



HAL
open science

Hierarchical Emergency Guidance Optimization for reusable Tossback vehicle landing

Hubert Ménou

► **To cite this version:**

Hubert Ménou. Hierarchical Emergency Guidance Optimization for reusable Tossback vehicle landing. Automatic Control Engineering. Université Paris sciences et lettres, 2023. English. NNT : 2023UPSLM004 . tel-04097551

HAL Id: tel-04097551

<https://pastel.hal.science/tel-04097551>

Submitted on 15 May 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à Mines Paris

**Hierarchical Emergency Guidance Optimization
for Reusable Tossback Vehicle Landing**

**Optimisation Hiérarchique pour le Guidage de Secours
pour un Véhicule Réutilisable "Tossback"**

Soutenu par

Hubert Ménou

Le 31 janvier 2023

École doctorale n°621

**Ingénierie des Systèmes,
Matériaux, Mécanique,
Energétique**

Spécialité

**Mathématiques et Automa-
tique**

Préparée au

Centre Automatique et Systèmes

Composition du jury :

Frédéric JEAN ENSTA	<i>Président du jury</i>
Hélène PIET-LAHANIER ONERA	<i>Rapporteur</i>
Emmanuel TRÉLAT Sorbonne Université	<i>Rapporteur</i>
Anouck GIRARD University of Michigan	<i>Examineur</i>
Gérald PIGNIÉ ArianeGroup	<i>Examineur</i>
Laurent PFEIFFER INRIA	<i>Examineur</i>
Eric BOURGEOIS CNES	<i>Examineur</i>
Nicolas PETIT Mines Paris	<i>Directeur de thèse</i>

“In the land of the blind, the autopilot is king.”

K. S. ROBINSON, Red Mars

Remerciements

“*Un stage ? Mais vous ne voulez pas un sujet de thèse plutôt ? J’ai un sujet sur l’atterrissage de fusées réutilisables qui pourrait vous plaire.*” C’est sur cette phrase de Nicolas Petit qu’est née l’idée de cette thèse, en 2016, lors d’une discussion à propos d’un sujet qui n’avait pas grand chose à voir. A l’époque, j’étais un jeune étudiant travaillant sur l’atténuation des vibrations pour les machines à laver, sous la supervision de Florent Di Meglio.

Je tiens à remercier Nicolas de m’avoir proposé ce sujet et encadré depuis tout ce temps. Merci d’avoir toujours répondu présent, même pendant cette longue période de crise sanitaire. Ce n’est pas seulement avec un chercheur que j’ai pu travailler, mais avec un passionné d’aérospatial. Je tiens aussi à remercier Eric Bourgeois de m’avoir encadré avec Nicolas.

Je tiens à remercier toute l’équipe du Centre Automatique et Systèmes pour ces nombreuses années de collaboration. Tous les permanents m’ont aidé à un moment ou à un autre de ma thèse. Merci à Delphine pour m’avoir fait confiance pour les cours d’optimisation, me laissant une grande liberté dans le choix de mes sujets, et d’avoir été toujours disponible quelle que soit la question. Merci à Florent, pour son soutien sans faille depuis toujours. Merci à Philippe, pour toutes ces discussions autour de la machine à café, et les nombreux coups de main techniques. Merci à Laurent, pour ses relectures précises, et ses conseils techniques avancés.

Merci infiniment à Dilshad, d’avoir été un co-doctorant et ami pendant 3 ans, de m’avoir initié au puit sans fond qu’est `tikz`, de m’avoir fait découvrir des sujets techniques comme littéraires, et d’avoir refait le monde autour d’un nombre peu raisonnable de cafés. Merci à tous les autres doctorants et anciens doctorants du laboratoire, Aurélien, Sijia, Maxime, Loris, Nils, Aradana, et tant d’autres.

Je tiens à remercier mes parents. Merci pour votre soutien depuis toujours. Avec les confinements à répétition, j’ai tout simplement passé plus de temps à la maison qu’au CNES. Merci à mes sœurs et à mon frère de m’avoir supporté pendant toutes ces années à parler de mathématiques, d’avion et de fusées à la maison.

Merci à tous mes amis, qu’ils soient là depuis 3 ans ou depuis 14 ans : Rémy, Xavier, Paul, Julie, Grégoire, Juliette et toute la clique de *Fondue*.

Et surtout, je tiens à remercier Mona, ma conjointe. Merci pour ta patience, toutes tes relectures, et pour toutes nos sessions de brainstorming confinées. Merci !

Abstract

This thesis studies emergency Powered Descent Guidance (PDG) for reusable launchers. PDG is a trajectory planning problem starting at some initial condition and ending at a given landing site. It is formulated as an Optimal Control Problem in free final-time with multiple constraints conveying operational goals, along the path or at the end-point. The launcher is under strong aerodynamic effects and has limited maneuverability. For off-nominal flight conditions, we wish to perform “emergency” trajectory planning by relaxing some negotiable parameters, such as the incidence safety bound, the normal acceleration load, or the landing site location. The relative importance of these parameters must however be taken into account. We differ from the classic penalty-based approaches whose heuristic tuning is tedious. A hierarchy between the parameters is introduced and a numerical method is developed to enforce it.

The contribution of the thesis is a methodology and an algorithm handling these parameters according to a user-specified hierarchy. The algorithm, denoted Hierarchical Emergency Guidance Optimization (H.E.G.O.), consists of a finite sequence of Linear Programs (LPs), called the Negotiation Problems, used to compute the necessary relaxations, followed by a Quadratic Program (QP), called the Refine Problem, which adjusts the landing trajectory optimally. The rocket is modeled by eight states, and three controls. The flight parameters are the initial conditions of the rocket states and several dimensioning parameters, such as the Engine Specific Impulse (ISP) and the wind profile. After discretization of the PDG problem, a sensitivity analysis of a parametric Non-Linear Program yields a QP, whose right-hand side depends linearly on both the flight and the negotiable parameters, and whose solution describes the optimal trajectory. The user-defined hierarchy is conveyed via a co-lexicographic order. The smallest negotiable parameter in the sense of this order is obtained by solving the sequence of LPs. The methodology is theoretically studied. Among others, the Lipschitz-continuity of the guidance trajectory with respect to the input flight parameters is established. Extensive numerical results serve to quantify the performance and relevance of the methodology.

Résumé

Cette thèse étudie le problème de Guidage d'urgence en Descente Propulsée (GDP) d'un lanceur réutilisable. Le GDP est un problème de planification de trajectoire depuis une condition initiale vers un site d'atterrissage. Il est formulé comme un problème de commande optimale en temps final libre sous plusieurs contraintes traduisant des souhaits opérationnels, le long du chemin ou au point final. Le lanceur est soumis à de forts effets aérodynamiques et dispose d'une manœuvrabilité limitée. Dans le cas de conditions de vol anormales, on souhaite calculer en temps réel une trajectoire de 'secours', en relâchant certains paramètres négociables, tels que la limite d'incidence, la limite d'accélération normale, ou encore le lieu d'atterrissage. L'importance relative de ces paramètres doit néanmoins être prise en compte. Nous nous distinguons des approches classiques de pénalisation dont le réglage heuristique est délicat. Nous introduisons une hiérarchie entre les paramètres et développons une méthode numérique pour la respecter strictement.

La contribution principale de la thèse est une méthode algorithmique déterminant les valeurs optimales de ces paramètres d'après une hiérarchie pré-définie par l'utilisateur. L'algorithme, Optimisation Hiérarchique pour le Guidage d'Urgence (H.E.G.O.), consiste en une suite finie de Problèmes Linéaires (LPs) de Négociation, utilisée pour calculer les relaxations nécessaires, suivie d'un Problème Quadratique (QP) de Raffinement, qui optimise la trajectoire. Le lanceur est modélisé par huit états et par trois commandes. Les paramètres de vol sont les conditions initiales de la fusée, et plusieurs paramètres caractéristiques, tels que l'Impulsion Spécifique du moteur (ISP) et le profil de vent. Après une étape de discrétisation, une analyse de sensibilité d'un Problème Non-Linéaire paramétrique amène à formuler un QP, dont le second membre dépend linéairement des paramètres de vol et des paramètres négociables, et dont la solution décrit la trajectoire optimale. La hiérarchie définie par l'utilisateur est exprimée via un ordre co-lexicographique. Résoudre la suite de LPs permet de calculer le plus petit paramètre négociable au sens de cet ordre. Cette méthode est analysée d'un point de vue théorique. Entre autres, la Lipschitz-continuité de la trajectoire re-planifiée vue comme une fonction des paramètres de vol est établie. Des résultats numériques permettent de quantifier la performance et la qualité de la méthode.

Contents

Remerciements	iv
Abstract	v
Résumé	vii
Nomenclature	xv
1 Introduction	1
1.1 Powered Descent Guidance (PDG) for reusable launchers	1
1.2 Mathematical programming for online PDG	5
1.2.1 Current state-of-the-art	5
1.2.2 Hierarchy and the emergency problem	8
1.3 Proposed contribution: Hierarchical Emergency Guidance Optimization	9
1.3.1 Fast PDG for fixed value of constraint parameters	9
1.3.2 Computing the best constraint alteration achieving feasibility	11
1.3.3 The HEGO algorithm	11
1.4 Manuscript outline	12
2 Dynamic models and the PDG problem	15
2.1 Atmospheric flight dynamics	16
2.1.1 Earth and atmosphere model	16
2.1.2 Aerodynamic model	16
2.1.3 Noteworthy particularities	17
2.2 Planar rocket model	20
2.3 Three-dimensional rocket model	22
2.3.1 Orientation frames	23
2.3.2 Dynamics	25
2.3.3 Normal acceleration, downrange and attitude	29
2.4 PDG as an Optimal Control Problem	30
2.4.1 Mission goals and constraints	30
2.4.2 Formulation as an optimal correction problem	32
3 Mathematical properties of the optimal vertical descent	35
3.1 Vertical descent	36
3.1.1 Single dimensional rocket model	37
3.1.2 Assumptions specific to the vertical descent	38

3.1.3	Optimal Control Problems	38
3.2	Preliminaries on the dynamics	39
3.3	Optimal thrust programs	42
3.3.1	Fuel Optimal Landing	42
3.3.2	Optimality of Min-Max Programs	47
3.3.3	Main result	49
3.4	Numerical illustrations	50
3.5	Comments	51
4	Nominal guidance via Quadratic Programming	53
4.1	Non-Linear Programming (NLP) formulation for PDG	54
4.1.1	Discretization of the decision variable	55
4.1.2	Formulation of the finite dimensional guidance problem	58
4.2	Sensitivity analysis for degenerate parametric NLP	60
4.2.1	An introductory toy example	61
4.2.2	Known results in parametric NLP sensitivity	62
4.3	Fast nominal guidance method	68
4.3.1	An offline/online approach for nominal guidance	68
4.3.2	Guidance law	69
4.3.3	Directional first-order estimate of waypoints	69
4.4	Numerical examples	70
4.4.1	Effectiveness of calculated guidance	71
4.4.2	Changes in the active set	71
4.4.3	Non-local constraint satisfaction	71
5	Emergency guidance via Linear and Quadratic Programming	77
5.1	Negotiable parameter choices	79
5.1.1	Negotiated constraints	79
5.1.2	On the relative importance of the parameters	80
5.2	A hierarchical negotiation	81
5.2.1	Algorithmic principle of HEGO	82
5.2.2	An illustrative toy example	85
5.2.3	Noteworthy remarks	87
5.3	Smoothness of the HEGO algorithm	90
5.3.1	Problem re-writing	91
5.3.2	Uniqueness of the optimal trajectory	93
5.3.3	Regularity w.r.t. the right-hand side of the constraints	95
5.3.4	Conclusion on the Lipschitz-continuity of HEGO	98
5.4	Monotonicity of the optimal negotiations	99
5.5	Non-monotonicity of the optimal trajectories	101
5.6	Emergency guidance method generalization	102
5.6.1	Generalized notations	102
5.6.2	Generalized emergency order	102
5.6.3	Generalized sequence of optimization problems	104
5.6.4	High-level description of safety margins	106
5.7	Illustrations	107
5.7.1	With the 2D model	107

5.7.2	With the 3D model	110
6	Performance evaluation	115
6.1	General comments	115
6.2	Input dispersion on 3D rocket model	117
6.2.1	Selection of figures	117
6.2.2	Observations and comments	118
6.2.3	Conclusion on the example	121
6.3	Comparison with vertical flight envelopes	121
Conclusion		145
A	Technical tools	147
A.1	Optimization results	147
A.1.1	Duality gap	147
A.1.2	Right-hand side sensitivity of Linear Programs	148
A.2	Differential Equations	150
A.2.1	Comparison theorem	150
A.2.2	Flow of Ordinary Differential Equations	150
B	Additional data	155
Bibliography		179

