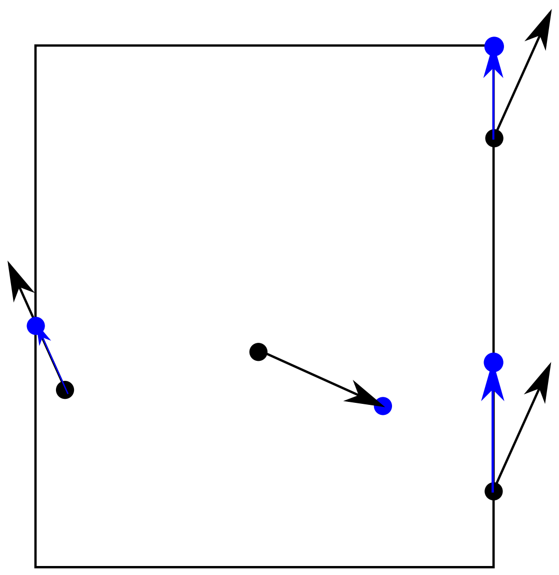


Figure 3.1: Inertia move with reduction of α. Black dots and black arrows indicate initial positions and $\mathbf{V}$, blue dot indicate final positions, blue arrow indicate modified $\mathbf{V}$

This strategy is repeated until either an improvement is found (i.e. a dominant solution is generated or Tchebycheff criterion is satisfied), or a specified maximum number of directions has been explored. In MACS2.1, the maximum number of directions is dynamically adjusted as

$$max\_dirs = round\left(n - (n-1)\frac{curr\_arch\_size}{max\_arch\_size}\right) \tag{3.8}$$

where max_dirs is the maximum number of dimensions to scan, n is the number of coordinates, $curr\_arch\_size$ is the current size of the archive and max_arch_size is the specified maximum size of the archive (lines 11-21). If a good sample is found, it is used to compute vector $\mathbf{V}_i$.

Differential Evolution If pattern search did not lead to an improvement, a differential evolution step is taken, by combining vector $\mathbf{x}_i$ with 3 randomly chosen agents $\mathbf{x_{i_1}}, \mathbf{x_{i_2}}$ and $\mathbf{x_{i_3}}$ (lines 24-28). The displacement vector is then given by:

$$\mathbf{dx}_i = \alpha\mathbf{e}\left((\mathbf{x}_i - \mathbf{x}_{i_1}) + F\left(\mathbf{x}_{i_2} - \mathbf{x}_{i_3}\right)\right) \tag{3.9}$$

where α is a random number between 0 and 1, F is a user specified constant and $\mathbf{e}$ is a mask vector whose elements are either 0 or 1 as follows:

$$e_j = \begin{cases} 1, & \text{if } \alpha_2 < CR \\ 0, & \text{otherwise} \end{cases} \tag{3.10}$$

where α_2 is a random number between 0 and 1, and CR is another user specified constant. The trial position for the differential evolution move finally reads:

$$\mathbf{x}_{trial} = \mathbf{x}_i + \mathbf{dx}_i \tag{3.11}$$

To ensure this new position is inside the domain, in MACS2.1 the same strategy as for the inertia case is employed: eventually suppressing some components of $\mathbf{dx}_i$ and reducing α.

Local neighbourhood size management If all local actions have failed, the local neighbourhood size ρ_i is reduced by a user defined factor ρ_{contr}. After a user defined maximum number of contractions $\rho_{max,contr}$, ρ_i is reset to ρ_{ini}. Conversely, if one action is successful, in MACS2.1 ρ_i is increased by a factor ρ_{contr}, up to the maximum value ρ_{ini} (lines 30-37).

Social actions

Social actions are implemented following the same principle as in Zuiani and Vasile [35]: a fraction p_{social} of the total population of the agents implements a DE type of heuristic by picking agents either from the population or from the archive (lines 4 or 6). The probability of picking agents from the archive or from the population is determined by:

$$p_{arch\_vs\_pop} = 1 - e^{-\frac{current\_size\_archive}{num\_agents}} \tag{3.12}$$

In MACS2, with Social Actions, each social agent was immediately moved to the trial position if the trial position satisfied the Tchebycheff criterion. In MACS2.1, instead, the archive is first updated with all trial vectors that are non-dominated by

Algorithm 2 Individualistic actions

```
for $i = 1 : n_{pop}$ do
  Set improved=FALSE
  if $||\mathbf{V}_i|| \neq 0$ then
    Perform Inertia move
    if successful then
      set improved=TRUE
    end if
  end if
  if not improved then
    counter=0
    while counter$\leq$ $max\_pat\_search\_dirs$ & not improved do
      counter=counter+1
      Pick random direction
      Perform Pattern Search
      if successful then
        set improved=TRUE
        set $\mathbf{V}_i = x_{i,old} - x_i$
      end if
    end while
  end if
  if not improved then
    Perform Differential Evolution
    if successful then
      set improved=TRUE
    end if
  end if
  if not improved then
    Contract $\rho_i$
    if $\rho_i$ has contracted more than $\rho_{max,contr}$ times then
      $\rho_i = \rho_{ini}$
    end if
  else
    Expand $\rho_i$ unless this would cause $\rho_i$ to be greater than $\rho_{ini}$
  end if
end for
```

any other element of the archive. After the archive is updated, each agent performing social actions is then moved to the location of the element of the archive that best improves the corresponding Tchebycheff sub-problem, unless that location is already occupied by another agent (lines 15-22).

This new heuristic better exploits the information in the archive and at the same time does not exclude non-dominated trial vectors that do not satisfy the Tchebycheff condition before checking the content of the archive. The pseudo-code for social actions is given in Algorithm 3.

Algorithm 3 Social actions

```
1: choose random number $r$ between 0 and 1
2: compute $p = 1 - e^{-\frac{curr\_arch\_size}{num\_agents}}$
3: if $r \leq p$ then
4:    perform DE between social agents and random points from archive
5: else
6:    perform DE between social agents and random points from current population
7: end if
8: add candidate solutions in archive through Algorithm 4
9: if there are at least as many agents in the archive as objective functions then
10:    if there's exactly as many agents in the archive as objective functions then
11:       $n_{move}$ = num objective functions
12:    else
13:       $n_{move}$ = min(num agents in archive, num agents performing social actions)
14:    end if
15:    create pool of $n_{move}$ agents to be moved. Agents following exclusively one of the
       objectives are always chosen
16:    move the $n_{move}$ agents to the positions in the archive better solving each agent's
       sub-problem
17: end if
```

3.2.2 A New Archiving Strategy

The archiving process is a fundamental part of the optimisation process. The archive not only stores an approximation of the Pareto set, but it also constitutes a source of information for the social actions. A well distributed archive is thus not just a desirable property, but can also improve exploration by increasing the diversity of the population in criteria space.

A thorough analysis of the behaviour of the algorithm revealed that the archiving procedure implemented in the previous versions of MACS was suboptimal as it was not retaining some isolated non-dominated solutions and was instead keeping solutions in densely populated regions of the Pareto front. The final archive was therefore not giving the most even representation possible of the Pareto front.

In order to address this issue, a new archiving strategy is here proposed. The new strategy, which from now on will be called Energy Based Archiving (EBA), is physically based on the simple idea of minimising the energy of a cloud of points which exert repulsion on each other. It draws inspiration from the fact that a set of equally charged particles in a sphere will move towards its surface and spread uniformly.

In this case, however, the particles are not free to move, but can only occupy specified positions. The algorithm described in the following attempts to generate the most evenly distributed Pareto front possible with the available set of solutions. This archiving strategy is not specific to MACS, but can be applied to any multi-objective optimisation algorithm, and, as will be shown, can improve the results obtained by other algorithms.

EBA algorithm implementation

Suppose that at iteration k the archive is full and is composed of r elements. Let $\mathbf{y}_i$ and $\mathbf{y}_j$ be the position of element i and j in objective space, then one can define the generalised energy of the archive as:

$$E = \sum_{i=1}^{r} \sum_{j=i+1}^{r} \frac{1}{(\mathbf{y}_i - \mathbf{y}_j)^T (\mathbf{y}_i - \mathbf{y}_j)} \tag{3.13}$$

This energy is simply the sum of the reciprocal of the squared distances of the points of the archive in the criteria space. To avoid problems when the objectives have significantly different scales, the solution vectors are first normalised so that the maximum distance along each coordinate is 1.

Suppose now that there are q non-dominated candidate solutions which also do not dominate any of the elements in the archive. The problem of choosing which candidate substitutes which element of the archive is reformulated as finding the subset of r elements from the set of $r + q$ elements that minimises the energy E.

A direct update of the archive using E is not feasible if r and q are large, or even moderate, because the total number of possible combinations is $\binom{r+q}{r}$. As an alternative, the following procedure is proposed, for which pseudo-code is given in Algorithm 4 and a flow diagram in 3.2.

Let the archive A be not full. If there is enough space in A to add all the candidates, the archive is simply updated adding those elements (lines 1-2). A symmetric matrix M, containing the reciprocals of the squared distances of all the elements in the archive, is updated (line 4). From M, the total energy of the archive E and a helper vector $\mathbf{E_2}$, are computed (lines 6-7). $\mathbf{E_2}$ is needed to simplify and speed up some computations, as will be explained later. If the archive is full, instead, the following procedure is applied.

Given the elements in the archive A and a set of candidate elements C, for each element in A the energy E is recalculated assuming that that element was replaced by an element in C (line 23). If the lowest variation of E is negative, the element of C that gives that variation and the element in the archive are swapped. If there has been at least one replacement, the whole process is repeated until no more improvements can be detected or a maximum specified number of iterations is exceeded (lines 19-29).

In case the archive is not full but there is not enough space to add all the candidates, the above mentioned algorithm adds, sequentially, the candidates which give the least increase of the total energy of the archive (lines 8-12). No swapping between candidates and agents in the archive is performed in this case, only addition of candidates until the archive is full, and the corresponding update of M and $\mathbf{E_2}$ (lines 13-14).

Algorithm 4 Energy Based Archiving

```
 1: if there's room for all candidates in archive then
 2:    Add them to the archive
 3:    for all candidates do
 4:       Update the symmetric matrix $M$ containing the reciprocal of the squared
          distance of each pair of elements
 5:    end for
 6:    Update the total energy $E$ of the archive
 7:    Update the vector $\mathbf{E_2}$
 8: else
 9:    if only some candidates can be added then
10:       while archive is not full do
11:          Choose the candidate which gives the least possible addition of energy to
             the archive and add it
12:          Update $M$, $E$ and $\mathbf{E_2}$
13:       end while
14:    else if the archive is full then
15:       Set improved=TRUE
16:       iterations=0
17:       while improved and $iterations < max_{it}$ do
18:          improved=FALSE
19:          iterations=iterations+1
20:          Create a matrix containing the energy that the archive would have if each
             element of the archive were substituted with each candidate
21:          Locate the minimum entry $E_{new}$ of this matrix
22:          if $E_{new} < E$ then
23:             $E_{new}$ is at position $(i^*, j^*)$
24:             Swap candidate $j^*$ with element $i^*$
25:             Set improved=TRUE
26:             Update $E$, $M$ and $\mathbf{E_2}$
27:          end if
28:       end while
29:    end if
30: end if
```

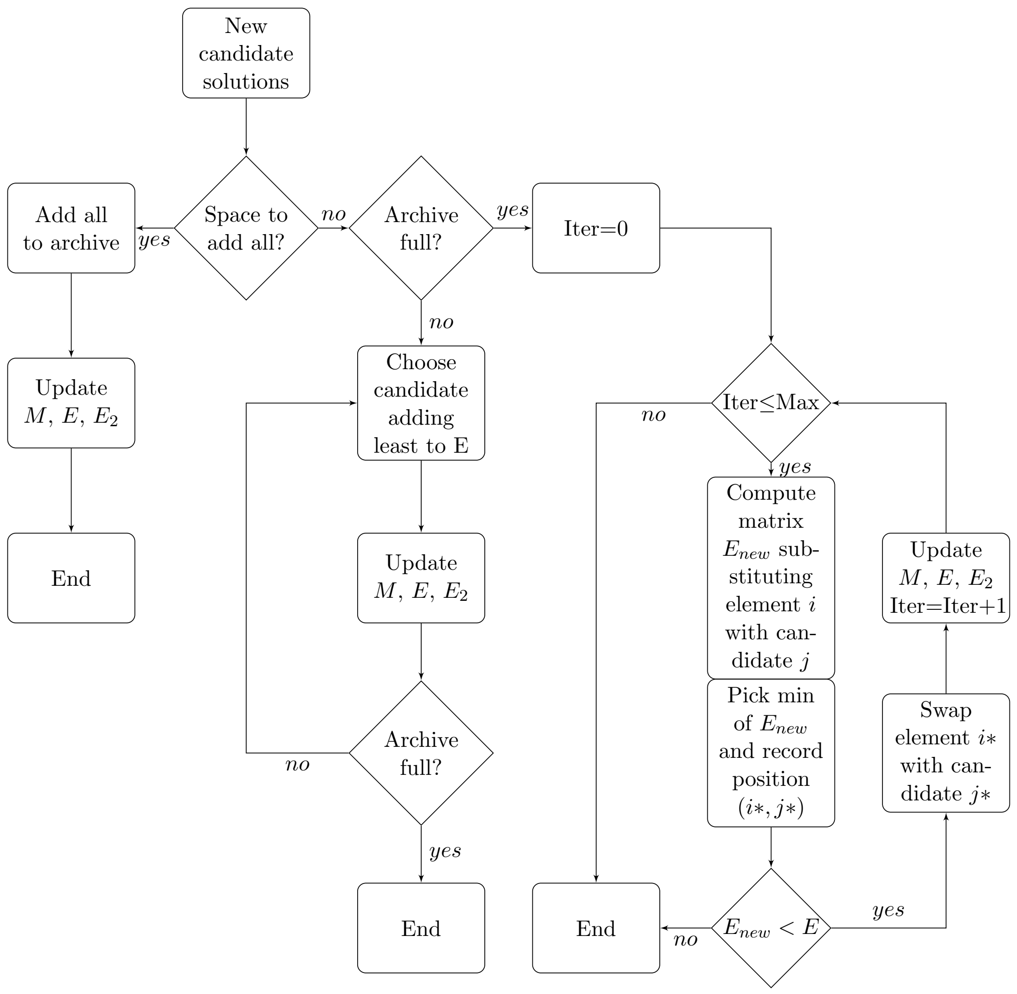

New candidate solutions
Space to add all?
yes
Add all to archive
Update M, E, E_2
End
no
Archive full?
yes
Iter=0
no
Choose candidate adding least to E
Update M, E, E_2
Archive full?
no
yes
End
Iter≤Max
no
yes
Compute matrix E_{new} substituting element i with candidate j
Pick min of E_{new} and record position $(i*, j*)$
$E_{new} < E$
no
End
yes
Swap element $i*$ with candidate $j*$
Update M, E, E_2 Iter=Iter+1

Figure 3.2: Diagram for the Energy based archiving strategy

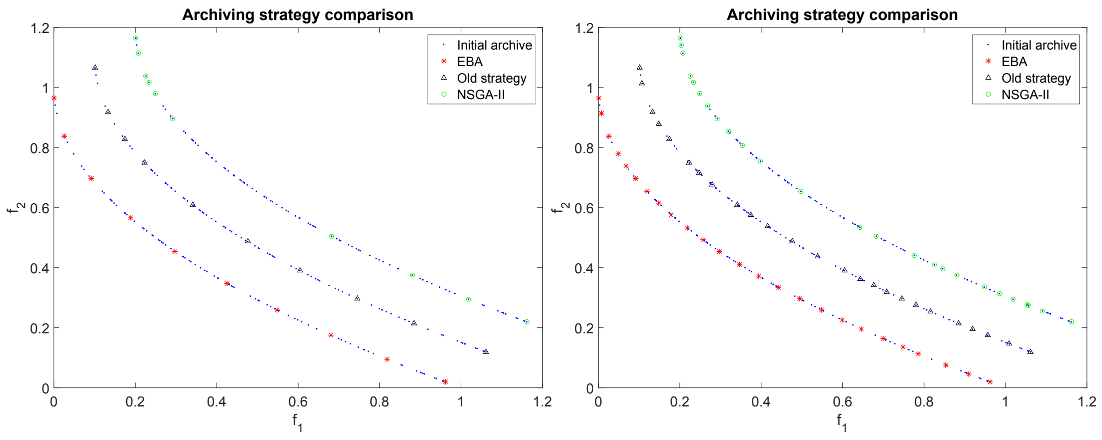
Table 3.1: IGD of the different archiving strategies

	MACS 2.1 archiver	MACS2 archiver	NSGA-II archiver
10 points	3.68e-2	3.90e-2	9.44e-2
25 points	1.52e-2	1.55e-2	2.44e-2

The actual Matlab implementation stores in a symmetric matrix the inverse of all pairwise squared distances between the elements currently in the archive. Deletion, addition and substitution of elements are performed as block matrix operations to save time. In the i-th entry of the $\mathbf{E_2}$ vector is stored the energy the archive would have if element i were removed. This way, the computation of the energy with a substitution of one element is linear in the number of candidates, because the baseline value (i.e the energy of the archive without replacement) is already stored, and only the contribution of the new candidate needs to be computed. The overall algorithm is called Energy Based Archiving (EBA).

As an example of the results provided by this archiving strategy, we considered a hypothetical Pareto front with 100 elements and tried to extract the q elements with the EBA algorithm, with the archiving algorithm employed in Zuiani and Vasile [35] and the one implemented in NSGA-II [26].

The hypothetical Pareto front is composed of a set of random samples taken from the Pareto front of the function ZDT4, a common benchmark function in multi-objective optimisation [98]. Figure 3.3 shows the results provided by the three archiving strategies for $q = 10$ and $q = 25$. The EBA strategy gives a good spreading of the extracted elements, slightly better than the one obtained with the strategy in MACS2 and much better than that obtained by the strategy implemented in NSGA-II. All the algorithms extract the same number of elements from this set. To quantify the effect of the algorithm, the Inverse Generational Distance metric (IGD), a measure of spreading commonly employed multi-objective optimisation, see [99], is shown in table 3.1.



(a) 10 points selected (b) 25 points selected

Figure 3.3: Outcomes of different archiving strategies from the same initial archive. The fronts have been shifted to enhance the comparison

3.2.3 Generation of the weight vectors

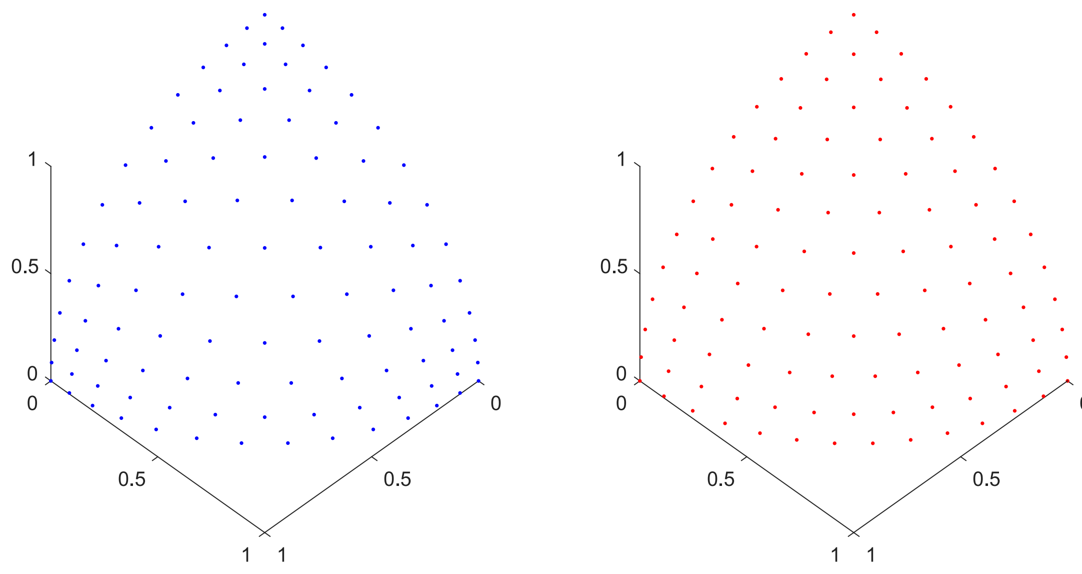
The weight vectors for are generated as follows: first, a simplex in objective space is generated through simplex lattice design [100]. Then, the points of this simplex lattice are projected on the unit sphere by dividing their position vectors by their distance to the origin. This gives a fairly uniform distribution of weight vectors (and thus descent directions) in any N_w dimensional space.

In order to generate a more uniform distribution, however, these weight vectors are used as an initial guess for the following optimisation problem:

$$\begin{aligned} &\min \ E(\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_{N_w}) \\ &s.t. \\ &\boldsymbol{\omega}_i^T \boldsymbol{\omega}_i = 1 \end{aligned} \tag{3.14}$$

where $E(\boldsymbol{\omega}_1, \cdots, \boldsymbol{\omega}_{N_w})$ is calculated using Eq. (3.13).

This optimisation problem can be quickly solved with a standard NLP solver. As a reference, from the initial lattice to the final optimised distribution the generation of 106 uniformly spread weight vectors for a three objective problem takes 22 iterations

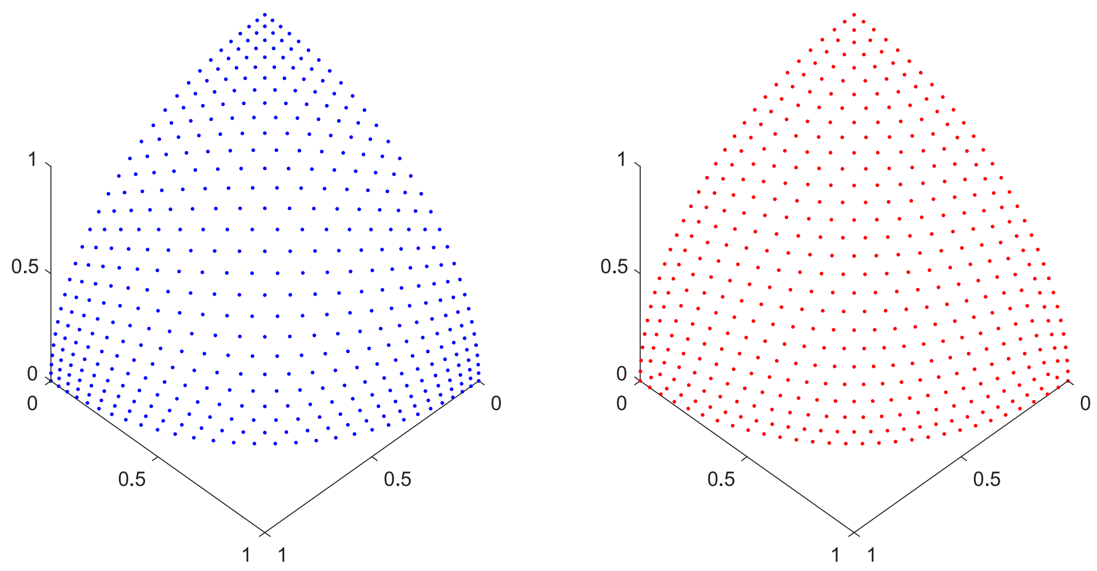


(a) Projection of symplex (b) After optimisation

Figure 3.4: Distribution of 106 Weight vectors

of the NLP solver with an SQP algorithm, which translates into approximately half a second in Matlab on an i7 laptop.

Figure 3.4 and 3.5 show the comparison between the distribution of weight vectors generated with the projection on the unit sphere of the simplex lattice design, and after the optimisation procedure, for the generation of 106 and 465 weight vectors. As it is evident, without the optimisation the points tend to be more clustered towards the vertices of the spherical triangle, while after the optimisation the points are evenly spread. Thus the optimisation of the weight vectors returns a better distribution at a negligible computational cost. While this approach is valid for general m-objective problems, for two objective problems it is simpler to directly generate uniformly angularly spaced weight vectors.



(a) Projection of symplex (b) After optimisation

Figure 3.5: Distribution of 465 Weight vectors

3.3 Benchmark Cases

MACS2.1 was tested on a set of benchmark functions: a mix of the first seven UF functions proposed in the CEC2009 competition [101] on multi-objective optimisation and the function ZDT4 proposed by Zitzler et al. [98].

3.3.1 CEC 2009 UF functions

The UF functions have a complex Pareto set and are a good benchmark to test the archiving procedure. MACS2.1 was tested and compared against the version of MOEA/D that won the CEC2009 competition [102] and against a previous version of MACS, called MACS2 [35].

On the UF test set, each algorithm was run 200 times for each of the functions UF1-7 on a Linux workstation with 8 GB of RAM and an Intel i7-4790 cpu. The settings for MACS2.1 are reported in Table 3.2 while for MOEA/D the parameters suggested by its authors in [102] were used. The algorithms are compared against the Inverse Generational Distance (IGD) metric, which was used to rank the solutions in

Table 3.2: Settings for MACS, CEC problems

$n_{feval,max}$	n_{pop}	ρ_{ini}	F	CR	p_{social}	ρ_{contr}	$\rho_{max,contr}$
300000	150	1	0.9	0.9	0.2	0.5	5

Table 3.3: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 and MOEA/D, CEC2009 problems. Also reported the unsigned Wilcoxon test results

Problem	MACS2.1	MOEA/D	Wilcoxon	MACS2.1	MOEA/D	Wilcoxon
	IGD	IGD	test IGD	Hausdorff	Hausdorff	test Hausdorff
UF1	**4.09e-3**	4.41e-3	3.70e-59	**1.65e-2**	1.12e-1	3.14e-11
	(9.58e-9)	(1.69e-8)		(5.53e-5)	(6.16e-2)	
UF2	**4.43e-3**	6.24e-3	3.70e-59	**2.09e-2**	7.48e-2	4.52e-53
	(1.23e-7)	(1.57e-6)		(4.41e-5)	(9.46e-3)	
UF3	1.84e-2	**7.16e-3**	3.70e-59	1.46e-1	**6.38e-2**	5.08e-34
	(1.09e-5)	(2.47e-5)		(2.61e-2)	(1.83e-2)	
UF4	**2.93e-2**	6.14e-2	3.70e-59	**4.99e-2**	1.11e-1	4.83e-67
	(7.50e-7)	(2.50e-5)		(2.74e-5)	(3.17e-4)	
UF5	**5.80e-2**	2.98e-1	3.70e-59	**1.32e-1**	7.96e-1	4.83e-67
	(5.58e-5)	(7.45e-3)		(9.56e-4)	(2.20e-1)	
UF6	**2.74e-2**	2.68e-1	3.70e-59	**8.86e-2**	6.27e-1	7.03e-67
	(6.10e-5)	(4.34e-2)		(2.06e-3)	(1.14e-1)	
UF7	**4.15e-3**	4.77e-3	1.46e-34	**2.96e-2**	1.67e-1	1.26e-08
	(5.61e-8)	(3.17e-6)		(9.40e-4)	(6.83e-2)	

the CEC2009 competition, and against the Averaged Hausdorff distance. Both metrics are described and extensively analysed in [103].

As pointed out by Schütze et al. [103], the IGD metric is sensistive to the number of elements in the reference Pareto front and in the computed one. Hence the inclusion of the Averaged Hausdorff distance in this comparison. Mean and variance of the IGD and Averaged Hausdorff distance for each problem and algorithm are reported in Table 3.3, together with result of the Wilcoxon hypothesis test. In 6 of the 7 cases analysed in this paper, the results obtained by MACS2.1 have lower mean IGD and mean Averaged Hausdorff distance, and the variances of those metrics are 1 to 3 orders of magnitude lower for MACS2.1, meaning that the results or MACS2.1 are much more repeatable. The low values of the Wilcoxon test confirm that the underlying distributions of the metrics are indeed different.

Effect of the Energy Based Archiver

To better appreciate the effect of the archiver, MACS2.1 was then run with the same settings but with the archiving strategy employed by MACS2 (denoted as MACS2.1 NO EBA), while the final results of MOEA/D were filtered with the EBA algorithm instead of using the filter employed by MOEA/D (denoted as MOEA/D+EBA). Tables 3.4 and 3.5 summarise the results of this test.

The EBA strategy improved the quality of the Pareto front found by MOEA/D in 4 cases without worsening the others, and improved the results of MACS2.1 in 3 cases with no significant variation in the other cases. The amount of the improvement depends on the quality and size of the the archive: a closer examination of the UF5 and UF6 cases showed that none of the 200 archives had 100 non-dominated elements, hence EBA simply gave the same result as the strategy implemented in MACS2.

For UF4 the high IGDs are caused by a relatively high distance between the computed front and the true one, more than by a poor distribution of the points, while for UF3 there is a relative lack of points in the upper left region of the Pareto front. For this comparison, the Averaged Hausdorff distance does not show any relevant improvement. This is due to the fact the Averaged Hausdorff distance penalizes the outliers (as clearly stated in [103]), and as such is a worst case measure. The EBA algorithm was conceived to maximise the spreading of the overall solution, but cannot guarantee the worst case distance from each point of the reference front to each point on the computed one, so the observed metrics are not surprising.

Effect of the heuristics and comparison with the previous version

MACS2.1 was then compared against MACS2 [35]. As it can be seen from Table 3.6, MACS2.1 improves over MACS2 in 5 out of 7 cases for the IGD, although it produces worse results on UF4 and UF5. The Averaged Hausdorff distance for this case favours MACS 2.1 only in 3 over 7 cases.

Table 3.4: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 with EBA archiving vs MACS2.1 without EBA archiving on the CEC2009 problems. Also reported the Wilcoxon test result

Problem	MACS2.1 IGD	MACS2.1 NO EBA IGD	Wilcoxon test IGD	MACS2.1 Hausdorff	MACS2.1 NO EBA Hausdorff	Wilcoxon test Hausdorff
UF1	**4.09e-3**	4.32e-3	1.88e-54	1.65e-2	1.61e-2	9.88e-1
	(9.58e-9)	(1.04e-8)		(5.53e-5)	(1.49e-5)	
UF2	**4.43e-3**	4.70e-3	3.84e-25	2.09e-2	2.09e-2	8.49e-1
	(1.23e-7)	(6.56e-8)		(4.41e-5)	(4.13e-5)	
UF3	1.84e-2	1.85e-2	6.56e-01	1.46e-1	1.41e-1	6.45e-1
	(1.09e-5)	(9.72e-6)		(2.61e-2)	(2.25e-2)	
UF4	2.93e-2	2.92e-2	5.29e-01	4.99e-2	5.04e-2	2.46e-1
	(7.50e-7)	(1.03e-6)		(2.74e-5)	(3.13e-5)	
UF5	5.80e-2	5.84e-2	7.84e-01	1.32e-1	1.35e-1	2.19e-1
	(5.58e-5)	(5.63e-5)		(9.56e-4)	(9.28e-4)	
UF6	2.74e-2	2.66e-2	2.58e-01	8.86e-2	9.61e-2	8.12e-2
	(6.10e-5)	(3.71e-5)		(2.06e-3)	(2.64e-3)	
UF7	**4.15e-3**	4.49e-3	1.40e-35	2.96e-2	2.79e-2	1.52e-1
	(5.61e-8)	(6.01e-8)		(9.40e-4)	(5.80e-5)	

Table 3.5: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MOEA/D with EBA archiving and standard MOEA/D, CEC2009 problems. Also reported the Wilcoxon test result

Problem	MOEA/D+EBA IGD	MOEA/D IGD	Wilcoxon test IGD	MOEA/D+EBA Hausdorff	MOEA/D Hausdorff	Wilcoxon test Hausdorff
UF1	**4.11e-3**	4.41e-3	8.16e-54	1.11e-1	1.12e-1	3.67e-1
	(1.71e-8)	(1.69e-8)		(6.16e-2)	(6.16e-2)	
UF2	**6.00e-3**	6.24e-3	2.53e-04	7.48e-2	7.48e-2	9.94e-1
	(1.58e-6)	(1.57e-6)		(9.46e-3)	(9.46e-3)	
UF3	**6.88e-3**	7.16e-3	2.54e-07	6.30e-2	6.38e-2	1.19e-1
	(2.54e-5)	(2.47e-5)		(1.83e-2)	(1.83e-2)	
UF4	6.13e-2	6.14e-2	7.92e-01	1.11e-1	1.11-e1	8.45e-1
	(2.49e-5)	(2.50e-5)		(3.12e-4)	(3.17e-4)	
UF5	2.98e-1	2.98e-1	9.97e-01	7.96e-1	7.96e-1	9.98e-1
	(7.45e-3)	(7.45e-3)		(2.20e-1)	(2.20e-1)	
UF6	2.68e-1	2.68e-1	9.95e-01	6.27e-1	6.27e-1	9.96e-1
	(4.34e-2)	(4.34e-2)		(1.14e-1)	(1.14e-1)	
UF7	**4.48e-3**	4.77e-3	1.24e-32	1.67e-1	1.67e-1	9.08e-1
	(3.19e-6)	(3.17e-6)		(6.83e-2)	(6.83e-2)	

To better understand which heuristic is contributing to give the different results of MACS2.1 with respect to MACS2, the results obtained by MACS2.1 with dynamic adjustment of the maximum number of directions in the pattern search was compaired against MACS2.1 with a fixed maximum number of direction equal to $2n$. This is because in MACS2 the number of directions scanned by pattern search is fixed and equal to $2n$.

Results in Table 3.7 show that in all cases except for UF4 and UF5, the dynamic adjustment of the maximum number of directions scanned by pattern search has a positive effect on the IGD. The comparison of the Averaged Hausdorff distance shows a substantial parity between the two approaches, meaning that the dynamic strategy does not improve the position of the outliers.

It can be also be appreciated that in the static case, both the IGD and the Averaged Hausdorff distance associated to the UF1 to UF5 cases are close to the values obtained by MACS2. This is not surprising since in MACS2.1 DE is performed after pattern search, so if pattern search scans all possible coordinates it will most probably find an improvement and thus DE will not be performed. This also means that in the UF6 and UF7 cases some other heuristic of MACS2.1 is instead contributing.

With the same rationale as the previous analysis, MACS2.1 with the new implementation of social moves was tested against MACS2.1 with the old implementation of the social moves. Table 3.8 shows that the IGD of the new version is better than the old version in 4 over 7 cases, statistically the same in 1 case and worse in 2 cases, while the Averaged Hausdorff distance of the new social moves is better in 3 cases, statistically the same in 1 case and worse in 3 cases.

Thus, this modification does not seem to give a clear contribution. However, the simple comparison with mean and variance of the metrics is not giving a full picture of the behaviour of the algorithms, since what is important in practice is that the algorithm is able to consistently find a good approximation of the Pareto front in a given computational budget.

For this reason, as a final rigorous performance test for the CEC cases, the success rate of each algorithm were computed for each of the UF functions. The success rate is defined as the number of runs in which the IGD falls below a given threshold over the total number of runs. Thresholds were chosen to differentiate the results as much as possible but using rather simple values.

Table 3.9 summarises the results. As it is evident, MACS2.1 has overall good performance, outperforming MOEA/D in all cases except for UF3. The introduction of EBA in MOEA/D can improve its results by 10-30% on some problems. MACS2.1 is also generally better than MACS2: although for UF2, UF4 and UF5 the latter has a success rate 20% higher than the former, MACS2.1 is more than 20% better in the other problems, up to 90% better for UF7.

In MACS2.1, an overall 5 to 30% improvement is given by the EBA archiving strategy, while the dynamic setting of the maximum number of coordinates can improve results up to 90% or worsen them up to 15%. Similarly, the new implementation of the social moves can improve results up to 50% or worsen them up to 20%. Overall, the proposed version of MACS2.1 seems to have more consistent results on the entire set of problems, never falling behind by more than 25% over any other algorithm on any problem.