Nomenclature

Mathematical nomenclature

$\ \cdot\ _1$	=	1-norm	
$\ \cdot\ _2$	=	Euclidean norm	Denoted $\ \cdot\ $ if there is no ambiguity.
$\ \cdot\ _\infty$	=	∞ -norm	
≥ 0	=	Positive	In \mathbb{R}^n , must be understood element-wise.
$\succ 0$	=	Positive definite	
$\succeq 0$	=	Positive semidefinite	
\succeq_e	=	Emergency order	Extended co-lexicographic order (1.1).
\times	=	Vector product	
Φ_f	=	Flow of f	See Appendix A.2.2.
$\dim x$	=	Dimension of x	
$\mathbb{1}$	=	Vector of ones	Dimensions: $\mathbb{1}_n \in \mathbb{R}^n$.
\mathbb{I}	=	Identity matrix	Dimensions: $\mathbb{I}_n \in \mathbb{R}^{n \times n}$.
\mathbb{O}	=	Null matrix	Dimensions: $\mathbb{O}_{n \times m} \in \mathbb{R}^{n \times m}$.
$\text{Sat}_a^b(x)$	=	Saturation	Saturates x between a and b .
$\text{Sgn}(\cdot)$	=	Sign function	For $a \in \mathbb{R}$, $\text{Sgn}(a) := \begin{cases} +1 & \text{if } a > 0, \\ 0 & \text{if } a = 0, \\ -1 & \text{if } a < 0. \end{cases}$
IVP	=	Initial Value Problem	
OCP	=	Optimal Control Problem	
STM	=	State Transition Matrix	See Appendix A.2.2.
LP	=	Linear Programming	
QP	=	Quadratic Programming	
NLP	=	Non Linear Programming	
RHS	=	Right-hand side	
OoP	=	Out of Plane	

Mathematical nomenclature specific to Chapter 2

\mathbf{e}_\star	=	Unit vector	In direction of \star .
\mathbf{X}, X	=	Vector, Norm	For \mathbf{X} a vector of \mathbb{R}^3 , then $X = \ \mathbf{X}\ _2$.
$\mathcal{R}(\mathbf{a}, \gamma)$	=	Rotation	3×3 rotation matrix of axis \mathbf{a} and angle γ .

Nomenclature for the 2D model

α	=	Incidence	(In 2D) α is signed : $-90^\circ < \alpha < 90^\circ$.
$(\mathbf{e}_z, \mathbf{e}_h)$	=	Earth frame	

$(\mathbf{C}_A, \mathbf{C}_N)$	=	Rocket body frame	
θ	=	Attitude	(In 2D) θ is signed : $-90^\circ < \theta < 90^\circ$.
h	=	Altitude	
v_h	=	Vertical speed	
z	=	Horizontal position	
v_z	=	Horizontal speed	
q_r	=	Engine flow	
m	=	Total mass	
a_{nor}	=	Normal acceleration	a_{nor} is signed.
x	=	Rocket states	$x = (h, v_h, z, v_z, m)^\top \in \mathbb{R}^5$.
u	=	Guidance controls	$u = (q_r, \alpha)^\top \in \mathbb{R}^2$.

Nomenclature for the 3D model

α	=	Incidence	(In 3D) α is unsigned : $0^\circ \leq \alpha < 90^\circ$.
Ψ	=	Yaw	
ξ	=	Pitch	
$(\mathbf{e}_z, \mathbf{e}_y, \mathbf{e}_h)$	=	Earth frame	
$(\mathbf{C}_N, \mathbf{C}_A, \mathbf{C}_P)$	=	Rocket body frame	
θ	=	Attitude	(In 3D) θ is unsigned : $0^\circ \leq \theta < 90^\circ$.
ζ_z, ζ_y	=	Projected attitudes	Defined in Figures 2.4 and 2.5.
α_z, α_y	=	Projected incidences	Defined in Figure 2.5.
z	=	Horizontal position	(With respect to \mathbf{e}_z .)
y	=	Horizontal position	(With respect to \mathbf{e}_y .)
h	=	Altitude	
v_z	=	Horizontal speed	(With respect to \mathbf{e}_z .)
v_y	=	Horizontal speed	(With respect to \mathbf{e}_y .)
v_h	=	Vertical speed	
m	=	Total mass	
q_r	=	Engine flow	(Real)
q_c	=	Engine flow	(Controlled)
a_{nor}	=	Normal acceleration	a_{nor} is signed.
x	=	Rocket states	$x = (z, y, h, v_z, v_y, v_h, m, q_r)^\top \in \mathbb{R}^8$.
u	=	Guidance controls	$u = (q, \alpha_z, \alpha_y)^\top \in \mathbb{R}^3$.

Nomenclature for the optimization methods

ξ	=	Input variable	Size N_ξ , s.t. $\xi = (\Delta x^{0^\top}, \Delta \eta^\top, \Delta u_{\text{init}}^\top)^\top$.
\bar{x}	=	Reference state	State of size n .
\bar{u}	=	Reference control	Control of size m .
$\bar{\eta}$	=	Reference parameter	Parameter of size n_η .
\bar{t}_f	=	Reference time-of-flight	
μ	=	Discretized control	Size N_μ . For Cubic Splines, $N_\mu = m(N + 3)$.
τ_i	=	Control time instance	N elements s.t. $0 = \tau_0 < \tau_1 < \dots < \tau_N = 1$.
τ'_i	=	Constraint time instance	N_c elements s.t. $0 = \tau'_0 < \tau'_1 < \dots < \tau'_{N_c} = 1$.
Δt_f	=	Final time change	
z	=	Decision variable	Size $N_z = N_\mu + 1$, s.t. $z = (\mu^\top, \Delta t_f)^\top$.

p = Negotiation parameter Size n_{neg} , split in R sub-vectors.

Aerospace-specific acronyms

- **AGS**: Abort Guidance System, as used for the Apollo lunar module.
- **DLR**: Deutsches Zentrum für Luft und Raumfahrt.
- **GNC / GN&C / G&C** : Guidance Navigation and Control.
- **RCS**: Reaction Control Systems.
- **RLV**: Reusable Launch Vehicle [44].
- **SSTO**: Single-Stage-To-Orbit.
- **TSTO**: Two-Stage-To-Orbit.
- **Pinpoint landing**: for Mars, it is "sub-100 m" accuracy according to [93], and for the Earth 30 m inland and 10 m offshore according to [15].
- **Downrange**: horizontal distance between a vehicle and its landing site.
- **Slenderness ratio**: in the context of this thesis, denotes the aspect ratio of a rocket body (i.e. quotient between the height and the width) [89], or the aspect ratio of a trajectory (i.e. quotient between the altitude and the downrange).

Chapter 1

Introduction

Résumé

Ce chapitre introduit les différents enjeux liés au calcul embarqué de trajectoires d'atterrissage d'urgence pour des lanceurs réutilisables. Après avoir présenté le contexte dans lequel se déroule le développement de ces lanceurs, les différentes familles de méthodes de guidage disponibles dans la littérature sont rappelées. Ceci permet de présenter le problème au coeur de cette thèse, et les outils nécessaires pour le résoudre : l'ordre co-lexicographique (ordre d'urgence), les problèmes (linéaires) de négociation et le problème (quadratique) de raffinement. Enfin, le plan du manuscrit est annoncé, et un résumé schématique est proposé.

1.1 Powered Descent Guidance for reusable launchers

Reusable launchers are autonomous vehicles tailored for complex missions consisting in a succession of distinct phases. Their development is driven by the need for a fast and cost-efficient solutions to put payloads into orbit. Reusability imposes that return to Earth of the launcher's first stage must be accomplished safely and reliably. Powered Descent Guidance (PDG) refers here to the action of making such a flying vehicle land autonomously on a horizontal surface using rocket-like engines for maneuvers and deceleration. PDG has emerged as a paramount topic during the 1960's space race, the prime example of PDG being the lunar module (LM) landing of the Apollo missions, which was successfully performed six times in a row on the moon. LM landing used surprisingly straightforward path planning methods [50,93], relying on non-optimized analytic calculations in an atmosphere-free environment. Though the space race slowed down drastically in the 1970's, the need for PDG on other planets re-emerged with automated exploration missions towards Mars, the Moon and even some large asteroids. These missions require a high level of autonomy, for they are conducted extremely far from Earth, and thus cannot rely on near-instantaneous communications used in teleoperation control. Accuracy is also a key factor, especially for Mars missions, which aims at landing in the vicinity of rich geological features where landing areas are scarce and narrow [15]. Several missions performed a series of increasingly accurate autonomous powered landings, peaking with the impressive performance of the Perseverance Rover which landed

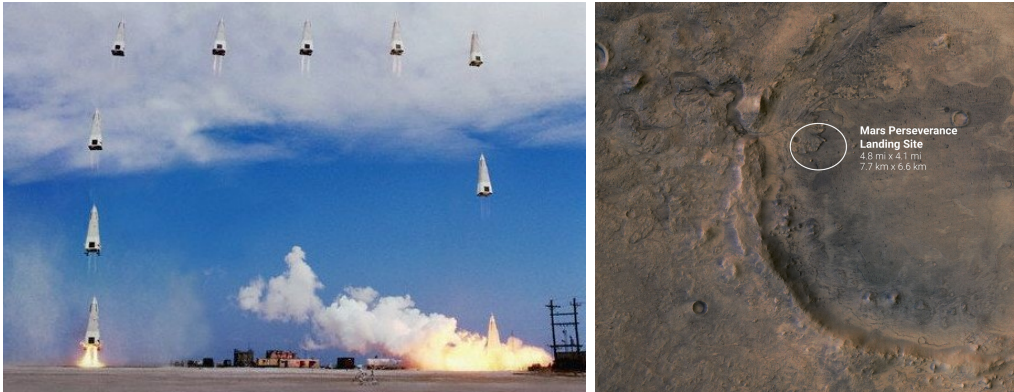


Figure 1.1: (*Left*) DC-X take-off and landing in 1993. (© New Mexico Museum of Space History) (*Right*) **Perseverance** landing site, 2021. (© ESA/DLR/FU-Berlin/NASA/JPL-Caltech)

onto a $7.7 \text{ km} \times 6.6 \text{ km}$ ellipse in 2021 (see Figure 1.1-(*Right*)). Interestingly, all of these missions operated in the absence of atmosphere or under negligible atmospheric density.

For PDG on Earth’s surface, where aerodynamic forces are not negligible, progress is more recent. At the end of the 1990’s, the DC-X project conducted collaboratively between McDonnell Douglas and NASA started to explore the possibilities for a reusable rocket that could land back on its “feet”, using the same engines for take-off and landing. As shown in Figure 1.1-(*Left*), this atypical tetrahedron shaped vehicle achieved several short flights, but never reached a significant altitude. It was necessary to wait until the early 2010’s to witness the first successful powered landings, achieved by Space X rocket Grasshopper. In parallel, some smaller prototypes were developed and successfully tested, such as the Xombie by Masten Space Systems and JPL¹. Today, several projects are under developments and have reached various Technology Readiness Levels, such as New Shepard and New Glenn (Blue Origins), FROG (CNES, [78]), Callisto (CNES/DLR/JAXA) and Themis (Ariane-Group/CNES/ONERA) or even Shenlan Nebula (Deep Blue Aerospace).

In this thesis, we are interested in *tossback* vehicles and, more precisely, in reusable Two-Stage-To-Orbit (TSTO) rockets, equipped with a single gimbaled engine, whose trajectory slenderness ratio² is medium to high. As shown in Figure 1.2, this type of launcher typically goes through six different flight phases: take-off, boost-back, ballistic, re-entry-burn, re-entry glide and final-burn. Our work focuses on the last flight phase, the final burn, which starts at a few kilometers of altitude, lasts for 10 seconds to a couple of minutes during which the engine is always turned on and (at least partially) controllable.

To land properly at a desired site, the typical Guidance and Control (G&C) strategy considers two objectives: *i*) the guidance problem i.e. determination of a high-level guidance trajectory , *ii*) the control problem i.e. tracking of this trajectory

¹See [3] and the associated video: youtube.com/watch?v=BqXFzVVCSCU.

²For trajectories, it is the ratio between maximum flight altitude and maximum downrange.

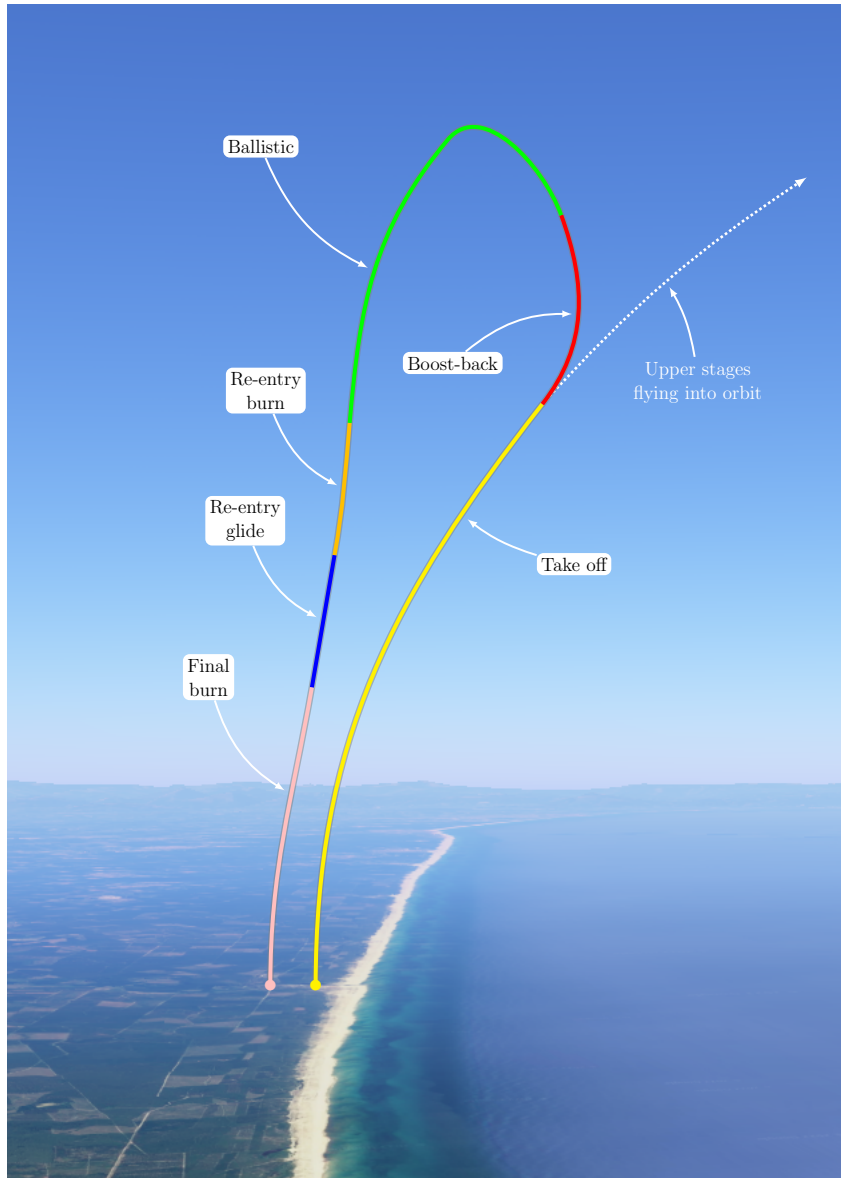


Figure 1.2: *Typical flight phases of a tossback vehicle, from take-off to landing (trajectory not to scale). The higher the slenderness ratio, the more vertical the flight trajectories.* (© Google Earth V 9.168.0.0, (July 28, 2022), France, Landsat/Copernicus)

using a low-level controller. This thesis addresses the guidance problem only under the assumption of full knowledge of the state of the system, making low-level control and other estimation topics out of scope.

Providing guidance for the final burn is a challenging and necessary task. Indeed, solving the PDG *in-flight* must be achieved using limited time and computational power. Only single-CPU methods are studied in the thesis, as opposed to methods requiring the use of GPUs. Further, PDG has to deal with the accumulation of the tracking errors - i.e. the distance between the actual rocket states and the expected reference trajectory - during the flight phases prior to the final burn, which have to be shrunk to zero during this last phase [15]. This must be done under strong disturbances (e.g. wind speed or changes of atmosphere density profile among others) and under multiple constraints: the incidence - or angle-of-attack - must be limited to avoid hazardous regions of the aerodynamic flight domain, the engine flow is mechanically bounded and its internal dynamics is not instantaneous, the low-level controllers actuating the rocket have limited capabilities which requires to bound the normal acceleration, the landing site is small, the available mass is limited, etc. Gathering all these detrimental effects, it is likely that the rocket starts its final burn *too far* from its reference trajectory to get a landing trajectory strictly meeting all these requirements. However, at the expense of sacrificing some requirements and/or loosening well-chosen constraints, it is possible to successfully land and maintain the reusability of the vehicle.

A key concept to preserve the launcher’s reusability is to “maximize the launcher’s integrity during landing”. To illustrate it, consider the following example, pictured in Figure 1.3. The reference trajectory **A** is shown in plain **black**. In a typical nominal scenario, the final burn would start reasonably close to the reference trajectory such that a proper guidance algorithm would provide the rocket **B** with a trajectory that reaches the desired landing at null speed while satisfying all the constraints listed earlier. However, if the rocket starts its descent farther away from the reference trajectory - e.g. with a large lateral displacement - then, at some point, it is impossible to land properly and to satisfy the whole system of constraints, which are in fact incompatible. Allowing sharper turns, which can be achieved by relaxing the incidence bound (or the normal acceleration bound) may make the system of constraints compatible again, at the cost of a trajectory alteration. Indeed, this relaxation enables the rocket **C** to land on the desired landing site. If we push this reasoning further, the rocket could start its final burn far enough from the reference trajectory so that even relaxing the incidence bound would not be enough. If the area neighboring the desired landing site is uninhabited and flat enough, one can allow the landing location to be relaxed too. This is what happens in the case of rocket **D**.

This first point highlighted by this example is that there is a list of quantifiable elements that can help to relax the constraints (to some extent). These will be referred to as *negotiable parameters* in this thesis. Maximizing the launcher’s integrity is choosing the values of these parameters in an optimal way. The negotiable parameters can refer to many different factors, such as the incidence bound, the normal acceleration bound, the landing site location or even the final vertical speed. The set of negotiable parameters can differ depending on the mission requirements,

the rocket structural capabilities or the landing site surroundings for instance. In the thesis we develop a generic methodology able to handle different sets of landing parameters, and not only a single particular set.

Further, the above-mentioned example illustrates why these parameters are not equally critical. Indeed, relaxing the incidence bound of a couple degrees will have a much lower impact on the rocket structure than relaxing the maximum final vertical speed by a few meters per second. A classic way to handle this relative importance while solving the PDG problem is to penalize the constraint incompatibility [28]. Computationally, it means that: *i*) the negotiable parameters are added to the decision variables of the optimization problem describing the descent trajectory, and that *ii*) the use of these parameter is penalized in the cost of the optimization problem. In this case, the relative importance of the parameters is partially ensured by the difference in the weights associated to each parameter. However, this is a heuristic, and it does not guarantee the relative importance of the parameter to be exactly satisfied. Instead, we propose to mathematically define the relative importance of the negotiable parameters via the introduction of a strict *hierarchy* between these parameters. In the thesis, this hierarchy is represented by a specific order relation over the set of negotiable parameters, which we call the *emergency order* below. A challenge for the guidance methodology to be developed is to compute the smallest constraint alteration that recovers feasibility in the sense of the latter order.

Following the discussion above, our main objective in this thesis is formulated as follows: *design a method to update the guidance trajectory at the beginning of the final-burn, while maximizing the launcher's integrity, in a computationally efficient way.* To address this objective, the thesis proposes a guidance method that performs online trajectory optimization for the final burn, while minimizing the use of negotiable parameters according to a certain hierarchy making sure that their relative importance is strictly respected.

1.2 Mathematical programming for online PDG

1.2.1 Current state-of-the-art

The Powered Descent Guidance (PDG) problem is a worked applied mathematics problem, most often tackled from the Optimal Control perspective.

The decision variables belong to infinite-dimensional function spaces (the control law u and the state x), and the scalar time-of-flight (t_f) over which these functions are defined. The goal in such Optimal Control Problems (OCPs) is to find the triplet (x, u, t_f) that minimizes a certain criterion, under multiple constraints.

Several performance indexes (or cost functions) have been considered in the literature: minimum-fuel [3, 27, 55, 64, 97, 107], minimum-acceleration [93], minimum time-of-flight [64, 93], minimum error-to-reference [89], minimum-landing-error [17], or any weighted combinations of these [33, 82].

The constraints depend on the rocket design and the mission requirements. They can take the form of inequalities (e.g. engine flow upper and lower bounds) as well as equalities (e.g. final horizontal and vertical speeds). As discussed earlier, these constraints can be quantified by several parameters. Among these, the ones that can eventually be modified are called the *negotiable parameters*. For instance, the

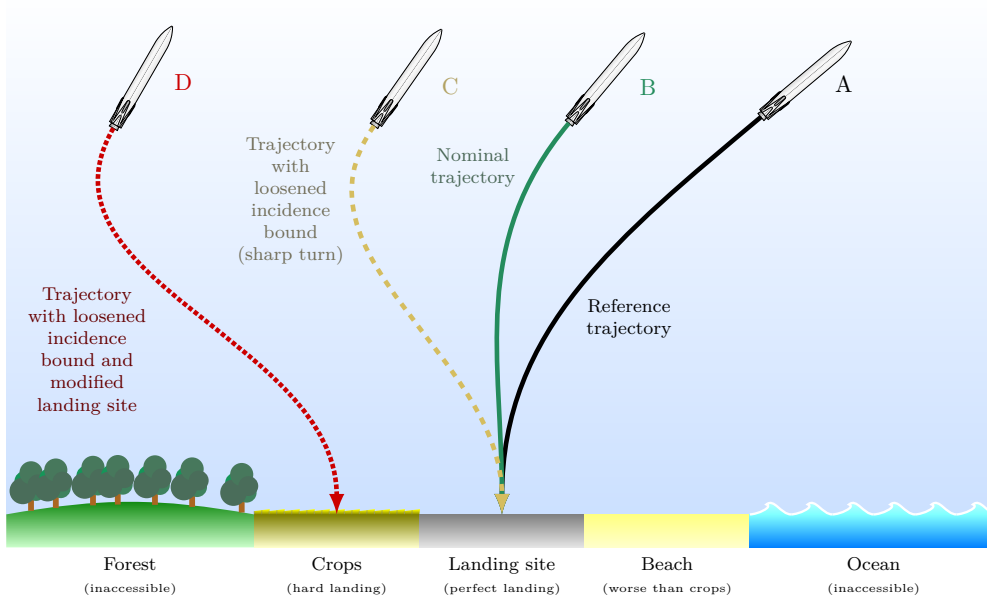


Figure 1.3: Trying to maximize launcher's integrity by relaxing well-chosen constraints.

incidence bound is a safety requirement regarding mechanical and flight quality aspects, whose value may be negotiated if necessary. On the contrary, the engine flow bound is a typical example of non-negotiable parameters, since it is a physical limitation. Mathematically, these cases are treated as follows: *the constraints can be tuned using a vector parameter denoted p* , which is defined such that $p = 0$ when the *nominal* requirements are met.

The PDG problem has several inputs: the initial state of the rocket - i.e. its initial position, speed, orientation and mass - and other external parameters, such as the wind speed for example. These cover all the above-mentioned sources of disturbances. Mathematically, *the inputs are conveyed by a parameter ξ* .

From a general perspective, the PDG problem, seeking a guidance trajectory for the final burn, is written as an OCP such that

$$\text{OCP}(\xi, p) := \left\{ \begin{array}{ll} \min_{x, u, t_f} \mathcal{J}(x, u, t_f, \xi) & \text{(Cost),} \\ \text{s.t. } \dot{x} = f(x, u) & \text{on } [0, t_f] \text{ (Rocket dynamics),} \\ x(0) = x^0 + \xi & \text{(Initial conditions),} \\ \Psi(x(t_f), p) = 0 & \text{on } [0, t_f] \text{ (Final state),} \\ C(x, u, p) \leq 0 & \text{on } [0, t_f] \text{ (Other constraints).} \end{array} \right.$$

where \mathcal{J} is the performance index (cost), f is the right-hand side of the ordinary differential equations, Ψ represents the terminal condition and C conveys the var-

ious constraints. The special case OCP $(\xi, 0)$ is the *nominal* PDG problem, and OCP $(0, 0)$ corresponds to the reference trajectory. This problem having an infinite dimensional decision variable and an infinite number of constraints, it is discretized for numerical resolution (for some given value of ξ and p). Two well-known and dual discretization approaches exist (see e.g. [14, 47, 76, 99]). On one hand, for *indirect* methods, stationary conditions are derived from OCP (ξ, p) in the form of Ordinary Differential Equations (ODEs) which are then discretized and solved. The celebrated Pontryagin Maximum Principle [46] is the fundamental tool to form the stationary conditions. On the other hand, for *direct methods*, OCP (ξ, p) is first discretized into the sub-problem DSC (ξ, p) , using a finite dimensional variable z representing the unknowns (x, u, t_f) of the initial problem, where DSC (ξ, p) is an optimization problem of the form

$$\text{DSC}(\xi, p) \quad := \quad \begin{cases} \min_z & J(z, \xi) \\ \text{s.t.} & h(z, \xi, p) \leq 0, \\ & g(z, \xi, p) = 0, \end{cases}$$

which is then solved using Non-Linear Programming (NLP).