Table 3.6: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 vs. MACS2 on the CEC2009 problems. Also reported the Wilcoxon test result

Problem	MACS2.1 IGD	MACS2 IGD	Wilcoxon test IGD	MACS2.1 Hausdorff	MACS2 Hausdorff	Wilcoxon test Hausdorff
UF1	**4.09e-3**	4.39e-3	1.50e-59	**1.65e-2**	2.80e-2	9.26e-35
	(9.58e-9)	(2.48e-8)		(5.53e-5)	(3.14e-4)	
UF2	**4.43e-3**	4.49e-3	6.33e-09	2.09e-2	**1.57e-2**	5.08e-24
	(1.23e-7)	(1.32e-8)		(4.41e-5)	(7.61e-6)	
UF3	**1.84e-2**	2.41e-2	8.25e-50	1.46e-1	**6.75e-2**	1.07e-29
	(1.09e-5)	(4.98e-6)		(2.61e-2)	(3.60e-4)	
UF4	2.93e-2	**2.63e-2**	2.15e-66	4.99e-2	**4.43e-2**	1.21e-26
	(7.50e-7)	(2.96e-7)		(2.74e-5)	(2.01e-5)	
UF5	5.80e-2	**5.29e-2**	2.81e-11	1.32e-1	**1.22e-1**	2.09e-05
	(5.58e-5)	(4.81e-5)		(9.56e-4)	(1.09e-3)	
UF6	**2.74e-2**	3.41e-2	7.30e-17	**8.86e-2**	1.02e-1	2.03e-04
	(6.10e-5)	(1.06e-4)		(2.06e-3)	(2.60e-3)	
UF7	**4.15e-3**	6.54e-3	3.84e-66	**2.96e-2**	4.93e-2	5.68e-39
	(5.61e-8)	(4.96e-6)		(9.40e-4)	(4.37e-4)	

Table 3.7: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 with EBA archiving and dynamic setting of maximum number of coordinates for pattern search vs MACS2.1 with EBA archiving and static setting of maximum number of coordinates for pattern search on the CEC 2009 problems. Also reported the Wilcoxon test result

Problem	MACS2.1 IGD	MACS2.1 static IGD	Wilcoxon test IGD	MACS2.1 Hausdorff	MACS2.1 static Hausdorff	Wilcoxon test Hausdorff
UF1	**4.09e-3**	4.40e-3	5.46e-61	**1.65e-2**	2.54e-2	4.74e-39
	(9.58e-9)	(1.98e-8)		(5.53e-5)	(1.29e-4)	
UF2	**4.43e-3**	4.47e-3	2.48e-06	2.09e-2	**1.60e-2**	2.64e-21
	(1.23e-7)	(2.14e-8)		(4.41e-5)	(1.24e-5)	
UF3	**1.84e-2**	2.51e-2	4.05e-09	1.46e-1	1.36e-1	4.67e-01
	(1.09e-5)	(1.09e-5)		(2.61e-2)	(1.43e-2)	
UF4	2.93e-2	**2.66e-2**	2.20e-65	4.99e-2	**4.54e-2**	2.70e-20
	(7.50e-7)	(3.42e-7)		(2.74e-5)	(2.26e-5)	
UF5	5.80e-2	**5.47e-2**	9.51e-06	1.32e-1	**1.24e-1**	1.21e-03
	(5.58e-5)	(4.98e-5)		(9.56e-4)	(7.51e-4)	
UF6	**2.74e-2**	3.04e-2	1.27e-05	**8.86e-2**	9.78e-2	3.48e-03
	(6.10e-5)	(7.43e-5)		(2.06e-3)	(2.13e-3)	
UF7	**4.15e-3**	5.08e-3	9.58e-66	**2.96e-2**	4.76e-2	9.84e-48
	(5.61e-8)	(1.59e-6)		(9.40e-4)	(2.33e-4)	

Table 3.8: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 vs MACS2.1 with old social moves on the CEC2009 problems. Also reported the Wilcoxon test result

Problem	MACS2.1 IGD	MACS2.1 old social IGD	Wilcoxon test IGD	MACS2.1 Hausdorff	MACS2.1 old social Hausdorff	Wilcoxon test Hausdorff
UF1	**4.09e-3**	4.14e-3	1.96e-04	**1.65e-2**	2.53e-2	2.71e-06
	(9.58e-9)	(2.63-e8)		(5.53e-5)	(1.54e-3)	
UF2	4.43e-3	**4.11e-3**	8.44e-36	2.09e-2	**1.54e-2**	2.36e-27
	(1.23e-7)	(1.83e-8)		(4.41e-5)	(9.18e-6)	
UF3	**1.84e-2**	1.95e-2	2.95e-04	1.46e-1	**8.41e-2**	5.16e-31
	(1.09e-5)	(4.03e-6)		(2.61e-2)	(1.24e-2)	
UF4	2.93e-2	**2.76e-2**	5.87e-48	4.99e-2	**4.64e-2**	2.22e-15
	(7.50e-7)	(7.04e-7)		(2.74e-5)	(2.39e-5)	
UF5	5.80e-2	5.75e-2	5.33e-01	1.32e-1	1.30e-1	4.49e-01
	(5.58e-5)	(5.01e-5)		(9.56e-4)	(8.26e-4)	
UF6	**2.74e-2**	3.05e-2	2.47e-05	**8.86e-2**	1.01e-1	5.35e-05
	(6.10e-5)	(6.63e-5)		(2.06e-3)	(2.67e-3)	
UF7	**4.15e-3**	4.60e-3	4.13e-39	**2.96e-2**	3.31e-2	3.39e-13
	(5.61e-7)	(1.14e-7)		(9.40e-4)	(7.54e-5)	

Table 3.9: Success rates for the IGD of the CEC2009 functions for all the tested algorithms and their variants

%IGD < τ	MACS2.1	MACS2.1 NO EBA	MOEA/D	MOEA/D + EBA	MACS2.1 static pat	MACS2.1 old social	MACS2
UF1 (τ=4.5e-3)	100	94.5	85.0	98.0	76.0	99.0	79.5
UF2 (τ=5.0e-3)	93.5	88.5	0.5	13.0	99.5	100	100
UF3 (τ=2.0e-2)	68.5	67.5	95.0	95.0	9.0	61.0	3.5
UF4 (τ=3.0e-2)	78.5	78.5	0	0	100	100	100
UF5 (τ=5.0e-2)	16.0	12.0	0	0	25.0	15.0	36.5
UF6 (τ=3.0e-2)	70.0	74.5	0	0	56.0	53.5	36.5
UF7 (τ=4.5e-3)	92.0	60.0	64.5	92.0	1.5	40.5	1.5

3.3.2 ZDT4 and 3 impulse problem

To further test the capabilities of MACS2.1 it was run 200 times on the ZDT4 test function and on a real space trajectory optimisation problem. In the space trajectory design problem the goal is to optimise three impulsive manoeuvres to transfer a spacecraft from a circular Low Earth Orbit, with a radius of 7000km, to a circular Geostationary orbit, with a radius of 42000km (for further details on the problem the interested reader can refer to [33]).

The motivation behind the choice of the ZDT4 and the 3 impulse test case is that both of them are characterised by many local Pareto fronts. The settings for MACS2.1 are reported in table 3.10, 200 solutions per run were maintained in the archive. The performance of MACS2.1 was compared against the perfomance of MACS2, whose settings were specified as in [35].

Note that, for the 3 impulse case the true Pareto front is unknown, thus for self consistence a global Pareto front was extracted from all the 400 runs and used as a reference Pareto front for the calculation of all the metrics. Tables 3.11 and 3.12 report the metrics computed for both the ZDT4 and 3 impulse problem and the corresponding success rates. On the ZDT4 case MACS2.1 outperforms MACS2.

The 3 impulse case gives less clear results instead. Although many interesting areas of the global Pareto front are due to MACS2.1, the mean IGD associated to its solutions is higher than the mean IGD of the solutions computed by MACS2. This is due to the fact (see Figure 3.6) that MACS2 is contributing to the global Pareto front with twice as many points than MACS2.1, and all those points are concentrated in a central area, while the contribution from MACS2.1 is as expected more widespread and surprisingly a bit scarce in the central area.

Thus, the typical run of MACS2 will generate many points close to the over represented region, while the typical run of MACS2.1 will generate less points in that area but more widely spread points, and this results in the averaged IGD of MACS2.1 to be

Table 3.10: Settings of MACS2.1 on the 3 impulse and ZDT4 problems (in brackets)

$n_{feval,max}$	n_{pop}	ρ_{ini}	F	CR	p_{social}	ρ_{contr}	$\rho_{max,contr}$
30000	10	1	0.9	0.9	1	0.5	5
(15000)	(10)	(1)	(0.9)	(0.9)	(1)	(0.5)	5

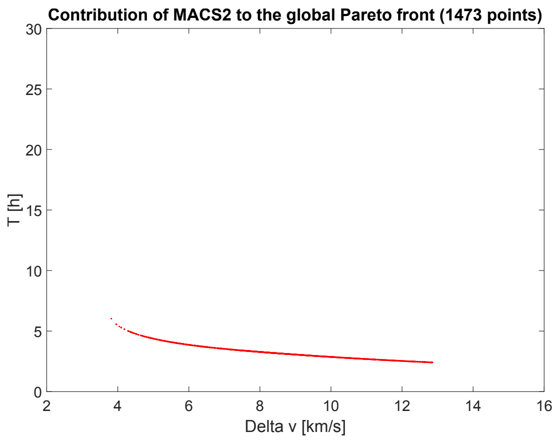
Table 3.11: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 vs MACS2 on zdt4 and triple impulse problems. Also reported the Wilcoxon test result

Problem	MACS2.1 IGD	MACS2 IGD	Wilcoxon test	MACS2.1 Hausdorff	MACS2 Hausdorff	Wilcoxon test
zdt4	**7.6e4-3**	8.01e-1	3.00e-64	**2.52e-2**	2.62e+0	1.20e-62
	(1.05e-4)	(2.48e-1)		(7.59e-4)	(3.60e+0)	
3 imp	2.78e-1	**1.17e-2**	1.32e-43	**2.89e+2**	3.45e+2	7.41e-31
	(2.81e-2)	(1.07e-3)		(7.11e+3)	(6.05e+3)	

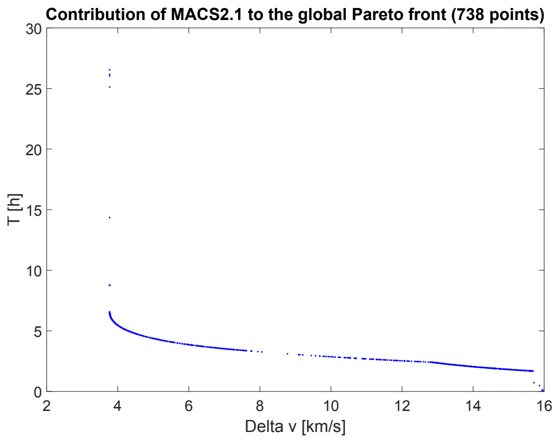
higher than that of MACS2. The Averaged Hausdorff distance instead does not suffer from this kind of bias, thus the lower value of this metric is associated to the fronts computed by MACS2.1.

Table 3.12: ZDT4 and triple impulse success rates for MACS2 and MACS2.1 and the various metrics

% IGD $<\tau$	MACS2	MACS2.1	% Hausdorff $<\tau$	MACS2	MACS2.1
zdt4(τ=1e-2)	1.5	83.5	zdt4(τ=5e-2)	3.0	97.5
3imp(τ=1e-1)	29.5	3.0	3imp(τ=3e+2)	5.0	82.5



(a) Contribution of MACS2



(b) Contribution of MACS2.1

Figure 3.6: Contribution of the different algorithms to the global Pareto front

3.4 Chapter summary

This chapter presented the multi-objective optimisation problem, the MACS2.1 algorithm, a new archiving strategy and some modified search heuristics for MACS2. The results computed by the new algorithm, are overall better than those computed by MOEA/D on the UF test set. MACS2.1 also outperformed the previous version in 5 over 7 of the UF functions and on the ZDT4 test case. In the 3 impulse case, MACS2.1 contributed with a wide spread of points to the Pareto front not concentrated in the central area, which was instead densely covered by the previous version.

The newly proposed archiving strategy also improved the results generated by MOEA/D, indicating both that is effective and that a higher quality of the solutions stored in the Pareto front can positively influence the search capabilities of an evolutionary algorithm.

Overall, the tests confirmed the high reliability of MACS2.1 as a global multi-objective optimisation algorithm. Its flexibility, inherited by the fact that it can employ different search heuristics of both global and local nature, will be employed in the next chapter in order to tackle the solution of multi-objective optimal control problems. For simplicity, in the following it will be simply referred to as MACS.

Chapter 4

Multi-objective Optimal Control[1]

Faber est suae quisque fortunae
Each man is the maker of his own fortune

Sallustius

This chapter combines the previous two chapters, proposing a novel approach to the solution of multi-objective optimal control problems. In addition, we here consider a problem which can have multiple phases either in series or in parallel. For example, a multi-stage vehicle can have one phase per vehicle stage with all phases connected in series for the ascent, and/or branching parallel phases for the upper stage ascent and first stage descent and return.

[1]The contents of this chapter were published in: L. A. Ricciardi, C. Maddock and M. Vasile. Direct Solution of Multi-Objective Optimal Control Prolem Applied to Spaceplane Mission Design. Journal of Guidance, Control, and Dynamics, Vol. 42, No. 1 (2019), pp. 30-46.

Multi-, or more generally many, objective optimal control problems can be formulated as follows:

$$\begin{aligned}
&\min_{\mathbf{u}\in U,\mathbf{b}\in B} \mathbf{J}\\
&s.t.\\
&\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x},\mathbf{u},\mathbf{b},t)\\
&\mathbf{g}(\mathbf{x},\mathbf{u},\mathbf{b},t) \leq 0\\
&\boldsymbol{\psi}(\mathbf{x}(t_0),\mathbf{x}(t_f),\mathbf{u}(t_0),\mathbf{u}(t_f),\mathbf{b},t_0,t_f) \leq 0\\
&t \in [t_0,t_f]
\end{aligned} \tag{4.1}$$

where $\mathbf{J} = [J_1, J_2, ..., J_i..., J_m]^T$ is, in general, a vector of objectives J_i that are functions of the state vector $\mathbf{x} : [t_0,t_f] \to \mathbb{R}^{n_x}$, control variables $\mathbf{u} \in L^\infty(U \subseteq \mathbb{R}^{n_u})$, static parameters $\mathbf{b} \in B \subseteq \mathbb{R}^{n_b}$ and time t. The functions $\mathbf{x}(t)$ belong to the Sobolev space $\mathcal{W}^{1,\infty}$ while the objective functions are $J_i : \mathbb{R}^{3n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_b} \times \mathbb{R}^2 \longrightarrow \mathbb{R}$. The objective vector is subject to a set of dynamic constraints with $\mathbf{F} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_b} \times [t_0,t_f] \longrightarrow \mathbb{R}^{n_x}$, algebraic constraints $\mathbf{g} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_b} \times [t_0,t_f] \longrightarrow \mathbb{R}^{n_g}$, and boundary conditions $\boldsymbol{\psi} : \mathbb{R}^{2n_x} \times \mathbb{R}^{2n_u} \times \mathbb{R}^{n_b} \times \mathbb{R}^2 \longrightarrow \mathbb{R}^{n_\psi}$, where $\mathbf{F}(\mathbf{x},\mathbf{u},\mathbf{b},t)$, $\mathbf{g}(\mathbf{x},\mathbf{u},\mathbf{b},t)$ and $\boldsymbol{\psi}(\mathbf{x}(t_0),\mathbf{x}(t_f),\mathbf{u}(t_0),\mathbf{u}(t_f),\mathbf{b},t_0,t_f)$ are vector fields.

When N_p phases are present, dynamic constraints, path and boundary constraints, and objective functions are defined on each timeline. In order to connect different timelines, a set of N_{ip} inter-phase constraints are introduced:

$$\psi_{s_p}\left(\mathbf{x}_{0,\mathbf{I}_{s_p}}, \mathbf{x}_{f,\mathbf{I}_{s_p}}, \mathbf{u}_{0,\mathbf{I}_{s_p}}, \mathbf{u}_{f,\mathbf{I}_{s_p}}, \mathbf{b}_{\mathbf{I}_{s_p}}, t_{0,\mathbf{I}_{s_p}}, t_{f,\mathbf{I}_{s_p}}\right) \leq 0 \quad s_p = 1, ..., N_{ip} \tag{4.2}$$

where the index vector $\mathbf{I}_{s_p}$ collects all the indexes of the phases that are connected by constraint ψ_{s_p}. Note that the number of phases is fixed, but their temporal order is actually defined by the inter-phase constraints (4.2).

If boundary states and times, $\mathbf{x}_0, \mathbf{x}_f, t_0, t_f$ are free decision variables, they can be included in the static parameter vector $\mathbf{b}$ and the union of controls and static parameters is called a decision vector. Thus, without loss of generality, one can say that the solution of problem (4.1) is a subset of $U \times B$ that satisfies the constraints and contains Pareto efficient decision vectors. This leads to the following definition.

Definition 2 *Given the subset $\Omega_U \subset U \times B$ of feasible decision vectors, a decision vector $[\mathbf{u}^*, \mathbf{b}^*] \in \Omega_U$ is said to be Pareto efficient if $[\mathbf{u}^*, \mathbf{b}^*] \not\succ [\mathbf{u}, \mathbf{b}]$, $\forall [\mathbf{u}, \mathbf{b}] \in \Omega_U$.*

One way to obtain necessary conditions for optimality for multi-objective optimal control problem is by employing the Pascoletti-Serafini scalarisation [44], thus transforming problem (4.1) in:

$$\begin{aligned}
&\min\nolimits_{sf \geq 0} s_f \\
&s.t. \\
&\omega_i(J_i - z_i) - s_f \leq 0 \\
&\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{u}, \mathbf{b}, t) \\
&\mathbf{g}(\mathbf{x}, \mathbf{u}, \mathbf{b}, t) \leq 0 \\
&\boldsymbol{\psi}(\mathbf{x}(t_0), \mathbf{x}(t_f), \mathbf{u}(t_0), \mathbf{u}(t_f), \mathbf{b}, t_0, t_f) \leq 0 \\
&t \in [t_0, t_f]
\end{aligned} \tag{4.3}$$

where s_f is a slack variable, $\boldsymbol{\omega}$ is a weight vector associated to the objectives, and $\mathbf{z}$ is a target point in criteria space. This scalarisation is constraining the solution of the multi-objective optimal control problem to lie on the diagonal of a rectangle, in criteria space, with ratios of the edges given by $\omega_i(J_i - z_i)$. s_f can be interpreted as being the final condition of an additional pseudo-state $\dot{s} = 0$, thus transforming the problem into a Mayer problem.

By adopting this scalarisation technique, it is possible to state the following

Theorem 3 *Consider the function $H = \boldsymbol{\lambda}^T \mathbf{F}(\mathbf{x}(t), \mathbf{u}(t), t) + \boldsymbol{\mu}^T \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t), t)$. If $\mathbf{u}(t)^*$ is a locally optimal solution for problem (4.3), with associated state vector $\mathbf{x}(t)^*$, and H is Frechet differentiable at $\mathbf{u}(t)^*$ and a regular point of the algebraic constraints,* *then there exist vectors* $\boldsymbol{\eta} \in \mathbb{R}^m, \boldsymbol{\lambda} \in \mathbb{R}^n$ *and* $\boldsymbol{\mu} \in \mathbb{R}^q$ *such that:*

$$\begin{aligned}
&\mathbf{u}^* = \operatorname{argmin}_{\mathbf{u}\in U} \boldsymbol{\lambda}^T \mathbf{F}\left(\mathbf{x}(t)^*, \mathbf{u}(t), t\right) + \boldsymbol{\mu}^T \mathbf{g}\left(\mathbf{x}(t)^*, \mathbf{u}(t), t\right) \\
&\boldsymbol{\lambda}^T \nabla_x \mathbf{F}\left(\mathbf{x}(t)^*, \mathbf{u}(t), t\right) + \boldsymbol{\mu}^T \nabla_x \mathbf{g}\left(\mathbf{x}(t)^*, \mathbf{u}(t), t\right) = 0 \\
&\dot{\lambda}_s = 0 \\
&\boldsymbol{\lambda} \geq 0; \boldsymbol{\mu} \geq 0
\end{aligned} \tag{4.4}$$

with transversality conditions:

$$\begin{aligned}
&1 - \textstyle\sum_{j=1}^m \eta_j = \lambda_s(t_f) \\
&\boldsymbol{\eta}^T \boldsymbol{\Omega} \nabla_x \mathbf{J} + \boldsymbol{\nu}^T \nabla_x \boldsymbol{\psi}(\mathbf{x}(t_0), \mathbf{x}(t_f), \mathbf{u}(t_0), \mathbf{u}(t_f), \mathbf{b}, t_0, t_f) = \boldsymbol{\lambda}_x(t_f) \\
&\boldsymbol{\eta} \geq 0; \boldsymbol{\nu} \geq 0
\end{aligned} \tag{4.5}$$

the proof is given in [104] and is reported for completeness in Appendix A.

4.1 Solution of the Transcribed Problem

The problem is initially discretised following the procedure for DFET transcription illustrated in Chapter 2. The resulting Multi Objective Nonlinear Programming (MNLP) problem coming from the transcription of problem (4.1), with the inclusion of interphase constraints (4.2), can be written, in vector form, as:

$$\begin{aligned}
&\min_{\mathbf{y}\in Y, \mathbf{p}\in\Pi} \tilde{\mathbf{J}} \\
&s.t. \\
&\mathbf{C}(\mathbf{y}, \mathbf{p}) \leq 0
\end{aligned} \tag{4.6}$$

where $\mathbf{y} = [\mathbf{x}_{0,1}, .., \mathbf{x}_{s,j}, ..., \mathbf{x}_{l_x,N}]^T$, Y is a box in $\mathbb{R}^{n_Y}$ with $n_Y = n(l_x+1)N$, $\mathbf{p} = [\mathbf{u}_{0,1}, .., \mathbf{u}_{s,j}, ..., \mathbf{u}_{l_u,N}, \mathbf{b}^*]^T$ collects all the static and discretised dynamic control variables, $\mathbf{b}^* = [\mathbf{b}, \mathbf{x}_0^b, \mathbf{x}_f^b, t_0, t_f]^T$, $\Pi \subseteq \mathbb{R}^{n_s} \times \mathbb{R}^{n_b^*}$, with $n_s = n_u(l_u+1)N$ (assuming that each element has the same number of control parameters) and $n_b^* = n_b + 2n + 2$, and $\mathbf{C}$ collects all constraints, including boundary and interphase ones.

Similar to problem (4.1), the solution of problem (4.6) is a subset of $\Omega_\Pi \subset \Pi$ that satisfies the constraints and contains vectors $\mathbf{p}$ that are Pareto efficient. For continuous functions, the subset Ω_Π is a manifold in $\mathbb{R}^{n_s+n_b^*}$ with dimension $\leq (m-1)$ [105]. In the following, the goal will be to identify a pre-defined countable number of Pareto efficient solutions contained in Ω_Π.

Problem (4.6) is solved with an adaptation of MACS called MACSoc, that combines the stochastic agent-based global search of MACS with a gradient-based treatment of the constraints, and refinement of the solutions exploiting Theorem 4. The overall solution process implemented in MACSoc is summarised in Algorithm 5.

Algorithm 5 MACS optimal control (MACSoc) framework

```
1: Initialise population $\mathcal{P}_0$ and global archive $\mathcal{A}_0$, $k = 0$, $\rho_{\mathcal{B}} = 1$
2: Initialise weight vectors $\boldsymbol{\omega}$
3: while $n\_fun\_eval < max\_fun\_eval$ do
4:    Run individualistic heuristics on $\mathcal{P}_k$ using bi-level formulation
5:    $\mathcal{P}_k \rightarrow \mathcal{P}_k^+$
6:    Update archive $\mathcal{A}_k$ with potential field filter
7:    Run social heuristics combining $\mathcal{P}_k^+$ and $\mathcal{A}_k$ using bilevel formulation
8:    Update archive $\mathcal{A}_k$ with potential field filter
9:    $\mathcal{P}_k^+ \rightarrow \mathcal{P}_k^\dagger$
10:   if local search triggered then
11:      Run gradient based refinement using single level formulation
12:      $\mathcal{P}_k^\dagger \rightarrow \mathcal{P}_k^*$
13:      Update archive $\mathcal{A}_k$ with potential field filter
14:      $\mathcal{P}_k^* \rightarrow \mathcal{P}_{k+1}$
15:   else
16:      $\mathcal{P}_k^\dagger \rightarrow \mathcal{P}_{k+1}$
17:   end if
18:   $k = k + 1$
19:   Update $\rho_{\mathcal{B}}$
20: end while
```

At the start of MACSoc, N_a candidate solutions are generated with a Latin Hypercube sampling, associated to a population $\mathcal{P}_0$ of N_a agents, and an attempt is made to make each candidate solution feasible before the optimisation process starts (line 1 in Algorithm 5).

Both the global search and local refinement strategies implemented in MACSoc require the definition of a set of descent directions in criteria space, thus, after initialising the agents, the algorithm generates N_w uniformly spread weight vectors $\boldsymbol{\omega}$ (line 2 in Algorithm 5) that define the components of N_w descent vectors; this is explained further in Subsection 4.1.4. Each agent will be associated to a different weight vector, allowing each agent to converge to a different part of the Pareto front.

The global search generates candidate solutions, for the decision vector, using a combination of social and individualistic actions (lines 4 and 7 in Algorithm 5). Each action generates a candidate decision vector, starting from the current solution allocated to a given agent j, and submits it to a bi-level optimisation problem.

Within the bi-level optimisation, the inner level makes the candidate decision vector feasible with respect to differential, path, and boundary constraints, while the outer level assesses whether the solution of the inner level represents an improvement with respect to the current solution allocated to agent j. All feasible and non-dominated solutions update the current population $\mathcal{P}_k$ and are saved in an archive $\mathcal{A}_k$ (lines 5, 6, 8, 9 and 13 in Algorithm 5).

After every user specified number of iterations, and as a last step before the algorithm ends, the local refinement is triggered, and the archive and population are updated with the refined solutions (lines 10 to 17 in Algorithm 5). The local refinement solves a single level scalarised version of problem (4.6).

The process proceeds alternating social and individualistic actions, with periodic local refinement, until a maximum number of calls to the objective vector *max_fun_eval* is reached. The overall algorithmic complexity is dominated by the NLP solver used in the bi-level problem and for the local refinement.

In the following both the bi-level and single level problems are explained in more detail together with the heuristics used to generate new candidate solutions. The combined use of the bi-level formulation for global exploration and single level formulation for local convergence within MACSoc is one of the distinctive features of the proposed approach compared to previous works such as [49], [45], [43], [37] and [50].

4.1.1 Bi-level Global Optimisation Problem

The global search part of the algorithm solves the following two-level problem:

$$\begin{aligned}
&\min_{\mathbf{p}^*} \tilde{\mathbf{J}}(\mathbf{y}^*, \mathbf{p}^*) \\
&s.t. \\
&(\mathbf{y}^*, \mathbf{p}^*) = \operatorname{argmin}_{(\mathbf{y},\mathbf{p})} \{\delta_p(\mathbf{y},\mathbf{p}) \,|\, \mathbf{C}(\mathbf{y},\mathbf{p}) \leq 0\}
\end{aligned} \tag{4.7}$$

Problem (4.7) defines two different optimisation sub-problems at two different levels. The outer level handles the objective vector $\tilde{\mathbf{J}}$ and generates tentative decision vectors $\mathbf{p}$. Tentative solutions are then submitted to the inner level, whose goal is to find the state and control vectors $\mathbf{y}^*$ and $\mathbf{p}^*$ that satisfy constraints $\mathbf{C}$ and minimise an inner cost function $\delta_p = \|\mathbf{p}^* - \mathbf{p}\|$. Thus, the inner level will look for the closest feasible solution to the tentative one generated by the outer level. The outer level then receives the solution $(\mathbf{y}^*, \mathbf{p}^*)$ and proceeds by evaluating the objective functions associated to $\mathbf{p}^*$. The inner level problem is solved with a generic NLP solver (Matlab `fmincon` in this case).

In order to reduce the number of iterations required by the inner level to converge, the outer level stores the feasible states $\mathbf{y}^*$ at iteration k to be used as a warm start for the inner level at iteration $k+1$. As illustrated in Fig. 4.1, the feasible states $\mathbf{y}_k^*$ are preserved from iteration k to iteration $(k+1)$ and the outer level only generates a new tentative vector $\mathbf{p}_{k+1}$. The inner level at iteration $(k+1)$ will thus use $\mathbf{y}_k^*$ and $\mathbf{p}_{k+1}$ as initial guesses for states and controls. Because of the way $\mathbf{p}_{k+1}$ is generated, even if $\mathbf{y}_k^*$ is associated to $\mathbf{p}_k^*$, it works well as an initial guess also when associated to $\mathbf{p}_{k+1}$.

When individualistic actions are applied, each agent generates one or more tentative vectors through three mechanisms that are triggered one after the other in this

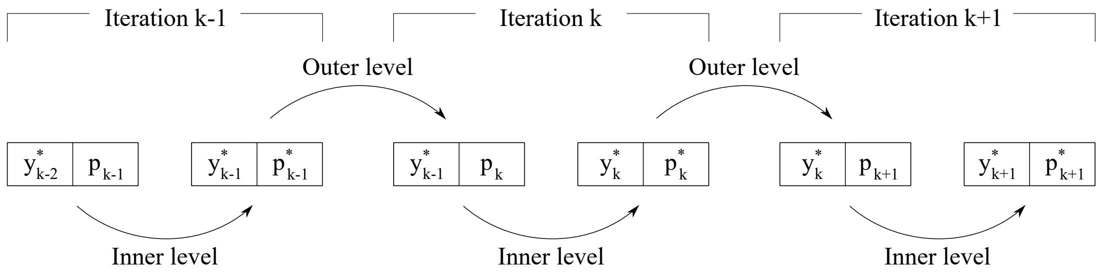


Figure 4.1: Schematic representation of the bilevel approach acting on a single solution.

order: Inertia → Pattern Search → Differential Evolution. If any of these mechanisms produces an improved solution, the following ones are not triggered and the process proceeds by updating the population and archive (line 5 and 6 of Algorithm 1).

Note that, if the inner level does not converge to the required tolerance, the objective functions of the outer level are recalculated to be the infinity norm of the constraint violation plus the maximum values of each objective functions in the archive and population. This creates an adaptive rejection mechanism: if none of the agents are feasible, the one that best satisfies the feasibility is temporarily entered in the archive with the next iterations trying to further improve feasibility.

Once an agent finds a feasible solution, it will explore the search space through the global bi-level approach, generating several feasible and non-dominated solutions. These solutions will enter in the archive because they will dominate many of the existing non-feasible ones, and thanks to the social actions some agents will be directly moved onto those solutions, allowing the whole population to converge to feasible solutions in a handful of iterations.

4.1.2 Single Level Local Search

The local refinement solves the following scalarised problem for each agent j:

$$\begin{aligned} &\min_{\epsilon \geq 0} \; \epsilon \\ &s.t. \\ &\boldsymbol{\omega}_{i,j}\,\vartheta_{i,j}(\mathbf{y},\mathbf{p}) \leq \epsilon \qquad \text{for } i = 1,\ldots,m \\ &\mathbf{C}(\mathbf{y},\mathbf{p}) \leq 0 \end{aligned} \tag{4.8}$$

where $\boldsymbol{\omega}_{i,j}$ is the i^{th} component the vector of weights $\boldsymbol{\omega}_j$, $\vartheta_{i,j}$ is the i^{th} component of the rescaled objective vector of the j^{th} agent, and ϵ is a slack variable. As it is evident, this problem is the discretised version of (4.3).

This reformulation of the problem is constraining the agent's movement, in criteria space, within the descent cone defined by the point $(\epsilon\mathbf{d}_j + \zeta_j)$ along the direction $\mathbf{d}_j = (1/\omega_{1,j},\ldots,1/\omega_{i,j},\ldots,1/\omega_{m,j})$.

The rescaled objective vector is defined as:

$$\boldsymbol{\vartheta}_j(\mathbf{y},\mathbf{p}) = \frac{\tilde{J}_{i,j}(\mathbf{y},\mathbf{p}) - \tilde{z}_i}{z^*_{i,j} - \tilde{z}_i} \qquad \text{for } i = 1,\ldots,m \tag{4.9}$$

where $\mathbf{z}^*_j$ is equal to $\tilde{\mathbf{J}}_j(\overline{\mathbf{y}},\overline{\mathbf{p}})$, $(\overline{\mathbf{y}},\overline{\mathbf{p}})$ being the initial guess for the solution of (4.8), $\tilde{\mathbf{z}} = \mathbf{z} - \mathbf{z}_A$ with $\mathbf{z}_A$ the nadir of the archive, or point whose components are the maximum values of all the components of the objective vectors in the archive.

From the normalisation one can derive the components of the vector $\boldsymbol{\zeta}_j$:

$$\zeta_{i,j} = \frac{z_i}{z^*_{i,j} - \tilde{z}_i} \qquad \text{for } i = 1,\ldots,m \tag{4.10}$$

This allows the components of $\boldsymbol{\vartheta}_j(\mathbf{y},\mathbf{p})$ to have values of 1 at the beginning of the local search, and value 0 if the agent converges to the target point $\tilde{\mathbf{z}}$. With this normalisation the single level approach avoids biases when the objectives have significantly different scales. The construction of the descent directions will be explained in

Subsection 4.1.4.

It is important to remark that Problem (3.5) and (4.8) are equivalent and lead to the same optimal solution if the target point for the Pascoletti-Serafini scalarisation coincides with the utopia point and the weight vectors are the same. As such, the following theorem holds true [15]:

Theorem 4 *A point* $(\epsilon, \mathbf{y}, \mathbf{p}) \in \mathbb{R} \times Y \times \Pi$ *is a minimal solution of* (4.8) *with* $\mathbf{z} \in \mathbb{R}^m$, $z_j < \min_{\mathbf{y} \in Y, \mathbf{p} \in \Pi} J_j(\mathbf{y}, \mathbf{p})$, $j = 1, \ldots, m$, *and* $\omega \in int(\mathbb{R}^m_+)$ *if and only if* $\mathbf{y}$ *and* $\mathbf{p}$ *are a solution of* (3.5).

While equivalent, the two formulations have different numerical implications: the Tchebychev scalarisation is unconstrained but not continuous, making it unsuitable to treat with a gradient based approach. The Pascoletti-Serafini scalarisation instead is smooth but constrained, and the tight satisfaction of those constraints might be difficult to obtain with an evolutionary algorithm. By employing (3.5) in the global search phase with (4.8) in the refinement phase, the algorithm employs the type of scalarisation most suitable for each case and realises a smooth transition from global exploration of the Pareto set to local convergence.

4.1.3 Generation of the Initial Feasible Population

Before the optimisation starts, MACSoc generates an initial population of agents $\mathcal{P}_0$ (see line 1 of Algorithm 1) with the following four-step automatic and unsupervised procedure:

1. A first guess for the decision vectors is generated with a Latin Hypercube sampling within the prescribed boundaries. State variables for each phase are initialised with a simple linear interpolation between initial and final conditions.

2. For each phase, each integral equation (2.9) is made feasible by solving only the inner level subproblem of problem (4.7) with an NLP solver.

3. Starting from the solution at step 2, for each phase, all constraints related to that phase are then included and the resulting problem is satisfied again applying the same NLP solver to the subproblem in (4.7).

4. All phases are connected together and the inter-phase constraints are satisfied applying again the same NLP solver to the subproblem in (4.7).

If, at the end of the initialisation phase, an agent is associated to a solution that is not feasible within the prescribed tolerance, that solution is still included in the initial population P_0 and submitted to the subsequent optimisation cycle.

In the following, the feasibility level required to the initial population is the same required for the rest of the algorithm, which by default is 10^{-6}, so if the NLP converges, the solution generated by this approach is a fully feasible solution. By default, the NLP solver is allowed to use a maximum number of calls to the constraint function that is equal to $10(n_b^* + n_s + n_Y)$. For this procedure of automatic guess generation, an Interior Point NLP algorithm was used because it delivered a more robust and consistent convergence to feasible solutions.

4.1.4 Definition of the descent directions and target points

The weight vectors for the bi-level global search are generated as described in Section 3.2.3. For the single level approach, the weight vectors $\boldsymbol{\omega}_j = [\sqrt{2}, \ldots, \sqrt{2}]^T$ are allocated to all agents except to those m agents that minimise each individual objective function. For these m agents the weight vectors are $\boldsymbol{\omega}_j = [0, \ldots, j, \ldots, 0]^T$ with $j = 1, ..., m$. These weights are called orthogonal because correspond to the m orthogonal directions in criteria space. If agent j associated to weight $\boldsymbol{\omega}_j$ does not generate any improvement after two iterations, a new random orthogonal weight is associated to j and problem (4.8) is solved with the added constraints:

$$\tilde{J}_i \leq z_i \;\; \forall i \neq j \tag{4.11}$$

The reason for the different choice of weight vectors between the bi-level and the single level formulation can be explained as follows: the bi-level formulation explores globally the search space with a population of agents, thus there is the need to maximise the spreading of the solutions. On the contrary, the single-level is used to improve the local convergence of each agent in a normalised criteria space. Thus the goal of the single level is to return dominating solutions without altering too much their spreading in criteria space.

4.2 Benchmark Cases

In this section the proposed approach is tested on the solution of two simple test cases: the first test case is a minimum transfer to rectilinear path and the second is the maximum energy orbit rise problem. For the former an analytical solution is available for a single objective optimisation problem (described in Chapter 2). For the latter the exact control law is known but an analytical expression for the Pareto front is not available.

4.2.1 Minimum transfers to rectilinear path

This problem is a multi-objective extension of the problem solved in Section 2.3:

$$\min_{t_f,u}(J_1, J_2)^T = (t_f, -v_x(t_f)) \tag{4.12}$$

subject to the dynamic constraints:

$$\begin{cases} \dot{x} &= v_x \\ \dot{v}_x &= a\cos u \\ \dot{y} &= v_y \\ \dot{v}_y &= -g + a\sin u \end{cases} \tag{4.13}$$

where g is the gravity acceleration, a the thrust acceleration, x and y are the components of the position vector, v_x and v_y the components of the velocity vector and u

the control. The dynamics is integrated from time $t = 0$ to time $t = t_f$. The boundary conditions are:

$$\begin{cases} x(0) = 0; & v_x(0) = 0 \\ y(0) = 0; & v_y(0) = 0 \\ y(t_f) = h; & v_y(t_f) = 0 \end{cases} \tag{4.14}$$

The parameters g, a and h were respectively set to $1.6 \cdot 10^{-3}$, $4 \cdot 10^{-3}$ and 10. The DFET method was applied splitting the time domain into 4 elements, with polynomials of order 6 for each control and state variable. Bounds on the states and control variables are reported in Table 4.1. This gives a total of 29 optimisation variables for the outer level, and 143 variables for the inner level and the refinement problem.

Table 4.1: Lower and upper bounds for the state, control and final time variables of the Minimum transfers to rectilinear path

Variable	Lower bound	Upper bound
x	-1	200
y	-10	10
v_x	-1	11
v_y	-10	10
u	$-\frac{\pi}{2}$	$\frac{\pi}{2}$
t_f	0	250

Table 4.2 summarises the settings of the optimiser: *max_fun_eval* the maximum number of objective functions evaluation, *pop_size* the number of agents performing the search, *ρ_ini* the initial radius of the local neighbourhood, F and CR the standard parameters for the Differential Evolution social actions, *p_social* the ratio between agents performing only social actions and the total number of agents, *max_arch* the number of solutions to be stored in A_g, *contr_ratio* contraction rate of the neighbourhood radius, and *max_contr_ratio* the maximum number of times ρ_j can contract before it is reset.

Settings reported in Table 4.3 instead refer to the parameters of *fmincon*: *max_con_eval* is the maximum number of constraints evaluation (for each call to the objective func-

Table 4.2: MACSoc settings

max_fun_eval	10000
pop_size	10
ρ_ini	1
F	0.9
CR	0.9
p_social	1
max_arch	10
max_contr_ratio	5

Table 4.3: *fmincon* settings

max_eval	100
tol_con	1e-6

tions) and *tol_con* is the threshold under which the solution is considered to be feasible. All other *fmincon* settings were left at default values.