Though the indirect methods are known to be more accurate, they are also very sensitive³ and usually are not used in real-time. Currently, for *online* applications, PDG problems are solved almost exclusively using direct methods. We follow this trend and will focus of the parametric problem DSC (ξ, p) .

Still, the main challenges for direct methods are the choice of the decision variable z , the design of the associated functions J , h and g , and the selection of the proper numerical method used to solve DSC (ξ, p) . These decisions directly determine whether DSC (ξ, p) can be used for online applications with sufficient accuracy.

A broad spectrum of direct methods have been used to solve OCP (ξ, p) for aerospace applications [14]. Direct trajectory optimization methods relying on NLP have been explored since the 1980's [45]. Since then, many methods have emerged. Pseudospectral methods - consisting in discretizing the states and the controls of OCP (ξ, p) at well-chosen time points using polynomial approximations [36] to transform OCP (ξ, p) into a NLP - have been investigated from various perspectives [79, 81, 91, 92, 106]. Sensitivity-based parametric optimization has been used for Mars atmospheric entry problems [85]. Recent work also focused on applying learning based methods to PDG [39]. A noteworthy state-of-the-art direct method aiming at solving OCP (ξ, p) for rocket landing problems is *Successive Convexification* [4]. This method solves OCP (ξ, p) by successively solving linearized and discretized versions of it [60, 77, 98]. In other words, this method is a variant of Successive Convex Programming (SCP) methods, tailored and optimized for the requirements of PDG. This method is presented in great details in [58]. Theoretical guarantees were established by B. Açikmeşe and his fellow researchers. Informally speaking, the main statement can be rephrased as follows: “*if the method converges and if some intermediate slack variables are sufficiently penalized and tend to zero, then the solution is the optimal landing method in the sense of the Karush-Kuhn-Tucker conditions*”

³An initial guess is needed for the adjoint state or the input u , t_f and numerical methods are sensitive to these guesses.

and the constraints are satisfied” [58, Thm. 8]. Superlinear convergence rates have been established under mild assumptions [61]. Specifically designed interior point solvers have also been implemented [34]. These results are rather powerful and the method proved to work correctly even on real vehicles [3]. It is often used along with *Lossless Convexification*, which aims at performing an exact convex relaxation of the thrust magnitude constraints [16, 27, 58]. The above-mentioned list of direct methods for PDG is not exhaustive. For further details on the available methods, see the recent survey by Song *et al.* [89].

All the above-mentioned methods tackle different versions of the PDG problem, but face the same bottleneck when it comes to landing on Earth’s surface: the presence of the aerodynamic forces. Without atmosphere, i.e. for Moon and Mars landing⁴, the problem is called *planetary landing*, and one can safely say that the above-mentioned literature is mature enough to provide efficient and proven PDG methods for real-time usage. A noteworthy example is the full characterization of the reachable set for Mars landing [35], obtained by combining convex analysis and Successive Convexification. Though a few papers have tackled PDG in the past decade, there are still many open questions regarding the best way to perform online PDG in the presence of non-negligible aerodynamic effects. In this thesis, this is the case we consider.

1.2.2 Hierarchy and the emergency problem

In this thesis we address cases when ξ is such that there exist *no* trajectories (x, u, t_f) that satisfy all the constraints of OCP (ξ, p) . This situation, referred to as the *emergency* problem, is also sometimes called the *abort planning* problem in the air and space field [26, 44, 52, 57, 87, 101]. To the best of our knowledge, a single paper has tackled this topic for planetary PDG, for Mars landings [17]. Using Lossless and Successive Convexification, the latter paper exposes a method that first minimizes the landing error - i.e. the minimum distance between the feasible landing sites and the targeted one - and then computes the fuel-optimal trajectory among the ones having a minimum landing-error. Our goal is to design a more general methodology to perform emergency guidance by dealing with *more than a single* parameter to relax.

From a modeling perspective, in this thesis, the above-mentioned vector of negotiable parameters p is decomposed into R vector parameters of possibly different dimensions, ranked with respect to their relative importance, such that

$$p = (p^{(1)}, \dots, p^{(R)}) = (\text{Least critical}, \dots, \text{Most critical}).$$

For instance, one can consider that $p^{(1)} = \Delta\alpha_{\max}$ (the incidence bound) is less critical than $p^{(2)} = \Delta v_h^f$ (the final vertical speed). The elements $p^{(j)}$ can be vectors. For instance, $p^{(j)} = (\Delta z^f, \Delta y^f)$ is the vector denoting the landing site location on a map.

The hierarchy among the parameters is defined using a variant of the *lexicographic* order: the negotiable parameters are compared using the 1-norm of their

⁴Mars atmosphere is often treated as a small disturbance, for it is more than 100 times thinner than Earth’s.

sub-parameters, by comparing their most critical sub-parameters first. In practice, a vector p_a is said to be *larger - or more negotiated -* than another vector p_b , which is denoted

$$p_a \succeq_e p_b$$

if and only if

$$\begin{aligned} \|p_a^{(R)}\|_1 > \|p_b^{(R)}\|_1 & \text{ or } \|p_a^{(R)}\|_1 = \|p_b^{(R)}\|_1 \text{ and } \|p_a^{(R-1)}\|_1 > \|p_b^{(R-1)}\|_1, \\ & \text{ or } \dots \\ & \text{ or } \|p_a^{(R)}\|_1 = \|p_b^{(R)}\|_1 \text{ and } \dots \text{ and } \|p_a^{(1)}\|_1 > \|p_b^{(1)}\|_1, \\ & \text{ or } \|p_a^{(R)}\|_1 = \|p_b^{(R)}\|_1 \text{ and } \dots \text{ and } \|p_a^{(1)}\|_1 = \|p_b^{(1)}\|_1. \end{aligned} \quad (1.1)$$

In the context of this thesis, the order \succeq_e is called the *extended colexicographic order* or simply the *emergency order*.

Maximizing the launcher's integrity by sacrificing the parameter p according to the hierarchy of importance between the sub-parameters translates into *finding the smallest p in the sense of \succeq_e such that there is a feasible trajectory (x, u, t_f) satisfying the constraints of OCP (ξ, p)* . There are no generic methods yet available that recovers feasibility in OCP (ξ, p) while enforcing such a hierarchy. This is the subject and main contribution of this thesis.

1.3 Proposed contribution: Hierarchical Emergency Guidance Optimization

The main contribution of this thesis is an online method that computes the optimal trajectory and the optimal values of the negotiable parameters, in the sense of the extended colexicographic order, using only Linear and Quadratic programming techniques. In the manuscript, this method is demonstrated on two rocket models of increasing complexity: from a planar model with non-trivial aerodynamic effects, to a richer three-dimensional model with non-negligible engine transients.

This contribution is obtained in two steps: first a nominal guidance method computing the landing trajectory z for a given p is presented, then an emergency guidance method aiming at computing the best value for p is studied.

1.3.1 Fast PDG for fixed value of constraint parameters : a sensitivity-based approach

To perform nominal guidance, i.e. to compute z for given values of ξ and p , we use a sensitivity-based approach.

In this approach, OCP (ξ, p) is described as an optimal *correction* problem PDG (ξ, p) , defined w.r.t. a reference trajectory $(\bar{x}, \bar{u}, \bar{t}_f)$ (e.g. rocket **A** in Figure 1.3). The latter trajectory is mission-specific, and its design is out-of-scope for this manuscript. Using handy notations⁵, PDG (ξ, p) is sketched below as an infinite

⁵The exact definition of problem PDG (ξ, p) , presented here in a simplified shape, is in Chapter 2.

dimensional optimization problem of the form

$$\text{PDG}(\xi, p) := \begin{cases} \min_{\delta u, \Delta t_f} & \mathcal{J}(\delta u, \Delta t_f) \\ \text{s.t.} & \dot{x}(t) = f(x(t), \bar{u}(t) + \delta u(t)), \\ & x(0) = \bar{x}^0 + \xi, \\ & \Psi(x(\bar{t}_f + \Delta t_f), p) = 0, \\ & C(x(t), \bar{u}(t) + \delta u(t), p) \leq 0. \end{cases}$$

The decision variable δu , the ODE of the dynamics and the constraints are discretized using a finite-dimensional variable z , which conveys a parametric description μ of the control change w.r.t. the reference control \bar{u} and the implicit time-of-flight change Δt_f . Through this discretization procedure, the state x is expressed as a function of ξ and z using the flow of the ODE defined by f . The problem $\text{PDG}(\xi, p)$ is approximated by its discrete non-linear version $\text{NLP}(\xi, p)$, which writes

$$\text{NLP}(\xi, p) := \begin{cases} \min_z & J(z, \xi) \\ & h(z, \xi) \leq H_p p, \\ & g(z, \xi) = B_p p. \end{cases}$$

where it is stressed that the negotiable parameter p has a linear influence on the constraint right-hand side of the guidance problem tackled in this thesis.

Then, sensitivity analysis is used to approximate the solution of $\text{NLP}(\xi, p)$. Revisiting results from the literature, we introduce a Quadratic Program (QP) which writes

$$\text{QP}(\xi, p) := \begin{cases} \min_z & \frac{1}{2} z^\top P z + q^\top z \\ \text{s.t.} & G z \leq h_0 + H_\xi \xi + H_p p \\ & A z = b_0 + B_\xi \xi + B_p p \end{cases}$$

and whose solution is shown to be a local approximation to the solution of $\text{NLP}(\xi, p)$, under mild assumptions. Interestingly, Strict Complementary Slackness, often assumed in the literature, is not needed here. Approximating $\text{NLP}(\xi, p)$ by $\text{QP}(\xi, p)$ is shown to be relevant and usable even for large - non-local - values of ξ , when used with the two above-mentioned rocket models. The problem $\text{QP}(\xi, p)$ contains a convenient linear description of the constraints which makes the emergency guidance method described thereafter tractable.

Problems such as $\text{QP}(\xi, p)$ can be solved in an online/offline fashion, where the defining constant matrices are computed before the flight for the prescribed reference trajectory, and where the QP itself is solved for given ξ and p on-board using off-the-shelf QP solvers. Numerical resolution can be achieved with confidence as QP solvers are now considered mature and reliable technology [21, 63, 94].

1.3.2 Computing the best constraint alteration achieving feasibility

When ξ is such that the nominal guidance problem $\text{QP}(\xi, 0)$ has no solutions, we propose a method computing a value of p such that $\text{QP}(\xi, p)$ is feasible.

Finding the smallest p in the sense of \succeq_e guaranteeing the existence of a trajectory is achieved by solving the following R different *negotiation problems*. The idea is to iteratively minimize each sub-parameter $p^{(j)}$ under the 1-norm (consistently with the mathematical definition of \succeq_e), and memorize the associated optimal value \mathcal{P}_j^* . For each negotiation problem, the constraints are those of NLP (ξ, p) , plus new constraints that ensure that the optimal values $\mathcal{P}_{j+1}^*, \dots, \mathcal{P}_R^*$ of the previous negotiation problems are reached. In details, the j^{th} negotiation problem writes

$$\mathcal{P}_j^* \longleftarrow \min_{z, p} \|p^{(j)}\|_1 \quad (1.2a)$$

$$\text{s.t. } Gz \leq h_0 + H_\xi \xi + H_p p \quad (1.2b)$$

$$Az = b_0 + B_\xi \xi + B_p p \quad (1.2c)$$

$$\|p^{(i)}\|_1 = \mathcal{P}_i^*, \quad i = j + 1, \dots, R. \quad (1.2d)$$

1.3.3 The HEGO algorithm: main contribution of the thesis

Once this problem has been solved for each j , starting from R down to 1, we minimize z under the original performance index J , such that

$$z^* \longleftarrow \arg \min_{z, p} \frac{1}{2} z^\top P z + q^\top z \quad (1.3a)$$

$$\text{s.t. } Gz \leq h_0 + H_\xi \xi + H_p p \quad (1.3b)$$

$$Az = b_0 + B_\xi \xi + B_p p \quad (1.3c)$$

$$\|p^{(i)}\|_1 = \mathcal{P}_i^*, \quad i = 1, \dots, R. \quad (1.3d)$$

This latter problem is the *refine* problem. It only returns the value of z . The value of p is not necessarily unique. The “negotiation and refine” problems are gathered into the HEGO algorithm, which is a nominal and emergency guidance algorithm, described using pseudo-code below.

Algorithm 1 Hierarchical Emergency Guidance Optimization (HEGO)

Require: Inputs ξ .

for $j = R, \dots, 1$ (decreasing indices) **do**

 Solve the negotiation problem (1.2) for j , and store \mathcal{P}_j^* .

end for

Solve the refine problem (1.3), using $\mathcal{P}_1^*, \dots, \mathcal{P}_R^*$ in (1.3d), which gives z^* .

return Optimal trajectory (x^*, u^*, t_f^*) , described by z^* .

Overall, HEGO is a numerical method that provides nominal and emergency guidance, relying on Linear and Quadratic programming solvers only.

1.4 Manuscript outline

A high-level summary of the approach with references to the associated chapters is presented in Figure 1.4.

Chapter 2 introduces the dynamic model of the class of reusable launchers under study, with two levels of complexity (2D and 3D rocket models). Among others, the non-trivial aerodynamic model is detailed, and special care is taken to define the 3D model. Then, after discussing the various mission constraints, the PDG problem is formulated as an OCP with respect to a reference trajectory.

Chapter 3 takes a side turn and tackles the problem of optimal thrust programming for the special case of a purely vertical atmospheric flight. By applying Pontryagin Maximum Principle, necessary and sufficient conditions are derived, showing that the optimal thrust program is min-max for the class of launchers that we study. The by-product of this result is a full characterization the reachable set of the rocket for the vertical landing problem. *For a first reading, this chapter can be skipped without loss of continuity.*

Chapter 4 details the method that performs nominal trajectory planning, sketched in Section 1.3.1. Using a finite-dimension decision variable to describe the trajectory of the PDG, a NLP is derived. Its optimal solution is then approximated using parametric sensitivity analysis, and solved using a single QP. The mathematical formulation of this QP is instrumental in the rest of the thesis.

Chapter 5 presents the core topic of this thesis. After discussing the available negotiation parameters, the Algorithm HEGO is introduced in the LP/QP framework. Its behavior is explained on a detailed toy example. Then, theoretical guarantees are proposed and proved. Among others, the Lipschitz-continuity of the optimal solution z^* is established, guaranteeing the absence of “jumps” in the solutions, a desirable property in practice. Also, high-level comments on the emergency guidance method are proposed, to distinguish what is generalizable from what comes from the underlying nominal guidance problem. Finally, several examples are presented to illustrate the various modeling possibilities offered by HEGO, and to visualize the quality of the results.

Chapter 6 offers a quantitative assessment of the performances of HEGO. The inputs of the algorithm are dispersed over wide uncertainty intervals, and the results are analyzed pair-wise. Also, a comparison with the vertical landing problem from Chapter 3 is presented.

A concluding chapter discusses a few topics that have not been detailed in the previous chapters, it also presents possible future research directions, and conclusions.

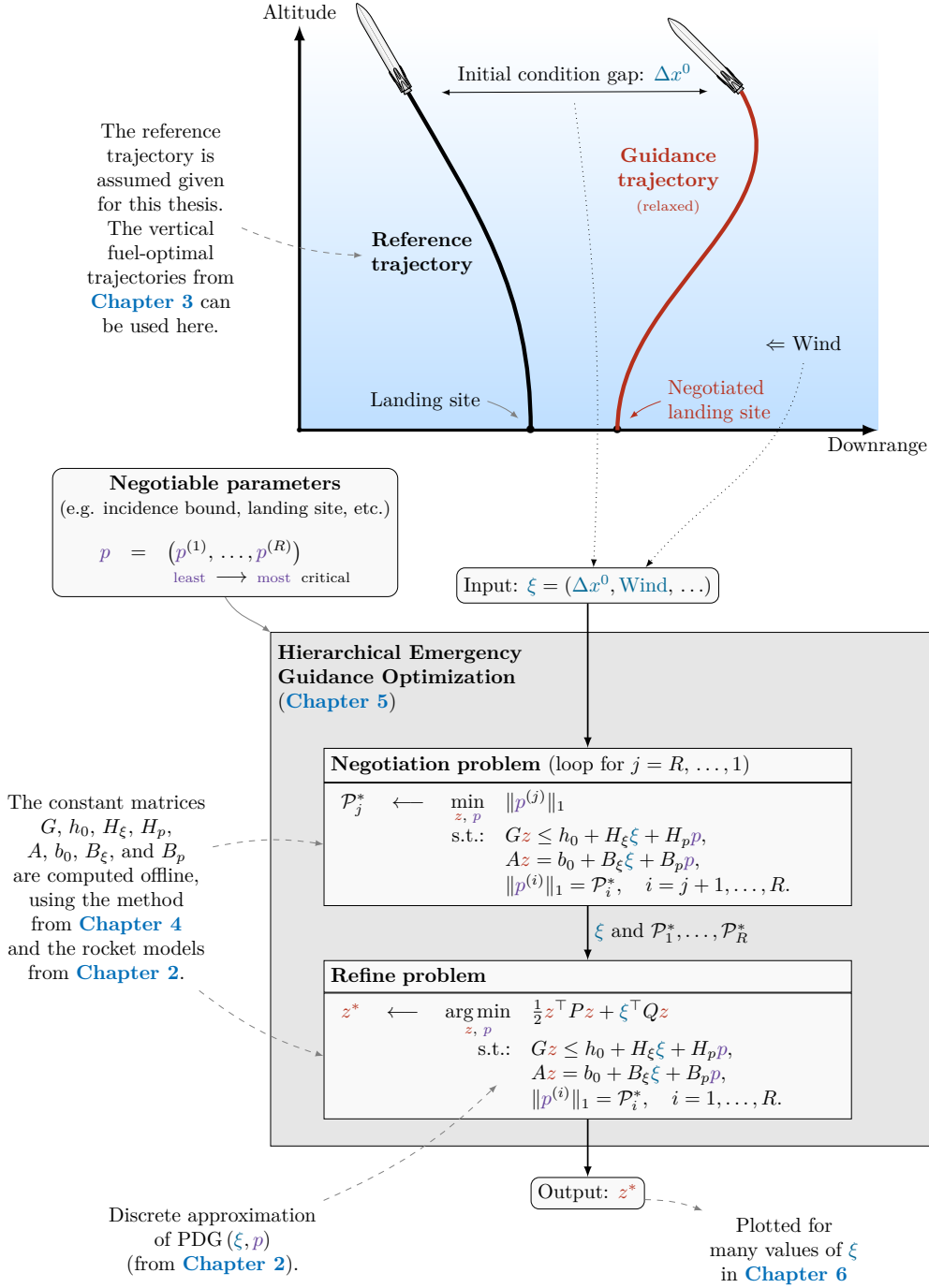


Figure 1.4: *High-level summary.* Nominal and emergency guidance methods presented in this thesis, with references to the associated chapters.

Chapter 2

Dynamic models and the PDG problem

Résumé

Ce chapitre contient une description mathématique du problème de guidage pour l'atterrissage (PDG). Il introduit les modèles dynamiques dérivant la fusée et les contraintes.

Tout d'abord, les choix généraux de modélisation sont discutés, ce qui permet de souligner le rôle de l'atmosphère, une question rarement abordée dans le cadre du guidage pour l'atterrissage. Deux modèles de fusée, avec différents niveaux de complexité, sont présentés. Un modèle de fusée dans le plan (2D) et un modèle de fusée tridimensionnel (3D) sont construits. Le modèle de fusée 2D suppose que la fusée reste toujours dans un seul plan. Il servira à illustrer les principes de guidage des chapitres suivants d'une manière beaucoup plus accessible que le modèle 3D. En revanche, le modèle 3D est utilisé dans les exemples avancés des Chapitres 4 et 5 et pour l'ensemble du Chapitre 6, ce dernier contenant une évaluation des performances numériques et vérifie l'applicabilité de la méthodologie proposée.

Un problème général de calcul de trajectoire est présenté à la fin de ce chapitre, sous la forme d'un OCP en temps final libre, défini par rapport à une trajectoire de référence. La résolution de ce problème sera l'objet principal du Chapitre 4. L'importance relative des contraintes du PDG sera examinée plus loin dans le Chapitre 5.

This chapter contains a mathematical description of the PDG problem. It introduces the dynamic models governing the rocket and the constraints.

First, general modeling choices are discussed, which serve to stress the role of the atmosphere, a seldomly looked at issue in PDG. Two rocket models, with different levels of complexity are presented. A planar (2D) and a three-dimensional (3D) rocket model are constructed. The 2D rocket model assumes that the rocket always remains in a single plane. It will serve to illustrate the guidance principles of the next chapters in a much more accessible way than the 3D model. On the other hand, the 3D model is used in the advanced examples of Chapter 4 and 5 and for the whole

Chapter 6 containing the numerical results assessing the numerical performance and the applicability of the proposed methodology.

A general trajectory design problem is presented at the end of this chapter, as a free-final time OCP defined w.r.t. a reference trajectory. The resolution of this problem will be the main concern of Chapter 4. The relative importance of the PDG constraints will be discussed later in Chapter 5.

2.1 Atmospheric flight dynamics

Here are presented the features shared by both 2D and 3D models. Some notions, such as those regarding the environment and the aerodynamic model are also used in Chapter 3.

The typical flight phases of a tossback vehicle are illustrated in Figure 1.2. In this thesis, we are interested in the last part of the flight: the final burn until landing. It starts a few kilometers above the ground [22].

In both 2D and 3D rocket models, which will be used in guidance algorithms, the rocket is assumed to be a punctual mass having an orientation (and thus having an aerodynamic incidence). This conceptual approach is presented in details below.

2.1.1 Earth and atmosphere model

Since we are only interested in the last flight phase, the Earth is considered locally *flat, non-rotating*, with a *constant gravity* field of magnitude g . The atmosphere is described via the pressure P_a , the density ρ , the temperature and the speed of sound S_{SP} are functions of the altitude. They are computed using linear interpolation of data samples.

Denoting by V_r the norm of the relative speed of the rocket, we define $M_a := V_r/S_{SP}(h)$ the Mach number at a given altitude h .

The wind is assumed to be horizontal with a speed depending on the altitude only. It is assumed null at null altitude. In practice, the wind map is described by its value at three reference altitudes¹, for each direction:

- $w_{z,0}, w_{z,1}$ and $w_{z,2}$ for the first horizontal direction (2D and 3D models),
- $w_{y,0}, w_{y,1}$ and $w_{y,2}$ for the second horizontal direction (3D model only).

For these wind parameters, index 0 (resp. 1, 2) corresponds to an altitude of $h = 2$ km (resp. 5 km, 10 km).

2.1.2 Aerodynamic model

The aerodynamic model of a rocket moving in the direction of its thrust flame is notoriously hard to determine. Early works from 1966 started to describe the aerodynamic effect of an air jet pushing in front of a body in a supersonic flow.

¹Picking three values to describe the wind profile is an arbitrary design choice. If it is needed to consider a finer description of the wind profile in practice, significantly increasing the number of variables describing the wind profile does not change the approach presented in the next chapters, as discussed in Chapter 6.

Later works from the early 2000's from JAXA have completed these observations [71]. They were followed and corroborated by recent experiments of the DLR [62]. The common findings of [71] and [62] are that for a sufficiently strong jet flow, its wrapping around the rocket body lowers drastically the drag along the body axis, though orthogonal effects remain strong for non-zero incidences. This can be partially explained by the fact that the air jet creates an air cushion in front of the rocket that helps the boundary layer to stick to the fuselage from its lower end. A qualitative explanation of this observation is presented in Figure 2.1.

For the sake of this thesis, the aerodynamic effects of the air flow around the rocket is taken into account via:

1. A non-trivial lift coefficient C_{Lift} , to account for the aerodynamic effect orthogonal to the drag and in the opposite direction to the relative speed,
2. A small-magnitude drag coefficient C_{Drag} , to account for the aerodynamic effect in the direction of the rocket body,
3. Drag and lift coefficients C_{Drag} and C_{Lift} depending on two parameters: the rocket incidence α and the Mach number M_a ,
4. An altitude-corrected expression of the thrust. Denoting by T the thrust magnitude along the rocket body, we will consider that

$$T = g \text{ISP} q - S_E P(h) \quad (2.1)$$

where g is the gravity acceleration, ISP the engine specific impulse, q is the engine flow, S_E the nozzle section surface and h the altitude.