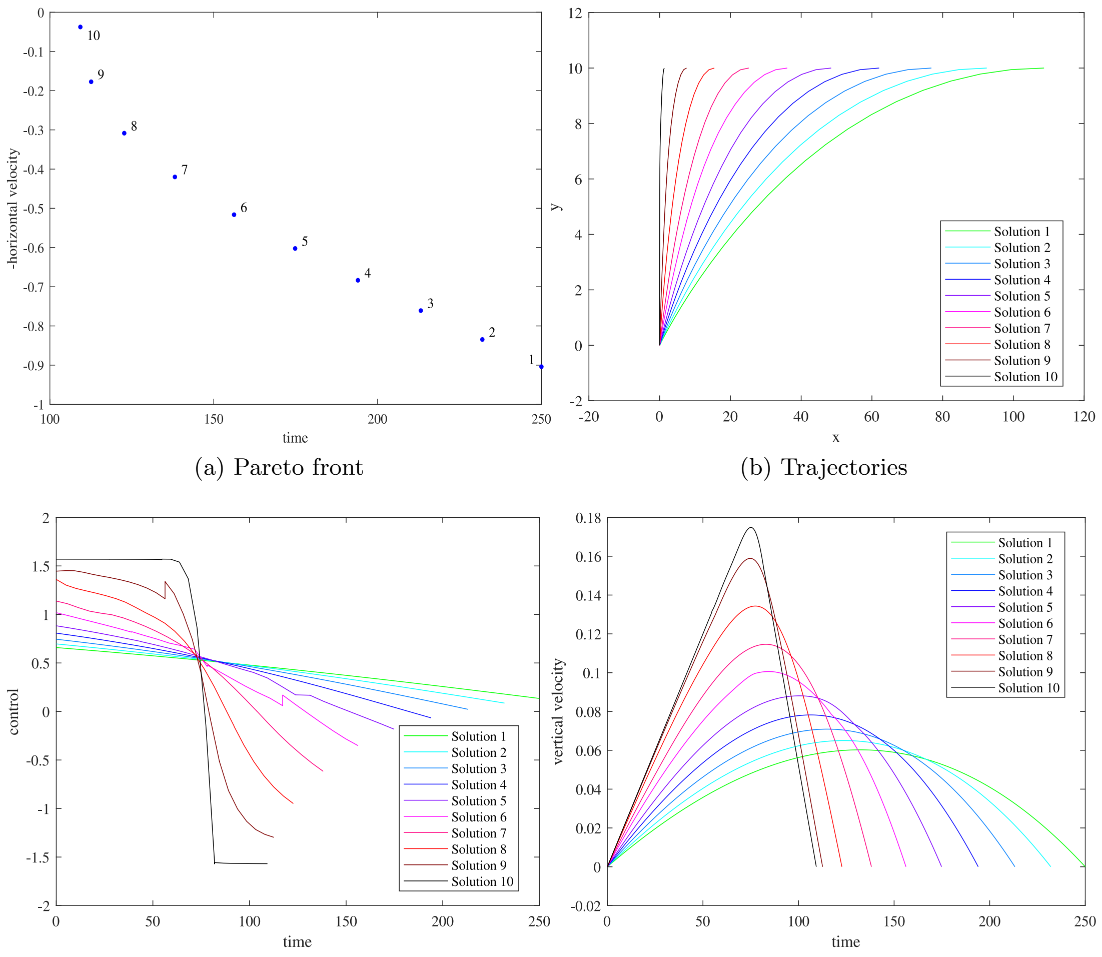
Algorithm 1 was run 30 times to collect some statistics on its convergence behaviour (see Table 4.4). Each run took approximately 10 minutes on an i7 laptop. The Generational Distance (GD) and Inverse Generational Distance (IGD) were used as accuracy metrics and were computed on a rescaled front in the interval $[0, 1]$. GD and IGD were computed using the analytical solution of the minimum time problem for different maximum v_x.

Figure 4.2a shows the Pareto front for one single run, figure 4.2b shows the trajectories of all solutions of the same run, while Figures 4.2c and 4.2d show the corresponding control laws and the vertical velocities. As it is evident, the algorithm found a well spread set of solutions on the Pareto front, which compose a family of trajectories and control laws. The minimum time solution was already obtained in Chapter 2, where it was shown that the DFET method is able to return solutions approaching to the analytical one even in the presence of discontinuities. Here, Figures 4.3a to 4.4d report the comparison between the solution computed with the proposed approach and the analytical solution with the same mission time. These comparison are very favourable, as for most part the solutions the control laws are indistinguishable, although it is

Table 4.4: Convergence and spreading statistics for the two problems

Problem	mean GD (variance)	mean IGD (variance)
Goddard	2.833e-2 (1.4232e-5)	2.9449e-2 (1.5498e-5)
Orbit	5.9444e-2 (1.96243e-4)	4.387e-2 (1.6173e-4)

possible to see some imperfections in the control where the NLP solver stopped before reaching full optimality. It is expected that the quality of the solutions and the values of the objectives will improve slightly with the use of a more refined mesh.



(a) Pareto front (b) Trajectories

(c) Controls (d) Vertical velocities

Figure 4.2: Pareto Front, trajectories, controls and vertical component of velocities for a single run of the minimum transfers to rectilinear path problem

4.2.2 Maximum Energy Orbit Rise

The original maximum energy orbit rise formulation and some solution strategies can be found in [106] and [70]. In this case, a spacecraft is orbiting around a celestial body, and it is required to increase its total energy by changing its altitude and velocity. The only control variable is the thrusting angle, and the only other force affecting the spacecraft is gravity.

The multi-objective extension, proposed here, maximises the final energy and minimises the manoeuvre time:

$$\min_{t_f,u}\left(J_1 = t_f, J_2 = -\frac{\left(v_r^2(t_f) + v_t^2(t_f)\right)}{2} + \frac{1}{r(t_f)}\right) \tag{4.15}$$

subject to the dynamic constraints:

$$\begin{cases} \dot{r} & = v_r \\ \dot{v}_r & = \dfrac{v_t^2}{r} - \dfrac{1}{r^2} + a\cos u \\ \dot{\theta} & = \dfrac{v_t}{r} \\ \dot{v}_t & = -\dfrac{v_t v_r}{r} + a\sin u \end{cases} \tag{4.16}$$

where r and θ are the polar coordinates of the spacecraft, v_r and v_t are the radial and transversal velocities, and a is the magnitude of the control acceleration. In this work, $a = 1e-2$.

The boundary conditions are:

$$\begin{cases} r(0) = 1.1; \quad v_r(0) = 0 \\ \theta(0) = 0; \qquad v_t(0) = \dfrac{1}{\sqrt{1.1}} \end{cases} \tag{4.17}$$

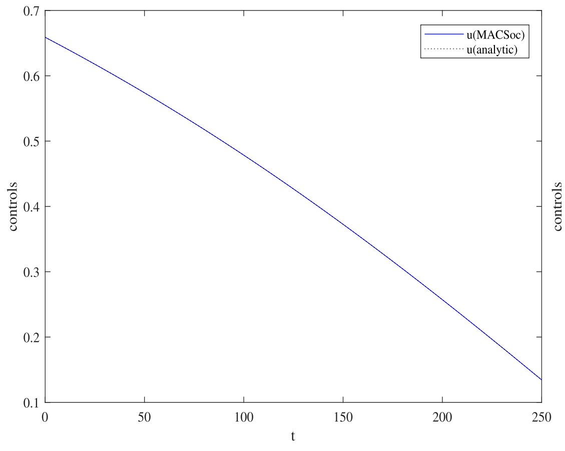
Following [70], the time domain was subdivided into 30 elements of order 1 for each state and control variable. Lower and upper bounds for the state and control variables are reported in Table 4.5. In total there are 61 optimisation variables for the outer level and 305 variables for the inner level and refinement problem. The transcribed problem was then optimised with MACSoc, with the same settings as in the previous case.

Table 4.5: Lower and upper bounds for the state, control and final time variables of the Maximum Energy Orbit Rise problem

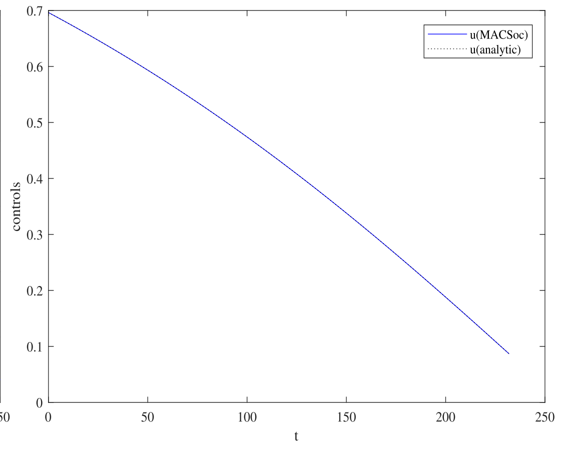
Variable	Lower bound	Upper bound
r	0.1	20
θ	$-\pi$	10π
v_r	-1	1
v_t	-1	1
u	$-\frac{\pi}{2}$	$\frac{\pi}{2}$
t_f	0	80

The GD and IGD of the combined 30 runs are reported in Table 4.4. Runtime for this case was similar to the previous case. Figure 4.5a shows the Pareto front for one of those runs, Figure 4.5b shows the corresponding trajectories, Figure 4.5c the control laws and Figure 4.5d the radial velocities. Also in this case, the algorithm automatically produced a well spread set of solutions.

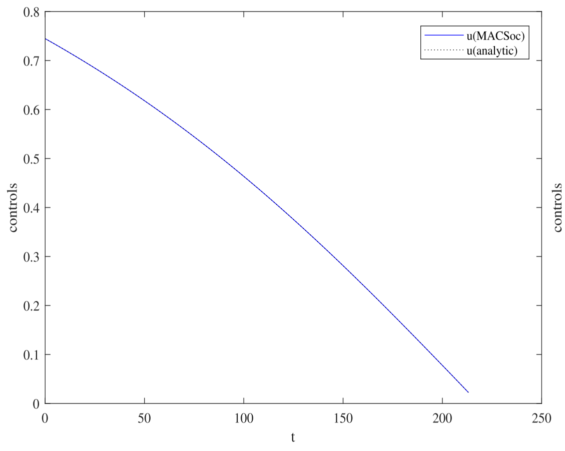
In order to assess the correctness of the computed solution, the problem was solved again in its single objective version, looking for the solution maximising the terminal energy with a transfer time equal to each of the solutions computed by the multi-objective case. Figures 4.6a to 4.7d show this comparison, where very good agreement can be found in most of the time laws. Also in this case, the imperfections are produced by an incomplete convergence of the NLP solver.



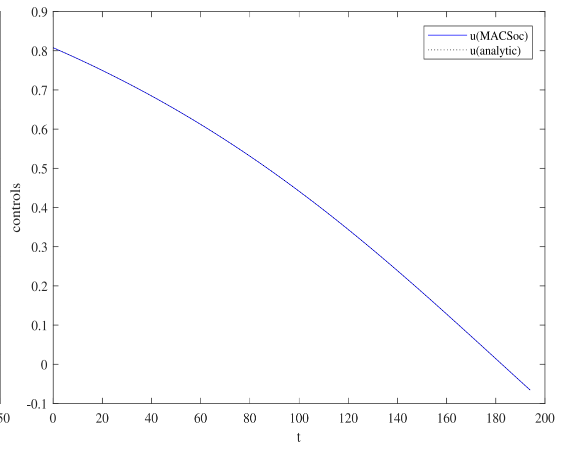
(a) Solution 1



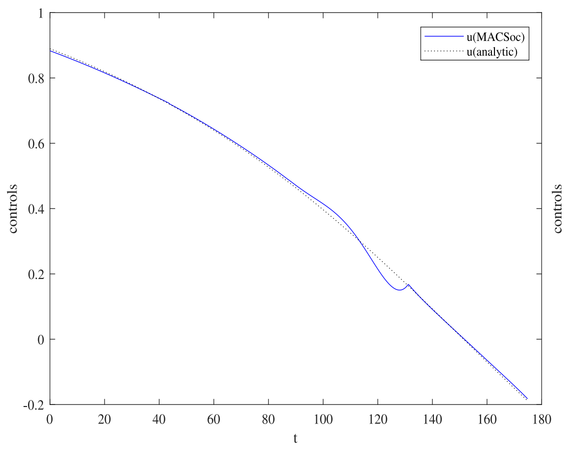
(b) Solution 2



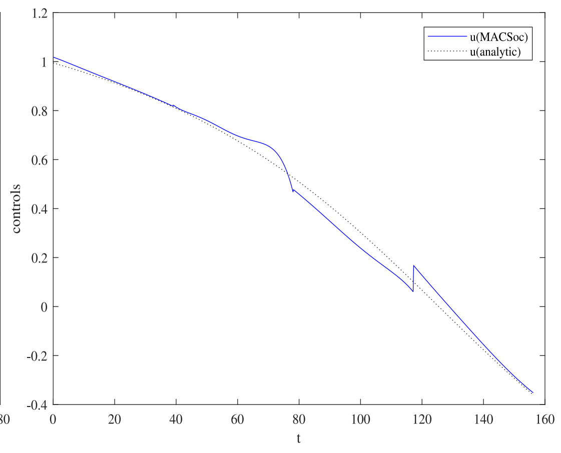
(c) Solution 3



(d) Solution 4

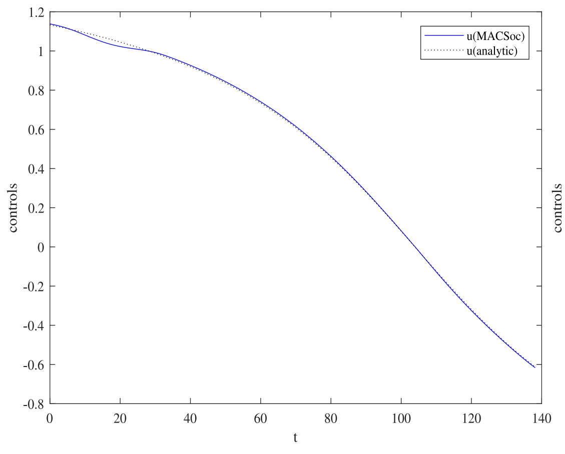


(e) Solution 5

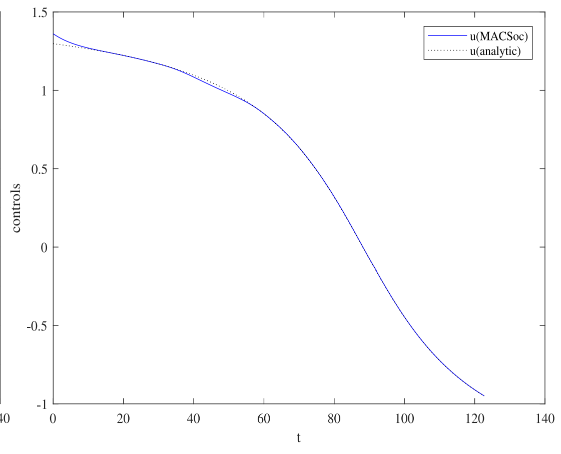


(f) Solution 6

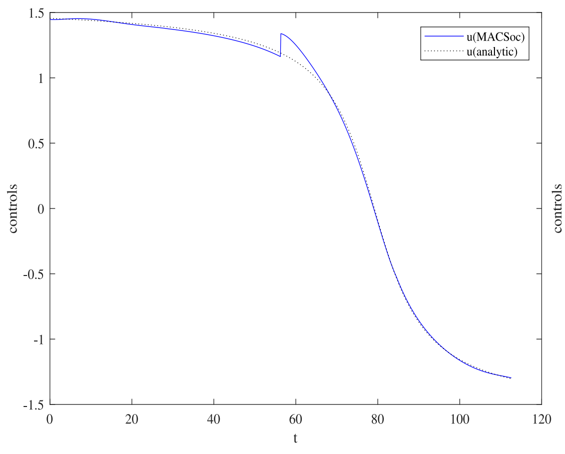
Figure 4.3: Comparison between solutions of the multi-objective problem vs the single objective problem



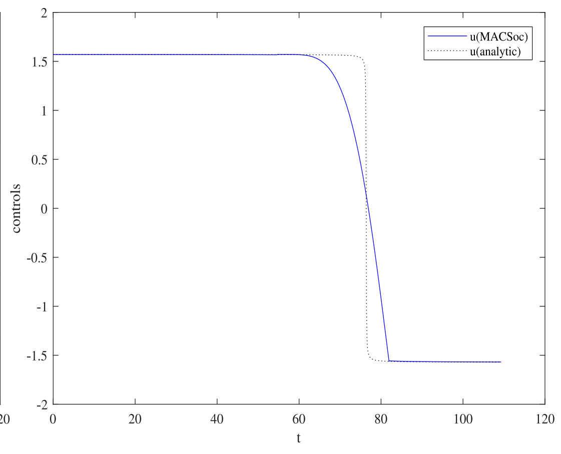
(a) Solution 7



(b) Solution 8

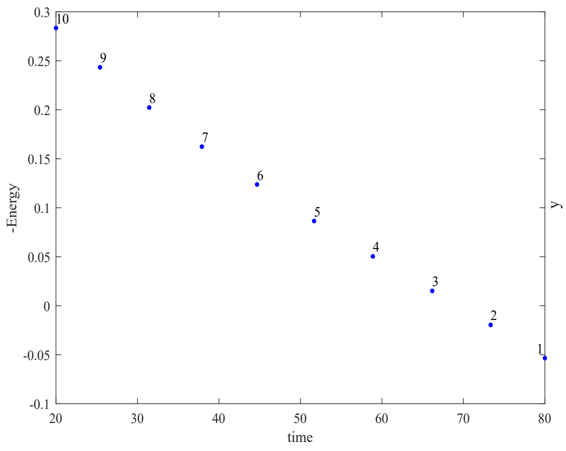


(c) Solution 9

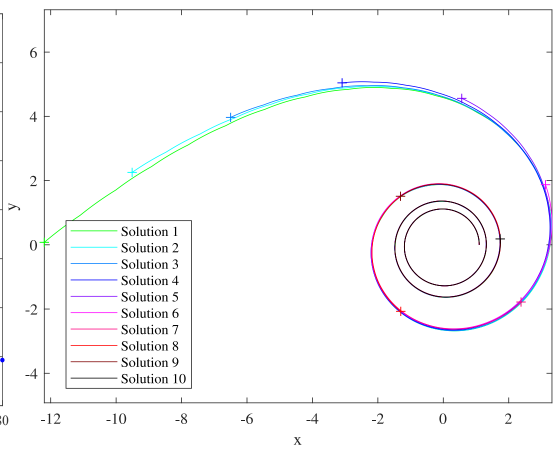


(d) Solution 10

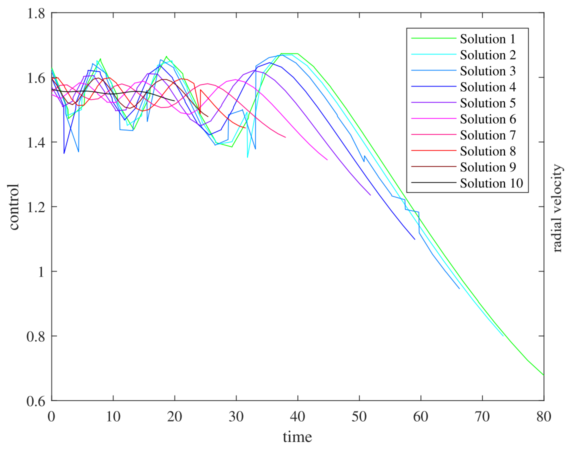
Figure 4.4: Comparison between the controls obtained with the present approach vs the analytic solution



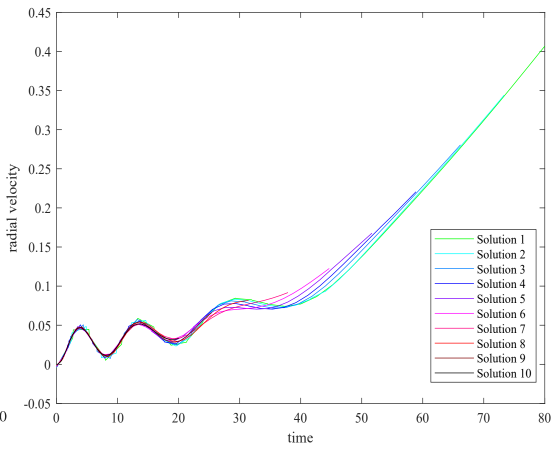
(a) Pareto front



(b) Trajectories

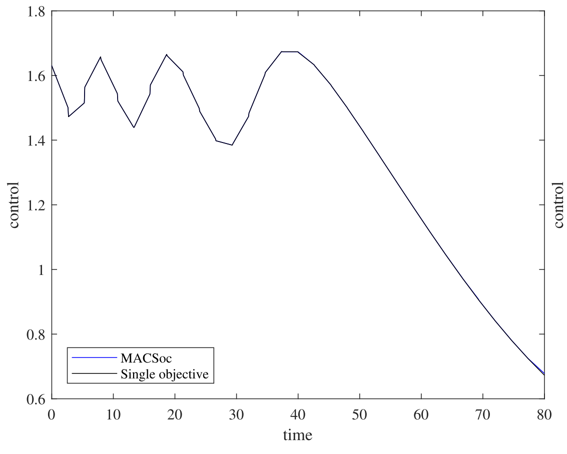


(c) Controls

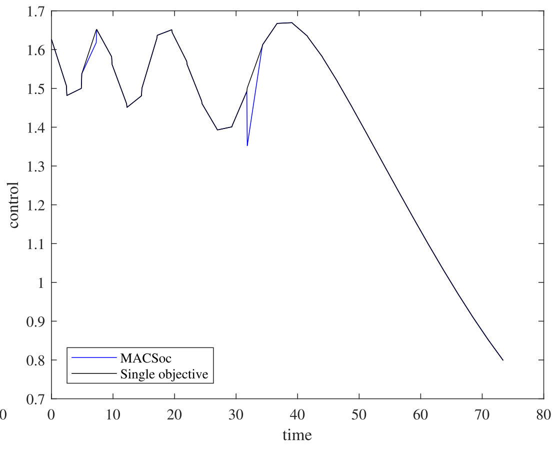


(d) Radial velocity

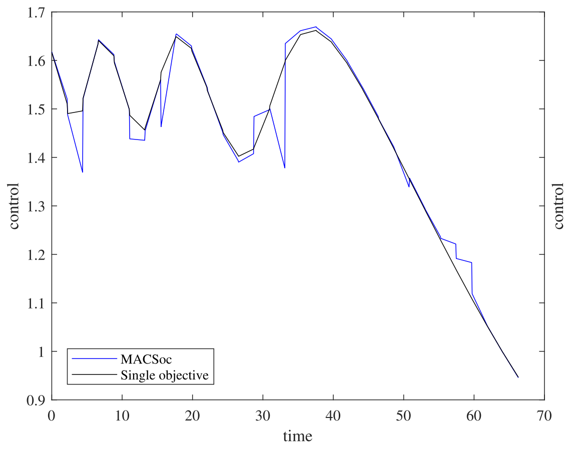
Figure 4.5: Pareto front, trajectories, controls and radial velocities for one single run of the orbit rise problem



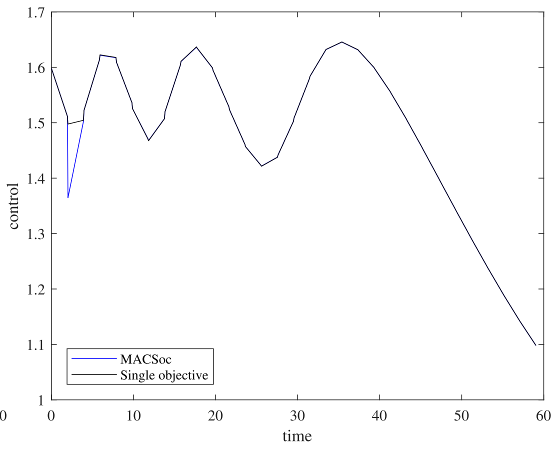
(a) Solution 1



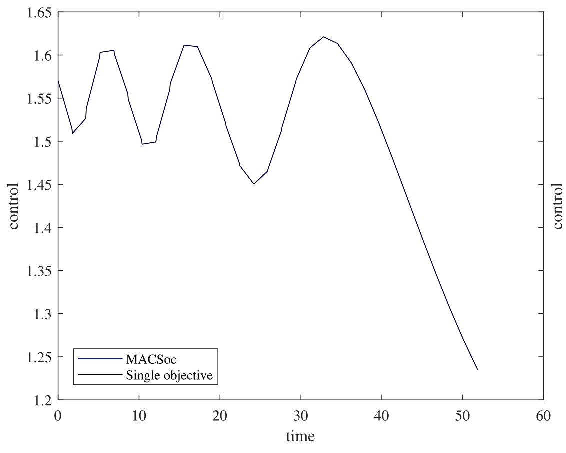
(b) Solution 2



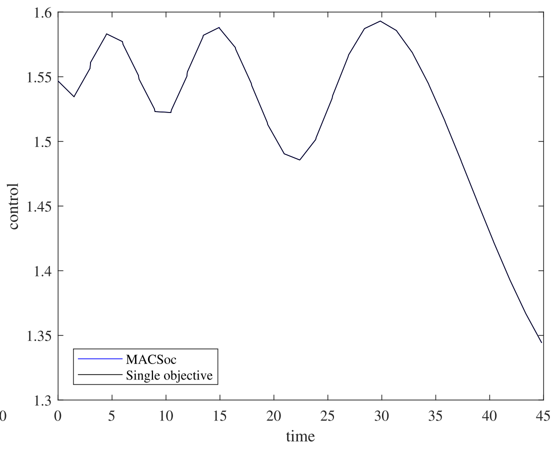
(c) Solution 3



(d) Solution 4

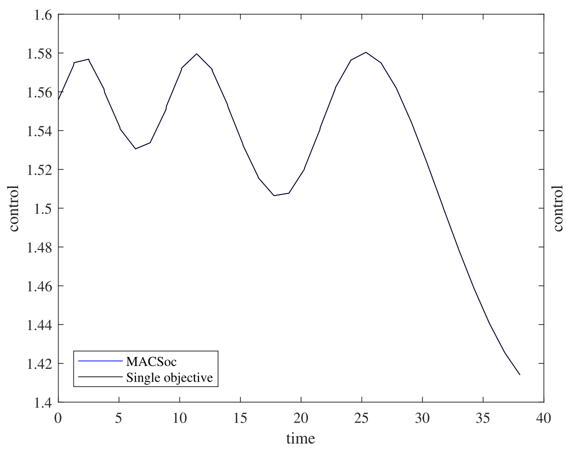


(e) Solution 5

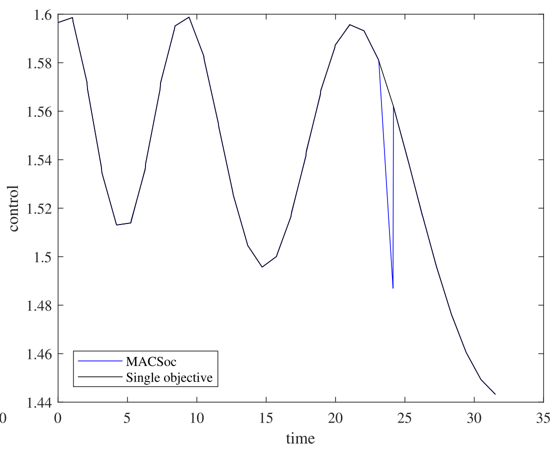


(f) Solution 6

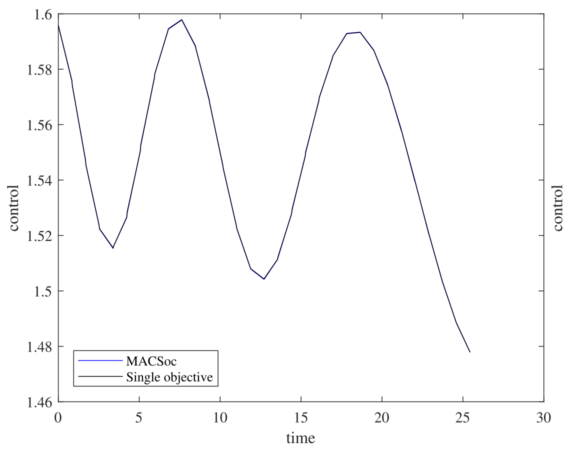
Figure 4.6: Comparison between controls of the multi-objective problem vs the single objective problem



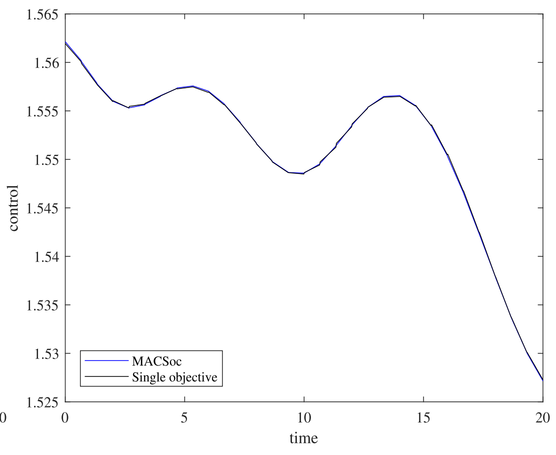
(a) Solution 7



(b) Solution 8



(c) Solution 9



(d) Solution 10

Figure 4.7: Comparison between solutions of the multi-objective problem vs the single objective problem

4.3 Chapter summary

This chapter presented some theoretical results for multi-objective optimal control and introduced MACSoc, a framework to solve Multi-objective Optimal Control problems exploiting some of those theoretical results. It is based on the coupling of DFET transcription described in Chapter 2 and the multi-objective optimisation algorithm MACS described in Chapter 3. DFET transcribes the multi-objective optimal control problem into a finite dimensional MNLP problem, which is then solved through MACSoc.

Two complementary strategies are employed to solve the MNLP problem: a bi-level approach, which allows for a global exploration of the search space and returns a well spread set of solutions, and a single level approach, which performs a gradient based refinement returning solutions with guaranteed local optimality. MACSoc is able to transition smoothly between the two approaches, thanks to the joint use of the Tchebychev and Pascoletti-Serafini scalarisation. In addition, MACSoc does not require any user specified guess.

As shown in the two numerical examples of this Chapter, MACSoc is able to return well spread set of solutions to simple problems. The solutions found match with either the analytical solution or with known and documented numerical solutions. Significantly more advanced applications of MACSoc will be described in Part II.

Chapter 5

Multi Objective Hybrid optimal control[1]

Flectere si nequeo superos,
Acheronta movebo
If I cannot bend the heavens
above, I will move Hell

Virgil

This chapter further extends the algorithm developed in Chapter 4 to introduce discrete decision variables, resulting in mixed-integer, or hybrid, optimal control problems. The simultaneous presence of discrete and continuous optimistion variables produces a significant increase in complexity of the problems to be solved: even for linear problems, the worst case scenario requires to evaluate a number of solutions which grows exponentially with the number of discrete variables. Moreover Leyffer [107] showed that even though a convex NLP has only one globally optimal solution, the same is not true for a strictly convex mixed integer NLP. These issues culminate in a Theorem, proven by Jeroslow [108], which states that the general mixed integer NLP problem can be undecidable or not computable. Despite this very discouraging result, solutions

[1]The contents of this chapter were published in: L. A. Ricciardi, M. Vasile, A Relaxation Approach for Hybrid Multi-Objective Optimal Control: Application to Multiple Debris Removal Missions. 29th AAS/AIAA Space Flight Mechanics Meeting, Ka'anapali, HI, Jan 2019, AAS 19-406.

of practical interest can often be obtained by the use of heuristic methods combining optimal control and global search methods.

5.1 The mixed integer optimal control problem

Problem 4.1, introduced in Chapter 4, can be extended to include the presence of integer variables as:

$$\begin{aligned}
&\min_{\mathbf{u}\in U,\boldsymbol{\sigma}\in\sigma,\mathbf{b}\in B,\boldsymbol{\mu}\in M} \mathbf{J}\\
&s.t.\\
&\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x},\mathbf{u},\boldsymbol{\sigma},\mathbf{b},\boldsymbol{\mu},t)\\
&\mathbf{g}(\mathbf{x},\mathbf{u},\boldsymbol{\sigma},\mathbf{b},\boldsymbol{\mu},t) \leq 0\\
&\boldsymbol{\psi}(\mathbf{x}(t_0),\mathbf{x}(t_f),\mathbf{u}(t_0),\mathbf{u}(t_f),\boldsymbol{\sigma}(t_0),\boldsymbol{\sigma}(t_f),\mathbf{b},\boldsymbol{\mu},t_0,t_f) \leq 0\\
&t \in [t_0,t_f]
\end{aligned} \tag{5.1}$$

where $\mathbf{J} = [J_1, J_2, ..., J_i..., J_m]^T$ is, in general, a vector of objectives J_i that are functions of the state vector $\mathbf{x} : [t_0, t_f] \to \mathbb{R}^{n_x}$, continuous control variables $\mathbf{u} \in L^\infty(U \subseteq \mathbb{R}^{n_u})$, discrete control variables $\boldsymbol{\sigma} \in \sigma \subseteq \mathbb{Z}^{n_\sigma}$, continuous static parameters $\mathbf{b} \in B \subseteq \mathbb{R}^{n_b}$, discrete static parameters $\boldsymbol{\mu} \in M \subseteq \mathbb{Z}^{n_z}$ and time t. The functions $\mathbf{x}(t)$ belong to the Sobolev space $\mathcal{W}^{1,\infty}$ while the objective functions are $J_i : \mathbb{R}^{3n_x} \times \mathbb{R}^{n_u} \times \mathbb{Z}^{n_\sigma} \times \mathbb{R}^{n_b} \times \mathbb{Z}^{n_\mu} \times [t_0, t_f]^2 \longrightarrow \mathbb{R}$. The objective vector is subject to a set of dynamic constraints with $\mathbf{F} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{Z}^{n_\sigma} \times \mathbb{R}^{n_b} \times \mathbb{Z}^{n_\mu} \times [t_0, t_f] \longrightarrow \mathbb{R}^{n_x}$, algebraic constraints $\mathbf{g} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{Z}^{n_\sigma} \times \mathbb{R}^{n_b} \times \mathbb{Z}^{n_\mu} \times [t_0, t_f] \longrightarrow \mathbb{R}^{n_g}$, and boundary conditions $\boldsymbol{\psi} : \mathbb{R}^{2n_x} \times \mathbb{R}^{2n_u} \times \mathbb{Z}^{2n_\sigma} \times \mathbb{R}^{n_b} \times \mathbb{Z}^{n_\mu} \times [t_0, t_f]^2 \longrightarrow \mathbb{R}^{n_\psi}$, where $\mathbf{F}(\mathbf{x},\mathbf{u},\boldsymbol{\sigma},\mathbf{b},\boldsymbol{\mu},t)$, $\mathbf{g}(\mathbf{x},\mathbf{u},\boldsymbol{\sigma},\mathbf{b},\boldsymbol{\mu},t)$ and $\boldsymbol{\psi}(\mathbf{x}(t_0),\mathbf{x}(t_f),\mathbf{u}(t_0),\mathbf{u}(t_f),\mathbf{b},\boldsymbol{\mu},t_0,t_f)$ are vector fields.

As for the previous Chapter, when N_p phases are present, dynamic constraints, path and boundary constraints, and objective functions are defined on each timeline. In order to connect different timelines, a set of N_{ip} inter-phase constraints are introduced:

$$\psi_{s_p}\left(\mathbf{x}_{0,\mathbf{I}_{s_p}}, \mathbf{x}_{f,\mathbf{I}_{s_p}}, \mathbf{u}_{0,\mathbf{I}_{s_p}}, \mathbf{u}_{f,\mathbf{I}_{s_p}}, \boldsymbol{\sigma}_{0,\boldsymbol{I}_{s_p}}, \sigma_{f,\mathbf{I}_{s_p}}, \mathbf{b}_{\mathbf{I}_{s_p}}, \boldsymbol{\mu}_{\mathbf{I}_{s_p}}, t_{0,\mathbf{I}_{s_p}}, t_{f,\mathbf{I}_{s_p}}\right) \leq 0 \quad s_p = 1, ..., N_{ip} \tag{5.2}$$

where the index vector $\mathbf{I}_{s_p}$ collects all the indexes of the phases that are connected by constraint ψ_{s_p}. Note that the number of phases is fixed, but their temporal order is actually defined by the inter-phase constraints (5.2).

The problem can be discretised following the procedure for DFET transcription illustrated in the previous chapters. This results in a Multi Objective Mixed Integer Nonlinear Programming (MOMINLP) problem coming from the transcription of problem (5.1), with the inclusion of interphase constraints (5.2). In vector form it can be written as:

$$\begin{aligned}&\min_{\mathbf{y}\in Y,\mathbf{p}\in\Pi,\mathbf{d}\in D} \tilde{\mathbf{J}}\\ &s.t.\\ &\mathbf{C}(\mathbf{y},\mathbf{p},\mathbf{d})\leq 0\end{aligned} \tag{5.3}$$

where $\mathbf{y}$ collects all the discretised state variables, $\mathbf{p}$ collects all the static and discretised dynamic control variables, $\mathbf{d}$ collects all the integer valued control variables and static parameters, and $\mathbf{C}$ collects all constraints, including boundary and interphase conditions.

5.2 Solution of the mixed integer optimal control problem

Problem 5.3 is solved by extending the bi-level and single level approaches defined in the previous chapter. In particular, for the outer layer of the bi-level approach, the heuristics of MACSoc will be extended to return integer values when operating on discrete parameters. For the inner level and single level approaches however this extension is not as straightforward since they operate with an NLP solver, which are generally not able to handle discrete parameters directly.

In the context of NLP solvers there are two possible strategies to deal with mixed integer problems. The first one is to remove the discrete variables from the solution vector and reintroduce them in the problem as constants. This has the undesirable consequence that the solution vector has to be manipulated very often to exclude and re-include the discrete variables. In addition, since the discrete variables would be kept constant, it might be impossible for the NLP solver to satisfy some constraints involving them.

The second strategy, adopted here, is to relax the discrete variables. This means that, within the inner level of the bi-level approach and the single level refinement, the discrete variables are treated as continuous variables with the same bounds as their discrete counterpart. This removes the need to exclude and re-include repeatedly those variables from the solution vector. Moreover, the NLP solver has more possibilities to satisfy the constraints because it can act upon the relaxed variables. However, the NLP solver will likely converge to a solution where the relaxed variables do not assume integer values. For this reason, the solution of the relaxed problem will be used as an initial guess for the solution of a new problem which will include additional constraints. These constraints are imposed to enforce the NLP solver to converge to a solution where the relaxed variables assume only integer values. The following sections will explain in detail the modifications applied to MACSoc in order to deal with mixed integer problems and the additional constraints imposed on the NLP solver.

5.2.1 Modification of the Heuristics of MACS

In order to preserve the behaviour and performances of the algorithm, the heuristics of MACS have been modified in the simplest, least invasive way possible. This means that no additional heuristic was introduced to deal with the discrete variables.

Inertia

The idea of this heuristic is to proceed searching along the same promising direction found at the previous iteration. The behaviour of this heuristic is thus strongly dependant on the heuristic which found the promising search direction at the previous iteration, i.e. either the Pattern Search or the Differential Evolution. However, since inertia produces a sample with a random step length along the chosen direction, it could result in the generation of non integer values for the discrete variables. For this reason, the inertia move is applied normally, but the components of the displacement vector corresponding to the discrete variables are rounded towards the closest integer. In case the rounding towards the closest integer would result in a zero displacement vector, a rounding towards the next larger integer is performed.

Pattern Search

This heuristic simply changes one component of the solution vector at a time, by a random amount:

$$x_{trial,j} = x_{ij} + \alpha \Delta_j \rho_i \tag{5.4}$$

where the component j of the solution vector x_i is perturbed by a random amount α scaled by the size of the interval for variable j Δ_j, and by a contraction factor ρ_i, which is decreased or increased at each iteration depending on whether a better solution is found or not.