5. A lift vector s.t. the effective aerodynamic forces on the rocket are contained in the plane defined by the relative speed vector and the rocket body longitudinal axis.

The functions $M_a, \alpha \mapsto C_{\text{Drag}}(M_a, \alpha)$ and $M_a, \alpha \mapsto C_{\text{Lift}}(M_a, \alpha)$ considered below for the 2D and 3D models convey the same aerodynamic model.

Remark 1. *Few articles have tackled PDG where drag is not negligible. See for instance [97], [55] or [22].*

2.1.3 Noteworthy particularities

2.1.3.1 Engine dynamics

The engine is assumed to generate the only control forces available on the rocket (i.e. aerodynamic grid fins or Reaction Control Systems (RCS) are not considered in this thesis). The output flow and the angular position of the engine are actuated. As far as the flow is concerned, its dynamic is not instantaneous and should not be neglected. It is assumed that its real flow is q_r , that the controlled flow - or input signal - is q_c , and that they are related via a first-order low-pass dynamics with time constant τ_q s.t.

$$\dot{q}_r = \frac{q_c - q_r}{\tau_q}. \quad (2.2)$$

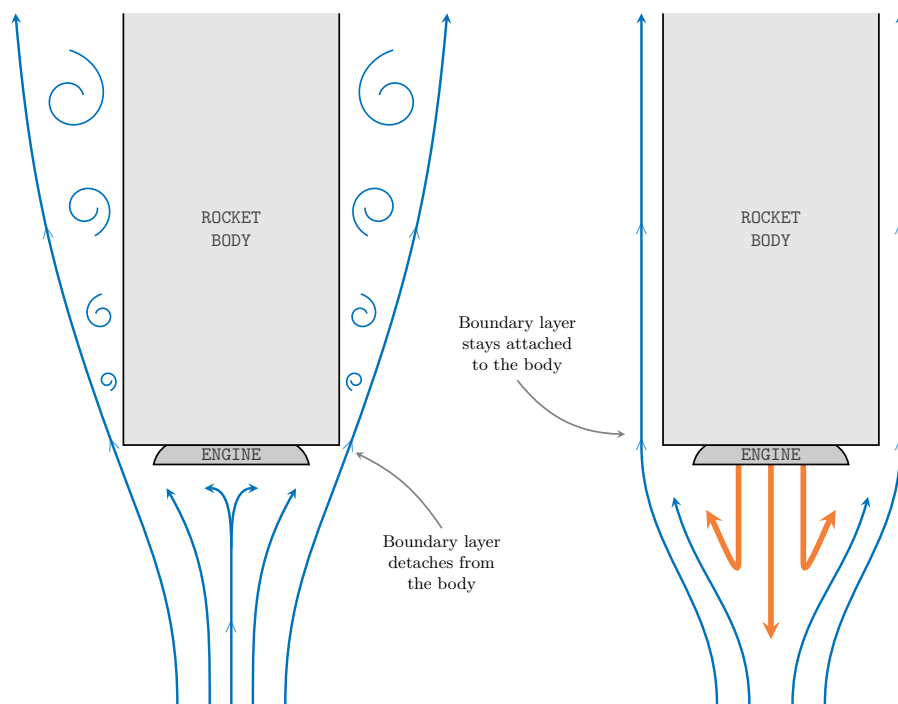


Figure 2.1: *Air jet influence*. Qualitative change of the wind flow with (*left*) and without (*right*) an air jet pushing in front of a moving rocket. See [71] and [62] for experimental results.

The dynamics of the angular position of the engine² is not rigorously instantaneous. However, the time constant of its transient is sufficiently small compared to the other time constants of the guidance problem to be neglected. In the following, it will only be required that the attitude of the rocket (that the rocket control system will track) be a continuous function of time.

Remark 2. Consider some lower and upper bounds q^- and q^+ on the engine flow. If $q_r(0) \in [q^-, q^+]$ and $q_c(t) \in [q^-, q^+]$, then from Equation (2.2) we get that $q_r(t)$ remains within these bounds for all $t \geq 0$. However, if $q_c(t)$ is not guaranteed to be between q^- and q^+ , the proper version of Equation (2.2) becomes

$$\dot{q}_r = \frac{\text{Sat}_{q^-}^{q^+}(q_c) - q_r}{\tau_q}.$$

2.1.3.2 Thrust direction

The thrust is assumed colinear to the rocket longitudinal axis. This assumption is an approximation, and corresponds to the equilibrium of the low-level controllers (out-of-the-scope of this thesis). More precisely, non-zero nozzle gimbal angles would slightly deviate the thrust vector, creating a momentum and in the end a rotation of the rocket. However, the rotation time constant is assumed significantly smaller than the translation time constant involved in the guidance problem.

2.1.3.3 Thrust dominance

The rocket engine is powerful compared to its weight, and the rocket incidence remains always sufficiently low s.t. the *vertical speed is always negative* and slowing down. Among others, this assumption prevents *hovering* maneuvers.

2.1.3.4 Dynamic equation and parameters choice

The dynamic equation of each model will be described by an ODE of the form

$$\dot{x} = f(x, u, \eta)$$

where x denotes the rocket states, u its controls, and η its dynamics parameters.

As will be detailed next, the rocket states are the position, the speed and the total mass, plus the real engine flow for the 3D rocket model only. The controls are the controlled engine flow, and one or two variables that convey the rocket incidence, depending on which model is considered (2D or 3D).

The states and controls chosen to describe each model will be made explicit. The choice of dynamics parameters presented in this Chapter is taken arbitrarily wide, to illustrate the various modeling possibilities. However, for the numerical examples of Chapters 4, 5 and 6, only relevant sub-sets of these parameters will be analyzed.

²One gimbal angle for the 2D model, two on the 3D model

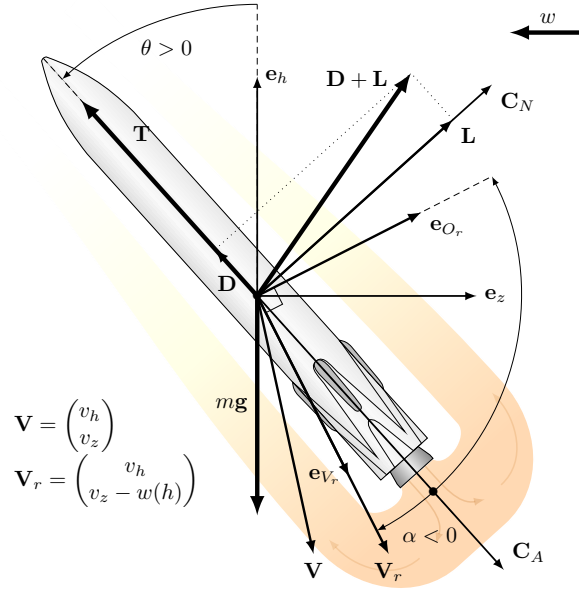


Figure 2.2: *Planar rocket model*. Axis, angles and forces. \mathbf{e}_{V_r} (respectively \mathbf{e}_{O_r}) denotes the unit vector parallel (resp. orthogonal) to \mathbf{V}_r .

2.2 Planar rocket model

The planar rocket model - a.k.a. 2D model - is described by its altitude h , its vertical speed v_h , its horizontal position z , its horizontal speed v_z and its total mass m . As mentioned above, $v_h \leq 0$.

The rocket is equipped with its own orthonormal frame $(\mathbf{C}_A, \mathbf{C}_N)$, where \mathbf{C}_A is parallel to the rocket body and oriented towards the engine, as pictured in Figure 2.2. To alleviate the writing, the vector pointing in the direction opposite to \mathbf{C}_A is noted $\mathbf{e}_A := -\mathbf{C}_A$.

The rocket orientation - or attitude - is defined by a single **signed** angle θ , formed by the angle between the vertical axis \mathbf{e}_h and the rocket main axis. Thus, when the rocket flies purely vertically, one has $\theta = 0^\circ$. The incidence is defined as the **signed** angle α between the relative speed and the vector \mathbf{C}_A . The unit vector associated to the relative speed is denoted \mathbf{e}_{V_r} . The unit vector orthogonal to \mathbf{e}_{V_r} is denoted \mathbf{e}_{O_r} , and is s.t. $(\mathbf{e}_{V_r}, \mathbf{e}_{O_r})$ is positively oriented, as shown in Figure 2.2. A geometric relation gives

$$\tan(\theta + \alpha) = \frac{v_z - w(h)}{|v_h|}.$$

As introduced in Section 2.1.1, the wind map is denoted $w(h)$ and parametrized by the three values $w_{z,0}$, $w_{z,1}$ and $w_{z,2}$. The equations of motion (EoM) are

$$\dot{h} = v_h \quad (2.3a)$$

$$\dot{v}_h = -g + \frac{L \sin \theta + (T + D) \cos \theta}{m} \quad (2.3b)$$

$$\dot{z} = v_z \quad (2.3c)$$

$$\dot{v}_z = \frac{L \cos \theta - (T + D) \sin \theta}{m} \quad (2.3d)$$

where

$$T := g \text{ISP} q_r - P_a(h) S_E, \quad (2.4a)$$

$$V_r := \sqrt{v_h^2 + (v_z - w(h))^2}, \quad (2.4b)$$

$$D := \frac{1}{2} \rho(h) V_r^2 S_{\text{ref}} C_{\text{Drag}}(M_a, \alpha), \quad (2.4c)$$

$$L := \frac{1}{2} \rho(h) V_r^2 S_{\text{ref}} C_{\text{Lift}}(M_a, \alpha). \quad (2.4d)$$

Adding the mass dynamics and using the fact that the vertical speed is assumed to be always negative (see Section 2.1.3.3), we can write the EoM

$$\begin{aligned} \dot{h} &= v_h \\ \dot{v}_h &= -g + \frac{((T + D)|v_h| + L v_z) \cos \alpha + ((T + D)v_z - L|v_h|) \sin \alpha}{V_r m} \\ \dot{z} &= v_z \\ \dot{v}_z &= \frac{-(T + D)v_z + F_L|v_h| \cos \alpha + ((T + D)|v_h| + L v_z) \sin \alpha}{V_r m} \\ \dot{m} &= -q_r \end{aligned}$$

The 2D rocket variables are

$$\begin{aligned} \text{(States)} \quad x &:= (h, v_h, z, v_z, m)^\top \in \mathbb{R}^5, \\ \text{(Controls)} \quad u &:= (q_r, \alpha)^\top \in \mathbb{R}^2, \\ \text{(Parameters)} \quad \eta &:= (\Delta \text{ISP}, w_{z,0}, w_{z,1}, w_{z,2})^\top \in \mathbb{R}^4, \end{aligned}$$

where the parameter ΔISP is incorporated into Equation (2.4a) s.t. it becomes

$$T = g(\text{ISP} + \Delta \text{ISP})q_r - S_E P(h).$$

This yields the dynamic function f_{2d} of the 2D rocket model:

$$\dot{x} = f_{2d}(x, u, \eta).$$

Remark 3. *Since the 2D rocket model serves illustrative purposes, it relies on the simplifying assumption that the engine flow dynamics (2.2) is instantaneous, and that q_c is its control variable. However, this engine dynamic will not be neglected in the 3D model.*

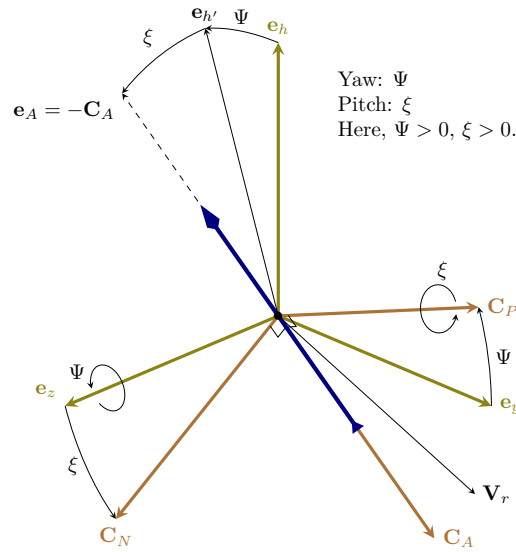


Figure 2.3: *Rocket orientation, based on yaw (Ψ) and pitch (ξ) angles.* First, $(\mathbf{e}_z, \mathbf{e}_y, \mathbf{e}_h)$ is rotated by Ψ around \mathbf{e}_z , giving $(\mathbf{e}_z, \mathbf{C}_P, \mathbf{e}_{h'})$. Then, the later is rotated by ξ around \mathbf{C}_P to give $(\mathbf{C}_N, \mathbf{C}_P, \mathbf{e}_A)$. Finally $\mathbf{C}_A = -\mathbf{e}_A$, leading to the orthonormal basis $(\mathbf{C}_N, \mathbf{C}_A, \mathbf{C}_P)$.

Normal acceleration

The normal acceleration a_{nor} is defined as the non-gravitational acceleration orthogonal to the relative speed. Thus, it equals

$$a_{\text{nor}} = \mathbf{e}_{O_r} \cdot \begin{pmatrix} \dot{v}_h + g \\ \dot{v}_z \end{pmatrix} \quad (2.6)$$

where \cdot is the inner product. For the 2D model, a_{nor} is **signed**.

2.3 Three-dimensional rocket model

The 3D rocket model has been designed to match the 2D model as closely as possible when the rocket trajectory remains in a plane. Thus, some concepts easily translate from one model to the other. Transposing the notions of incidence and attitude in 3D is, however, a delicate part.

First, we introduce a series of frames that are necessary to define the rocket orientation in 3D. Then, we express the aerodynamic model, and formulate the associated EoM. Finally, some comments on the specifics of the 3D model are provided.

2.3.1 Orientation frames

The rocket is axially symmetric, making the notion of roll irrelevant. Two angles are used to describe its orientation.

First, we introduce the rocket yaw and pitch using Euler angles, which enables us to define a frame attached to the rocket body. Then, this frame is re-written using projected angles, which are more convenient for our applications.

2.3.1.1 Rocket orientation frame

As shown in Figure 2.3, the Earth's frame is $(\mathbf{e}_z, \mathbf{e}_y, \mathbf{e}_h)$. The rocket, initially³ positively colinear to \mathbf{e}_h , is oriented using the yaw Ψ first, and then using the pitch ξ , as explained in Figure 2.3. Mathematically, this translates into

$$\begin{aligned}\mathbf{C}_P &:= \mathcal{R}(\mathbf{e}_z, \Psi) \mathbf{e}_y \\ \mathbf{C}_N &:= \mathcal{R}(\mathbf{C}_P, \xi) \mathbf{e}_z \\ \mathbf{e}_A &:= \mathcal{R}(\mathbf{C}_P, \xi) \mathcal{R}(\mathbf{e}_z, \Psi) \mathbf{e}_h \\ \mathbf{C}_A &:= -\mathbf{e}_A\end{aligned}$$

where the vectors $(\mathbf{C}_N, \mathbf{C}_A, \mathbf{C}_P)$ define a new direct orthonormal frame, attached to the rocket body. Expressed in the frame $(\mathbf{e}_z, \mathbf{e}_y, \mathbf{e}_h)$, the latter gives

$$\mathbf{C}_N = \begin{pmatrix} \cos \xi \\ \sin \xi \sin \Psi \\ -\sin \xi \cos \Psi \end{pmatrix}, \quad \mathbf{C}_A = \begin{pmatrix} -\sin \xi \\ \cos \xi \sin \Psi \\ -\cos \xi \cos \Psi \end{pmatrix}, \quad \mathbf{C}_P = \begin{pmatrix} 0 \\ \cos \Psi \\ \sin \Psi \end{pmatrix}. \quad (2.7)$$

2.3.1.2 Projected angles

The above-defined angles Ψ and ξ define the orientation of \mathbf{e}_A . When projected onto the plane $(\mathbf{e}_h, \mathbf{e}_z)$ (respectively $(\mathbf{e}_y, \mathbf{e}_h)$), the vector \mathbf{e}_A has an angle ζ_y (resp. ζ_z) with the vector \mathbf{e}_h . Note that, in terms of vector labeling, the angle on the plane $(\mathbf{e}_h, \mathbf{e}_z)$ corresponds to a rotation on the axis \mathbf{e}_y . The angles are illustrated in Figure 2.4. The expression of \mathbf{e}_A writes

$$\mathbf{e}_A = \frac{1}{\sqrt{1 + \tan^2 \zeta_z + \tan^2 \zeta_y}} \begin{pmatrix} \tan \zeta_y \\ -\tan \zeta_z \\ 1 \end{pmatrix} \quad (2.8)$$

and is valid for $|\zeta_z| < 90^\circ$ and $|\zeta_y| < 90^\circ$.

The steps used to derive formula (2.8) are as follows. Consider the square pyramid defined by a square on plane $(\mathbf{e}_z, \mathbf{e}_y)$ with its summit being at the top of \mathbf{e}_A . As shown in Figure 2.4, denote by v its height, and h_z (resp. h_y) its base length

³In the sense of the successive rotations defining the rocket body frame.

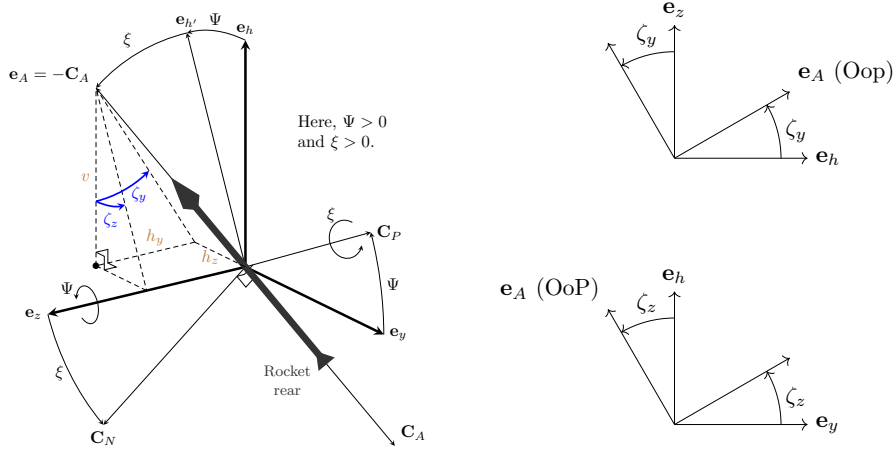


Figure 2.4: Angles ζ_z and ζ_y in the frame $(\mathbf{e}_z, \mathbf{e}_y, \mathbf{e}_h)$, for the proof of Equation (2.8).

orthogonal to \mathbf{e}_z (resp. \mathbf{e}_y). Here, v , h_z and h_y are taken unsigned. Then, using basic geometry, one has

$$1 = v^2 + h_z^2 + h_y^2, \quad \tan \zeta_z = \frac{h_z}{v}, \quad \tan \zeta_y = \frac{h_y}{v}$$

which gives

$$v = \frac{1}{\sqrt{1 + \tan^2 \zeta_z + \tan^2 \zeta_y}}, \quad h_z = v \tan \zeta_z, \quad h_y = v \tan \zeta_y.$$

Using the proper signs for \mathbf{e}_A , as shown in Figure 2.4, one has $\mathbf{e}_A = (h_y, -h_z, v)^\top$, hence (2.8).

Remark 4. Equation (2.7) is well defined whatever the values of Ψ and ξ . However, Equation (2.8) has a singular definition when $\zeta_z = 90^\circ$ or $\zeta_y = 90^\circ$. This raises multiple comments:

1. In all the scenarios studied in the thesis, only close-to-vertical trajectories are considered, where the rocket remains far from these singularities.
2. The singularity in Equation (2.8) is only a problem of definition. Indeed, \mathbf{e}_A can be extended by continuity everywhere, except when ζ_z and ζ_y equal $\pm 90^\circ$ simultaneously. For example, for a given $|\zeta_y| < 90^\circ$, then $\mathbf{e}_A(\zeta_z) \xrightarrow{\zeta_z \uparrow 90^\circ} \mathbf{e}_z$.
3. One could legitimately wonder whether a numerical method could fall into one of these singularities during intermediate computations. As far as this thesis is concerned, the guidance methods exposed below only require the evaluation of the dynamic function f and its derivatives along a prescribed reference trajectory before the flight (more details on this topic in Section 4.3). Since the evaluation of f is not required on-board, this singular definition at $\pm 90^\circ$ does not present any risk for our applications.

Equation (2.7) and (2.8) and $\mathbf{e}_A = -\mathbf{C}_A$ convey enough information to establish the change of variables between (Ψ, ξ) and (ζ_z, ζ_y) . By taking the ratio of the first two components of \mathbf{e}_A , the following relations are obtained

$$\Psi = \zeta_z \quad \text{and} \quad \tan \xi = \cos \Psi \tan \zeta_y. \quad (2.9)$$

Then, using the shortcuts $t_z := \tan \zeta_z$, $t_y := \tan \zeta_y$ and $t_{z,y} := \sqrt{1 + t_z^2 + t_y^2}$, we get

$$\mathbf{C}_N = \frac{1}{t_{z,y}} \begin{pmatrix} \frac{1}{\cos \zeta_z} \\ t_y \sin \zeta_z \\ -t_y \cos \zeta_z \end{pmatrix}, \quad \mathbf{C}_A = \frac{1}{t_{z,y}} \begin{pmatrix} -t_y \\ t_z \\ -1 \end{pmatrix} \quad \text{and} \quad \mathbf{C}_P = \begin{pmatrix} 0 \\ \cos \zeta_z \\ \sin \zeta_z \end{pmatrix}. \quad (2.10)$$

2.3.2 Dynamics

To express the aerodynamic forces, the orientation of the relative speed vector has to be defined. Then, lift and drag vectors are formulated. Finally, the dynamic equations are expressed.

2.3.2.1 Relative speed definition

To describe the orientation of the relative speed vector w.r.t. the rocket body, we introduce the incidence variables for the 3D model.

The speed vector is $\mathbf{V} := (\dot{z}, \dot{y}, \dot{h})^\top = (v_z, v_y, v_h)^\top$. The horizontal wind vector is $\mathbf{w} := (w_z(h), w_y(h), 0)^\top$. Thus, the relative speed is

$$\mathbf{V}_r := \mathbf{V} - \mathbf{w} = \begin{pmatrix} v_z - w_z(h) \\ v_y - w_y(h) \\ v_h \end{pmatrix}.$$

The unit vector of the relative speed vector \mathbf{V}_r is denoted \mathbf{e}_{V_r} . We can define the orientation of \mathbf{e}_{V_r} using projected angles. The angles α_y and α_z are introduced s.t. the angles defining the position of \mathbf{e}_{V_r} w.r.t. the base frame using projected angles are $\zeta_y + \alpha_y$ and $\zeta_z + \alpha_z$. They are represented in Figure 2.5. Since \mathbf{e}_{V_r} is defined equivalently as \mathbf{e}_A in Equation (2.8), its expression is

$$\mathbf{e}_{V_r} = \frac{1}{\sqrt{1 + \tan^2(\alpha_z + \zeta_z) + \tan^2(\alpha_y + \zeta_y)}} \begin{pmatrix} -\tan(\alpha_y + \zeta_y) \\ \tan(\alpha_z + \zeta_z) \\ -1 \end{pmatrix}. \quad (2.11)$$

Then, knowing the expressions of \mathbf{C}_A and \mathbf{e}_{V_r} , we can define the incidence as the **unsigned** angle between these vectors, leading to the expression

$$\alpha = \arcsin \left(\frac{\sqrt{(T_z - t_z)^2 + (T_y - t_y)^2 + (T_z t_y - T_y t_z)^2}}{\sqrt{(1 + t_z^2 + t_y^2)(1 + T_z^2 + T_y^2)}} \right). \quad (2.12)$$

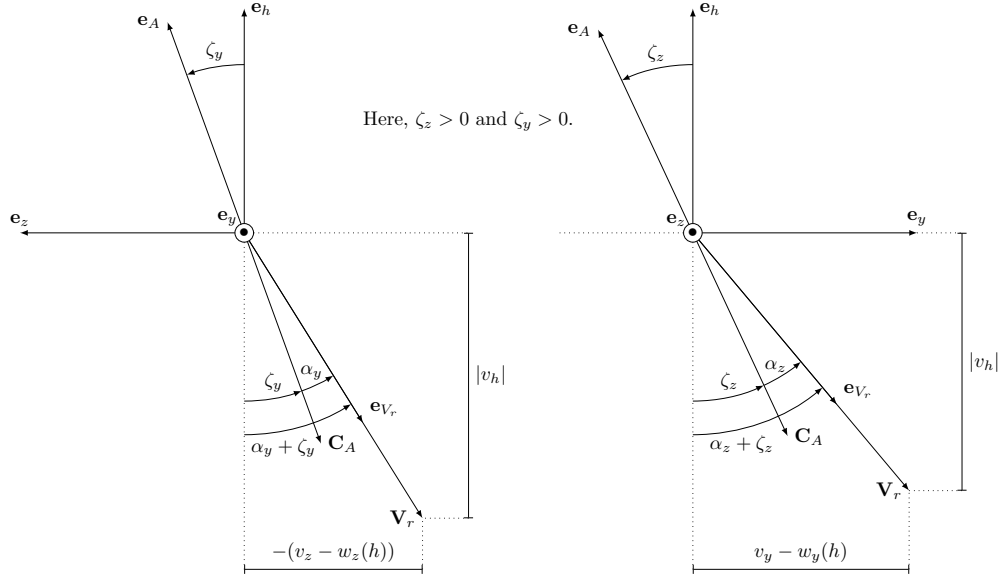


Figure 2.5: Representation of the projected angles: $\zeta_z, \zeta_y, \alpha_z, \alpha_y$. Note that $\mathbf{C}_A, \mathbf{e}_{V_r}$ and \mathbf{V}_r are not coplanar to $(\mathbf{e}_z, \mathbf{e}_h)$ nor $(\mathbf{e}_y, \mathbf{e}_h)$.