In case it is currently operating on a discrete variable, the modification consists in rounding the perturbation $\alpha\Delta_j\rho_i$ to the closest integer. Similarly to the Inertia case, if this rounding would result in a zero length move, a rounding towards the next integer is performed instead.

Differential Evolution

This heuristic simultaneously changes all the components of the solution vector by combining it with other three solution vectors.

$$\mathbf{x}_{trial} = \mathbf{x}_i + \alpha \mathbf{e} \left((\mathbf{x}_i - \mathbf{x}_{i_1}) + F \left(\mathbf{x}_{i_2} - \mathbf{x}_{i_3} \right) \right) \tag{5.5}$$

where solution vector x_i is perturbed by a random amount α scaled by a displacement vector generated by three randomly chosen solution vectors x_1, x_2 and x_3, which must all be different, and multiplied by a mask vector $\mathbf{e}$ which randomly assumes component values of 0 or 1.

In order to ensure that the discrete variables remain discrete, the same procedure for the Inertia case is adopted: the components of the displacement vector associated to the discrete variables are rounded towards the closest integer. In case this rounding would result in a zero displacement, a rounding by excess would be performed instead.

5.2.2 Relaxation and Integrality constraints

As previously mentioned, within the inner level and the single level refinement the discrete variables are relaxed and treated as continuous. This will likely generate a solution where the relaxed variables do not assume integer values. In order to force the NLP solver to converge to solutions where the relaxed variables assume integer values, the following constraint is imposed

$$\sin(\pi\tilde{\sigma}_j) = 0 \tag{5.6}$$

where $\tilde{\sigma}_j$ is a relaxed variable.

This simple and smooth constraint is satisfied only for integer values of the relaxed variable σ_j. However, since this constraint has many possible solutions, it can introduce several local optima. For this reason it is enforced only after a solution to the relaxed problem is found. This delayed imposition of the constraint should allow the NLP solver to avoid the local minima introduced by this constraint.

5.3 Benchmark Cases

This section presents a test case for mixed integer optimal control problems, with two variants. The problem is a multi-objective extension of the one described by Von Stryk and Glocker [109]: a vehicle starts from the origin of the plane and has to visit three target positions before returning to the initial position. The order in which the targets are visited is not specified. Thus the problem can be considered as an extended Travelling Salesman Problem which includes the dynamics of the car of the Salesman. As will be evident, albeit this problem is extremely simple both in terms of combinatorial complexity and in terms of the dynamical model of the vehicle, the coupling of the two aspects results in a surprisingly rich and interesting problem. In the second version of the test case, the targets do not have a fixed position in space but follow a predefined trajectory. This makes the problem even more challenging and rich.

The objectives are the minimisation of the total time for the tour, and the minimisation of the energy required:

$$\min_{t_f,\mathbf{u},\boldsymbol{\mu}} (J_1, J_2)^T = \left(t_f, \int_{t_0}^{t_f} u_1^2 dt \right)^T \tag{5.7}$$

The dynamics of the vehicle are the following:

$$\begin{cases} \dot{x} & = v\cos(\alpha) \\ \dot{y} & = v\sin(\alpha) \\ \dot{v} & = u_1 \\ \dot{\alpha} & = u_2 \end{cases} \tag{5.8}$$

where x and y denote the position of the vehicle on the plane, v the magnitude of its velocity and α the direction of the velocity vector. The vehicle is controlled by the magnitude of the acceleration u_1 and by the steering rate u_2, both of which are limited: $u_1^2 \leq 1$, $u_2^2 \leq 1$

The vehicle starts at the origin at rest, and has to pass on 3 targets, whose locations are

$$\begin{cases} \mathbf{P}_1 = & (1,2) \\ \mathbf{P}_2 = & (2,2) \\ \mathbf{P}_3 = & (2,1) \end{cases} \tag{5.9}$$

The order in which the targets need to be visited is not specified, and no restriction is imposed on the velocity of the vehicle when visiting the targets. At the end of the journey, the vehicle has to return at the origin with 0 velocity. The problem can be formulated as a 4 phase problem: the first phase starts from the origin and has unknown target, the second and third phases start from the destination of the previous phase with the same velocity and direction and have unknown target, and the last phase has to return to the origin with 0 final velocity.

The following final conditions are imposed for the first three phases:

$$\mathbf{x}(t_f;i) = \mu_{1,i}\mathbf{P}_1 + \mu_2\mathbf{P}_{2,x} + \mu_{3,i}\mathbf{P}_3 \tag{5.10}$$

where $\mathbf{x}(t_f;i)$ indicate the positions at the final time of the phase i, and $\mu_{1,i}$, $\mu_{2,i}$ and $\mu_{3,i}$ are three static optimisation variables for phase i which can only assume a value of 0 or 1. In total, there are thus 9 such optimisation variables, three for each phase. Those variables can be considered as a choice on the target for each phase. In fact, if $\mu_{1,i} = 1$ while $\mu_{2,i} = 0$ and $\mu_{3,i} = 0$, the constraint will impose phase i to terminate at the position of target $\mathbf{P}_1$. However, the problem requires to visit each target exactly once. To achieve that, a set of constraints needs to be imposed. The next section will discuss these constraints in detail.

This formulation of the final conditions was given by [109], and for positive and real values of $\mu_{i,j}$ between 0 and 1 can be seen as a convex linear combination of the final states. This strategy can be applied in general when the problem involves functions of discrete parameters that cannot be relaxed. This could happen for example when one function is an external simulation or compiled code $s(x, u, t, d) = 0$ that can only take discrete inputs or specific keywords and flags d. In those cases, it is still possible to employ a relaxation approach and to express the final function as a linear combination, or superposition, of all possible function calls each with a different discrete input [59]. The disadvantage of this approach is that all possible combinations of discrete inputs have to be called each time, even if most terms of the linear combination will then be multiplied by a $\mu_i = 0$. This could become prohibitive if the number of combinations of discrete parameters is very large, but is not a major issue otherwise.

Treatment of the constraints on the targets

Von Stryk and Glocker [109] proposed to arrange the discrete variables $\mu_{i,j}$ in a matrix, relax the variables $\mu_{i,j}$ into $\tilde{\mu}_{i,j}$ and impose the following set of constraints

$$\begin{cases} \sum_i \tilde{\mu}_{i,j} = 1 \ \forall j \\ \sum_j \tilde{\mu}_{i,j} = 1 \ \forall i \end{cases} \tag{5.11}$$

where $\tilde{\mu}_{i,j}$ is the relaxed i^{th} binary choice for phase j. For an n targets problem, this results in a total of $2n$ constraints. The idea behind this set of constraints is to ensure that, if the variables assume a value of 0 or 1, only one element per row and column can be 1, thus resulting in a unique choice of target per each phase.

In the work of Von Stryk and Glocker, these constraints were employed to obtain a relaxed solution which constituted the upper bound for a branch and bound method: one relaxed variable was then split into a branch where it had a fixed value of 0 and a value of 1 in the other. The branch and bound method progressed until all relaxed variables were given a value of either 0 or 1, progressively discarding regions of the search space which were considered unpromising.

This method has the advantage of not requiring any further special treatment for the discrete variables, which are treated separately to the continuous ones. However, it has two main disadvantages: the first is that in order to obtain a solution to the full non-relaxed problem it has to solve many NLPs, in the best case equal to twice the number of binary variables and in the worst case equal to performing a full enumeration of the 2^n possible combinations.

The second, less obvious disadvantage, is that the method relies on the NLP to provide the upper and lower bounds for the solution. If the problem has multiple local solutions, the bounds computed by the NLP might be incorrect and the method might discard the region of the search space where the true optimal solution lays. This will

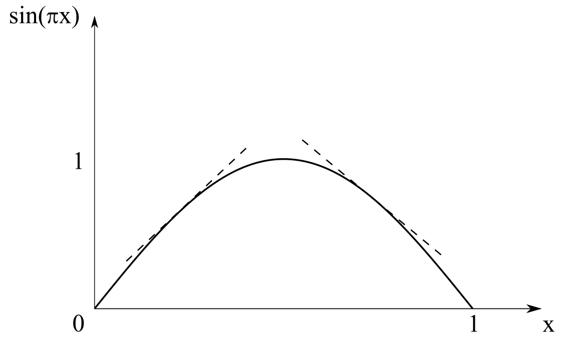


Figure 5.1: Graphic representation of constraints 5.6

be further discussed in the results section.

Unfortunately, the set of constraints (5.11) has two major problems when applied in the current framework: the first one is that it is composed by $2n$ equality constraints. When adding constraints (5.6) to enforce each $\tilde{\mu}_{i,j}$ to assume a value of 0 or 1, the problem becomes overdetermined for the $\tilde{\mu}_{i,j}$, given that constraints (5.6) already impose an equality constraint per relaxed variable. Even if the constraints are redundant at the solution points, the NLP solver might have serious difficulties in handling this situation.

A second less obvious problem is shown in Figure 5.1: one should guarantee that the $\tilde{\mu}_{i,j}$ involved in each of constraints (5.11) are not all greater, or all less, of 0.5. The reason for this is that, in order to satisfy the constraints, the NLP solver performs a linearisation of the constraints. Thus, it will try to satisfy the constraints by pushing all the $\tilde{\mu}_{i,j}$ towards either 0 or 1 depending on which side of the constraint the current solution happens to be. However, if all $\tilde{\mu}_{i,j}$ are on the same side of the constraint, the fact that the NLP will try to bring them all to 0 or 1 conflicts with the constraints $\sum_i \tilde{\mu}_{i,j} = 1$. In other words, with a poor guess it might be impossible for the NLP to find a feasible solution to an otherwise simple problem.

In order to overcome all these obstacles, another approach is proposed. First, equality constraints (5.11) are rewritten as inequality constraints

$$\begin{cases} \sum_i \tilde{\mu}_{i,j} \le 1 + \epsilon \ \forall j \\ \sum_i \tilde{\mu}_{i,j} \ge 1 - \epsilon \ \forall j \\ \sum_j \tilde{\mu}_{i,j} \le 1 + \epsilon \ \forall i \\ \sum_j \tilde{\mu}_{i,j} \le 1 - \epsilon \ \forall i \end{cases} \tag{5.12}$$

where ϵ is a fixed positive parameter lower than 1. This set of linear inequality constraints solves the issue of overdetermination of the system, because the only equality constraints remaining are from Eq. (5.6). Moreover it creates a feasible region of thickness 2ϵ around the hyperplane defined by the constraints (5.11), which should be easier to satisfy than a strict equality constraint.

However, it is still possible that all elements $\tilde{\mu}_{i,j}$ of a row or column are on the same side of the constraint (5.6), i.e. they are all lower or greater than 0.5. In order to solve this issue, an additional set of constraints is imposed:

$$\begin{cases} \sum_i \left(\tilde{\mu}_{i,j}^2 - \frac{1}{2}\right) \ge r^2 \ \forall j \\ \sum_j \left(\tilde{\mu}_{i,j}^2 - \frac{1}{2}\right) \ge r^2 \ \forall i \end{cases} \tag{5.13}$$

where r is a parameter that can be chosen between $r_{min} = \frac{1}{2}$ and $r_{max} = \frac{\sqrt{n}}{2}$ and n is the number of targets. For each row and column equality of Eq. (5.11) this additional set of constraints is creating a keep out sphere of radius r centred around the point $\left(\frac{1}{2}, \cdots, \frac{1}{2}\right)$. As shown in Figure 5.2 for the $n = 2$ case, this combination of constraints (5.12) and (5.13) has a feasible region of finite size around the vertices of a unit hypercube that also lie along the hyperplane parallel to the vector $(1, 1, \cdots, 1)$. Thus, when the integrality constraints (5.6) are imposed, the NLP solver has a good

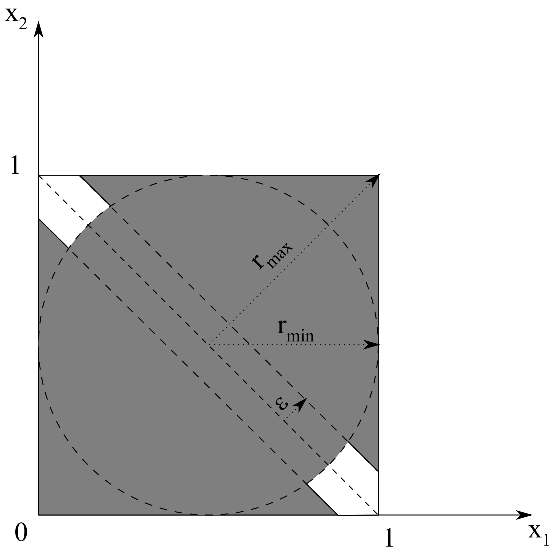


Figure 5.2: Feasible region of constraints 5.12 and 5.13

initial guess and can converge more easily towards a fully feasible solution.

It is important to remark that the feasible region of constraints (5.12) and (5.13) is disconnected and has a number of islands equal to the number of possible sequences. Thus, many different feasible and locally optimal solutions are possible and a global exploration approach is necessary.

Results

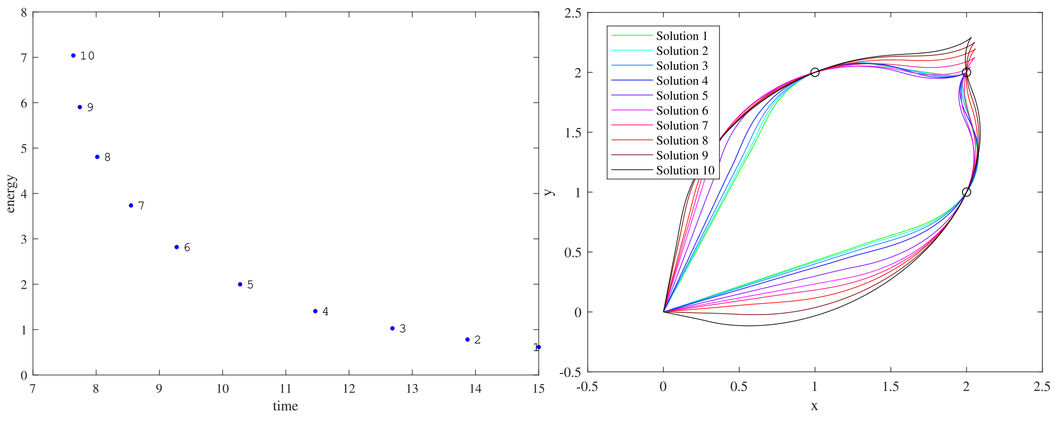
The problem was discretised using 3 DFET elements of order 7 for both states and controls and each phase, resulting in a problem with 638 variables. Bounds for the optimisation variables are reported in Table 5.1.

MACSoc was run with 10 agents for a total of 40000 function evaluations with standard settings, and the single level gradient based refinement every 10 iterations. 10 solutions were archived. Figure 5.3 shows the Pareto front and the trajectories computed. Lower time solutions require more energy, as expected. Moreover, the order in which the target locations are visited is the same for all the trajectories, starting from P_3 and proceeding to P_2 and P_1. Rather unexpectedly, lower time solutions have longer trajectories with a marked cusp, while longer time solutions have shorter quasi

Table 5.1: Lower and upper bounds for the state, control and final time variables of the motorised Travelling Salesman Problem

Variable	Lower bound	Upper bound
x (m)	-5	5
y (m)	-5	5
v ($\mathrm{m\,s^{-1}}$)	-10	10
α (rad)	$-\pi$	π
u_1 ($\mathrm{m/s^2}$)	-1	1
u_2 ($\mathrm{rad\,s^{-1}}$)	-1	1
t_f (s)	0	15

rectilinear trajectories.

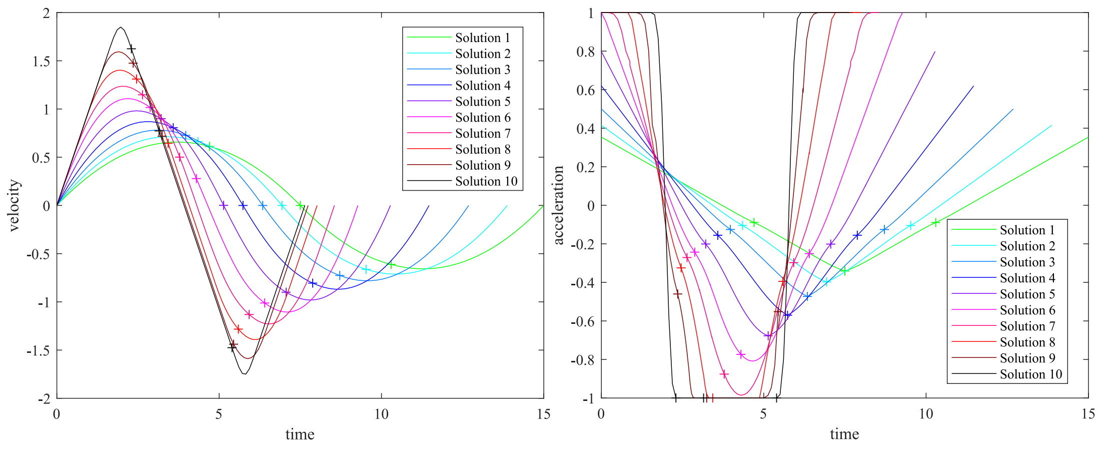


(a) Pareto front (b) Trajectories

Figure 5.3: Pareto front and Trajectories for the dynamic Travelling Salesman Problem

To explain this unintuitive result it is useful to look at Figure 5.4, which shows the time histories of the magnitude of the velocity and the corresponding control, the acceleration. As expected, the minimum time trajectories are the result of a bang-bang control profile for the accelerations. In this case, the bang-bang switch happens twice: a full acceleration is followed by a full deceleration until velocity reaches zero. After that, the acceleration is still kept negative, meaning that the vehicle will proceed backwards, until a full acceleration will stop the vehicle again when it reaches the origin. The reason behind this unintuitive backwards arc is due to the limitations in the steering rate: the maximum steering rate is not sufficient to allow the vehicle to smoothly turn and visit all three targets while also accelerating. Thus, in order to reduce time, the optimiser found a solution which is overcoming this limitation.

Low energy solutions instead have a smooth double linear profile, corresponding to a milder acceleration/deceleration profile. Also for these solutions the vehicle proceeds backwards after the second target. In particular, the vehicle stops at that point even if it was not necessary to do so. Since the velocities are much lower for these solutions, a smaller curvature radius is achievable, resulting in rectilinear trajectories for most of the interval connecting two targets.



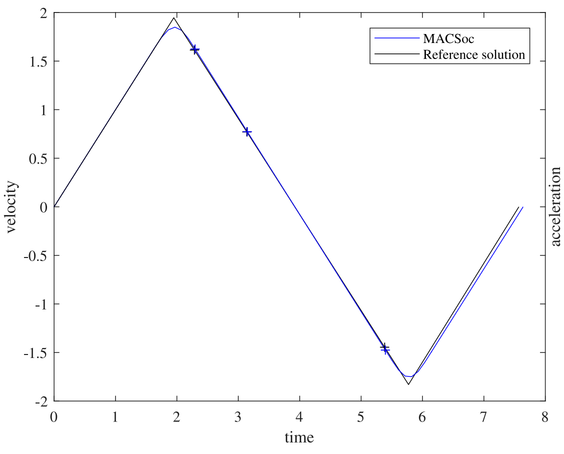
(a) Time history of the velocity (b) Time history of the acceleration

Figure 5.4: Time histories of velocity and acceleration for the Travelling Salesman Problem. + marks when a target is visited

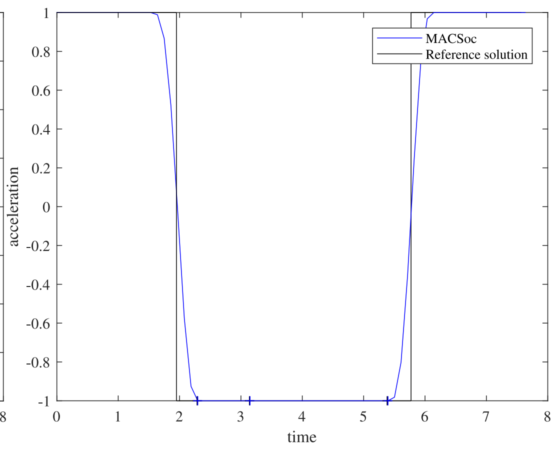
Solution 10 can be compared with the solution found in the reference [109], which was solving the minimum time problem. Table 5.2 reports a comparison of the times when each solution visits each target and the total mission time. The difference in total mission time is below 0.3% and can be attributed to the fact that the DFET transcription with Bernstein polynomials smooths the sharp discontinuities of bang-bang profiles, resulting in a slight penalisation of the objective function. This is also evident from Figure 5.5, which compares the reference solution with the solution obtained with the proposed method and shows a very good agreement.

Table 5.2: Comparison of intermediate and final times between the reference solution and solution 10 on the Pareto front

Solution	T_1	T_2	T_3	T_f
Reference	2.286	3.1390	5.3830	7.6166
MACSoc 10	2.296	3.1475	5.3958	7.639



(a) Time histories of the velocity



(b) Time histories of the acceleration

Figure 5.5: Comparison of the time histories between the reference solution and the minimum time solution computed by MACS. + marks when a target is visited

An important fact is mentioned in the reference: multiple local solutions exist even for a given order in which the targets are visited. It was not specified how the different solutions were generated, but the authors mentioned that different initial guesses had to be provided to the NLP solver in order to obtain different solutions. Since the approach proposed in this work is global and treats simultaneously both the discrete and the continuous variables, this kind of exploration is performed automatically, and, as shown, is able to return the best solution reported in the literature without requiring the user to generate different initial guesses for the NLP problem. Moreover, since the approach proposed in the reference was based on a deterministic branch and bound, it requires to compute all the solutions at the leaf level of the tree associated to the branch and bound. If the number of targets increases, this number can potentially become exceedingly large, resulting in intractable problems. With the approach proposed here, the maximum number of trials is specified a priori. Thus, even if larger problems become more and more difficult, the approach here proposed should be able to return a solution with a bounded number of function evaluations and computational time.

5.3.1 A time dependent motorised Travelling Salesman Problem

The problem described in the previous section was solved again, but this time the position of the targets was time dependent:

$$\begin{cases} \mathbf{P}_1(t) = & (0.5 + 0.5\cos(t), 1.5 + 0.5\sin(t)) \\ \mathbf{P}_2(t) = & (2 + \cos(2t), 2 - \cos(2t)) \\ \mathbf{P}_3(t) = & (1.5 + 0.5\cos(3t), 1) \end{cases} \tag{5.14}$$

The time dependence of the position of the targets makes the problem particularly challenging, since the choice of the sequence in which the targets are to be visited cannot be decoupled from time. The time laws for the position of the targets were chosen in order to have periodic motions with different amplitudes and frequencies. The trajectories of the targets are shown in Figure 5.6.

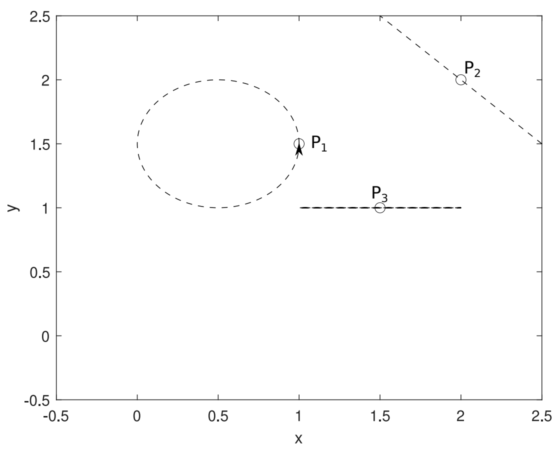
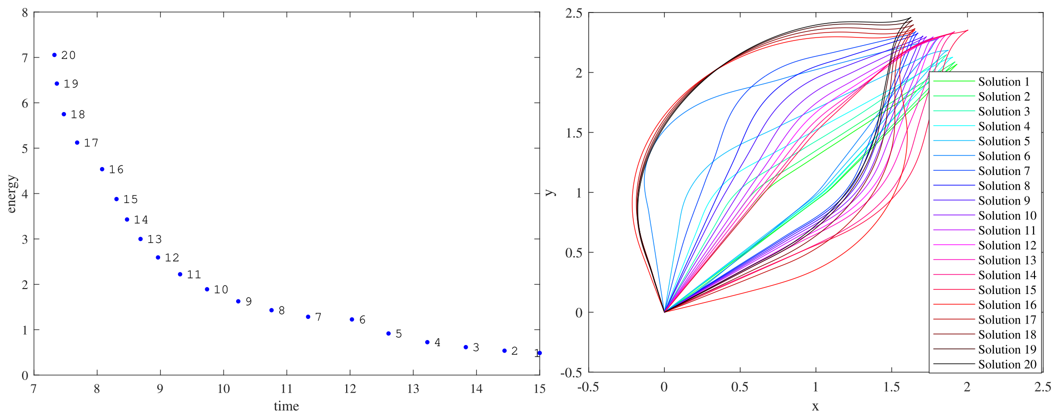


Figure 5.6: Trajectories of the targets for the time dependent motorised Travelling Salesmen Problem. Hollow circles indicate the position of the targets at $t = 0$

The problem was solved with the same settings as the previous case, except that the number of agents was increased to 20, 20 solutions were kept in the archive, and a total of 100000 function evaluations was allowed.



(a) Pareto front (b) Trajectories

Figure 5.7: Pareto front and Trajectories for the time dependent motorised Travelling Salesman Problem

Figure 5.7 shows the Pareto front and the trajectories computed. As it is evident, there is still a trade-off between time and energy, but it appears to be composed of different branches. Solutions $1-5$ and $7-15$ visit the targets in the order ($P_3 \rightarrow P_2 \rightarrow P_1$), while solutions 6 and $16-20$ visit the targets in the order ($P_1 \rightarrow P_2 \rightarrow P_3$).

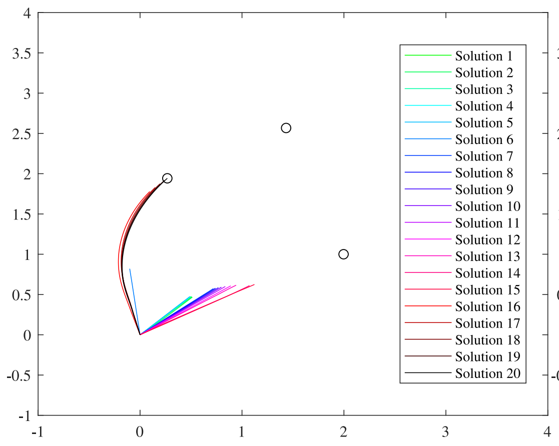
Table 5.3: Comparison of minimum time and minimum energy solution features between fixed and time dependent motorised Travelling Salesman Problem

Targets	Solution	$T(P_1)$	$T(P_2)$	$T(P_3)$	T_f	Energy
Fixed	Min T_f	5.3958 (s)	3.1475 (s)	2.296 (s)	7.639 (s)	7.040
Moving	Min T_f	2.0546 (s)	4.1399 (s)	5.5458 (s)	7.324 (s)	7.055
Fixed	Min E	10.623 (s)	7.4999 (s)	4.7023 (s)	15 (s)	0.616
Moving	Min E	11.389 (s)	7.1031 (s)	3.3213 (s)	15 (s)	0.487

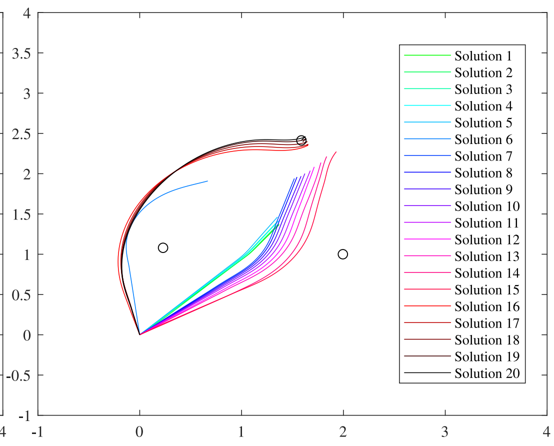
Interestingly, the minimum time solution takes less time to visit all targets than the minimum time solution of the previous case, indicating that the algorithm was able to exploit the relative motions of the targets to find a favourable timing for the various encounters, whose sequence is also different from the equivalent case of the previous problem. In order to do so, the solution also requires slightly more energy than the equivalent fixed targets case. Similarly, the minimum energy solution takes a sligthtly lower amount of energy than the fixed target minimum energy solution, while both trajectories take the same time to complete, equal to the upper bound. Table 5.3 summarises these findings.

The trajectories followed by the various solutions are also more complex than in the previous case, although they still retain a similar overall shape. In order to convey a sense of the motion of the targets and of the vehicle, Figure 5.8 shows different snapshots of all trajectories. The snapshots were taken at the instants when the fastest solution (solution 20) reaches each target and finally when it returns to the origin.

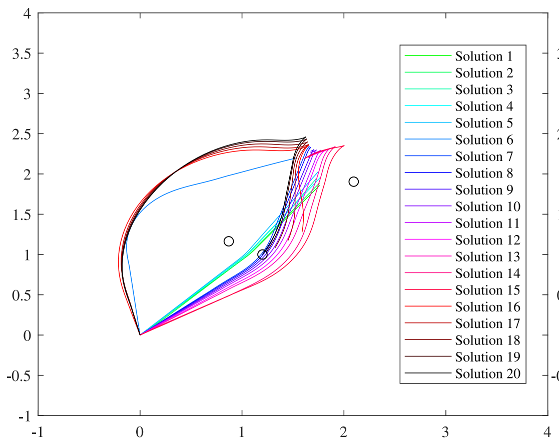
Figure 5.9 shows the time histories of the velocity and of the acceleration. The profiles are similar to the previous case, although it is possible to see some differences. In particular, solutions $1-5$, $7-15$ and $16-20$ seem to belong to different families, with solution 6 making a class of its own. The same applies for the controls, where it is also possible to detect some defects where the NLP did not manage to converge to a locally optimal solution but stopped for small step size. Again, it would be possible to improve the quality and the resolution of these solutions by a mesh refinement procedure, but the overall trade-off and the main characteristics of the solutions were



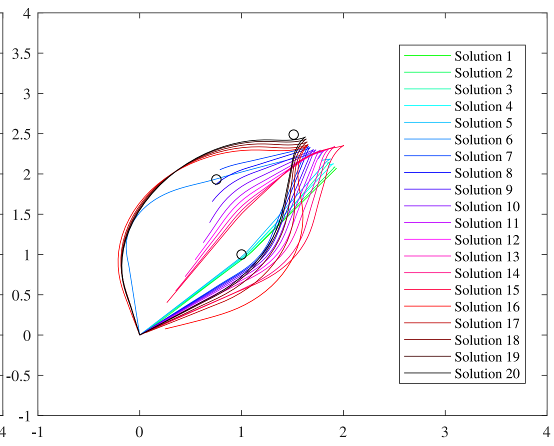
(a) t = 2.0546



(b) t = 4.1399



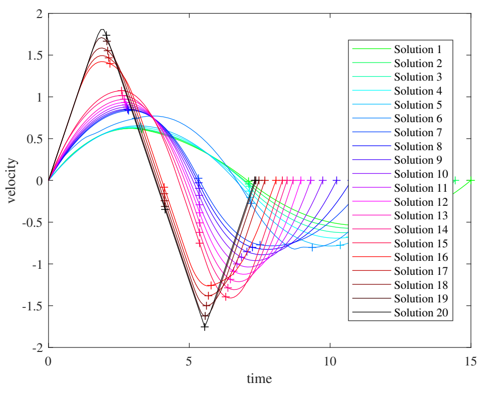
(c) t = 5.5458



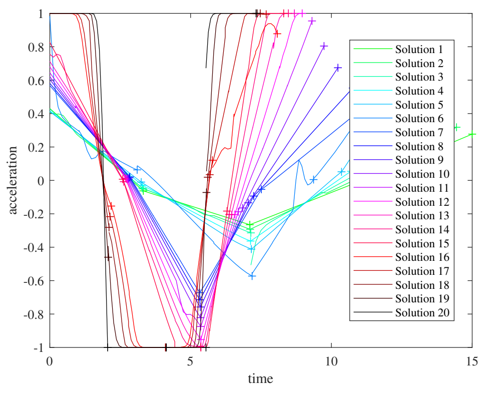
(d) t = 7.3237

Figure 5.8: Different snapshots of the trajectories for the time dependent motorised Travelling Salesman Problem

already captured at this stage.



(a) Time history of the velocity



(b) Time history of the acceleration

Figure 5.9: Time histories of velocity and acceleration for the time dependent Travelling Salesman Problem. + marks when a target is visited

5.4 Chapter summary

This chapter extended the algorithm presented in the previous chapter to include the treatment of discrete parameters. This results in mixed-integer, or hybrid, optimal control problems, which are extremely difficult to solve. In order to tackle the solution of those problems, both the multi-objective optimisation algorithm and the single and bi-level approaches were extended.

First, the heuristics of MACS were modified to deal with integer variables. The logic of the extension was to perform the minimum modifications possible, in order to retain the behaviour of the heuristics when operating on the continuous variables but ensure that their result was always integer when operating on the discrete variables.

The bi-level and single level approaches were also modified. The discrete variables were first relaxed, allowing to operate normally with the NLP solver without excluding any variable. After the relaxed problem is solved, an additional set of smooth constraints was imposed in order to enforce the relaxed variables to assume only integer values. The introduction of these constraints is delayed in order to allow the NLP solver more freedom and avoid getting trapped in local minima or in regions where no feasible solution exists.

Finally, a new set of constraints was introduced to tackle problems where the discrete variables have special exclusivity constraints, resulting in a combinatorial kind of complexity. These constraints are typical of problems like the Travelling Salesman problem, where the choice of targets is constrained by the fact that each target can be visited only once. The proposed approach, together with the new constraints, was able to solve a multi-objective extension of a Dynamic Travelling Salesman Problem with three cities, and even a more complex time-dependent version of the same problem.

Part II

Application to Space Systems

Chapter 6

Transatmospheric Vehicle and Mission Design[1]

Ipsa scientia potestas est

Knowledge itself is power

Francis Bacon

The approach to the solution of multi-objective optimal control problems described in Chapter 4 is applied to three problems of increasing complexity: the multi-objective extension of a known atmospheric re-entry problem, a 2 phase ascent problem with 3 objectives, and a 3 phase branched ascent and abort problem with 2 objectives.

The code and all the test cases were implemented in Matlab 2017b and run on a laptop with an i7 processor under Windows 10. The gradient based refinement was run every 10 iterations in all cases, and the NLP solver used in both the bi-level and single level formulations employed an SQP algorithm. The maximum number of calls to the constraint function for the NLP solver in the bi-level formulation was set equal to the number of optimisation variables for the inner level NLP, while for the single level refinement it was increased to 10 times the number of the variables.

[1]The contents of this chapter were published in: L. A. Ricciardi, C. Maddock and M. Vasile. Direct Solution of Multi-Objective Optimal Control Prolem Applied to Spaceplane Mission Design. Journal of Guidance, Control, and Dynamics, Vol. 42, No. 1 (2019), pp. 30-46.

6.1 Physical models

Before presenting the test cases, the physical models that are common to all three problems are here described.

6.1.1 Dynamical model

The vehicle dynamics are modelled as a 3DOF point mass moving in an Earth Centred Inertial reference frame (adapted from [69]),

$$\dot{r} = \dot{h} = v \sin\gamma \tag{6.1a}$$

$$\dot{\theta} = \frac{v}{r} \cos\gamma \cos\chi \tag{6.1b}$$

$$\dot{\lambda} = \frac{v}{r\cos\theta} \cos\gamma \sin\chi \tag{6.1c}$$

$$\dot{v} = \frac{T(\delta_T)\cos\alpha - D}{m} - g\sin\gamma \tag{6.1d}$$

$$\dot{\gamma} = \frac{T(\delta_T)\sin\alpha + L}{mv}\cos\sigma + \left(\frac{v}{r} - \frac{g}{v}\right)\cos\gamma \tag{6.1e}$$

$$\dot{\chi} = \frac{T(\delta_T)\sin\alpha + L}{mv\cos\gamma}\sin\sigma + \frac{v}{r\cos\theta}\cos\gamma\sin\chi\sin\theta \tag{6.1f}$$

$$\dot{m} = -\dot{m}_p(\delta_T) \tag{6.1g}$$

where $r = \|\mathbf{r}\|$ is the modulus of the position vector $\mathbf{r}$, $h = r - R_E$ is the altitude, λ and θ are longitude and Geocentric latitude, $v = \|\mathbf{v}\|$ is the magnitude of velocity vector, γ and χ are the flight path and heading angles, m is the mass of the vehicle, L and D are the aerodynamic lift and drag forces, $g = \mu_E/r^2$ is the gravitational acceleration, T is the thrust produced by the engine and $\dot{m}_p$ is the propellant mass flow rate. The control variables are the angle of attack α, bank angle σ, and the throttling δ_T of the engine.

6.1.2 Atmospheric and aerodynamic models

The International Standard Atmosphere model was used to model the atmospheric temperature, pressure p_a, and density ρ_a as a function of the radius r assuming a spherical Earth model with a radius $R_E = 6371\,\text{km}$ and constant angular rotational velocity $\omega_E = 7.292\,118 \times 10^{-5}\,\mathrm{rad\,s^{-1}}$.

The atmosphere is assumed to rotate with the same angular velocity as the Earth, so the aerodynamic forces were computed using the velocity of the vehicle relative to the air with no wind (or other disturbances),

$$L = \frac{C_L \rho_a S_{ref} v_{rel}^2}{2} \tag{6.2}$$

$$D = \frac{C_D \rho_a S_{ref} v_{rel}^2}{2} \tag{6.3}$$

where $\mathbf{v}_{rel} = \mathbf{v} - \boldsymbol{\omega_E} \mathbf{r}$ and given the aerodynamic coefficients for lift C_L and drag C_D, and the reference area of the vehicle S_{ref}.

6.1.3 Propulsion model

The thrust vector is assumed to be always aligned with the longitudinal body axis of the vehicle, proportional to the vacuum thrust T_{vac} of the engine and modulated by the throttle control $\delta_T \in [0,\ 1]$. An additional term is added to account for the losses due to the difference between the nozzle's exit pressure and the external atmospheric pressure p_a. The resulting model is,

$$T = \delta_T \left(T_{vac} - A_e p_a\right) \tag{6.4}$$

where A_e is the nozzle exit area. The mass flow rate of the propellant $\dot{m}_p$ is given by

$$\dot{m}_p = \frac{\delta_T T_{vac}}{I_{sp} g_0} \tag{6.5}$$

where I_{sp} is the specific impulse of the engine and g_0 is the Earth gravitational acceleration at sea level.

Table 6.1: Lower bounds, upper bounds and boundary conditions for the Optimal Descent Trajectory case

Variable [units]	Lower bound	Upper Bound	Initial value	Final value
Radius r [kft]	20900	21300	21163	20983
Latitude θ [rad N]	$-\pi/2$	$\pi/2$	0	≥ 0.2618
Longitude λ [rad E]	$-\pi$	π	0	Free
Velocity v [ft s^{-1}]	1	30000	25600	2500
Flight path angle γ [rad]	$-\pi/2$	$\pi/2$	-0.0175	-0.0873
Heading angle χ [rad]	$-\pi$	π	$\pi/2$	Free
Angle of attack α [rad]	$-\pi/2$	$\pi/2$	Free	Free
Bank angle σ [rad]	$-\pi/2$	0	Free	Free

6.2 Optimal Descent Trajectory

The first test case is based on a problem proposed by Betts [69] who analysed the unpowered re-entry of a Space Shuttle-like vehicle controlled by changing the angle of attack α and bank angle σ. To be consistent with the reference, the throttle was set to $\delta_T = 0$ for all t, $\omega_E = 0$ and units are in the Imperial system. Furthermore, the following models and reference values were taken from Betts [69]: the exponential model for the atmosphere, the linear model for C_L, the parabolic model for C_D, the aerodynamic reference area of $S_{ref} = 2690$ ft^2 (249.9 m^2) and the vehicle mass of 6309 sl (92 t). Bounds on the rate of change of the flight path and heading angles were imposed: $|\dot{\gamma}| \leq 0.035\,\mathrm{rad\,s^{-1}}$ and $|\dot{\chi}| \leq 0.035\,\mathrm{rad\,s^{-1}s}$. A semi-empirical correlation for the heat flux at the nose was given as

$$q_{flux} = \left(c_0 + c_1\alpha + c_2\alpha^2 + c_3\alpha^3\right) c_4 \sqrt{\rho_a}\,(c_5 v)^{c_6} \leq 70 \text{ btu ft}^{-2} \text{ s}^{-1} \qquad (6.6)$$

with the coefficients $c_0, c_1, c_2, c_3, c_4, c_5, c_6$ as reported in [69]. Boundary conditions, lower bounds and upper bounds for the optimisation variables are listed in Table 6.1.

Objectives

In [69] two different single objective problems were proposed: a maximum cross-range (equivalent to maximising the final latitude θ_f since $\theta_0 = 0$) and a minimum peak heat flux $\max q_{flux}$ problem. Here, we propose, instead, the following multi-objective optimisation problem:

$$\begin{aligned}
&\min_{t_f,\mathbf{u}} \, [J_1,\, J_2]^T = [-\theta_f,\, q_u]^T \\
&s.t \\
&q_{flux} \leq q_u
\end{aligned} \tag{6.7}$$

Numerical settings

The problem was formulated as a single time phase, with boundary conditions defined in Table 6.1, discretised using 6 finite elements and order 9 Bernstein polynomials for both states and controls, resulting in 121 optimisation variables for the outer level (120 for the control variables, and 1 for the free final time), and 484 total variables for the single level and inner level NLP. A limit of 20000 calls to the objective vector was given to MACSoc, a population of 10 agents was deployed in the search space and the size of the archive $\mathcal{A}$ was limited to 10 elements. The choice of these values for the parameters of the solver requires some experience and knowledge about the problem to be solved: since this is a two objective problem, a 10 points approximation of the Pareto front was deemed sufficient to get a good idea of the shape of the Pareto front. Moreover, since the problem was not expected to be particularly challenging from a global optimisation point of view, the number of agents was taken equal to the number of desired solutions on the Pareto front. For the same reason it was expected that the global exploration heuristics would have been able to take the agents close to an optimal solution with few function evaluations. The NLP solver would then have a good starting guess and would converge quite easily to the locally optimal solution. With 20000 function evaluations and 10 agents, each agent would have a budget of 2000 function evaluations, more than 4 times higher than the number of optimisation variables for the single level and inner level NLP, and more than 10 times the number of optimisation variables seen by the outer level. These values were considered sufficient without being excessive.

The initialisation of the population required approximately 1 second for 8 of the 10 agents, while for 2 agents it took approximately 3 minutes because the NLP solver did not converge to a feasible solution in the maximum number of iterations. However, as soon as the optimisation loop started, all solutions immediately became feasible thanks to the bi-level approach. The total runtime was approximately 1 hr.

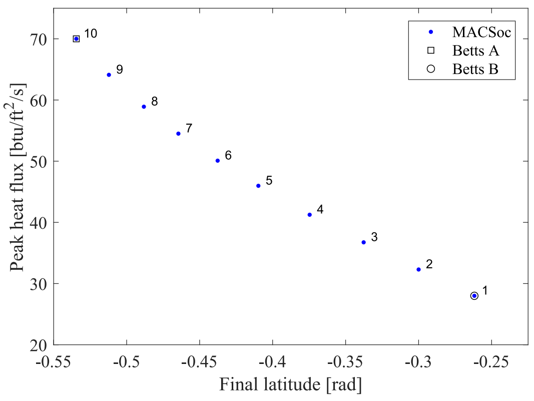


Figure 6.1: Optimal descent trajectory: Pareto optimal solutions stored in the archive $\mathcal{A}$ at the end of the optimisation process and the two published single objective solutions from Betts [69].

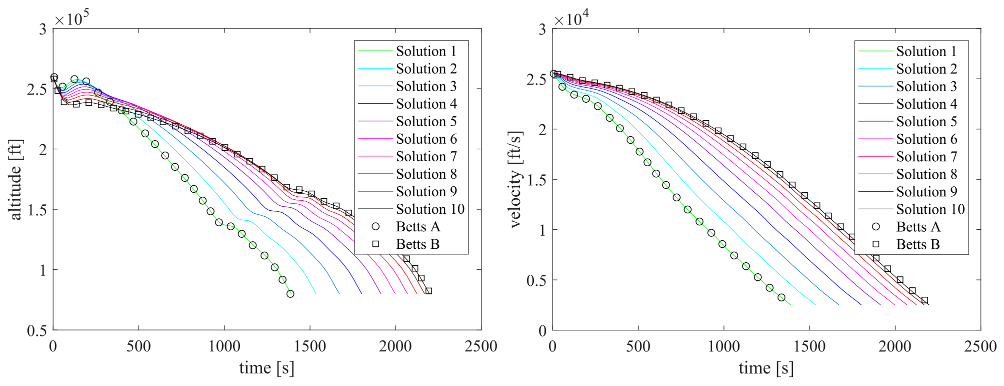


Figure 6.2: Optimal descent trajectory: time-history of a) the altitude and b) the velocity for each of the 10 solutions in the Pareto front in Fig. 6.1 plus the two published single objective solutions from Betts [69].

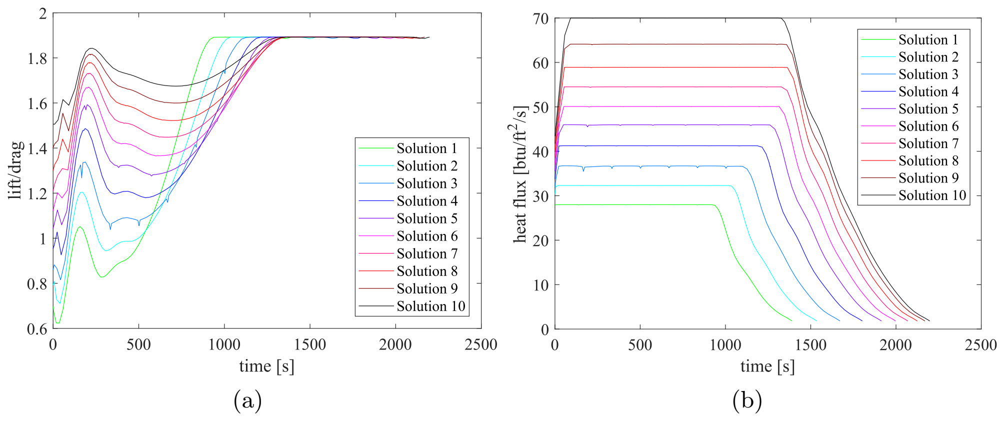


Figure 6.3: Optimal descent trajectory: time-history of a) the lift to drag ratio and b) the heat flux for each of the 10 solutions in the Pareto front in Fig. 6.1.

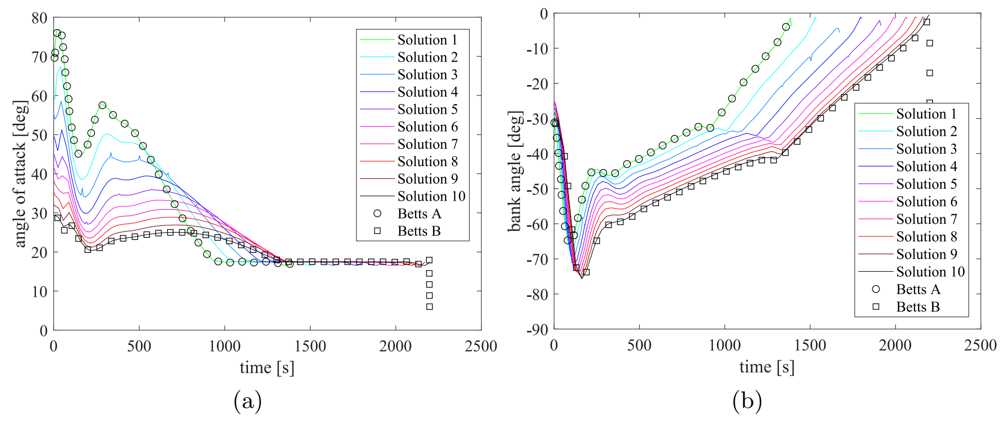


Figure 6.4: Optimal descent trajectory: time-history of a) the angle of attack and b) the bank angle for each of the 10 solutions in the Pareto front in Fig. 6.1 plus the two published single objective solutions from Betts [69].

6.2.1 Results

Figure 6.1 shows the 10 Pareto optimal solutions in the archive $\mathcal{A}$ at the end of the optimisation process, while Figs. 6.2–6.5 show altitude, velocity, lift-to-drag ratio L/D, heat flux, angle of attack, bank angle and angular velocities for the flight path and heading angles for each of the 10 solutions in $\mathcal{A}$.

Figure 6.1 shows that the method is able to find an even spread of elements belonging to the Pareto front including the two single objective solutions given in [69], here

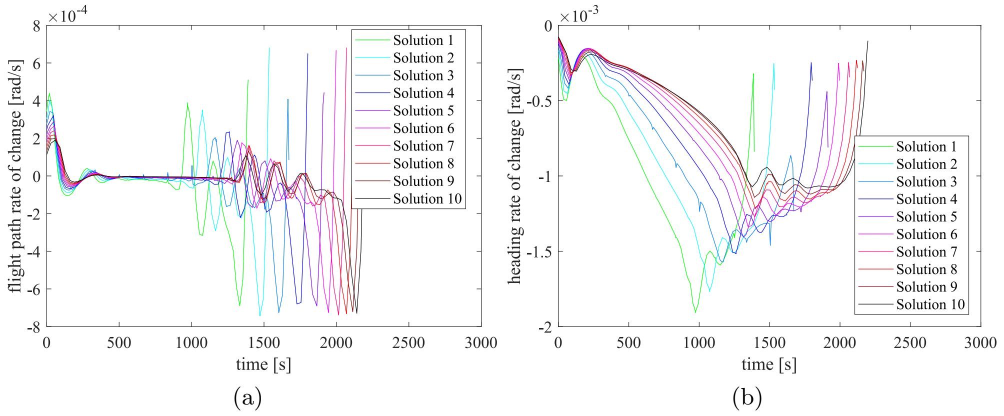


Figure 6.5: Optimal descent trajectory: time-history of a) the rate of change of the flight path angle $\dot{\gamma}$ and b) the heading angle $\dot{\chi}$ for each of the 10 solutions in the Pareto front in Fig. 6.1.

labelled as Betts A and Betts B, corresponding to Solutions 1 and 10, without any externally supplied guesses. Solution 1 has objective values $(-0.2618,\ 28.0016)$ and Solution 10 has objective values $(-0.5345,\ 69.9997)$, while the corresponding solutions from the reference are $(-0.2618,\ 27.9982)$ and $(-0.5345,\ 70)$ respectively.

The relative difference in the second objectives is below 10^{-4} and can be attributed to the different NLP solvers and settings employed, and the presence of an iterative mesh refinement procedure in the reference solutions. In Figs. 6.2 and 6.4 the circles and squares represent the solutions from [69] for the two individual objective functions. The figures show a very good match between the result generated with MACSoc.

A clear trend emerges from the solutions of the multi-objective problem: the minimum peak heat flux solution is also the shortest in time with longer re-entries imposing higher peak heat fluxes. This is due to the initial altitude skip in the trajectory (see Fig. 6.2a in the first 200 s), which is more pronounced for the shorter time re-entries and causes the velocity to decrease faster but at the price of having less energy and time to maximise the cross-range. Since the initial conditions are fixed, the corresponding heat flux is also fixed. Thus it is not possible to further reduce the peak heat flux.

All the trajectories present a similar control profile: an initial peak heat flux containment phase, with high bank and angle of attack, followed by a maximum aerodynamic efficiency phase to maximise the cross-range when heat flux is no longer a concern. The angle of attack during the initial descent is limited by the objective to minimise the heat flux, which is a function of the velocity, altitude and angle of attack, while a high bank angle is used to maximise the cross-range with no penalty to the heat flux. Angular rates for the flight path and heading angles are well below the imposed constraints along all the trajectories.

6.3 Three-objective Ascent Problem

This test case is the multi-objective, multidisciplinary design of a rocket-powered, two-stage launch vehicle optimised for the ascent to orbit. The vehicle is air dropped from a carrier aeroplane flying at 200 m s^{-1} at an altitude of 10 km eastbound along the equator, with an initial flight path angle of 10°. It has to deliver a 500 kg payload to a 650 km altitude circular equatorial orbit.

The aim of this test case is to minimise the initial gross mass of the vehicle, examining the trade-off between the engine sizing and dry masses of each of the two stages. The vacuum thrust ratings of the two rocket engines are set as optimisation variables, which through the mass model directly affect the dry masses of the two vehicle stages. Similarly, the mass of propellant used in each stage also affects the dry mass of each stage by altering the mass of the tanks.

As the focus here is on the vehicle design of the mass and propulsion systems, a simple aerodynamic model was used for both stages: for the first $C_L = 0$, $C_D = 0.1$ and $S_{ref} = 73.73\,\mathrm{m}^2$, while for the second $C_L = 0$, $C_D = 0.01$ and $S_{ref} = 1\,\mathrm{m}^2$.

The ascent trajectory was divided into two phases: Phase 1 is the ascent of the integrated vehicle (combined first and second stage vehicles), and Phase 2 is the ascent of only the second stage vehicle. The initial and final conditions and bounds for optimisation variables are listed in Table 6.2.

Table 6.2: Lower bounds, upper bounds and boundary conditions for the Three Objective Ascent case

Variable [units]	Lower bound (All phases)	Upper Bound (All phases)	Initial value (Phase 1)	Final value (Phase 2)
Radius r [km]	6371	7171	6381	7021
Latitude θ [rad E]	$-\pi/2$	$\pi/2$	0	Free
Longitude λ [rad N]	0	π	0	Free
Velocity v [$\mathrm{m\,s^{-1}}$]	465	10000	665	7535
Flight path angle γ [rad]	$-\pi/2$	$\pi/2$	0.1745	0
Heading angle χ [rad]	0	π	$\pi/2$	$\pi/2$
Vehicle mass m [t]	0	100	Free	Free
Angle of attack α [rad]	0	0.3491	Free	Free
Bank angle σ [rad]	-0.1745	0.1745	Free	Free
Throttle δ_T	0	1	Free	Free
$T_{vac,1}$ [MN]	0	2	N.A.	N.A.
$T_{vac,2}$ [kN]	0	200	N.A.	N.A.
$m_{p,1}$ [t]	0	100	N.A.	N.A.
$m_{p,2}$ [t]	0	1	N.A.	N.A.

6.3.1 Structural mass models

For the first stage, the dry mass is a function of the engine mass $m_{eng,1}$ and propellant mass required for the first phase $m_{p,1}$,

$$m_{dry,1} = -l_3\tilde{m}_{p,1}^3 + l_2\tilde{m}_{p,1}^2 + l_1\tilde{m}_{p,1} + l_0 + m_{eng,1} \tag{6.8}$$

$$\tilde{m}_{p,1} = \frac{m_{p,1} - l_4}{l_5} \tag{6.9}$$

For the second stage vehicle, the dry mass was assumed to be

$$m_{dry,2} = \frac{0.1}{0.9}m_{p,2} + m_{eng,2} \tag{6.10}$$

The gross vehicle masses for each phase are then given by,

$$m_{0,1} = m_{dry,1} + m_{p,1} + m_{dry,2} + m_{p,2} \tag{6.11}$$

$$m_{0,2} = m_{dry,2} + m_{p,2} \tag{6.12}$$

The vacuum thrust of the engines was used to estimate their structural mass based on an empirical linear relationship of existing commercial engines,

$$m_{eng} = l_6 T_{vac} + l_7 \tag{6.13}$$

where l_6, l_7 are constants. The mass model was developed in parallel for an industrial vehicle and the coefficients l_0 to l_7 cannot be released publicly [110]. However, for other applications those coefficients could be estimated by employing historical data for the masses of similar vehicles and engines.

For the first stage engine, $0 \leq T_{vac} \leq 2\,\mathrm{MN}$ and $I_{sp} = 332\,\mathrm{s}$, while for the second stage engine $0 \leq T_{vac} \leq 200\,\mathrm{kN}$ and $I_{sp} = 352\,\mathrm{s}$. Propellant masses were limited to 100 t for the first stage and 20 t for the second.

Matching conditions between the phases were imposed on all state variables except for the mass, for which the following instantaneous drop was imposed at the stage separation:

$$m_{0,2} = m_{f,1} - m_{dry,1} \tag{6.14}$$

6.3.2 Objectives

The aim of the optimisation is to study the trade off between propellant efficient designs and designs that require relatively small engines. The objective functions were to minimise the gross vehicle mass $m_{0,1}$ and the two ratios between the vacuum thrust of the stage engine, and the gross weight at the beginning of each phase.

The thrust-to-weight metric also gives an indication of the vehicle loads or induced accelerations the vehicle experiences during flight. The higher the ratio between thrust and mass, the higher the loads imposed on the vehicle, thus one option is to minimise loading by minimising the thrust to weight ratio. Reducing the vacuum thrust reduces the engine performance however, which requires often longer duration trajectories and more propellant, which in turn increase the vehicle mass.

$$\min_{t_f, \mathbf{u}, T_{vac,1}, T_{vac,2}, m_{p,1}, m_{p,2}} [J_1,\, J_2,\, J_3]^T = \left[m_{0,1},\, \frac{T_{vac,1}}{g_0 m_{0,1}},\, \frac{T_{vac,2}}{g_0 m_{0,2}} \right]^T \tag{6.15}$$

6.3.3 Numerical settings

The problem was discretised using 4 DFET elements of order 7 for both states and controls, and both phases, resulting in a total of 207 optimisation variables for the outer level and 666 optimisation variables for the single level and inner level NLP. A limit of 80000 calls to the objective vector was given to the optimiser, 106 agents were deployed in the search space and the same maximum number of solutions were kept in the Archive. The considerations that lead to the choice of these parameters for the solver were similar to the ones taken for the previous case. Since for problems with more than two objectives the number of weight vectors is given by the simplex lattice design algorithm (3.2.3), it was decided than 13 subdivisions per objective for a three objective problem would have been sufficient but not excessive, resulting in 106 points in the Archive. Since the problem was expected to be relatively simple from the global optimisation point of view, one agent per point in the Archive was deemed sufficient. For the same reason, 80000 function evaluations were considered sufficient for a first try. The initialisation of the population required on average 5 seconds, but for some agents it took 5 minutes as the randomly generated guess was poor and the NLP did not manage to find a feasible solution. At the next iteration, all solutions were feasible and the Archive quickly filled up. The runtime of the problem was approximately 30h.

Results

Figure 6.6 shows the 106 Pareto optimal solutions in the archive at the last iteration, with an additional colorbar indicating gross take-off mass. The shape of this 3D Pareto front resembles a smooth half cup. The figure shows the 3D surface in the middle, and the three orthogonal projections. As can be seen, the algorithm found a very good spread set of solutions, all of which are feasible and locally Pareto optimal up to the requested 10^{-6} threshold.

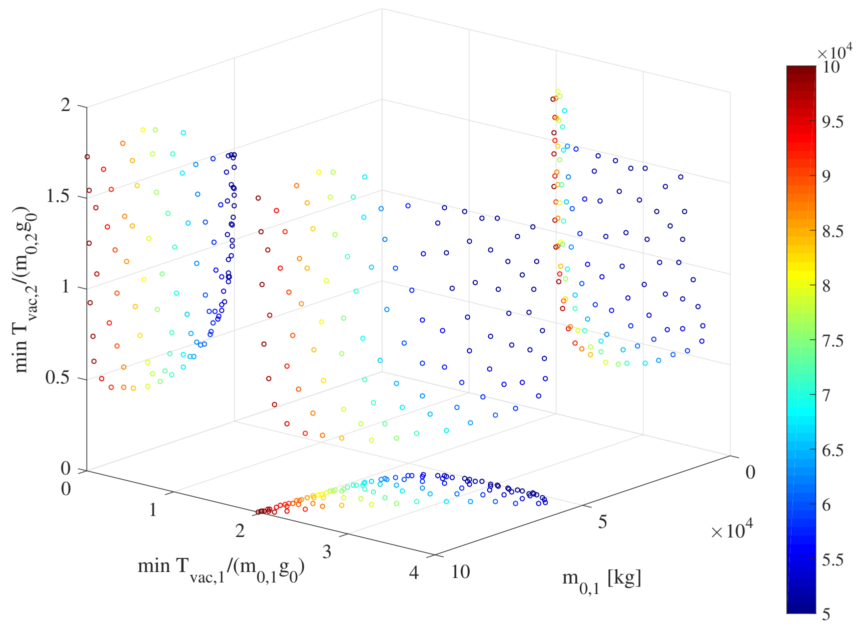


Figure 6.6: Three-Objective Ascent: set of Pareto-optimal solutions, colorbar indicates gross mass of the vehicle. The 3D Pareto front is in the middle, with orthographic projections shown on each coordinate plane.

Figures 6.7 and 6.8 show the altitude and velocity profiles plus the flight path angle and throttle time histories of the three extreme solutions of the Pareto front. The altitude, velocity and throttle profiles of the minimum gross mass and minimum first stage ($T_{vac,1}/m_{0,1}g_0$) solutions are similar, while their flight path angles differ substantially during the initial ascent: in both cases the first stage engine is working at full throttle and for a comparable time, but given the relatively lower thrust engine of the minimum ($T_{vac,1}/m_{0,1}g_0$) case, the resulting flight path angle dips and becomes negative causing the vehicle to briefly lose altitude.

The minimum second stage ($T_{vac,2}/m_{0,2}g_0$) solution is instead quite different: the first stage engine has to compensate for the relatively small second stage engine by pushing the vehicle to a higher altitude, velocity and flight path angle at the separation point. The second stage engine has to operate at maximum throttle for a comparatively longer length of time after the separation, and has a higher throttle setting during the final circularisation burn in order to compensate for its lower thrust, as shown in Fig.

6.8b. The total flight duration is also slightly longer than the other two.

Table 6.3: Design parameters for the three extreme cases of the Three-Objective Ascent case

Solution	Stage	Vacuum thrust [kN]	Thrust weight ratio	Δv [$\mathrm{km\,s^{-1}}$]
$\min(m_{0,1})$	1	1682.611	3.432	2.836
	2	126.100	1.467	5.664
$\min(T_{vac,1}/m_{0,1}g_0)$	1	1930.137	1.968	4.115
	2	200.000	1.730	6.003
$\min(T_{vac,2}/m_{0,2}g_0)$	1	2000.000	2.571	4.565
	2	15.124	0.351	4.606

Table 6.4: Mass breakdown for the three extreme cases of the Three-Objective Ascent case

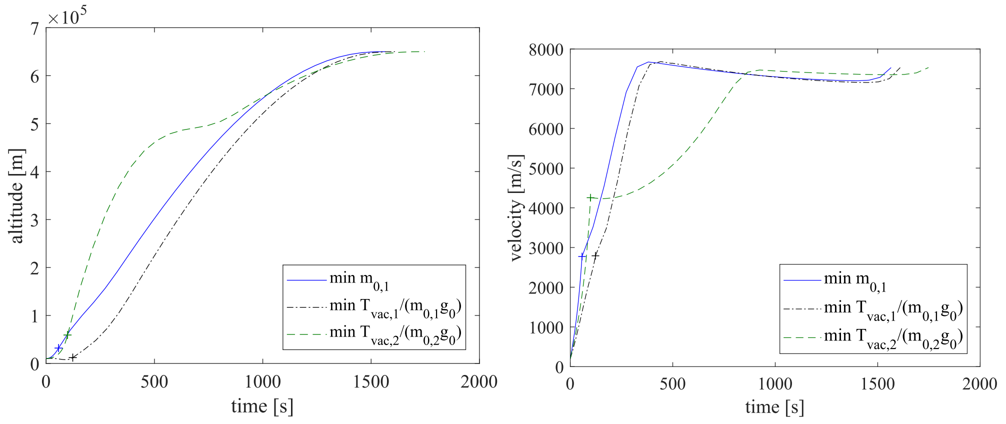
Solution	Stage	Initial mass [t]	Propellant mass [t]	Dry mass [t]
$\min(m_{0,1})$	1	49.995	29.632 (59.27%)	20.363 (40.73%)
	2	8.765	7.063 (80.59%)	1.699 (19.38%)
$\min(T_{vac,1}/m_{0,1}g_0)$	1	100.000	72.830 (72.83%)	27.170 (27.17%)
	2	11.789	9.717 (82.42%)	2.071 (17.57%)
$\min(T_{vac,2}/m_{0,2}g_0)$	1	79.318	60.629 (76.44%)	18.689 (23.56%)
	2	4.390	3.234 (73.66%)	1.156 (26.34%)

Tables 6.3 and 6.4 report the design parameters for the Pareto extrema (i.e., the solutions that minimise each objective individually) and a breakdown of the vehicle masses with the relative percentage values with respect to the stage's initial mass, engine vacuum thrust, thrust to weight ratio, and resulting Δv contribution.

The solution with minimum initial mass requires high ratios of vacuum thrust to initial weight, though the vacuum thrust of the engines does not reach the maximum allowed values. Propellant mass is approximately 60% of the total mass of the first stage and approximately 80% of the total of the second stage. Total Δv is of $8.5\,\mathrm{km\,s^{-1}}$, with the first stage contributing approximately for $2.8\,\mathrm{km\,s^{-1}}$ or 33% of the total, and the rest coming from the second stage. The ratio between the payload and gross vehicle mass is approximately 1%.

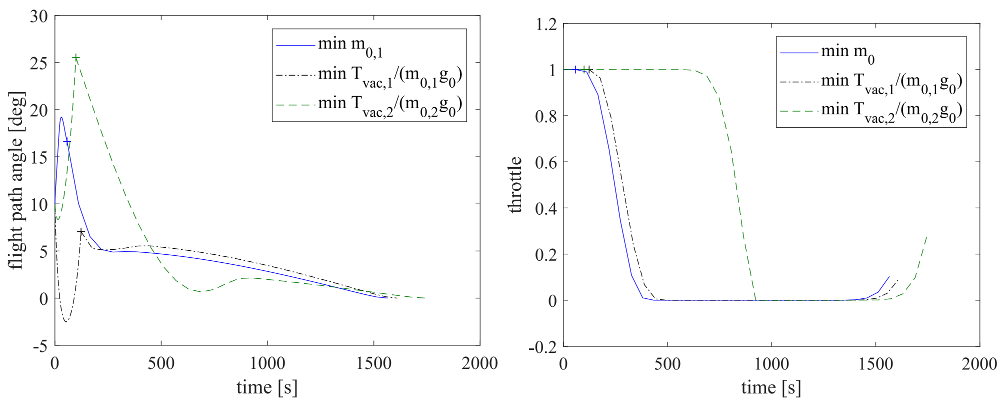
The solution corresponding to the minimum thrust to weight ratio of the first stage requires a larger vehicle with a substantially higher amount of propellant: its initial mass reaches the maximum allowed value for the mass of the vehicle (see Table 6.2), and is double the value of the previous case. Of this gross mass, approximately 70% is propellant for the first stage. The ratio between the payload mass and the initial mass is 0.5%. The total required Δv is $10.1\,\mathrm{km\,s^{-1}}$, with $6\,\mathrm{km\,s^{-1}}$ coming from the second stage. The second stage engine also has the maximum possible vacuum thrust and consumes more propellant than the previous case leading to a high $(T_{vac,2}/m_{0,2}g_0)$ at the cost of a minimised first stage $(T_{vac,1}/m_{0,1}g_0)$.

The solution corresponding to the minimum thrust to weight ratio of the second stage requires an intermediate initial mass, approximately 60% more than the minimum initial mass case. The ratio between the payload mass and the initial mass is 0.63%, and the required Δv totals $9.1\,\mathrm{km\,s^{-1}}$, evenly spread between the two stages. This is true also for the propellant mass, representing approximately 75% of the total of each stage and totalling twice as much as the minimum gross take-off mass case. The first stage engine has to compensate by taking the maximum allowed value of vacuum thrust, with the resulting thrust to weight ratio being higher than in the previous case, leading to higher induced accelerations. However, the second stage is significantly lighter than the other solutions both in terms of dry mass and propellant mass, and its engine has a vacuum thrust one order of magnitude smaller than the previous solutions.



(a) (b)

Figure 6.7: Three-Objective Ascent: time-history of a) the altitude and b) the velocity for the three extreme solutions of the Pareto front in Fig. 6.6. The + indicates the stage separation point.



(a) (b)

Figure 6.8: Three-Objective Ascent: time-history of a) the flight path angle and b) the throttle for the three extreme solutions of the Pareto front in Fig. 6.6. The + indicates the stage separation point.

6.4 Optimal Ascent and Abort Scenarios

The third test case is the multidisciplinary design of a spaceplane accounting for abort scenarios. Other studies [111] have looked at optimising the abort descent for an independently determined optimal ascent trajectory. Here, the design of the vehicle and trajectory is formulated as a multi-objective optimisation to account for the vehicle design and performance during the ascent to orbit under nominal conditions and the descent under abort conditions, for different abort scenarios.

The vehicle is a single stage to orbit spaceplane designed to be air-dropped from a carrier aircraft, similar in design to the X-34. As in the three-objective ascent case, the drop point from the carrier aircraft is set at 10 km altitude and 200 m/s. The mission is to reach a 100 km altitude, circular equatorial parking orbit. The propulsion system is rocket-based with $T_{vac} = 1340$ kN and $I_{sp} = 450$ s. An abort is designed to occur after complete engine failure at a specific time t_{fail}. To study the worst case, no propellant dumping was allowed.

The optimisation problem is configured to find the optimal control for the trajectories, and the optimal sizing of the wing area, which affects the downrange performance during the abort and the dry mass of the vehicle. The initial flight path angle is also set as an optimisable design parameter.

The timeline was divided into three phases: Phase 1 considers the normal ascent trajectory before the failure occurs, from t_0 to t_{fail}. At t_{fail}, the timeline branches into two parallel phases: in Phase 2 the engine is assumed to be working normally and the spaceplane ascends to its target orbit, while instead in Phase 3 the engine has failed and the spaceplane attempts an emergency landing. The boundary conditions for the states and controls are given in Table 6.5. Inter-phase constraints were introduced at the branching point to impose the continuity of states and controls between Phase 1 and Phase 2 and continuity of the states only between Phases 1 and 3. The throttle in Phase 3 was imposed to be zero.

Table 6.5: Lower bounds, upper bounds and boundary conditions for the Optimal Ascent and Abort Scenarios case

	Lower	Upper	Initial value	Final value	
Variable [units]	bound	Bound	Phase 1	Phase 2	Phase 3
Radius r [km]	6371	6496	6381	6471	6373
Latitude θ [rad N]	$-\pi/2$	$\pi/2$	0	0	0
Longitude λ [rad E]	0	π	0	Free	Free
Velocity v [m s^{-1}]	465	10000	665	7848	Free
Flight path angle γ [rad]	$-\pi/2$	$\pi/2$	$\lvert\gamma_{0,1}\rvert \leq 0.1745$	0	$\gamma_{f,3} \geq 0.1745$
Heading angle χ [rad]	0	π	$\pi/2$	$\pi/2$	Free
Vehicle mass m [t]	0	100	Free	Free	Free
Angle of attack α [rad]	0	0.6109	Free	Free	Free
Bank angle σ [rad]	-0.1745	0.1745	Free	Free	Free
Throttle δ_T	0	1	Free	Free	N.A.
S_{ref} [m^2]	20	400	N.A.	N.A.	N.A.

Aerodynamic model

Polynomial models of the lift and drag coefficients, as functions of angle of attack α and Mach number M, were built with a non-linear least square best fit of the aerodynamic data of the X-34 vehicle [112, 113]. The polynomial models are in the form:

$$\begin{aligned} C_L(\alpha, M) &= P_{2,1}(\alpha) + P_{2,2}(\alpha)W_1(M) + P_{2,3}(\alpha)W_2(M) \\ C_D(\alpha, M) &= P_{3,1}(\alpha) + P_{3,2}(\alpha)W_3(M) + P_{3,3}(\alpha)W_3(M) \end{aligned} \tag{6.16}$$

where $P_{i,j}$ is the j^{th} polynomial of degree i of α with monomial coefficients $(a_{j,0}, \cdots, a_{j,i+1})$ and W_i are Weibull distributions over Mach with parameters (ς_j, κ_j) shifted by ϱ_j, i.e.,

$$W_i = \frac{\kappa_i}{\varsigma_i} \left(\frac{M - \varrho_i}{\varsigma_i} \right)^{\kappa_i - 1} e^{-\left(\frac{M-\varrho_i}{\varsigma_i}\right)^{\kappa_i}} \tag{6.17}$$

The upper and lower limits on the coefficient of the Weibull functions were chosen so that they had a maximum at around $M = 1$, went to zero for $M = 0$ and for $M \to \infty$ went to zero with first derivative equal to zero. Coefficients for this aerodynamic model are given in Table C.1 in Appendix C. Figure 6.9 shows the overall agreement between the model and the data points. The R^2 value of the non-linear fit is over 0.99 for both models. As no data was provided in the rectangular area defined by $\alpha \geq 20°$ and $M \leq 3$, the smooth constraint

$$\left(\frac{\alpha - 35}{15}\right)^8 + \left(\frac{M - 30}{27}\right)^8 - 1 \leq 0 \tag{6.18}$$

was imposed to exclude that area.

The additional path constraint $M \geq 0.3$ was imposed on all trajectories to exclude Mach numbers for which the models extrapolated poorly. As the vehicle is not expected to fly in either of these conditions, these constraints should not affect the optimality of the results.

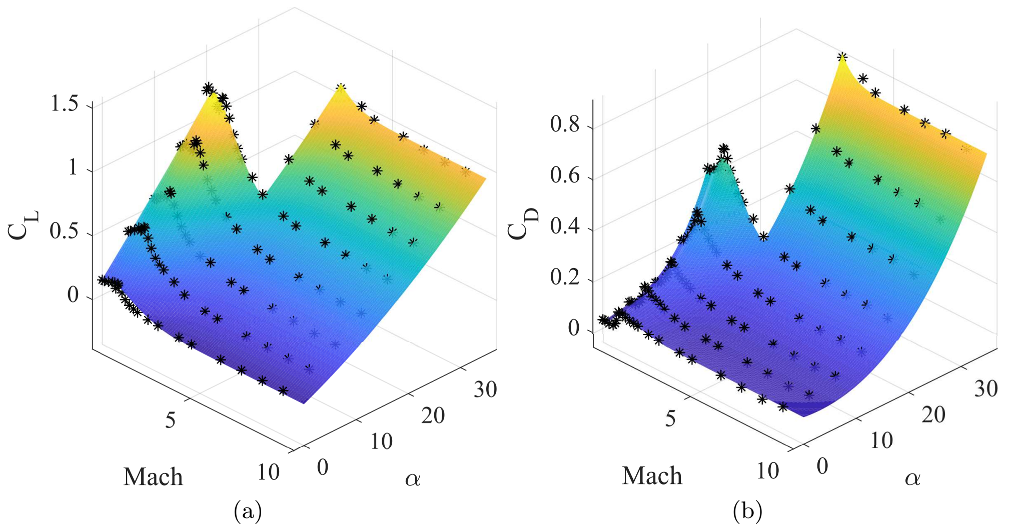


Figure 6.9: Data points, black dots, and non-linear fit surfaces of the aerodynamic coefficients: a) C_L as a function of M and α and b) C_D as a function of M and α.

Wing mass model

In order to account for the change in mass due to a change of wing surface, the following mass relationship from Rohrschneider [114] was used:

$$m_{wing} = \left(N_z m_{land} \frac{1}{1 + \eta \frac{S_{body}}{S_{exp}}} \right)^{0.386} \left(\frac{S_{exp}}{t_{root}} \right)^{0.572} \left(K_{wing} b_{str}^{0.572} + K_{ct} b_{body}^{0.572} \right) \quad (6.19)$$

where N_z is the ultimate load factor, m_{land} is the landed mass, S_{body} is the surface of the fuselage, S_{exp} is the surface of the exposed part of the wing, t_{root} is the wing thickness at the root, b_{str} is the structural wing span at half chord line, b_{body} is the fuselage width and η, K_{wing}, K_{ct} are constants depending on the wing structure.

The landed mass m_{land} was set equal to the unknown initial mass of the vehicle $m_{0,1}$, while the final mass was set equal to $m_{f,2} = m_{wing} + m_{str}$, where m_{str} is the structural mass of the vehicle plus the payload, without the wings, and was imposed to be a fixed value of 10 t.

Given the options available in [114], the following values of the constants in Eq. (6.19) were used: $N_z = 4.5$ corresponding to a maximum loading of 3 multiplied by the safety factor of 1.5 for the ultimate load, $\eta = 0.15$ corresponding to a control configured vehicle, $K_{wing} = 0.214$ for organic composite honeycomb wing without TPS, and $K_{ct} = 0.0267$ for integral dry carry-through of the wing beam, the lightest option available.

The geometric parameters were derived from the geometry of the X-34 [112, 113]: $S_{exp} = 0.7283 S_{ref}$, $S_{body} = 0.6970 S_{ref}$, $b_{str} = 1.486\sqrt{S_{ref}}$, $b_{body} = 0.2785\sqrt{S_{ref}}$, $t_{root} = 0.087\sqrt{S_{ref}}$, where S_{ref} is a static optimisation variable.

Loading Constraints

As an optimisation of the gross initial mass could result in a reduction of the wing area, the following constraint on the wing loading was imposed:

$$\frac{m_{0,1}}{S_{ref}} \leq 700\,\mathrm{kg/m^2} \tag{6.20}$$

together with the limit on the dynamic pressure experienced by the vehicle in all three phases:

$$\frac{1}{2}\rho_a v_{rel}^2 \leq 60\,\mathrm{kPa} \tag{6.21}$$

In order to limit the maximum static structural stresses, the total acceleration in all three phases was constrained to be:

$$\dot{v}^2 + v^2\left(\dot{\gamma}^2 + \dot{\theta}^2\right) \leq (3g_0)^2 \tag{6.22}$$

The final flight path angle for the abort phase was required to be greater than $-10°$, and the final velocity of the abort phase had to be such that $M = 0.4$.

Objectives

The optimisation criteria are the minimisation of the spaceplane initial mass and the maximisation of the downrange in case of an abort. As the vehicle is flying along the equator, the downrange can be measured in terms of the angular difference in longitude. The problem can thus be formulated as

$$\min_{t_f,\mathbf{u},S_{ref},\gamma_0} [J_1,\ J_2]^T = [m_{0,1},\ -(\lambda_{f,3} - \lambda_{0,3})]^T \tag{6.23}$$

where $m_{0,1}$ indicates the mass of the vehicle at the beginning of phase 1, while $\lambda_{0,3}$ and $\lambda_{f,3}$ indicate the longitude of the vehicle at the beginning and end of phase 3 respectively.

This coupled multi-objective, multi-phase formulation allows for a robust optimisation of the ascent trajectory because the abort scenario is included in the sizing process and the expected result is an optimal trade-off between ascent and landing performance in the case of a failure.

Numerical settings

The first analysis performed was for $t_{fail} = 0$ s. This is the worst case scenario in which the engine does not start. In this case there are only two phases as phase 1 has 0 length. The problem was discretised with 6 elements in the first phase and 3 in the second, using polynomials of order 7 for all states and controls. MACSoc was run for 20000 calls to the objective vector, 10 agents and a maximum archive size of 10. These values were chosen following the guideline mentioned in the previous sections. The initialisation took between 10 seconds and 40 seconds for each agent, all of which managed to find a feasible solution directly at the initialisation stage. The whole process ran for approximately 2h.

Results

The set of Pareto-optimal solutions is shown in Fig. 6.10a. The solutions are well spread and problems (4.8) are solved down to an accuracy of 10^{-6} in both optimality and feasibility. The associated wing loading and downrange for all the 10 Pareto-optimal solutions are shown in Fig. 6.10b.

As the ascent and abort trajectories start concurrently, the main trade-off is on the vehicle design and aerodynamic performance. The wing loading varies inversely to the downrange, as expected. This has an effect on the flight path angle and the throttle, which in turn affect the heat flux and the total acceleration. The result is that all solutions reach the maximum possible initial flight path angle.

High downrange solutions favour larger wing areas, which translates into a higher initial mass, due to the increased drag and higher mass of the wings themselves. However the increase in wing area is such that the wing loading actually decreases. Thus,

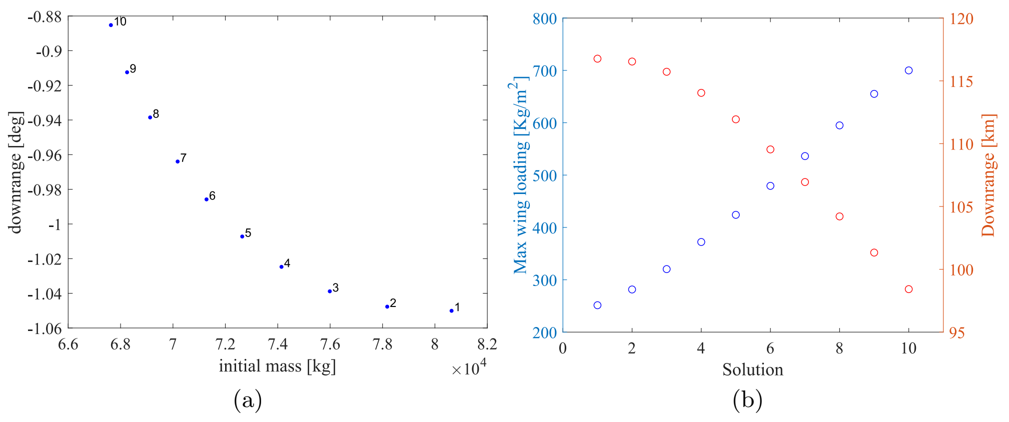


Figure 6.10: Abort at 0 seconds: a) Pareto front and b) wing loading vs downrange.

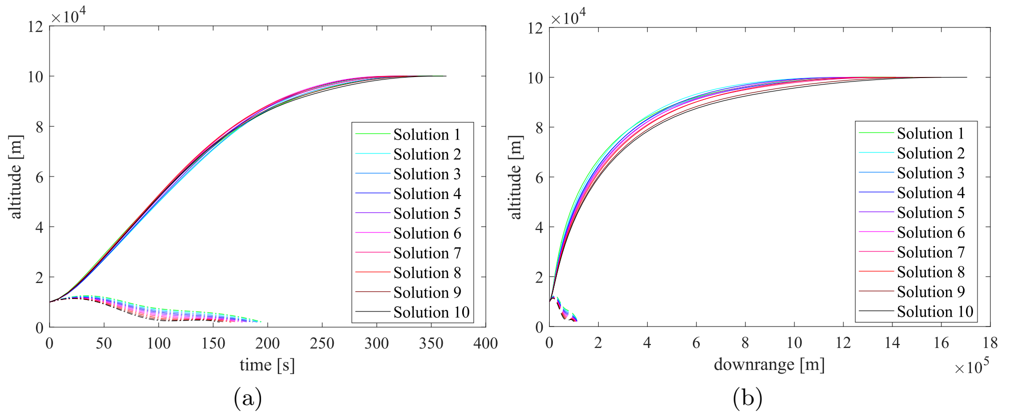


Figure 6.11: Abort at 0 seconds: a) altitude vs time and b) altitude vs downrange for ascent (solid lines) and descent (dashed lines).

solutions with larger wings have a lower wing loading and are able to generate longer downrange abort trajectories.

Figure 6.11 shows the ascent and abort trajectory profiles over time and downrange. Figure 6.12a shows the time history of the total acceleration experienced by the vehicle. All ascent phases are characterised by a maximum acceleration region in approximately the same time interval. The throttle profile, shown in Fig. 6.12b, differs from minimum mass solutions to maximum downrange solutions. For minimum mass solutions (small wing area) the throttle starts at maximum and remains there for a period of time, then progressively reduces down to zero, to comply with the limits on the accelerations,

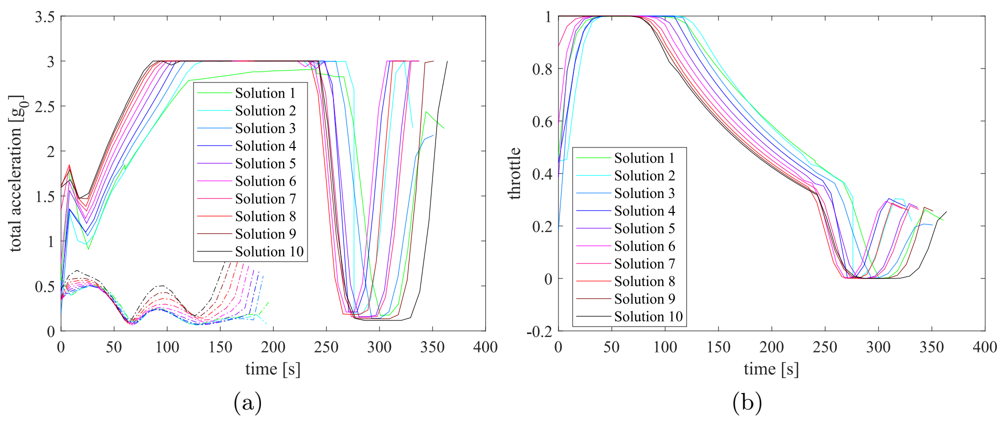


Figure 6.12: Abort at 0 seconds: time histories for a) total acceleration and b) engine throttling for ascent (solid lines) and descent (dashed lines).

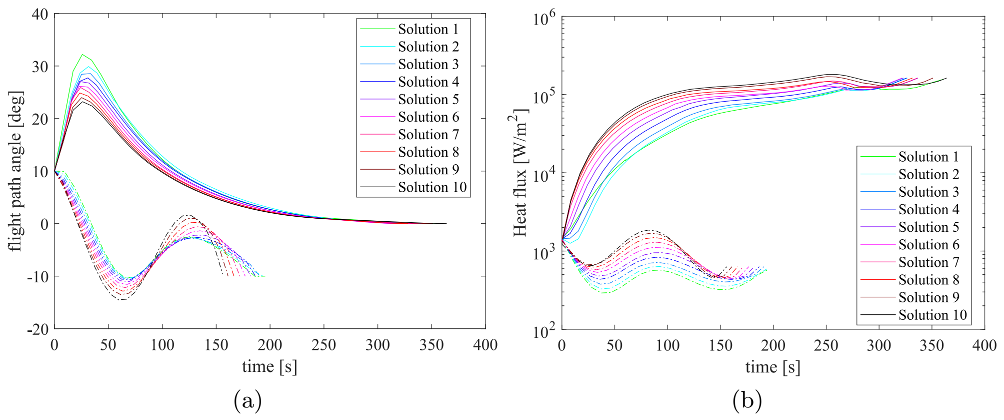


Figure 6.13: Abort at 0 seconds: time histories of a) flight path angle and b) heat flux for ascent (solid lines) and descent (dashed lines).

before finally increasing again to inject the spaceplane into orbit. Maximum downrange solutions, instead, favour a more gradual increase of the thrust and a better use of the aerodynamics.

The same analysis was then repeated for abort points at 5, 10, 15, 20, 25, 30, 35 and 40 seconds, changing the discretisation to 3 elements for each of the 3 phases. Figure 6.14a shows the corresponding approximations of the Pareto fronts. Interestingly, the different abort cases generate Pareto fronts which progressively converge to a single point at $t_{fail} = 30$ s. For t_{fail} between 30 s and 40 s, only one design solution exists for increasing values of the downrange. The increase of downrange is due to the increase of velocity and altitude while the size of the wings remains unchanged to keep the initial mass to the minimum. Thus, if the vehicle operates properly for at least 30 seconds, it gains so much velocity and altitude that, in the case of an abort, the downrange does not benefit from large wings. The same conclusion can be drawn from Fig. 6.14b that shows the wing loading of all solutions for all abort points.

The collapse of the Pareto front to a single point implies that there exists a single design solution that minimises the mass and simultaneously maximises the downrange. This design solution is both the most reliable and the most mass efficient if the abort happens between 30 s and 40 s from the drop point.

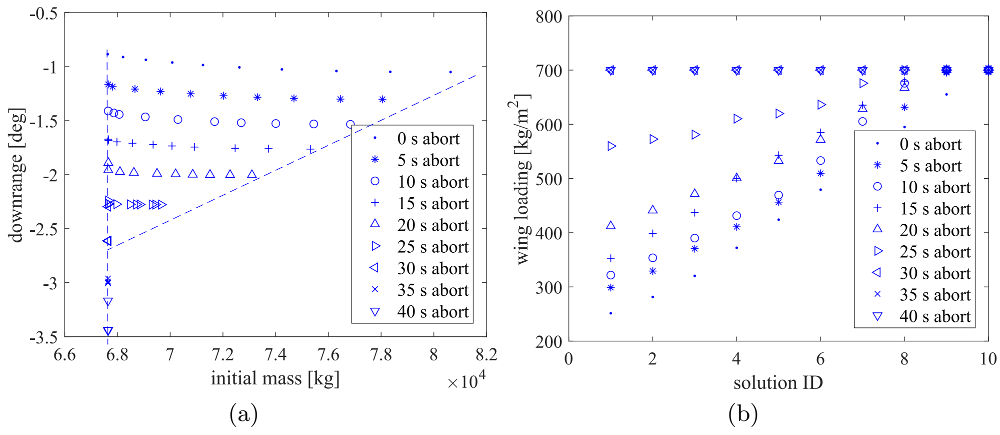


Figure 6.14: Comparison of Pareto optimal solutions for different abort times: a) downrange versus initial mass, b) wing loading for each solution ID.

6.5 Chapter summary

This chapter applied the approach presented in Chapter 4 to the optimisation of the ascent, re-entry and abort scenarios for different hypothetical vehicles. The approach was first validated on a known case from the literature confirming the ability to accurately identify the optimal values for each individual objective function and to reconstruct a well spread set of locally Pareto optimal solutions.

The application to the three objective case demonstrated the ability of the algorithm to generate a well spread Pareto front even in the case of more than two objectives. Furthermore, the solutions in the Pareto set give a unique insight on the impact of system design choices and an optimal trade-off between system sizing and control law, allowing the decision maker to make entirely different strategic choices depending on what is considered more important.

The abort scenario case provided an unexpected result that could not be derived from a single objective optimal control formulation of the problem: the trade-off on the wing loading affects the ascent trajectory as the aerodynamics of the vehicle changes due to the need to improve flight performance during descent, however this is true up to a limit abort time of about 30 s. Beyond that point there appears to be no trade-off and only one optimal configuration exists. The reason for this is that a high wing surface solution is also heavier, and the increase in gliding capabilities does not compensate for the increase of mass and thus the lower starting velocity and altitude of the abort phase.

Chapter 7

Multiple Debris Removal Mission[1]

Satius est supervacua scire quam nihil
It is better, of course, to know useless things than to know nothing

Seneca

Over the last 60 years, almost 9000 spacecraft had been launched using disposable launch vehicles. Many spent upper stages are still orbiting the Earth in LEO. In addition to those spent upper stages, around 3000 satellites are no longer operative due to obsolescence or failure. Given the relative orbital velocities, if two of these unresponsive objects collide, they will produce a large number of fragments, or debris. This creates a risk of a cascading failure mode, known as Kessler syndrome [115], which could cause complete loss of access to space. Mitigation strategies and guidelines have thus been considered to avoid this problem. On orbit servicing and active debris removal have thus become important areas of research.

[1]The contents of this chapter were published in: L. A. Ricciardi, M. Vasile, A Relaxation Approach for Hybrid Multi-Objective Optimal Control: Application to Multiple Debris Removal Missions. 29th AAS/AIAA Space Flight Mechanics Meeting, Ka'anapali, HI, Jan 2019, AAS 19-406.

Tadini et al. [116] estimated that the total mass in LEO, excluding the ISS, is approximately 2650 t, 97% of which is concentrated in spacecraft and rocket bodies. The authors also estimated that removing around 0.2-0.5% of these objects every year would be sufficient, provided that the highest priority targets were removed first. Following this principle two strategic targets were chosen and a spacecraft was designed specifically to remove those targets, albeit the design philosophy was to have a common design that could be reused many times for similar missions. The trajectory analysis part of that work considered a simple Hohmann transfer and did not include particular consideration for the order in which the targets would have been visited, given the small number.

Jing et al. [117] instead considered the problem of visiting a larger number of spacecraft in GEO employing multiple spacecraft. The problem was formulated as a multi-objective Travelling Salesman Problem minimising ΔV and total mission time, and was solved using MOPSO [27], a Particle Swarm optimisation approach. Also in this case, the cost of the transfers in term of ΔV was computed using simple analytical formulas.

As a final example of application, the framework proposed in this thesis is here applied to design a space mission to remove three defunct satellites on the Geostationary orbit by applying a de-orbiting kit to each of them. Differently from the previously mentioned works, no simple analytical models are employed for the trajectories, computation of propellant mass or time of flight. A multi-objective hybrid optimal control problem is solved instead, as illustrated in Chapter 5. Operative time constraints are included for the rendez-vous and docking operations, and the application of the de-orbiting kits.

7.1 Problem formulation

The spacecraft is assumed to start from an equatorial orbit. Thus it is convenient to express its dynamics using the full nonlinear equations of relative motion in Hill's frame[118], restricted to a planar motion:

$$\begin{cases} \dot{x} = v_x \\ \dot{y} = v_y \\ \dot{v_x} = 2\dot{f}\left(v_y - y\dfrac{\dot{r_c}}{r_c}\right) + x\dot{f}^2 + \dfrac{\mu}{r_c^2} - \dfrac{\mu(r_c + x)}{r_d^3} + \dfrac{uT\cos(\alpha)}{m} \\ \dot{v_y} = -2\dot{f}\left(v_x - x\dfrac{\dot{r_c}}{r_c}\right) + y\dot{f}^2 - \dfrac{\mu y}{r_d^3} + \dfrac{uT\sin(\alpha)}{m} \\ \dot{m} = \dfrac{-uT}{g_0 I_{sp}} \end{cases} \tag{7.1}$$

where x and y are the positions of the spacecraft relative to a reference point on the Geostationary orbit, v_x and v_y are the relative velocities, $\dot{f}$ is the rate of change of true anomaly of the Geostationary orbit, r_c is the radius of the Geostationary orbit, r_d is the distance of the current position to the centre of the Earth and and m is the mass of the spacecraft, which is controlled by an engine with maximum thrust T and a specific impulse I_{sp}. The thrust vector is expressed in terms of throttling u and direction α. g_0 is the gravitational acceleration at sea level, while μ is the gravitational parameter of Earth.

The spacecraft has an initial mass of 1000 kg, a minimum mass of 100 kg and is equipped with 3 de-orbiting kits, each with a mass of 30 kg. The propulsion system is able to generate up to 10 N of thrust with a specific impulse of 300 s.

The spacecraft starts from an equatorial circular orbit with radius of 42 000 km, which is at a safe distance from the target orbit. This corresponds in the selected Hill's frame to the following initial conditions:

$$\begin{cases} x(0) = -164\,\mathrm{km} \\ y(0) = 0\,\mathrm{km} \\ v_x(0) = 0\,\mathrm{m\,s^{-1}} \\ v_y(0) = 5.9971\,\mathrm{m\,s^{-1}} \end{cases} \tag{7.2}$$

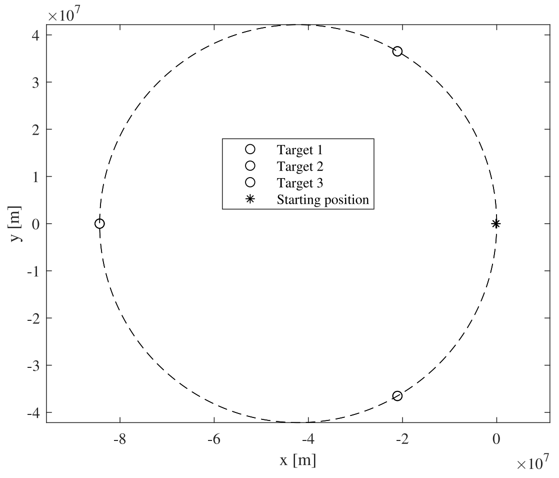


Figure 7.1: Positions of the targets and initial position of the spacecraft. Starting position lays inside the circle.

Since the reference frame is collocated on a point on the Geostationary orbit, the targets have a fixed position in time. This significantly simplifies the treatment of the boundary conditions, which no longer have a periodic time dependency over time. In this reference frame, they are assumed to be uniformly spread, forming an angle with the reference point on the Geostationary orbit of 60 deg, 180 deg and 300 deg. Since in this reference frame their positions are constant, their velocities are $0\,\mathrm{m\,s^{-1}}$. Figure 7.1 shows position of the targets and the initial position of the spacecraft in the relative frame.

The spacecraft has to rendez-vous and dock with all targets in order to apply the de-orbit kits, the application of each is assumed to take 5 h. The order in which targets are to be visited is not specified a priori. The objectives are the minimisation of the total mission time, and the maximisation of the final mass, corresponding to the minimisation of propellant consumption:

$$\min_{t_f,\mathbf{u},\boldsymbol{\mu}} (J_1, J_2)^T = (t_f, -m_f) \tag{7.3}$$

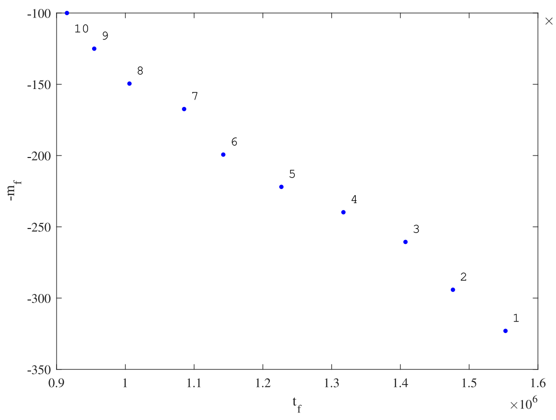
7.2 Results

The problem was formulated as a 3 phase problem, each phase was discretised with 3 DFET elements of order 7, resulting in a problem with 551 variables. The upper and lower bounds for state and control variables are reported in Table 7.1. The algorithm was run for 100000 function evaluations with standard settings, 10 agents storing 10 points on the Pareto front. These values for the number of points on the Pareto front and number of agents follow from the consideration that the problem has only two objectives and thus 10 points should provide a good approximation of the Pareto front. However, since this is a mixed integer problem, multiple local solutions are expected, thus a larger budget of function evaluations per agents was considered necessary: with 100000 function evaluations and 10 agents, each would have on average 10000 function evaluations, equivalent to 18 times the number of variables. The algorithm ran for approximately 2 days on a normal workstation. Part of this significant computational time is due to the implementation of the SQP solver in *fmincon*: the solution of the quadratic subproblem step can take very different amounts of time between two consecutive iterations of SQP, sometimes several minutes against an average of a fraction of a second for the entire run of *fmincon*.

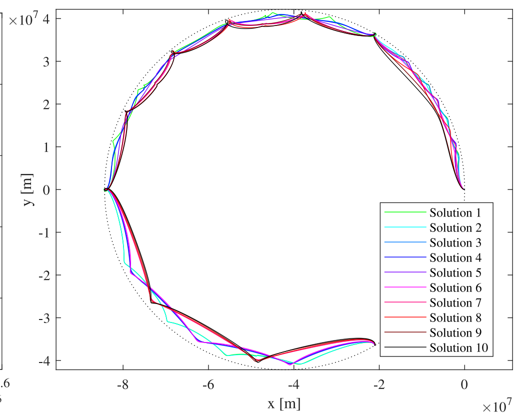
Table 7.1: Lower and upper bounds for the state and control variables of the multi debris removal mission

Variable	Lower bound	Upper bound
x (km)	-126492	42164
y (km)	-84328	84328
v_x (km s^{-1})	-20	20
v_y (km s^{-1})	-20	20
m (kg)	100	1000
u	0	1
α (rad)	$-\pi$	π
t_f (d)	0	40

Figure 7.2 shows the Pareto front and the trajectories of the spacecraft in the relative frame. A tradeoff between mission time and final mass is present, as expected. Faster trajectories tend to be closer to the centre of the circle, meaning that in order to reduce



(a) Pareto front



(b) Trajectories

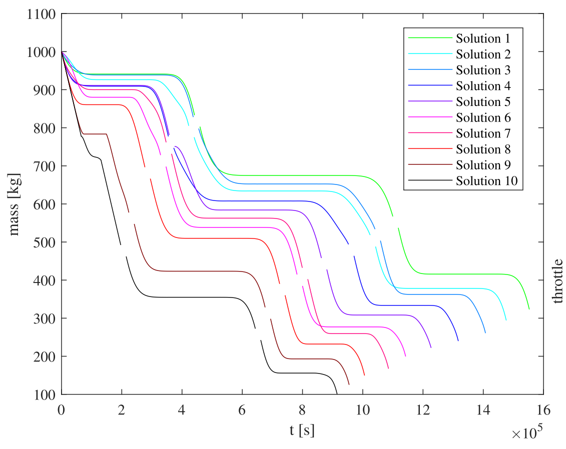
Figure 7.2: Pareto front and Trajectories for the multiple debris removal mission

mission time the optimiser exploited the relative rotation rate between the two orbits. All targets are visited in the same order. The epicycloidal shape of the trajectories in the relative frame means that the trajectories are more eccentric, while more circular shaped arcs mean that the corresponding trajectory is closer to circular.

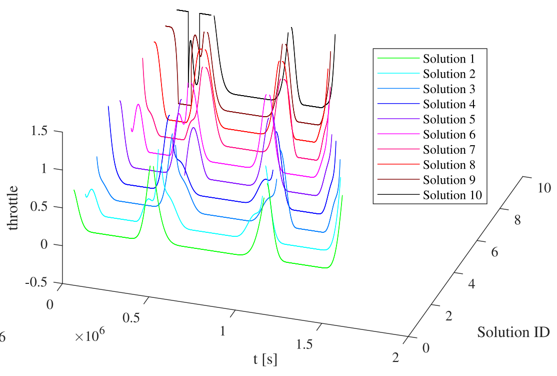
Figure 7.3 shows the time histories of the mass of the vehicle and the throttling of the engine. The minimum time solution requires approximately 10 days and reached the lower bound of the final mass, meaning that it employed all available propellant. Interestingly, the minimum time solution does not have enough propellant to proceed at full throttle all the time, thus the optimiser had to save propellant and increase the mission time in order to reach all targets.

In general, all solutions have a bang-zero-bang structure, as expected, where shorter mission times solutions have comparatively longer burns. The throttle and mass profiles are not very sharp, meaning that the solutions would benefit from mesh refinement and that some improvements in the values of the objective functions are expected. However, the overall trend and trade-off between the solutions is well captured.

Starting from the existing solutions, the problem was solved again doubling the number of elements per phase. The same maximum number of function evaluations



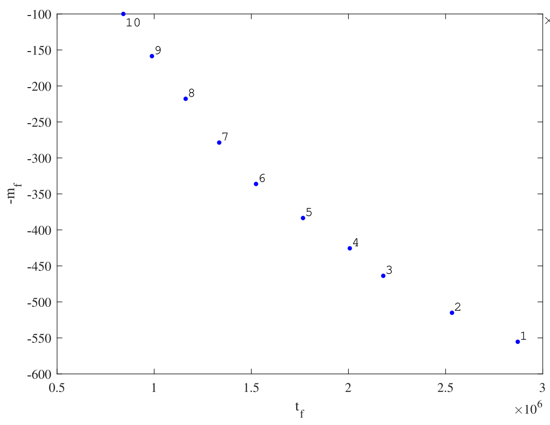
(a) Time history of the mass



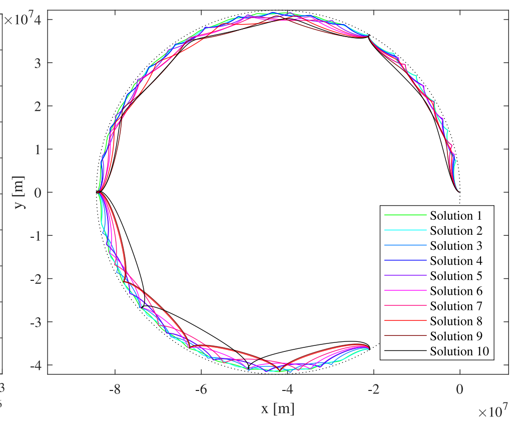
(b) Time history of the throttle

Figure 7.3: Time histories of velocity and acceleration for the time dependent Travelling Salesman Problem

was kept the same as in the previous run, since the initial solutions were considered good guesses but the number of variables doubled. Figure 7.4a shows the Pareto front for this refined run. The shape of the Pareto front is similar to the previous case, but its extension is longer, since now long mission time solutions have a much longer duration than from the previous case, and this in turn allows to save more propellant. This behaviour is not surprising: a more refined mesh allows not only to have sharper control profiles, but also to describe more accurately the trajectories of the spacecraft. Figure 7.4b shows the trajectories for this refined case. As it is possible to see, with more elements the are more points to represent the states and thus the trajectories can now describe narrower epicycloids. Physically, this means that the trajectory is performing more revolutions than in the previous case. The longer time missions are thus made possible by the higher resolution of the mesh, which can now describe more revolutions. Figure 7.5 shows the mass as a function of time, which has now sharper profiles than in the previous case.



(a) Pareto front



(b) Trajectories

Figure 7.4: Pareto front and Trajectories for the multiple debris removal mission, refined mesh

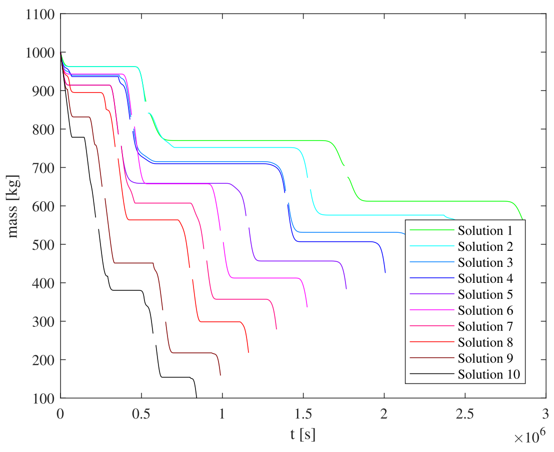


Figure 7.5: Time history of the mass with refined mesh

7.3 Chapter summary

This chapter applied the approach presented in Chapter 5 to the design of a multi-target debris removal space mission, in which the order of the targets was not specified a priori. The proposed approach was able to find feasible mission schedules and trajectories and construct a uniformly spread Pareto front, composed of solutions minimising mission time and propellant consumption.

The problem was solved again using a higher number of elements for the transcription. Unsurprisingly, a higher number of elements allows not only to describe sharper control profiles, but also to describe more complex trajectories in the Hill rotating reference frames. In an inertial frame these shapes are trajectories with multiple revolutions around the Earth. A higher number of elements allows thus enough flexibility to integrate longer term dynamics and represent correctly solutions that were not possible to describe with less elements. It seems then necessary to have a good idea of the number of revolutions or in order to choose and adequate number of elements. This is in common with other direct methods and is indeed one of the main difficulties of many revolutions trajectory optimisation, especially when low thrust propulsion is employed. Automatic mesh refinement procedures would alleviate this problem. The topic of automatic mesh refinement in the current framework is thus considered to be a challenging but important development. Alternatively, with the knowledge of the problem just gained, it could be possible to reformulate it by adding some intermediate phases describing the long coasting arcs. Those long coasting arcs could then be more efficiently integrated by expressing the dynamics with keplerian or equinoctial elements, or even by using simple analytical formulas.

Whereas in the test cases of the previous Chapter the difficulty lied in the presence of many constraints and of a system design aspect, the difficulty of the test case treated in this Chapter lies purely on the trajectory aspect. In fact, in terms of combinatorial complexity it is equivalent to the motorised TSP problem solved in Chapter 5. The main difference is the presence of sensitive nonlinear dynamics that need to be integrated for long times, resulting in trajectories with many revolutions.

Albeit the problem was posed more as a conceptual investigation and test of the algorithm than a real life mission design problem, it should be evident that this approach to the automated solution of mission design problems would bring useful insight on the solution structure and characteristics of the problem. Comparing it to other approaches in the literature it is more expensive to run but it does not require any simplification of the dynamical model. By changing the values for the specific impulse and for the thrust accordingly, the same set up could be used to plan low thrust missions, provided that an adequate number of elements is provided.

The main limitation of the formulation here adopted lies in its computational cost. Solving problems with several dozens of targets, like for the 9th edition of the GTOC competition [119], would require to set up the same number of phases as there are targets, and the number of elements would also increase accordingly. The computational cost of solving those NLPs would thus be exceptional, although the use of large scale solvers, an implementation using a compiled language and distributed computing could perhaps allow to tackle those kinds of problems in reasonable times. The practical applications described in the literature however seem to indicate that at the moment there is more interest in designing debris removal missions for a small number of strategic targets, than in designing a mission de-orbiting a very large number of generic targets.

Chapter 8

Conclusion

Per Aspera Sic Itur Ad Astra

Through hardships to the stars

Seneca

This dissertation treated the problem of the design optimisation of complex systems, when their dynamics and control is included in the process, multiple conflicting objectives are considered and some optimisation variables are continuous while others are integer. This results in the formulation of multi-objective hybrid optimal control problems, which are very complex problems requiring specific theoretical and numerical tools in order to be tackled. The goal was to create a unified approach applicable to general problems, validate it against known problems and finally test it on more complex space related problems.

Part I of the dissertation was focused on the theoretical and algorithmic developments necessary to obtain a working framework with provable characteristics. Each chapter in this part dealt with one major building block of the framework. New theoretical results were given, the algorithms were described with particular emphasis on their interactions, and each contribution was tested against known benchmark problems in order to validate it.

Chapter 2 treated the optimal control aspect of the problem, which was solved by a direct transcription method based on finite elements. With the adoption of a new class of polynomials for the basis functions, it was shown that the DFET method is able to generate smooth monotonic approximations to bang-bang control laws, which appear very often in practical problems. These approximations can improve arbitrarily in accuracy as the order of the polynomials increases. The drawback of the method is mainly its slower convergence rate to the asymptotic value of the objective function with respect to the typically used bases. Despite this fact, the approximation of the control laws can be significantly closer to the analytical ones if discontinuities are present. In addition, a theoretical result was given in form of a Theorem, stating that the proposed scheme is able to guarantee the satisfaction of inequality path constraints for all times even for a finite approximation, provided the feasible set is convex. Finally, it was shown that the number of iterations required by the NLP solver to converge to an optimal solution can be significantly lower than with the traditional bases, indicating a more favourable numerical conditioning.

Chapter 3 treated the general multi-objective optimisation aspect of the problem, which was solved by the Multi Agent Collaborative Search, a population based global memetic algorithm. The chapter described some modifications of the existing heuristics, in most of the cases outperforming both its previous version and MOEA/D, one of the leading multi-objective optimisation algorithms. A new archiving strategy was introduced in order to improve the spreading of the solutions in criteria space. The archiving strategy is based on the physical concept of the minimisation of the potential energy of a system of electrically repelling particles. This concept was also used to improve the distribution of the search directions in criteria space.

Chapter 4 treated the solution of multi-phase multi-objective optimal control problems. Through the DFET transcription, the problem was transformed into a multi-objective NLP. The resulting problem was then reformulated in two complementary approaches: one bi-level and the other single level. With the bi-level approach, the outer level is handled by MACS, which employs its heuristics to generate tentative control laws and static optimisation parameters. The tentative solution vector is then passed to the inner level, which tries to satisfy the constraints by solving a NLP. Once a feasible solution is found, it is passed back to the outer level, which evaluates the objectives and decides if the current solution is to be added to the archive of non-dominated solutions. This approach is able to perform a global search and still guarantees tight satisfaction of the constraints. The single level approach instead employs the Pascoletti-Serafini scalarisation, resulting in a single objective problem along a particular direction in criteria space. This approach is employed as a refinement method, since it can be directly tackled by an NLP solver, thus giving guarantees about the local optimality of the solution. To corroborate this aspect, necessary conditions for optimality were derived for a Pascoletti-Serafini scalarised multi-objective optimal control problem. A remarkable aspect of the two approaches is that it was shown that it is possible to smoothly transition from one to the other. An additional aspect described in this chapter is an approach to automatically generate initial guesses. Finally, the method was tested against two problems with known solution, where it was shown its ability to automatically generate a uniform spread of Pareto optimal solutions agreeing with the known solutions.

Chapter 4 treated the solution of multi-phase multi-objective optimal control problems. Through the DFET transcription, the problem was transformed into a multi-objective NLP. The resulting problem was then reformulated in two complementary approaches: one bi-level and the other single level. With the bi-level approach, the outer level is handled by MACS, which employs its heuristics to generate tentative control laws and static optimisation parameters. The tentative solution vector is then passed to the inner level, which tries to satisfy the constraints by solving a NLP. Once a feasible solution is found, it is passed back to the outer level, which evaluates the objectives and decides if the current solution is to be added to the archive of non-dominated solutions. This approach is able to perform a global search and still guarantees tight satisfaction of the constraints. The single level approach instead employs the Pascoletti-Serafini scalarisation, resulting in a single objective problem along a particular direction in criteria space. This approach is employed as a refinement method, since it can be directly tackled by an NLP solver, thus giving guarantees about the local optimality of the solution. To corroborate this aspect, necessary conditions for optimality were derived for a Pascoletti-Serafini scalarised multi-objective optimal control problem. A remarkable aspect of the two approaches is that it was shown that it is possible to smoothly transition from one to the other. An additional aspect described in this chapter is an approach to automatically generate initial guesses. Finally, the method was tested against two problems with known solution, where it was shown its ability to automatically generate a uniform spread of Pareto optimal solutions agreeing with the known solutions.

Chapter 5 extended the approach of the previous chapter in order to treat mixed-integer problems. As these problems are exceptionally complex and it is known that they can be unsolvable in the general case, the approach adopted is mostly heuristic. The heuristics of MACS were modified in order to treat transparently discrete and continuous variables, while a relaxation approach was employed when solving NLP problems within the inner level or the single level approach. This allows the NLP the flexibility to change the value of the relaxed discrete variables, thus giving it more chances of solving problems with complex mixed-integer constraints. In order to ensure that the relaxed variables assume only the allowed integer values, a simple additional constraint is added, but it is enforced only after a relaxed solution is found. This way, integrality constraints are imposed systematically while minimising the additional complexity to be handled by the NLP. A special set of constraints was also introduced to treat optimal control problems where the final target is not assigned but has to be chosen among a discrete set of possibilities. The method was tested against a known problem: the Motorised Travelling Salesman problem, which showcases all the difficulties encountered in this chapter. It was shown that the proposed approach is able to automatically return the best solution known. In addition, a more complex variant of the problem was solved, where the targets visited by the salesman are not fixed but move along a prescribed trajectory.

Part II of the dissertation is focused on the application of the framework described in Part I to challenging system, mission and trajectory design problems in the aerospace sector. These problems are characterised by having highly non-linear dynamics, multiple controls, the presence of static design variables that can significantly interact with the trajectory of the vehicle, and multiple objectives. Each of these applications can be seen both as an original result and an example on how the possibility to evaluate optimal trade-offs can allow the designers to take more informed decisions.

Chapter 6 shows the application of the proposed approach to the optimisation of the trajectory of three transatmospheric vehicles. The first case was a multi-objective extension of a known case, the re-entry trajectory of a Space Shuttle like vehicle. The objectives to be optimised were the crossrange and the peak heat flux. This example showed how it is possible to reduce the peak heat fluxes by sacrificing some steering capabilities, and demonstrated that the approach can return solutions with an excellent agreement with known solutions even for more complex problems. The second case was an integrated vehicle and trajectory design of a two stage launch vehicle. The objectives to optimise were the propellant mass and the thrust to weight ratios of each stage, in order to find designs with relatively cheap and small engines and reduce the accelerations experienced by each stage. This case showed the usefulness of the proposed approach, since it generated a trade-off surface of entirely different design strategies, allowing the decision maker to gain a much better understanding of the possibilities. The final test case involved the ascent and abort trajectories of an air dropped single stage vehicle in case its engine stopped working. In this case, the problem was to find designs which were mass efficient but also guaranteed a certain inherent safety in case of main engine failure. The safety was measured by the length of the downrange of the eventual abort trajectory. The method allowed to generate families of trajectories, trading off mass efficiency for downrange distance, for a given assumed main engine failure time. By repeating this analysis with increasing spans of correct functioning for the main engine, it was discovered that a high wing size contributes to the downrange of the vehicle only until a certain critical point, after which the added mass and drag penalise the downrange instead of increasing it.

Chapter 7 dealt with an integrated mission design and trajectory optimisation to design a hypothetical space mission, whose task was to apply de-orbiting kits to multiple defunct satellites in Geostationary Orbit. Since the order of the targets is not important and can be freely optimised, the problem resembles a dynamic Travelling Salesman Problem, albeit with a much more complex dynamics. The objectives to optimise were the mass of the propellant and the mission time. The algorithm found sensible mission plans and trajectories that exploited the natural dynamics of the problem. For this problem, assuming the spacecraft is already located close to the GEO orbit, this study indicates it is possible to visit three equally spaced targets in less than 10 days by consuming all available propellant, or saving substantial amount of propellant while still visiting all targets in less than one month. Given the increasing economic interest in placing, maintaining and disposing satellites in GEO orbit, the usefulness of this test case should be evident.

Despite having achieved all the desired goals, there is ample room for improvements to this work. One first aspect that should be mentioned is that the quality of the trajectories and control laws obtained via the DFET transcription depends on the number of elements and the order of the polynomials employed. This is especially true for many revolutions trajectory optimisation, or if the solutions contain bang-bang profiles, for which the proposed Bernstein basis returns a smooth and monotonic approximation that uniformly, but slowly, approaches to the true solution. In order to improve the accuracy of those solutions, an h-p adaptive mesh strategy should be employed. The technique has already been developed and demonstrated for the DFET method, and additional work is already ongoing exploiting the properties of the Bezier curves originating by the choice of adopting the Bernstein basis. The main difficulty with mesh refinement in a multi-objective context is that each Pareto optimal solution will probably require a different refinement pattern. Albeit it would be certainly possible to refine each solution individually after having found the Pareto front, it would also be interesting to investigate the benefits of performing the mesh refinement within the multi-objective optimisation loop. This approach shares a difficulty in common with a class of problems that was purposefully omitted from this work: the treatment of problems with variable size. Future work will probably have to deal with this characteristic, both because it allows to treat an even wider class of problems, and because it could allow the desired functionality of online mesh refinement.

Another interesting future work direction is the dynamic adaptation of the search directions in criteria space for MACS, especially for problems with more than three objectives. Since it has been amply demonstrated that for many objective problems the dominance criterion becomes progressively ineffective, scalarisation methods have gained increasing popularity. However, one major research topic in that field is how to adapt those search directions depending on the current shape of the approximated of the Pareto front. As it stands, MACS has a way to generate a well spread set of search directions, but those directions are kept constant. Previous versions of MACS generated a high number of random directions and periodically changed the association between agent and search direction, but the results of that approach were mixed. New research is already ongoing on how to dynamically adapt the search directions in criteria space in a more systematic fashion.

Finally, the treatment of the uncertainties was almost completely omitted from this work. Robust design and resilience engineering are new major trends, since they try to design a system with reliable performances even when some of its design parameters, or the environment in which it operates, turn out to be different than expected. It would be interesting to see whether the proposed framework could benefit the field of robust design by allowing the treatment of multiple criteria and including the optimal control aspect in a unified fashion. Several possibilities exist on how to introduce those important considerations within the proposed framework, both at the level of the control laws and at the level of the system components. Some research ideas in this direction are already being considered.

Although the above mentioned aspects are of great importance, this work has proven the usefulness of a unified and consistent framework able to treat complex systems in a holistic fashion and considering multiple conflicting objectives. Its applicability is not restricted to a specific domain, like aerospace, and it is built on strong theoretical foundations. As such, it is hoped that it will be used as a backbone on which the above mentioned aspects are built upon.

Appendix A

Necessary conditions for optimality

This appendix includes the derivation of the necessary conditions for optimality employed in the thesis. In particular, it includes the necessary conditions for optimality for the ordinary single objective optimal control problem, the necessary conditions for optimality for the Pascoletti-Serafini scalarised multi-objective optimal control problem, and the necessary conditions for optimality for the discretised optimal control problem originating from the application of the DFET transcription. By following these three derivations, it is possible to prove the convergence of the method employed in this thesis.

A.1 Optimal control of time-continuous systems

The Bolza form of optimal control problems reads:

$$\begin{aligned}
&\min_{\mathbf{u}} J = \phi\left(\mathbf{x}(t_0), \mathbf{x}(t_f), t_0, t_f\right) + \int_{t_0}^{t_f} L\left(\mathbf{x}(t), \mathbf{u}(t), t\right) dt \\
&s.t \\
&\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}(t), \mathbf{u}(t), t) \\
&\mathbf{g}(\mathbf{x}(t), \mathbf{u}(t), t) \leq 0 \\
&\boldsymbol{\psi}(\mathbf{x}(t_0), \mathbf{x}(t_f), t_0, t_f) \leq 0 \\
&t \in [t_0, t_f]
\end{aligned} \tag{Bolza}$$

One standard procedure when dealing with constrained optimisation is to transform the problem into an unconstrained one by augmenting the objective function with the constraints:

$$\min_{\mathbf{x},\mathbf{u},\boldsymbol{\lambda},\boldsymbol{\nu}} J^{\dagger} = \phi + \boldsymbol{\nu}^T \boldsymbol{\psi} + \int_{t_0}^{t_f} \left(L + \boldsymbol{\mu}^T(t)\mathbf{g} + \boldsymbol{\lambda}^T(t)(\mathbf{F} - \dot{\mathbf{x}})\right) dt \tag{A.1}$$

where $\boldsymbol{\nu}$, $\boldsymbol{\mu}(t)$ and $\boldsymbol{\lambda}(t)$ are Lagrange multipliers, $\boldsymbol{\nu} \geq 0, \boldsymbol{\mu}(t) \geq 0$, and all other functional dependencies have been dropped to improve readability.

Computing the first variation of $J^{\dagger}$ gives:

$$\begin{aligned}
\delta J^{\dagger} =& \delta\phi + \delta\boldsymbol{\nu}^T \boldsymbol{\psi} + \boldsymbol{\nu}^T \delta\boldsymbol{\psi} + \\
&\int_{t_0}^{t_f} \left(\delta L + \delta\boldsymbol{\mu}(t)^T \mathbf{g} + \boldsymbol{\mu}(t)^T \delta\mathbf{g} + \delta\boldsymbol{\lambda}^T(t)(\mathbf{F} - \dot{\mathbf{x}}) + \boldsymbol{\lambda}^T(t)(\delta\mathbf{F} - \delta\dot{\mathbf{x}})\right) dt
\end{aligned} \tag{A.2}$$

It is useful to integrate by parts the term $\boldsymbol{\lambda}^T(t)\delta\dot{\mathbf{x}}$

$$\int_{t_0}^{t_f} -\boldsymbol{\lambda}^T(t)\delta\dot{\mathbf{x}} = -\left[\boldsymbol{\lambda}^T(t)\delta\mathbf{x}\right]_{t_0}^{t_f} + \int_{t_0}^{t_f} \dot{\boldsymbol{\lambda}}^T \delta\mathbf{x} dt \tag{A.3}$$

this way, the first order variation $J^\dagger$ becomes

$$\begin{aligned}\delta J^\dagger =& \delta\phi + \delta\boldsymbol{\nu}^T\boldsymbol{\psi} + \boldsymbol{\nu}^T\delta\boldsymbol{\psi} - \boldsymbol{\lambda}^T(t_f)\delta\mathbf{x}(t_f) + \boldsymbol{\lambda}^T(t_0)\delta\mathbf{x}(t_0) + \\ & \int_{t_0}^{t_f}\Big(\delta L + \delta\boldsymbol{\mu}(t)^T\mathbf{g} + \boldsymbol{\mu}(t)^T\delta\mathbf{g} + \delta\boldsymbol{\lambda}^T(t)(\mathbf{F}-\dot{\mathbf{x}}) + \boldsymbol{\lambda}^T(t)\delta\mathbf{F} + \dot{\boldsymbol{\lambda}}^T\delta\mathbf{x}\Big)\,dt\end{aligned} \tag{A.4}$$

At this point, by assuming the differentiability of $\phi, L, \boldsymbol{\psi}, \mathbf{F}$ and $\mathbf{g}$

$$\begin{aligned}\delta J^\dagger =& \nabla_x\phi\delta\mathbf{x}(t_0) + \nabla_x\phi\delta\mathbf{x}(t_f) + \frac{\partial\phi}{\partial t}\delta t_0 + \frac{\partial\phi}{\partial t}\delta t_f + \\ & \delta\boldsymbol{\nu}^T\boldsymbol{\psi} + \boldsymbol{\nu}^T\nabla_x\boldsymbol{\psi}\delta\mathbf{x}(t_0) + \boldsymbol{\nu}^T\nabla_x\boldsymbol{\psi}\delta\mathbf{x}(t_f) + \boldsymbol{\nu}^T\frac{\partial\boldsymbol{\psi}}{\partial t}\delta t_0 + \boldsymbol{\nu}^T\frac{\partial\boldsymbol{\psi}}{\partial t}\delta t_f - \\ & \boldsymbol{\lambda}^T(t_f)\delta\mathbf{x}(t_f) + \boldsymbol{\lambda}^T(t_0)\delta\mathbf{x}(t_0) + \\ & \int_{t_0}^{t_f}\Big(\nabla_x L\delta\mathbf{x} + \nabla_u L\delta\mathbf{u} + \delta\boldsymbol{\mu}^T(t)\mathbf{g} + \boldsymbol{\mu}^T(t)\nabla_x\mathbf{g}\delta\mathbf{x} + \boldsymbol{\mu}^T(t)\nabla_u\mathbf{g}\delta\mathbf{u} + \\ & \qquad \delta\boldsymbol{\lambda}^T(t)(\mathbf{F}-\dot{\mathbf{x}}) + \boldsymbol{\lambda}^T(t)\nabla_x\mathbf{F}\delta\mathbf{x} + \boldsymbol{\lambda}^T(t)\nabla_u\mathbf{F}\delta\mathbf{u} + \dot{\boldsymbol{\lambda}}^T\delta\mathbf{x}\Big)dt + \\ & \big[L + \boldsymbol{\mu}^T(t)\mathbf{g} + \boldsymbol{\lambda}^T(t)\mathbf{F}\big]_{t_f}\delta t_f - \big[L + \boldsymbol{\mu}^T(t)\mathbf{g} + \boldsymbol{\lambda}^T(t)\mathbf{F}\big]_{t_0}\delta t_0\end{aligned} \tag{A.5}$$

Now, by collecting the terms with the same variation

$$\begin{aligned}\delta J^\dagger =& \left(\nabla_x\phi + \boldsymbol{\nu}^T\nabla_x\boldsymbol{\psi} + \boldsymbol{\lambda}^T(t_0)\right)\delta\mathbf{x}(t_0) + \left(\nabla_x\phi + \boldsymbol{\nu}^T\nabla_x\boldsymbol{\psi} - \boldsymbol{\lambda}^T(t_f)\right)\delta\mathbf{x}(t_f) + \delta\boldsymbol{\nu}^T\boldsymbol{\psi} + \\ & \left(\frac{\partial\phi}{\partial t} + \boldsymbol{\nu}^T\frac{\partial\boldsymbol{\psi}}{\partial t} - L - \boldsymbol{\mu}^T(t)\mathbf{g} - \boldsymbol{\lambda}^T(t)\mathbf{F}\right)\delta t_0 + \\ & \left(\frac{\partial\phi}{\partial t} + \boldsymbol{\nu}^T\frac{\partial\boldsymbol{\psi}}{\partial t} + L + \boldsymbol{\mu}^T(t)\mathbf{g} + \boldsymbol{\lambda}^T(t)\mathbf{F}\right)\delta t_f + \\ & \int_{t_0}^{t_f}\Bigg(\Big(\nabla_x L + \boldsymbol{\mu}^T(t)\nabla_x\mathbf{g} + \boldsymbol{\lambda}^T(t)\nabla_x\mathbf{F} + \dot{\boldsymbol{\lambda}}^T\Big)\delta\mathbf{x} + \\ & \qquad \Big(\nabla_u L + \boldsymbol{\mu}^T(t)\nabla_u\mathbf{g} + \boldsymbol{\lambda}^T(t)\nabla_u\mathbf{F}\Big)\delta\mathbf{u} + \\ & \qquad \delta\boldsymbol{\mu}^T(t)\mathbf{g} + \delta\boldsymbol{\lambda}^T(t)\left(\mathbf{F}-\dot{\mathbf{x}}\right)\Bigg)dt\end{aligned} \tag{A.6}$$

Finally, by requiring the total variation $\delta J^\dagger$ to be 0 for all admissible variations, it is possible to derive the necessary first order conditions for local optimality:

$$
\begin{aligned}
&\boldsymbol{\lambda}^T(t_0) = -\left(\nabla_x \phi + \boldsymbol{\nu}^T \nabla_x \boldsymbol{\psi}\right)_{t=t_0} \\
&\boldsymbol{\lambda}^T(t_f) = \left(\nabla_x \phi + \boldsymbol{\nu}^T \nabla_x \boldsymbol{\psi}\right)_{t=t_f} \\
&\left(\frac{\partial \phi}{\partial t} + \boldsymbol{\nu}^T \frac{\partial \boldsymbol{\psi}}{\partial t} - L - \boldsymbol{\mu}^T(t)\mathbf{g} - \boldsymbol{\lambda}^T(t)\mathbf{F}\right)_{t_0} = 0 \\
&\left(\frac{\partial \phi}{\partial t} + \boldsymbol{\nu}^T \frac{\partial \boldsymbol{\psi}}{\partial t} + L + \boldsymbol{\mu}^T(t)\mathbf{g} + \boldsymbol{\lambda}^T(t)\mathbf{F}\right)_{t_f} = 0 \\
&\psi_i \leq 0; \quad \nu_i = 0 \text{ where } \psi_i < 0 \\
&\dot{\boldsymbol{\lambda}}^T = -\Big(\nabla_x L + \boldsymbol{\mu}^T(t)\nabla_x \mathbf{g} + \boldsymbol{\lambda}^T(t)\nabla_x \mathbf{F}\Big) \\
&\dot{\mathbf{x}} = F \\
&\Big(\nabla_u L + \boldsymbol{\mu}^T(t)\nabla_u \mathbf{g} + \boldsymbol{\lambda}^T(t)\nabla_u \mathbf{F}\Big) = 0 \\
&g_i \leq 0; \quad \mu_i = 0 \text{ where } g_i < 0
\end{aligned}
\tag{A.7}
$$

The first four equations, also known as transversality constraints, are required only for the free initial and final states and times, while they are not required for fixed initial and final states and times (because the allowed variation would be zero). The fifth equation is again the boundary conditions, the sixth equation is also called the adjoint equation, the seventh equation is the equation for the dynamics of the system, the eighth equation is the controls equation, and the last equation is again the path constraints inequality. It must be stressed that all the multipliers $\boldsymbol{\nu}, \boldsymbol{\mu}$ must be positive when the constraints are active and zero when the constraints are inactive.

A.2 Pascoletti-Serafini Scalarised MOCP

If function J in (Bolza) was replaced by the vector function $\mathbf{J} = [J_1, ..., J_j, ...J_m]^T$ one could use the Pascoletti-Serafini scalarisation approach to obtain:

$$\begin{aligned}
&\min_{s_f \geq 0} s_f \\
&s.t. \\
&\omega_j(J_j(\mathbf{x}_f, t_f) - z_j) - s_f \leq 0 \qquad \forall j = 1, ..., m \\
&\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{u}, t) \\
&\mathbf{g}(\mathbf{x}, \mathbf{u}, t) \leq 0 \\
&\boldsymbol{\psi}(\mathbf{x}_0, \mathbf{x}_f, t_0, t_f) \leq 0 \\
&t \in [t_0, t_f]
\end{aligned} \tag{PSOCP}$$

If s is a slack variable with final condition s_f and zero time variation $\dot{s} = 0$, then problem (PSOCP) presents itself in a form similar to Mayer's problem, which is a Bolza problem where the running cost function L is zero. The major difference is the mixed boundary constraint on x_f, t_f and s_f for every $j = 1, ..., m$. It is now possible to prove the following:

Theorem 5 *Consider the function $H = \boldsymbol{\lambda}^T \mathbf{F}(\mathbf{x}, \mathbf{u}, t) + \boldsymbol{\mu}^T \mathbf{g}(\mathbf{x}, \mathbf{u}, t)$. If $\mathbf{u}^*$ is a locally optimal solution for problem (PSOCP), with associated state vector $\mathbf{x}^*$, and H is Frechet differentiable at $\mathbf{u}^*$ and a regular point of the algebraic constraints, then there exist vectors $\boldsymbol{\eta} \in \mathbb{R}^m$, $\boldsymbol{\lambda} \in \mathbb{R}^n$ and $\boldsymbol{\mu} \in \mathbb{R}^q$ such that:*

$$\begin{aligned}
&\mathbf{u}^* = \operatorname{argmin}_{\mathbf{u} \in U} \boldsymbol{\lambda}^T \mathbf{F}(\mathbf{x}^*, \mathbf{u}, t) + \boldsymbol{\mu}^T \mathbf{g}(\mathbf{x}^*, \mathbf{u}, t) \\
&\boldsymbol{\lambda}^T \nabla_{\mathbf{x}} \mathbf{F}(\mathbf{x}^*, \mathbf{u}^*, t) + \boldsymbol{\mu}^T \nabla_{\mathbf{x}} \mathbf{g}(\mathbf{x}^*, \mathbf{u}^*, t) + \dot{\boldsymbol{\lambda}}^T = 0 \\
&\dot{\lambda}_s = 0 \\
&\boldsymbol{\mu} \geq 0
\end{aligned} \tag{A.8}$$

with transversality conditions:

$$\begin{aligned}
&1 - \textstyle\sum_j^m \eta_j + \lambda_{s_f}(t_f) = 0 \\
&\boldsymbol{\eta}^T \boldsymbol{\Omega} \nabla_{\mathbf{x}_f} \mathbf{J} + \boldsymbol{\nu}^T \nabla_{\mathbf{x}_f} \boldsymbol{\psi} + \lambda_x(t_f)^T = 0 \\
&\boldsymbol{\eta} > 0; \boldsymbol{\nu} \geq 0
\end{aligned} \tag{A.9}$$

and

$$\frac{\partial H}{\partial t_f} - \boldsymbol{\eta}^T \boldsymbol{\Omega} \frac{\partial \mathbf{J}}{\partial t_f} - \boldsymbol{\nu}^T \frac{\partial \boldsymbol{\psi}}{\partial t_f} = 0 \tag{A.10}$$

where $\boldsymbol{\Omega}$ *is a diagonal matrix with the components* ω_i, $i = 1, ..., m$ *along the diagonal and* U *is the space of admissible controls that satisfy respectively the algebraic and differential constraints* $\mathbf{g}(\mathbf{x}, \mathbf{u}, t) \leq 0$ *and* $\dot{\mathbf{x}} - \mathbf{F}(\mathbf{x}, \mathbf{u}, t) = 0$.

Proof 3 *A possible proof comes from the direct application of Pontryagin's maximum principle to problem (PSOCP). Alternatively, at the solution, one can take the first variation of the functional:*

$$L = s_f + \boldsymbol{\eta}^T(\boldsymbol{\Omega}(\mathbf{J} - \mathbf{z}) - s_f \mathbf{1}) + \boldsymbol{\nu}^T \boldsymbol{\psi} + \int_{t_0}^{t_f} [\boldsymbol{\lambda}^T(\mathbf{F} - \dot{\mathbf{x}}) + \lambda_s \dot{s} + \boldsymbol{\mu}^T \mathbf{g}] dt \tag{A.11}$$

with $\mathbf{1}$ *a vector of ones and* $\boldsymbol{\Omega}$ *a diagonal matrix with elements* ω_i*, which gives:*

$$\begin{aligned}
&\delta L = \delta s_f + \delta \boldsymbol{\eta}^T(\boldsymbol{\Omega}(\mathbf{J} - \mathbf{z}) - s_f \mathbf{1}) + \boldsymbol{\eta}^T \boldsymbol{\Omega} \delta \mathbf{J} + \boldsymbol{\eta}^T \delta(s_f \mathbf{1}) + \delta \boldsymbol{\nu}^T \boldsymbol{\psi} + \boldsymbol{\nu}^T \delta \boldsymbol{\psi} + \\
&\int_{t_0}^{t_f} [\delta \boldsymbol{\lambda}^T(\mathbf{F} - \dot{\mathbf{x}}) + \boldsymbol{\lambda}^T(\delta \mathbf{F} - \delta \dot{\mathbf{x}}) + \delta \lambda_s \dot{s} + \lambda_s \delta \dot{s} + \delta \boldsymbol{\mu}^T \mathbf{g} + \boldsymbol{\mu}^T \delta \mathbf{g}] dt = 0.
\end{aligned} \tag{A.12}$$

We can now collect terms with equal variation δ*:*

$$\begin{aligned}
&\delta L = \delta s_f(1 - \boldsymbol{\eta}^T \mathbf{1}) + \delta \boldsymbol{\eta}^T(\boldsymbol{\Omega}(\mathbf{J} - \mathbf{z}) - s_f \mathbf{1}) + \delta \boldsymbol{\nu}^T \boldsymbol{\psi} + (\boldsymbol{\eta}^T \boldsymbol{\Omega} \nabla_{\mathbf{x}_f} \mathbf{J} + \boldsymbol{\nu}^T \nabla_{\mathbf{x}_f} \boldsymbol{\psi}) \delta \mathbf{x}_{t_f} + \\
&\int_{t_0}^{t_f} [\delta \boldsymbol{\lambda}^T(\mathbf{F} - \dot{\mathbf{x}}) - \boldsymbol{\lambda}^T \delta \dot{\mathbf{x}} + (\boldsymbol{\lambda}^T \nabla_{\mathbf{x}} \mathbf{F} + \boldsymbol{\mu}^T \nabla_{\mathbf{x}} \mathbf{g}) \delta \mathbf{x} + (\boldsymbol{\lambda}^T \nabla_{\mathbf{u}} \mathbf{F} + \boldsymbol{\mu}^T \nabla_{\mathbf{u}} \mathbf{g}) \delta \mathbf{u} + \\
&\delta \lambda_s \dot{s} + \lambda_s \delta \dot{s} + \delta \boldsymbol{\mu}^T \mathbf{g}] dt + [\boldsymbol{\lambda}^T \mathbf{F} + \boldsymbol{\mu}^T \mathbf{g}]_{t_f} \delta t_f + \boldsymbol{\nu} \tfrac{\partial \boldsymbol{\psi}}{\partial t_f} \delta t_f + \boldsymbol{\eta}^T \boldsymbol{\Omega} \tfrac{\partial \mathbf{J}}{\partial t_f} \delta t_f = 0
\end{aligned} \tag{A.13}$$

and after integrating by parts the terms $\boldsymbol{\lambda}^T\delta\dot{\mathbf{x}}$ and $\lambda_s\delta\dot{s}$ we get:

$$\begin{aligned}
&\delta L = \delta s_f(1-\boldsymbol{\eta}^T\mathbf{1}) + \delta\boldsymbol{\eta}^T(\boldsymbol{\Omega}(\mathbf{J}-\mathbf{z}) - s_f\mathbf{1}) + \\
&\delta\boldsymbol{\nu}^T\boldsymbol{\psi} + (\boldsymbol{\eta}^T\boldsymbol{\Omega}\nabla_{\mathbf{x}_f}\mathbf{J} + \boldsymbol{\nu}^T\nabla_{\mathbf{x}_f}\boldsymbol{\psi})\delta\mathbf{x}_{t_f} - \boldsymbol{\lambda}(t_f)^T\delta\mathbf{x}_{t_f} + \lambda_{s_f}(t_f)\delta s_f \\
&\int_{t_0}^{t_f}[\delta\boldsymbol{\lambda}^T(\mathbf{F}-\dot{\mathbf{x}}) + \dot{\boldsymbol{\lambda}}^T\delta\mathbf{x} + (\boldsymbol{\lambda}^T\nabla_{\mathbf{x}}\mathbf{F} + \boldsymbol{\mu}^T\nabla_{\mathbf{x}}\mathbf{g})\delta\mathbf{x} + (\boldsymbol{\lambda}^T\nabla_{\mathbf{u}}\mathbf{F} + \boldsymbol{\mu}^T\nabla_{\mathbf{u}}\mathbf{g})\delta\mathbf{u} + \\
&\delta\lambda_s\dot{s} - \dot{\lambda}_s\delta s + \delta\boldsymbol{\mu}^T\mathbf{g}]dt + [\boldsymbol{\lambda}^T\mathbf{F} + \boldsymbol{\mu}^T\mathbf{g}]_{t_f}\delta t_f + \boldsymbol{\nu}\tfrac{\partial\boldsymbol{\psi}}{\partial t_f}\delta t_f + \boldsymbol{\eta}^T\boldsymbol{\Omega}\tfrac{\partial\mathbf{J}}{\partial t_f}\delta t_f = 0
\end{aligned} \tag{A.14}$$

Now in order for the variation to be zero for every value of the δ and d quantities the following equations must be satisfied:

$$\begin{aligned}
&\dot{\mathbf{x}} - \mathbf{F} = 0 \\
&\boldsymbol{\lambda}^T\nabla_{\mathbf{u}}\mathbf{F}(\mathbf{x}^*,\mathbf{u}^*,t) + \boldsymbol{\mu}^T\nabla_{\mathbf{u}}\mathbf{g}(\mathbf{x}^*,\mathbf{u}^*,t) = 0 \\
&\boldsymbol{\lambda}^T\nabla_{\mathbf{x}}\mathbf{F}(\mathbf{x}^*,\mathbf{u}^*,t) + \boldsymbol{\mu}^T\nabla_{\mathbf{x}}\mathbf{g}(\mathbf{x}^*,\mathbf{u}^*,t) + \dot{\boldsymbol{\lambda}}^T = 0 \\
&\mathbf{g}(\mathbf{x}^*,\mathbf{u}^*,t) \le 0 \\
&\dot{\lambda}_s = 0 \\
&\boldsymbol{\mu} \ge 0 \\
&1 - \textstyle\sum_j^m \eta_j + \lambda_{s_f}(t_f) = 0 \\
&\boldsymbol{\eta}^T\boldsymbol{\Omega}\nabla_{\mathbf{x}_f}\mathbf{J} + \boldsymbol{\nu}^T\nabla_{\mathbf{x}_f}\boldsymbol{\psi} - \boldsymbol{\lambda}(t_f)^T = 0 \\
&\boldsymbol{\Omega}(\mathbf{J}-\mathbf{z}) - s_f\mathbf{1} \le 0 \\
&\boldsymbol{\psi} \le 0 \\
&\boldsymbol{\eta} \ge 0; \boldsymbol{\nu} \ge 0
\end{aligned} \tag{A.15}$$

and

$$[\boldsymbol{\lambda}^T\mathbf{F} + \boldsymbol{\mu}^T\mathbf{g}]_{t_f} + \boldsymbol{\eta}^T\boldsymbol{\Omega}\frac{\partial\mathbf{J}}{\partial t_f} + \boldsymbol{\nu}^T\frac{\partial\boldsymbol{\psi}}{\partial t_f} = 0 \tag{A.16}$$

If now one introduces the function $H = \boldsymbol{\lambda}^T\mathbf{F}(\mathbf{x},\mathbf{u},t) + \boldsymbol{\mu}^T\mathbf{g}(\mathbf{x},\mathbf{u},t)$, Eqs. (A.15) reduce *to:*

$$\begin{aligned}
&\boldsymbol{\Omega}(\mathbf{J}-\mathbf{z}) - s_f\mathbf{1} \le 0\\
&\dot{\mathbf{x}} = \tfrac{\partial H}{\partial \boldsymbol{\lambda}}\\
&\dot{\boldsymbol{\lambda}}^T = -\tfrac{\partial H}{\partial \mathbf{x}}\\
&\tfrac{\partial H}{\partial \mathbf{u}} = 0\\
&\tfrac{\partial H}{\partial \boldsymbol{\mu}} \le 0\\
&\dot{\lambda}_s = -\tfrac{\partial H}{\partial s}\\
&\boldsymbol{\psi} \le 0\\
&\boldsymbol{\mu} \ge 0\\
&s_f \ge 0
\end{aligned} \tag{A.17}$$

with transversality conditions:

$$\begin{aligned}
&\tfrac{\partial H}{\partial t_f} + \boldsymbol{\eta}^T\boldsymbol{\Omega}\tfrac{\partial \mathbf{J}}{\partial t_f} + \boldsymbol{\nu}^T\tfrac{\boldsymbol{\psi}}{\partial t_f} = 0\\
&1 - \textstyle\sum_j^m \eta_j + \lambda_{s_f}(t_f) = 0\\
&\boldsymbol{\eta}^T\boldsymbol{\Omega}\nabla_{\mathbf{x}_f}\mathbf{J} + \boldsymbol{\nu}^T\nabla_{\mathbf{x}_f}\boldsymbol{\psi} - \boldsymbol{\lambda}(t_f)^T = 0\\
&\boldsymbol{\eta} \ge 0; \boldsymbol{\nu} \ge 0
\end{aligned} \tag{A.18}$$

Remark 1. At the solution $(\mathbf{x}^*, \mathbf{u}^*)$ all constraints are assumed to be active, which means that $\boldsymbol{\psi} = 0$ and $\mathbf{g}(\mathbf{x}^*, \mathbf{u}^*) = 0$. Furthermore, it is assumed that also the constraints on the objective functions are active, which means that at the solution $\boldsymbol{\Omega}(\mathbf{J}-\mathbf{z}) - s_f\mathbf{1} = 0$.

Remark 2. If initial conditions and time were not given but had to satisfy some constraint functions, one would need to add other three transversality conditions similar to (A.18) but at the initial time t_0.

A.2.1 Convergence of the Transcribed Problem

It is now possible to prove that the transcribed problem converges asymptotically to the necessary conditions for local optimality (4.4) and (4.5).

Theorem 6 *If* $\mathbf{c}$ *is Frechet differentiable at* $\mathbf{u}^*$ *and both* $\mathbf{x}$, $\mathbf{F}$ *and are* $\mathbf{g}$ *integrable functions, then the necessary conditions for local optimality of problem (4.8) converge asymptotically to (4.4) and (4.5) for* $k \to \infty$ *and* $s \to \infty$.

Proof 4 *We start from the augmented Lagrangian of the related NLP problem:*

$$L = s_f + \boldsymbol{\eta}^T(\boldsymbol{\Omega}(\mathbf{J} - \mathbf{z}) - s_f\mathbf{1}) + \hat{\boldsymbol{\lambda}}^T\mathbf{c}_d + \hat{\boldsymbol{\mu}}^T\mathbf{g} + \lambda_{s_f}s_f \tag{A.19}$$

where $\mathbf{c}_d$ *is the part of the constraint vector* $\mathbf{c} = [\mathbf{c}_d, \mathbf{g}]$ *that does not contain path constraints. If one differentiates the Lagrangian, the result is the necessary conditions for local optimality:*

$$\frac{\partial L}{\partial s_f} = 1 - \sum_{j=1}^{m}\eta_j + \lambda_{s_f} = 0 \tag{A.20}$$

$$\frac{\partial L}{\partial \mathbf{u}} = \hat{\boldsymbol{\lambda}}^T\nabla_{\mathbf{u}}\mathbf{c}_d + \hat{\boldsymbol{\mu}}^T\nabla_{\mathbf{u}}\mathbf{g} = 0 \tag{A.21}$$

$$\frac{\partial L}{\partial \mathbf{x}} = \hat{\boldsymbol{\lambda}}^T\nabla_{\mathbf{x}}\mathbf{c}_d + \hat{\boldsymbol{\mu}}^T\nabla_{\mathbf{x}}\mathbf{g} = 0 \tag{A.22}$$

$$\frac{\partial L}{\partial \mathbf{x}_f} = \boldsymbol{\eta}^T\boldsymbol{\Omega}\nabla_{\mathbf{x}_f}\mathbf{J} + \hat{\boldsymbol{\lambda}}_f^T\nabla_{\mathbf{x}_f}\mathbf{c}_d = 0 \tag{A.23}$$

$$\frac{\partial L}{\partial t_f} = \boldsymbol{\eta}^T\boldsymbol{\Omega}\frac{\partial \mathbf{J}}{\partial t_f} + \hat{\boldsymbol{\lambda}}_f^T\frac{\partial \mathbf{c}_d}{\partial t_f} + \hat{\boldsymbol{\mu}}_f^T\frac{\partial \mathbf{g}}{\partial t_f} = 0 \tag{A.24}$$

where the first equation corresponds to the transversality conditions on λ_{s_f}*, the fourth equation corresponds to the transversality conditions on* $\boldsymbol{\lambda}$ *that were derived in (4.5) and the firth equation is the transversality condition (A.16). We used the symbol* $\hat{\boldsymbol{\lambda}}_f^T$ *to indicate the Lagrange multipliers that correspond to the boundary constraints and to the dynamic constraints at the boundaries. In fact, one of the constraint equations* $\mathbf{c}$ *is* $\boldsymbol{\psi} \leq 0$*, which defines the boundary conditions on* $\mathbf{x}_0$ *and* $\mathbf{x}_f$*, and other two correspond to the first and last finite element (see Eq. (2.9)), which contain again the vectors* $\mathbf{x}_0$ *and* $\mathbf{x}_f$*. If we call* $\boldsymbol{\nu}$ *the* $\hat{\boldsymbol{\lambda}}_f$ *that corresponds to* $\boldsymbol{\psi}$ *and expand the last three equations, we get:*

$$\frac{\partial L}{\partial \mathbf{u}_{s,j}} = \sum_{k=1}^{l+1}\sigma_k\left[\hat{\boldsymbol{\lambda}}^T h_{s,j}\nabla_{\mathbf{u}_{s,j}}\mathbf{F} + \hat{\boldsymbol{\mu}}^T\nabla_{\mathbf{u}_{s,j}}\mathbf{g}\right] = 0 \tag{A.25}$$

$$\frac{\partial L}{\partial \mathbf{x}_{s,j}} = \sum_{k=1}^{l+1}\sigma_k\left[\dot{\hat{\boldsymbol{\lambda}}}^T h_{s,j}f_{s,j} + h_{s,j}\nabla_{\mathbf{x}_{s,j}}\mathbf{F} + \hat{\boldsymbol{\mu}}^T\nabla_{\mathbf{x}_{s,j}}\mathbf{g}\right] = 0 \tag{A.26}$$

$$\frac{\partial L}{\partial \mathbf{x}_f} = \boldsymbol{\eta}^T\boldsymbol{\Omega}\nabla_{\mathbf{x}_f}\mathbf{J} + \boldsymbol{\nu}^T\nabla_{\mathbf{x}_f}\boldsymbol{\psi} - \hat{\lambda}_f^T = 0 \tag{A.27}$$

The third equation is the transversality condition on the terminal states in (4.5). The second equation becomes:

$$\begin{aligned}\textstyle\sum_{k=1}^{l+1}\sigma_k\Big[\hat{\boldsymbol{\lambda}}^T\dot{h}_{s,j}f_{s,j} + h_{s,j}\nabla_{\mathbf{x}}\mathbf{F}\frac{\partial \mathbf{x}}{\partial \mathbf{x}_{s,j}} + \hat{\boldsymbol{\mu}}^T\nabla_{\mathbf{x}}\mathbf{g}\frac{\partial \mathbf{x}}{\partial \mathbf{x}_{s,j}}\Big] = \\ \textstyle\sum_{k=1}^{l+1}\sigma_k\Big[\hat{\boldsymbol{\lambda}}^T\dot{h}_{s,j}f_{s,j} + h_{s,j}\nabla_{\mathbf{x}}\mathbf{F}f_{s,j} + \hat{\boldsymbol{\mu}}^T\nabla_{\mathbf{x}}\mathbf{g}f_{s,j}\Big] = 0\end{aligned} \tag{A.28}$$

Here we made use of the fact that $\mathbf{g}(\mathbf{x}_{s,j},\mathbf{u}_{s,j},t_s) \leq 0 \Rightarrow \sum_{k=1}^{l+1}\sigma_k\mathbf{g}(\mathbf{x}_{s,j},\mathbf{u}_{s,j},t_s) \leq 0$. *If one now takes the limit for an infinite number of integration points the sums become continuous integrals:*

$$\int\Big[\sum_s \hat{\boldsymbol{\lambda}}_{s,j}h_{s,j}\nabla_{\mathbf{u}}\mathbf{F} + \hat{\boldsymbol{\mu}}^T\nabla_{\mathbf{u}}\mathbf{g}\Big]dt = 0 \tag{A.29}$$

$$\int\Big[\sum_s \hat{\boldsymbol{\lambda}}_{s,j}\dot{h}_{s,j} + \sum_s \hat{\boldsymbol{\lambda}}_{s,j}h_{s,j}\nabla_{\mathbf{x}}\mathbf{F} + \hat{\boldsymbol{\mu}}^T\nabla_{\mathbf{x}}\mathbf{g}\Big]dt = 0 \tag{A.30}$$

$$\tag{A.31}$$

Now if we make use of the fact that $\boldsymbol{\lambda}$ *is approximated by the polynomial* $\boldsymbol{\lambda} \simeq \sum \hat{\boldsymbol{\lambda}}_{s,j}h_{s,j}$, *for an infinite number of collocation points, which would correspond to an infinite number of integration points, we have:*

$$\int\Big[\boldsymbol{\lambda}^T\nabla_{\mathbf{u}}\mathbf{F} + \boldsymbol{\mu}^T\nabla_{\mathbf{u}}\mathbf{g}\Big]dt = 0 \tag{A.32}$$

$$\int\Big[\dot{\boldsymbol{\lambda}}^T + \boldsymbol{\lambda}^T\nabla_{\mathbf{x}}\mathbf{F} + \boldsymbol{\mu}^T\nabla_{\mathbf{x}}\mathbf{g}\Big]dt = 0 \tag{A.33}$$

$$\tag{A.34}$$

which are satisfied if the quantities in brackets are identically zero. These equations correspond to the optimality condition and to the differential equations on $\boldsymbol{\lambda}$ *in (4.4).*

Remark 5 *This suggests that the test functions* $\mathbf{w}(t)$ *employed in the DFET transcription can also be interpreted as a polynomial approximation of the Lagrange multipliers* $\boldsymbol{\lambda}(t)$. *Moreover, this means that from the Lagrange multipliers* $\hat{\boldsymbol{\lambda}}$ *of the NLP problem transcribed with the DFET method can be employed to recover an approximated time history for the Lagrange multipliers of the original non discretised problem. This is a*

DFET equivalent of the covector mapping principle [120].

Appendix B

Computational complexity of the Energy Based Archiving Algorithm

The EBA strategy requires two sets of operations corresponding to two steps: the fill-in of the archive and computation of the energy E, and its minimisation. The two steps never occur at the same time when the archive is updated.

The computation of each reciprocal of squared distance, in (3.13), for an m dimensional space, requires $3m$ operations: m differences of homologous coordinates, m squares of differences, $m-1$ sums of squares and 1 reciprocal of sums. If the reciprocal of pairwise squared distances of the r elements of the archive is stored in an r by r symmetric matrix M with zero diagonal elements, the number of reciprocal of squared distances to be computed is $\frac{r(r-1)}{2}$, for a total of $\frac{3mr(r-1)}{2}$ operations. With this matrix, the computation of the total energy E of the archive requires the sum of the elements of the upper (or lower) triangular part of the matrix, for a total of $\frac{r(r-1)}{2}$ sums.

Starting from the energy E and matrix M the already mentioned r components vector $\mathbf{E_2}$ is computed. $\mathbf{E_2}$ contains in its i-th entry the energy the archive would have if element i were excluded from the archive. The computation of the elements of this vector is conveniently performed by subtracting the sum of all elements of the i-th row of M from the energy E. Thus, a total of r^2 operations is required: 1 subtraction and $r-1$ sums for each component of $\mathbf{E_2}$. This completes the fill-in step of the archive, for a total of $\frac{3mr(r-1)}{2} + \frac{r(r-1)}{2} + r^2 \sim \mathcal{O}(mr^2)$ operations. Note that during a run of MACS the archive grows gradually, so the construction of the matrix M, energy E and vector $\mathbf{E_2}$ is performed incrementally, rather than all at once.

The energy minimisation step requires the computation of the reciprocal of square distances from each element of A to each candidate in C, for a total of rq combinations or $3mrq$ operations. At this point, the energy the archive would have if element i were substituted with the candidate j is computed: this is conveniently performed by summing $\mathbf{E_2}(i)$ to the reciprocal of the square distances from the j-th candidate to each other element in the archive, for a total of r^2 operations: r sums for each of element of the archive.

Now, the minimum energy over all possible combinations of candidates E_{new} is compared against the energy of the archive E. Suppose all the tentative energies are stored in an r by q matrix and its lower value entry E_{new} is located in the position (i^*, j^*). If E_{new} is lower than E, then a new minimum is found, and the candidate in position j^* substitutes the element in position i^*.

At this point, it is required to update all elements of the i-th row (or column) of the matrix M with the new reciprocal squared distances. This requires again $3m(r-1)$ operations, but can be avoided with proper bookkeeping (i.e if the matrix containing the reciprocal of square distances from each candidate to each element of the archive is stored). Finally, the update of the vector $\mathbf{E_2}$ is performed exactly as before. If the energy minimisation step is performed n_{it} times, the total number of operations is $3mrq + n_{it}\left(r^2 + r^2\right)$. Thus, the total cost of the EBA archiving algorithm is $\mathcal{O}\left(3mr^2/2\right)$ for the fill-in step, and $\mathcal{O}(n_{it}r^2)$ for the minimisation step (assuming $q \leq r$).

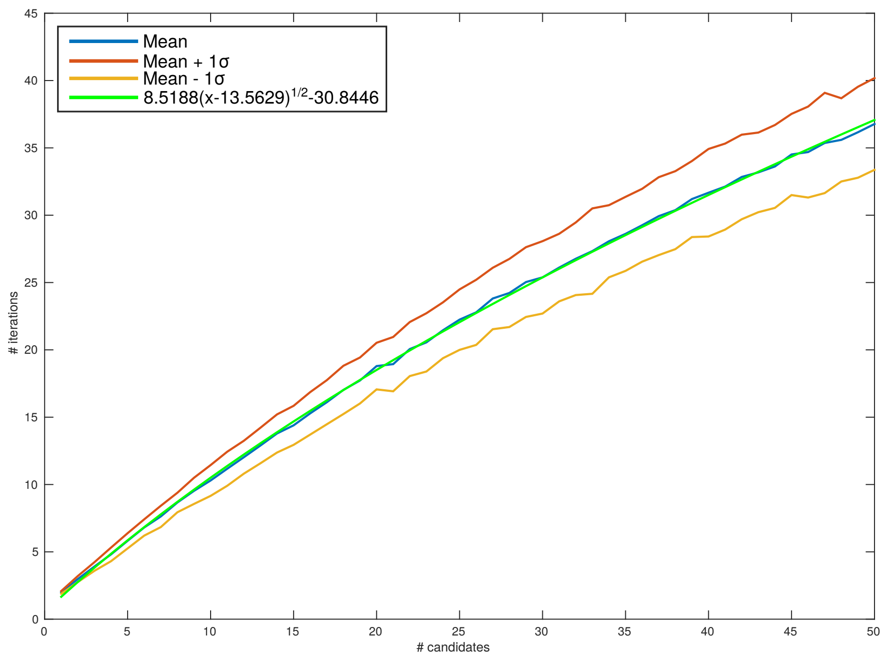


Figure B.1: Statistical evaluation of the number of iterations n_{it} as a function of the number of candidates q

In order to verify the computational complexity of the EBA strategy, a statistical analysis was performed on the dependence of n_{it} on q by sampling the Pareto front of the ZDT4 benchmark function with 100 points plus an increasing number of additional candidates. 200 independent runs were made for each number of additional candidates, and an $\mathcal{O}(q^{\frac{1}{2}})$ relation was discovered (see Figure B.1). Thus, the average cost of the minimisation step was found to be $\mathcal{O}(r^2 q^{\frac{1}{2}})$. It is also important to note that the computational complexity is linear in the number of dimensions of the Pareto front, making EBA a promising method for many-objective optimisation problems.

Appendix C

Aerodynamic model for Chapter 6

Table C.1 reports the coefficients used for the aerodynamic model for the Optimal Ascent and Abort Scenarios case of Chapter 6.

Table C.1: Coefficients for the aerodynamic model

Coeff.	Value	Coeff.	Value	Coeff.	Value
$a_{1,0}$	-6.378335936032101e-02	ς_1	1.344774373794846e+00	$a_{5,3}$	-1.683702441539749e-04
$a_{1,1}$	1.976923220591986e-02	ς_2	1.542614382500486e+00	$a_{6,0}$	8.869412111210242e+00
$a_{1,2}$	2.973963976446931e-04	ϱ_1	-3.100574615791786e+01	$a_{6,1}$	7.482402265848023e-01
$a_{2,0}$	-3.349264465313049e+05	ϱ_2	-9.123166455265215e-02	$a_{6,2}$	2.604898066115986e-02
$a_{2,1}$	6.001178833841350e+05	$a_{4,0}$	4.941304084167961e-02	$a_{6,3}$	2.689898581212407e-04
$a_{2,2}$	-2.862064269172794e+04	$a_{4,1}$	-2.771101541363621e-03	κ_3	1.348770573128532e+00
$a_{3,0}$	4.398289382706495e-01	$a_{4,2}$	4.195518696508440e-04	κ_4	1.376496163657956e+00
$a_{3,1}$	6.555024080116524e-02	$a_{4,3}$	6.144719928912810e-06	ς_3	1.004292397340135e+00
$a_{3,2}$	-4.816546419307238e-04	$a_{5,0}$	-8.408713897701869e+00	ς_4	1.053245533842992e+00
κ_1	1.088199979257513e+00	$a_{5,1}$	-6.942588805953275e-01	ϱ_3	-2.715562764314760e-01
κ_2	1.523372394108941e+00	$a_{5,2}$	-2.602381527566895e-02	ϱ_4	-3.464086848960685e-01

Bibliography

[1] J. F. McCloskey, "The beginnings of operations research: 1934-1941," *Operations research*, pp. 143–152, 1987.

[2] G. E. Moore *et al.*, "Cramming more components onto integrated circuits," 1965.

[3] H. D. Benington, "Production of large computer programs," *Annals of the History of Computing*, vol. 5, no. 4, pp. 350–361, 1983.

[4] W. W. Royce, "Managing the development of large software systems: concepts and techniques," in *Proceedings of the 9th international conference on Software Engineering.* IEEE Computer Society Press, 1987, pp. 328–338.

[5] K. J. Schlager, "Systems engineering-key to modern development," *IRE Transactions on Engineering Management*, vol. EM-3, no. 3, pp. 64–66, jul 1956.

[6] M. Avriel and R. S. Dembo, Eds., *Engineering Optimization.* Springer Berlin Heidelberg, 1979. [Online]. Available: https://doi.org/10.1007/bfb0120855

[7] A. E. Bryson, "Optimal control-1950 to 1985," *IEEE Control Systems Magazine*, vol. 16, no. 3, pp. 26–33, 1996.

[8] H. J. Pesch, M. Plail, and D. Munich, "The maximum principle of optimal control: a history of ingenious ideas and missed opportunities," *Control and Cybernetics*, vol. 38, no. 4A, pp. 973–995, 2009.

[9] H. J. Sussmann and J. C. Willems, "300 years of optimal control: from the brachystochrone to the maximum principle," *IEEE Control Systems Magazine*, vol. 17, no. 3, pp. 32–44, 1997.

[10] P. E. Gill, V. Kungurtsev, and D. P. Robinson, “A stabilized sqp method: superlinear convergence,” *Mathematical Programming*, vol. 163, no. 1-2, pp. 369–410, 2017.

[11] G. Hornby, A. Globus, D. Linden, and J. Lohn, “Automated antenna design with evolutionary algorithms,” in *Space 2006*, 2006, p. 7242.

[12] G. Rudolph, “Convergence rates of evolutionary algorithms for quadratic convex functions with rank-deficient hessian,” in *International Conference on Adaptive and Natural Computing Algorithms.* Springer, 2013, pp. 151–160.

[13] ——, “Convergence of evolutionary algorithms in general search spaces,” in *Proceedings of IEEE international conference on evolutionary computation.* IEEE, 1996, pp. 50–54.

[14] J. T. Betts and S. O. Erb, “Optimal low thrust trajectories to the moon,” *SIAM Journal on Applied Dynamical Systems*, vol. 2, no. 2, pp. 144–170, 2003.

[15] G. Eichfelder, *Adaptive Scalarization Methods in Multiobjective Optimization.* Springer-Verlag Berlin Heidelberg, 2008.

[16] E. Mezura-Montes and C. A. C. Coello, “Constrained optimization via multiobjective evolutionary algorithms,” in *Multiobjective problem solving from nature.* Springer, 2008, pp. 53–75.

[17] J. Lee and S. Leyffer, *Mixed integer nonlinear programming.* Springer Science & Business Media, 2011, vol. 154.

[18] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan, “Mixed-integer nonlinear optimization,” *Acta Numerica*, vol. 22, pp. 1–131, 2013.

[19] B. A. Earl, *Applied optimal control: optimization, estimation and control.* CRC Press, 1975.

[20] I. M. Ross and M. Karpenko, “A review of pseudospectral optimal control: From theory to flight,” *Annual Reviews in Control*, vol. 36, no. 2, pp. 182

– 197, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1367578812000375

[21] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 773–791, Oct 2016.

[22] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, Aug 2014.

[23] Y. Qi, X. Ma, F. Liu, L. Jiao, J. Sun, and J. Wu, "Moea/d with adaptive weight adjustment," *Evolutionary computation*, vol. 22, no. 2, pp. 231–264, 2014.

[24] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," *SIAM Journal on Optimization*, vol. 8, no. 3, pp. 631–657, 1998.

[25] A. Messac and C. A. Mattson, "Normal constraint method with guarantee of even representation of complete pareto frontier," *AIAA journal*, vol. 42, no. 10, pp. 2101–2111, 2004.

[26] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 2, pp. 182–197, 2002.

[27] C. A. C. Coello and M. S. Lechuga, "Mopso: a proposal for multiple objective particle swarm optimization," in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, vol. 2, May 2002, pp. 1051–1056 vol.2.

[28] C. Audet, G. Savard, and W. Zghal, "A mesh adaptive direct search algorithm for multiobjective optimization," *European Journal of Operational Research*, vol. 204, no. 3, pp. 545 – 556, 2010.

[29] N. Beume, B. Naujoks, and M. Emmerich, "Sms-emoa: Multiobjective selection based on dominated hypervolume," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1653 – 1669, 2007.

[30] J. Bader and E. Zitzler, "Hype: An algorithm for fast hypervolume-based many-objective optimization," *Evolutionary Computation*, vol. 19, no. 1, pp. 45–76, 2011, pMID: 20649424.

[31] A. Auger, J. Bader, D. Brockhoff, and E. Zitzler, "Theory of the hypervolume indicator: Optimal μ-distributions and the choice of the reference point," in *Proceedings of the Tenth ACM SIGEVO Workshop on Foundations of Genetic Algorithms*, ser. FOGA '09. New York, NY, USA: ACM, 2009, pp. 87–102.

[32] C. M. Fonseca, L. Paquete, and M. Lopez-Ibanez, "An improved dimension-sweep algorithm for the hypervolume indicator," in *2006 IEEE International Conference on Evolutionary Computation*, July 2006, pp. 1157–1163.

[33] M. Vasile and F. Zuiani, "Multi agent collaborative search: an agent-based memetic multi-objective optimization algorithm applied to space trajectory design," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 225, no. 11, pp. 1211–1227, 2011.

[34] F. Zuiani and M. Vasile, "Improved individualistic actions for multi agent collaborative search," *In EVOLVE*, July 2013.

[35] ——, "Multi agent collaborative search based on tchebycheff decomposition," *Computational Optimization and Applications*, vol. 56, no. 1, pp. 189–208, 2013. [Online]. Available: http://dx.doi.org/10.1007/s10589-013-9552-9

[36] ——, "Multi agent collaborative search with thecycheff decomposition and monotonic basin hopping," *In BIOMA*, May 2012.

[37] C. Y. Kaya and H. Maurer, "A numerical method for nonconvex multi-objective optimal control problems," *Computational Optimization and Applications*, vol. 57, no. 3, pp. 685–702, 2014.

[38] Q. J. Zhu, "Hamiltonian necessary conditions for a multiobjective optimal control problem with endpoint constraints," *SIAM Journal on Control and Optimization*, vol. 39, no. 1, p. 97–112, 2000.

[39] V. de Oliveira, G. Silva, and M. Rojas-Medar, "A class of multiobjective control problems," *Optimal Control Applications and Methods*, vol. 30, pp. 77–86, 2009.

[40] B. Kien and J. Y. N.C. Wong, "Necessary conditions for multiobjective optimal control problems with free end-time," *SIAM Journal on Control and Optimization*, vol. 47, no. 5, pp. 2251–2274, 2010.

[41] V. A. Oliveira and G. N. Silva, "On sufficient optimality conditions for multiobjective control problems," *Journal of Global Optimization*, vol. 64, no. 4, pp. 721–744, 2016.

[42] T.-N. Ngo and N. Hayek, "Necessary conditions of pareto optimality for multiobjective optimal control problems under constraints," *Optimization*, vol. 66, no. 2, pp. 149–177, 2017.

[43] S. Ober-Blöbaum, M. Ringkamp, and G. zum Felde, "Solving multiobjective optimal control problems in space mission design using discrete mechanics and reference point techniques," in *51st IEEE Annual Conference on Decision and Control*, Maui, Hawaii, 10-13 Dec 2012, pp. 5711–5716.

[44] A. Pascoletti and P. Serafini, "Scalarizing vector optimization problems," *Journal of Optimization Theory and Applications*, vol. 42, no. 4, pp. 499–524, 1984.

[45] F. Logist, B. Houska, M. Diehl, and J. F. Van Impe, "A toolkit for multi-objective optimal control in bioprocess engineering," *IFAC Proceedings Volumes*, vol. 43, no. 6, pp. 281–286, 2010.

[46] A. Pagano and E. Mooij, "Global launcher trajectory optimization for lunar base settlement," in *AIAA/AAS Astrodynamics Specialist Conference, Guidance, Navigation, and Control and Co-located Conferences*, Toronto, Canada, 2-5 Aug 2010.

[47] B. Bairstow, O. de Weck, and J. Sobieszczanski-Sobieski, “Multiobjective optimization of two-stage rockets for earth-to-orbit launch,” in *47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2006, p. 1720.

[48] J. Roshanian, A. A. Bataleblu, and M. Ebrahimi, “Robust ascent trajectory design and optimization of a typical launch vehicle,” *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, jan 2018.

[49] V. Coverstone-Carroll, J. W. Hartmann, and W. J. Mason, “Optimal multi-objective low-thrust spacecraft trajectories,” *Computer methods in applied mechanics and engineering*, vol. 186, no. 2, pp. 387–402, 2000.

[50] J. A. Englander, M. A. Vavrina, and A. R. Ghosh, “Multi-objective hybrid optimal control for multiple-flyby low-thrust mission design,” in *AAS/AIAA Space Flight Mechanics Meeting*, Williamsburg, Virginia, 11-15 Jan 2015.

[51] R. H. Leary, “Global optimization on funneling landscapes,” *Journal of Global Optimization*, vol. 18, no. 4, pp. 367–383, Dec 2000.

[52] A. H. Land and A. G. Doig, “An automatic method of solving discrete programming problems,” *Econometrica: Journal of the Econometric Society*, pp. 497–520, 1960.

[53] M. A. Duran and I. E. Grossmann, “An outer-approximation algorithm for a class of mixed-integer nonlinear programs,” *Mathematical programming*, vol. 36, no. 3, pp. 307–339, 1986.

[54] R. Fletcher and S. Leyffer, “Solving mixed integer nonlinear programs by outer approximation,” *Mathematical programming*, vol. 66, no. 1-3, pp. 327–349, 1994.

[55] A. M. Geoffrion, “Generalized benders decomposition,” *Journal of optimization theory and applications*, vol. 10, no. 4, pp. 237–260, 1972.

[56] C. Audet and J. E. Dennis Jr, "Mesh adaptive direct search algorithms for constrained optimization," *SIAM Journal on optimization*, vol. 17, no. 1, pp. 188–217, 2006.

[57] M. A. Abramson, C. Audet, J. W. Chrissis, and J. G. Walston, "Mesh adaptive direct search algorithms for mixed variable optimization," *Optimization Letters*, vol. 3, no. 1, p. 35, 2009.

[58] M. Glocker and O. von Stryk, "Hybrid optimal control of motorized traveling salesmen and beyond," in *Proc. 15th IFAC World Congress on Automatic Control*, 2002, pp. 21–26.

[59] S. Sager, *Numerical methods for mixed-integer optimal control problems.* Der Andere Verlag Tönning, 2005.

[60] M. Schlueter, "Nonlinear mixed integer based optimization technique for space applications," Ph.D. dissertation, University of Birmingham, 2012.

[61] L. A. Ricciardi and M. Vasile, "Improved archiving and search strategies for multi agent collaborative search," in *Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences.* Springer, 2019, pp. 435–455.

[62] ——, "Direct transcription of optimal control problems with finite elements on bernstein basis," *Journal of Guidance, Control, and Dynamics*, vol. 42, no. 2, pp. 229–243, Feb. 2019. [Online]. Available: https://doi.org/10.2514/1.g003753

[63] L. A. Ricciardi, C. A. Maddock, and M. Vasile, "Direct solution of multi-objective optimal control problems applied to spaceplane mission design," *Journal of Guidance, Control and Dynamics*, August 2018.

[64] L. A. Ricciardi, M. Vasile, and C. Maddock, "Global solution of multi-objective optimal control problems with multi agent collaborative search and direct finite elements transcription," in *2016 IEEE Congress on*

Evolutionary Computation (CEC). IEEE, jul 2016. [Online]. Available: https://doi.org/10.1109/cec.2016.7743882

[65] L. A. Ricciardi, M. Vasile, F. Toso, and C. A. Maddock, "Multi-objective optimal control of the ascent trajectories of launch vehicles," in *AIAA/AAS Astrodynamics Specialist Conference.* American Institute of Aeronautics and Astronautics, sep 2016. [Online]. Available: https://doi.org/10.2514/6.2016-5669

[66] M. Vasile and L. Ricciardi, "A direct memetic approach to the solution of multi-objective optimal control problems," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI).* IEEE, dec 2016. [Online]. Available: https://doi.org/10.1109/ssci.2016.7850103

[67] L. A. Ricciardi and M. Vasile, "A relaxation approach for hybrid multi-objective optimal control: Application to multiple debris removal missions (aas 19-406)," in *29th AAS/AIAA Space Flight Mechanics Meeting.* Ka'anapali, Hawaii, United States: American Institute of Aeronautics and Astronautics, 11-17 Jan 2019.

[68] L. Ricciardi, C. Maddock, and M. L. Vasile, "Multi-objective optimal control of re-entry and abort scenarios," in *2018 Space Flight Mechanics Meeting.* American Institute of Aeronautics and Astronautics, jan 2018. [Online]. Available: https://doi.org/10.2514/6.2018-0218

[69] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming.* Society for Industrial and Applied Mathematics, jan 2010. [Online]. Available: https://doi.org/10.1137/1.9780898718577

[70] M. Vasile, "Finite elements in time: A direct transcription method for optimal control problems," in *AIAA/AAS Astrodynamics Specialist Conference.* American Institute of Aeronautics and Astronautics, aug 2010. [Online]. Available: https://doi.org/10.2514/6.2010-8275

[71] M. Vasile and A. E. Finzi, "Direct lunar descent optimisation by finite elements in time approach," *International Journal of Mechanics and Control*, vol. 1, no. 1, 2000.

[72] D. H. Hodges and R. R. Bless, "Weak hamiltonian finite element method for optimal control problems," *Journal of Guidance, Control, and Dynamics*, vol. 14, no. 1, pp. 148–156, jan 1991. [Online]. Available: https://doi.org/10.2514/3.20616

[73] M. Borri and C. Bottasso, "A general framework for interpreting time finite element formulations," *Computational Mechanics*, vol. 13, no. 3, pp. 133–142, Dec 1993.

[74] C. L. Bottasso and A. Ragazzi, "Finite element and runge-kutta methods for boundary-value and optimal control problems," *Journal of Guidance, Control, and Dynamics*, vol. 23, no. 4, pp. 749–751, jul 2000. [Online]. Available: https://doi.org/10.2514/2.4595

[75] M. Vasile and F. Bernelli-Zazzera, "Optimizing low-thrust and gravity assist maneuvers to design interplanetary trajectories," *Journal of the Astronautical Sciences*, vol. 51, no. No. 1, 2003.

[76] ——, "Targeting a heliocentric orbit combining low-thrust propulsion and gravity assist manoeuvres," *Operational Research in Space & Air*, vol. 79, 2003.

[77] C. L. Bottasso, "A new look at finite elements in time: a variational interpretation of runge-kutta methods," *Applied Numerical Mathematics*, vol. 25, no. 4, pp. 355–368, dec 1997. [Online]. Available: https://doi.org/10.1016/s0168-9274(97)00072-x

[78] G. Henryk and P. J. Luis, "On the approximation properties of bernstein polynomials via probabilistic tools," *Boletın de la Asociación Matemática Venezolana*, vol. 10, no. 1, pp. 5–13, 2003.

[79] T. Rogalsky, "Bézier parameterization for optimal control by differential evolution," in *Proceedings of the International Conference on Genetic and Evolutionary Methods (GEM).* Citeseer, 2011, p. 1.

[80] F. Ghomanjani, M. Farahi, and M. Gachpazan, “Bézier control points method to solve constrained quadratic optimal control of time varying linear systems,” *Computational & Applied Mathematics*, vol. 31, no. 3, pp. 433–456, 2012. [Online]. Available: https://doi.org/10.1590/s1807-03022012000300001

[81] F. Ghomanjani and M. HadiFarahi, “Optimal control of switched systems based on bezier control points,” *International Journal of Intelligent Systems and Applications*, vol. 4, no. 7, pp. 16–22, jun 2012. [Online]. Available: https://doi.org/10.5815/ijisa.2012.07.02

[82] M. Darehmiraki, M. H. Farahi, and S. Effati, “A novel method to solve a class of distributed optimal control problems using bezier curves,” *Journal of Computational and Nonlinear Dynamics*, vol. 11, no. 6, p. 061008, jul 2016. [Online]. Available: https://doi.org/10.1115/1.4033755

[83] N. Mirkov and B. Rasuo, “A bernstein polynomial collocation method for the solution of elliptic boundary value problems,” *arXiv preprint arXiv:1211.3567*, 2012.

[84] M. I. Bhatti and P. Bracken, “Solutions of differential equations in a bernstein polynomial basis,” *Journal of Computational and Applied Mathematics*, vol. 205, no. 1, pp. 272–280, aug 2007. [Online]. Available: https://doi.org/10.1016/j.cam.2006.05.002

[85] R. Farouki and V. Rajan, “On the numerical condition of polynomials in bernstein form,” *Computer Aided Geometric Design*, vol. 4, no. 3, pp. 191–216, nov 1987. [Online]. Available: https://doi.org/10.1016/0167-8396(87)90012-4

[86] R. Ait-Haddou, T. Nomura, and L. Biard, “A refinement of the variation diminishing property of bézier curves,” *Computer Aided Geometric Design*, vol. 27, no. 2, pp. 202–211, feb 2010. [Online]. Available: https://doi.org/10.1016/j.cagd.2009.12.001

[87] G. M. Phillips, *Interpolation and approximation by polynomials.* New York: Springer, 2003.

[88] A. Pallini, "Bernstein-type approximations of smooth functions," *Statistica; Vol 65, No 2 (2005); 169-191*, 2007.

[89] W. V. Loock, G. Pipeleers, and J. Swevers, "Optimal motion planning for differentially flat systems with guaranteed constraint satisfaction," in *2015 American Control Conference (ACC)*. IEEE, jul 2015. [Online]. Available: https://doi.org/10.1109/acc.2015.7171996

[90] V. Cichella, I. Kaminer, C. Walton, and N. Hovakimyan, "Optimal motion planning for differentially flat systems using bernstein approximation," *IEEE Control Systems Letters*, vol. 2, no. 1, pp. 181–186, jan 2018. [Online]. Available: https://doi.org/10.1109/lcsys.2017.2778313

[91] B. Açıkmeşe and L. Blackmore, "Lossless convexification of a class of optimal control problems with non-convex control constraints," *Automatica*, vol. 47, no. 2, pp. 341–347, feb 2011. [Online]. Available: https://doi.org/10.1016/j.automatica.2010.10.037

[92] L. Blackmore, B. Açıkmeşe, and J. M. Carson, "Lossless convexification of control constraints for a class of nonlinear optimal control problems," *Systems & Control Letters*, vol. 61, no. 8, pp. 863–870, aug 2012. [Online]. Available: https://doi.org/10.1016/j.sysconle.2012.04.010

[93] Y. Mao, D. Dueri, M. Szmuk, and B. Açıkmeşe, "Successive convexification of non-convex optimal control problems with state constraints," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 4063–4069, jul 2017. [Online]. Available: https://doi.org/10.1016/j.ifacol.2017.08.789

[94] M. R. documentation. Choosing the algorithm. [Online]. Available: https://uk.mathworks.com/help/optim/ug/choosing-the-algorithm.html#brppuoz

[95] C. Büskens and D. Wassel, "The esa nlp solver worhp," in *Modeling and Optimization in Space Engineering*, G. Fasano and J. D. Pintér, Eds. Springer New York, 2013, vol. 73, pp. 85–110.

[96] M. R. documentation. Worhp website. [Online]. Available: https://worhp.de/

[97] M. Vasile and M. Locatelli, "A hybrid multiagent approach for global trajectory optimization," *Journal of Global Optimization*, vol. 44, no. 4, pp. 461–479, Aug 2009.

[98] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary computation*, vol. 8, no. 2, pp. 173–195, 2000.

[99] O. Schutze, X. Esquivel, A. Lara, and C. A. C. Coello, "Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 4, pp. 504–522, Aug 2012.

[100] S. D. Chasalow and R. J. Brand, "Algorithm as 299: Generation of simplex lattice points," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 44, no. 4, pp. 534–545, 1995.

[101] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari, "Multi-objective optimization test instances for the cec 2009 special session and competition," *University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report*, vol. 264, 2008.

[102] Q. Zhang, W. Liu, and H. Li, "The performance of a new version of moea/d on cec09 unconstrained mop test instances." in *IEEE Congress on Evolutionary Computation*, vol. 1, 2009, pp. 203–208.

[103] O. Schütze, X. Esquivel, A. Lara, and C. A. C. Coello, "Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 16, no. 4, pp. 504–522, 2012.

[104] M. Vasile, "Multi-objective optimal control: a direct approach," in *Satellite Dynamics and Space Missions.* Springer INDaM, 2019.

[105] C. Hillermeier, *Nonlinear Multiobjective Optimization.* International Series of Numerical Mathematics, Birkhäuser Basel, 2001.

[106] A. L. Herman and B. A. Conway, "Direct optimization using collocation based on high-order gauss-lobatto quadrature rules," *Journal of Guidance, Control, and Dynamics*, vol. 19, no. 3, pp. 592–599, 1996.

[107] S. Leyffer, "Deterministic methods for mixed integer nonlinear programming," Ph.D. dissertation, Citeseer, 1993.

[108] R. C. Jeroslow, "There cannot be any algorithm for integer programming with quadratic constraints," *Operations Research*, vol. 21, no. 1, pp. 221–224, feb 1973. [Online]. Available: https://doi.org/10.1287/opre.21.1.221

[109] O. Von Stryk and M. Glocker, "Numerical mixed-integer optimal control and motorized traveling salesmen problems. commande numerique optimale a entiers mixtes et problemes de voyageur de commerce motorise," *Journal Européen des Systèmes Automatisés*, vol. 35, no. 4, pp. 519–534, 2001.

[110] C. Maddock, F. Toso, L. Ricciardi, A. Mogavero, K. H. Lo, S. Rengarajan, K. Kontis, A. Milne, D. Evans, M. West *et al.*, "Vehicle and mission design of a future small payload launcher," in *21st AIAA International Space Planes and Hypersonics Technologies Conference*, Xiamen, China, 6-9 Mar 2017.

[111] F. Toso and C. Maddock, "Return and abort trajectory optimisation for reusable launch vehicles," in *21st AIAA/IAA International Space Planes and Hypersonic Systems and Technologies Conferences*, Xiamen, China, 6-9 Mar 2017.

[112] B. N. Pamadi and G. J. Brauckmann, "Aerodynamic characteristics and development of the aerodynamic database of the x-34 reusable launch vehicle," in *International Symposium on Atmospheric Reentry Vehicles and Systems*, Arcachon, France, 16-18 Mar 1999.

[113] G. J. Brauckmann, "X-34 vehicle aerodynamic characteristics," *Journal of Spacecraft and Rockets*, vol. 36, no. 2, pp. 229–239, 1999.

[114] R. Rohrschneider, "Development of a mass estimating relationship database for launch vehicle conceptual design," School of Aerospace Engineering, Georgia Institute of Technology, Tech. Rep., 2002.

[115] D. J. Kessler and B. G. Cour-Palais, "Collision frequency of artificial satellites: The creation of a debris belt," *Journal of Geophysical Research: Space Physics*, vol. 83, no. A6, pp. 2637–2646, 1978.

[116] P. Tadini, U. Tancredi, M. Grassi, L. Anselmo, C. Pardini, A. Francesconi, F. Branz, F. Maggi, M. Lavagna, L. DeLuca *et al.*, "Active debris multi-removal mission concept based on hybrid propulsion," *Acta Astronautica*, vol. 103, pp. 26–35, 2014.

[117] Y. Jing, X.-q. Chen, and L.-h. Chen, "Biobjective planning of geo debris removal mission with multiple servicing spacecrafts," *Acta Astronautica*, vol. 105, no. 1, pp. 311–320, 2014.

[118] H. Schaub and J. L. Junkins, *Analytical Mechanics of Space Systems, Fourth Edition.* American Institute of Aeronautics and Astronautics, Inc., jan 2018. [Online]. Available: https://doi.org/10.2514/4.105210

[119] D. Izzo, "Problem description for the 9th global trajectory optimisation competition," *GTOC Portal*, 2017.

[120] I. M. Ross and F. Fahroo, "A pseudospectral transformation of the convectors of optimal control systems," *IFAC Proceedings Volumes*, vol. 34, no. 13, pp. 543–548, aug 2001. [Online]. Available: https://doi.org/10.1016/s1474-6670(17)39048-1

[121] D. Preller and M. K. Smart, "Reusable launch of small satellites using scramjets," *Journal of Spacecraft and Rockets*, pp. 1–13, 2017.

[122] T. Tsuchiya and T. Mori, “Optimal design of two-stage-to-orbit space planes with airbreathing engines,” *Journal of Spacecraft and Rockets*, vol. 42, no. 1, pp. 90–97, 2005.

[123] S. Yang, T. Cui, X. Hao, and D. Yu, “Trajectory optimization for a ramjet-powered vehicle in ascent phase via the gauss pseudospectral method,” *Aerospace Science and Technology*, vol. 67, pp. 88–95, 2017.

[124] D. J. Bayley, R. J. Hartfield, J. E. Burkhalter, and R. M. Jenkins, “Design optimization of a space launch vehicle using a genetic algorithm,” *Journal of Spacecraft and Rockets*, vol. 45, no. 4, pp. 733–740, 2008.

[125] P. Lu and B. Pan, “Highly constrained optimal launch ascent guidance,” *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 2, pp. 404–414, 2010.

[126] P. Lu, H. Sun, and B. Tsai, “Closed-loop endoatmospheric ascent guidance,” *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 2, pp. 283–294, 2003.

[127] G. Dukeman, “Atmospheric ascent guidance for rocket-powered launch vehicles,” in *AIAA Guidance, Navigation, and Control Conference and Exhibit, Guidance, Navigation, and Control and Co-located Conferences*, Monterey, USA, 5-8 August 2002.

[128] G. Dukeman and A. Hill, “Rapid trajectory optimization for the ares i launch vehicle,” in *AIAA Guidance, Navigation, and Control Conference and Exhibit, Guidance, Navigation, and Control and Co-located Conferences*, Honolulu, USA, 18-21 August 2008.

[129] P. F. Gath and A. J. Calise, “Optimization of launch vehicle ascent trajectories with path constraints and coast arcs,” *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 2, pp. 296–304, 2001.

[130] A. J. Calise and N. Brandt, “Generation of launch vehicle abort trajectories using a hybrid optimization method,” *Journal of Guidance, Control, and Dynamics*, vol. 27, no. 6, pp. 930–937, 2004.

[131] K. Graichen and N. Petit, "Solving the goddard problem with thrust and dynamic pressure constraints using saturation functions," in *17th International Federation of Automatic Control (IFAC) World Congress*, July 2008.

[132] F. Zuiani, Y. Kawakatsu, and M. Vasile, "Multi-objective optimisation of many-revolution, low-thrust orbit raising for destiny mission," in *23rd AAS/AIAA Space Flight Mechanics Conference*, Kauai, Hawaii, 10-14 Feb 2013.

[133] I. M. Ross and M. Karpenko, "A review of pseudospectral optimal control: From theory to flight," *Annual Reviews in Control*, vol. 36, no. 2, pp. 182–197, 2012.

[134] O. V. Stryk, "Numerical solution of optimal control problems by direct collocation," in *Optimal Control Calculus of Variation, Optimal Control Theory and Numerical Methods.* Birkhauser Verlag, 1993.

[135] M. Vasile and F. Bernelli-Zazzera, "Optimizing low-thrust and gravity assist maneuvres to design interplanetary trajectories," *The Journal of the Astronautical Sciences*, vol. 51, no. 1, 2003, january-March 2003.

[136] M. Vasile, "Robust optimization of trajectory intercepting dangerous neo," in *Proceedings of the AAS/AIAA Aerodynamics Specialist Conference*, Monterey, California, Aug 2002.

[137] D. A. Garg, M. Patterson, W. Hagger, A. V. Rao, D. A. Benson, and G. T. Huntington, "A unified framework for the numerical solution of optimal control problem using pseudospectral methods," *Automatica*, 2010.

[138] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari, "Multi-objective optimization test instances for the cec 2009 special session and competition," *University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report*, pp. 1–30, 2008.

[139] F. Zuiani and M. Vasile, "Preliminary design of debris removal missions by means

of simplified models for low-thrust, many-revolution transfers," *International Journal of Aerospace Engineering*, vol. 2012, 2012.

[140] F. Zuiani, Y. Kawakatsu, and M. Vasile, "Multi-objective optimisation of many-revolution, low-thrust orbit raising for destiny mission," in *23rd AAS/AIAA Space Flight Mechanics Conference*, 2013.

[141] P. Fortescue, *Spacecraft systems engineering.* Hoboken, N.J: Wiley, 2011.

[142] D. Vallado, *Fundamentals of astrodynamics and applications.* Hawthorne, CA: Microcosm Press, 2013.

[143] ArianeSpace, "Vega user's manual," http://www.arianespace.com/wp-content/uploads/2015/09/Vega-Users-Manual_Issue-04_April-2014.pdf, accessed: May 2016.

[144] ——, "Ariane 5 user's manual," http://www.arianespace.com/wp-content/uploads/2015/09/Ariane5$_$users$_$manual$_$Issue5$_$July2011.pdf, accessed: May 2016.

[145] R. Battin, *An introduction to the mathematics and methods of astrodynamics.* Reston, VA: American Institute of Aeronautics and Astronautics, 1999.

[146] J. Niebling and G. Eichfelder, "A branch-and-bound based algorithm for non-convex multiobjective optimization," *Preprint-Series of the Institute for Mathematics*, 2018.

[147] S. Shapiro, "Lagrange and mayer problems in optimal control," *Automatica*, vol. 3, no. 3-4, pp. 219–230, jan 1966. [Online]. Available: https://doi.org/10.1016/0005-1098(66)90014-8

[148] E. S. Agency. Space debris by the numbers. [Online]. Available: https://m.esa.int/Our_Activities/Space_Safety/Space_Debris/Space_debris_by_the_numbers