where $t_z = \tan \zeta_z$, $t_y = \tan \zeta_y$, $T_z = \tan(\alpha_z + \zeta_z)$ and $T_y = \tan(\alpha_y + \zeta_y)$, to alleviate the writing. This expression stems from Equations (2.10) and (2.11) and

$$\|\mathbf{C}_A \times \mathbf{e}_{V_r}\| = \|\mathbf{C}_A\| \|\mathbf{e}_{V_r}\| \sin \widehat{\mathbf{C}_A, \mathbf{e}_{V_r}} = \sin \alpha$$

where

$$\mathbf{C}_A \times \mathbf{e}_{V_r} = \frac{1}{\sqrt{(1+t_z^2+t_y^2)(1+T_z^2+T_y^2)}} \begin{pmatrix} T_z - t_z \\ t_y - T_y \\ T_y t_z - T_z t_y \end{pmatrix}.$$

where \times is the cross product. Then, from the definition of \mathbf{V}_r and the definition of the projected angles, we have

$$\tan(\alpha_z + \zeta_z) = \frac{v_y - w_y(h)}{|v_h|} \quad \text{and} \quad \tan(\alpha_y + \zeta_y) = -\frac{v_z - w_z(h)}{|v_h|}. \quad (2.13)$$

Note that, in Equation (2.14), the “z” indices in the angles correspond to the “y” indices in the speed (and vice-versa). Also, beware of the fact that the signs are different. Inverting the previous relations w.r.t. ζ_z and ζ_y yields

$$\zeta_z = -\alpha_z + \arctan \frac{v_y - w_y(h)}{|v_h|} \quad \text{and} \quad \zeta_y = -\alpha_y - \arctan \frac{v_z - w_z(h)}{|v_h|}. \quad (2.14)$$

2.3.2.2 Aerodynamic effects

Lift and drag can be defined using $(\mathbf{C}_N, \mathbf{C}_A, \mathbf{C}_P)$. Indeed, the drag \mathbf{D} is colinear to \mathbf{C}_A , in the opposite direction to \mathbf{V}_r . Only low-incidence flight is considered, and \mathbf{D} is positively colinear with $-\mathbf{C}_A$.

Moreover, in consistency with the aerodynamic model described in Section 2.1.2, the axial symmetry of the 3D rocket model implies that \mathbf{L} , \mathbf{C}_A and \mathbf{V}_r are linearly dependent, and that \mathbf{L} must belong to the plane $(\mathbf{C}_P, \mathbf{C}_N)$. Therefore, as illustrated in Figure 2.6, one can define the lift and drag vectors as

$$\mathbf{L} = L \mathbf{e}_L \quad \text{and} \quad \mathbf{D} = -D \mathbf{C}_A,$$

where the magnitudes L and D equal

$$L = \frac{1}{2} \rho(h) V_r^2 S_{\text{ref}} C_{\text{Lift}}(M_a, \alpha) \quad \text{and} \quad D = \frac{1}{2} \rho(h) V_r^2 S_{\text{ref}} C_{\text{Drag}}(M_a, \alpha).$$

The direction \mathbf{e}_L of the lift requires further attention, since the lift orientation is defined only when the incidence is not zero. When it is well defined, the vector \mathbf{e}_L is a unit vector, positively colinear to the projection of $-\mathbf{e}_{V_r}$ lying in the plane $(\mathbf{C}_P, \mathbf{C}_N)$. Thus, it can be expressed as

$$\mathbf{e}_L := -\frac{(\mathbf{e}_{V_r} \cdot \mathbf{C}_N) \mathbf{C}_N + (\mathbf{e}_{V_r} \cdot \mathbf{C}_P) \mathbf{C}_P}{\|(\mathbf{e}_{V_r} \cdot \mathbf{C}_N) \mathbf{C}_N + (\mathbf{e}_{V_r} \cdot \mathbf{C}_P) \mathbf{C}_P\|} = -\frac{\mathbf{e}_{V_r} - (\mathbf{e}_{V_r} \cdot \mathbf{C}_A) \mathbf{C}_A}{\|\mathbf{e}_{V_r} - (\mathbf{e}_{V_r} \cdot \mathbf{C}_A) \mathbf{C}_A\|} \quad (2.15)$$

when \mathbf{e}_{V_r} is not colinear to \mathbf{C}_A and $\mathbf{e}_L = \mathbf{0}$ otherwise. This apparent discontinuity is actually not troublesome. Indeed, for $\alpha > 0$, the vector \mathbf{e}_L can be equivalently defined as

$$\mathbf{e}_L = \mathbf{C}_A \times \frac{\mathbf{C}_A \times \mathbf{e}_{V_r}}{\|\mathbf{C}_A \times \mathbf{e}_{V_r}\|} = \frac{\mathbf{C}_A \times (\mathbf{C}_A \times \mathbf{e}_{V_r})}{\sin \alpha}$$

which yields the following expression for the lift

$$\mathbf{L} = L \mathbf{e}_L = \frac{1}{2} \rho(h) V_r^2 S_{\text{ref}} \frac{C_{\text{Lift}}(M_a, \alpha)}{\sin \alpha} \mathbf{C}_A \times (\mathbf{C}_A \times \mathbf{e}_{V_r}).$$

For any fixed Mach number M_a , the map $\alpha \mapsto C_{\text{Lift}}(M_a, \alpha)$ is assumed continuously differentiable and it equals zero at $\alpha = 0$. Thus, the ratio $\frac{C_{\text{Lift}}(M_a, \alpha)}{\sin \alpha}$ remains bounded when α tends to zero. Also, $\mathbf{C}_A \times \mathbf{e}_{V_r}$ tends to zero when \mathbf{e}_{V_r} tends to \mathbf{C}_A . Consequently, the expression of \mathbf{e}_L does not matter when $\alpha = 0$, which is a false singularity.

Remark 5. Here, *the coefficient C_{Lift} is positive and only evaluated for positive values of α . However, note that in the 2D model, $\alpha \mapsto C_{\text{Lift}}(M_a, \alpha)$ is taken odd for any fixed Mach number M_a and is evaluated on signed values of α .*

2.3.2.3 Rocket dynamics in 3D

With $\mathbf{g} = (0, 0, -g)^\top$ denoting the gravity vector, the acceleration vector \mathbf{a} equals

$$\mathbf{a} := \frac{d}{dt} \mathbf{V} = \mathbf{g} + \frac{\mathbf{T} + \mathbf{L} + \mathbf{D}}{m}$$

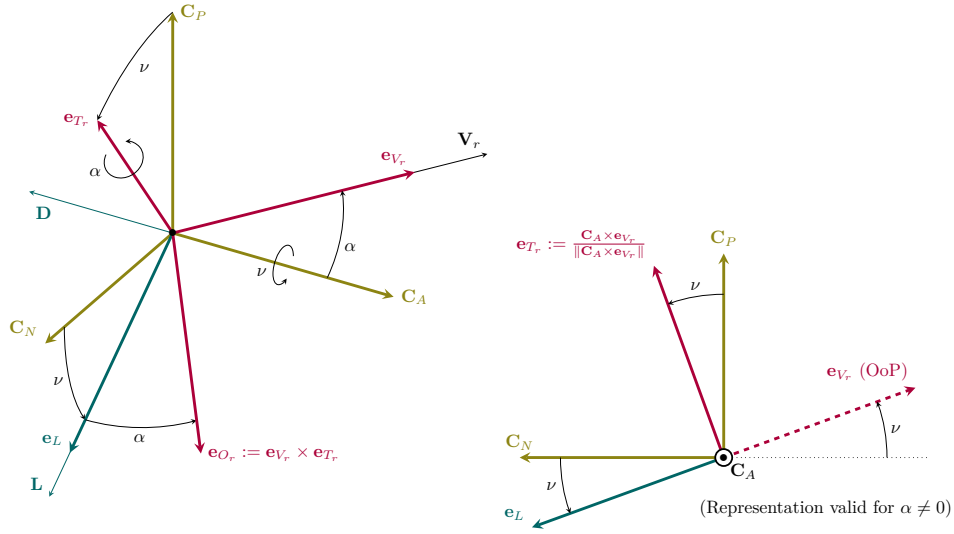


Figure 2.6: Relations between the lift, the drag and the relative speed vectors. Here, the angle ν denotes the oriented angle between \mathbf{C}_N and \mathbf{L} , defined only for $\alpha \neq 0$.

where the thrust vector is $\mathbf{T} = -T\mathbf{C}_A$ and its magnitude T is defined in Equation (2.1).

The dynamics parameters conveyed by the variable η are the ISP via ΔISP , multiplicative factors for the aerodynamic coefficients (m_L , m_D) and the wind parameters ($w_{z,0}, w_{z,1}, w_{z,2}$) and ($w_{y,0}, w_{y,1}, w_{y,2}$) defined in Section 2.1.1. The first three parameters must be incorporated in the equations s.t.

$$\begin{aligned} T = g\text{ISP}q_r - S_E P(h) & \text{ becomes } g(\text{ISP} + \Delta\text{ISP})q_r - S_E P(h) \\ L = \frac{1}{2}\rho(h)V_r^2 S_{\text{ref}} C_{\text{Lift}}(M_a, \alpha) & \text{ becomes } \frac{1}{2}\rho(h)V_r^2 S_{\text{ref}}(1 + m_L)C_{\text{Lift}}(M_a, \alpha) \\ D = \frac{1}{2}\rho(h)V_r^2 S_{\text{ref}} C_{\text{Drag}}(M_a, \alpha) & \text{ becomes } \frac{1}{2}\rho(h)V_r^2 S_{\text{ref}}(1 + m_D)C_{\text{Drag}}(M_a, \alpha) \end{aligned}$$

It allows us to define the 3D rocket variables as

$$\begin{aligned} (\text{States}) \quad x &:= (z, y, h, v_z, v_y, v_h, m, q_r)^\top \in \mathbb{R}^8, \\ (\text{Controls}) \quad u &:= (q_r, \alpha_z, \alpha_y)^\top \in \mathbb{R}^3, \\ (\text{Parameters}) \quad \eta &:= (\Delta\text{ISP}, m_L, m_D, w_{z,0}, w_{z,1}, w_{z,2}, w_{y,0}, w_{y,1}, w_{y,2})^\top \in \mathbb{R}^9 \end{aligned}$$

Then, written using blocks, the dynamic equation equals

$$\dot{x} = f_{3d}(x, u, \eta) = \begin{pmatrix} \mathbf{V} \\ \mathbf{g} + \frac{\mathbf{T} + \mathbf{L} + \mathbf{D}}{m} \\ -q_r \\ \frac{q_c - q_r}{\tau_q} \end{pmatrix}. \quad (2.16)$$

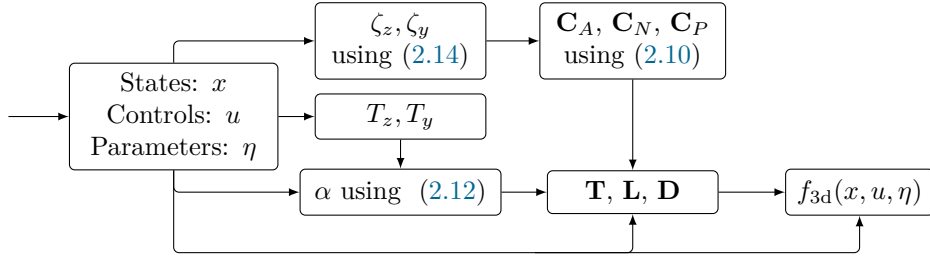


Figure 2.7: Summary of the workflow computing $f_{3d}(x, u, \eta)$.

The sequential method used to compute all the expressions involved in f_{3d} is summarized in Figure 2.7. Its implementation has served to produce all the numerical results of the thesis.

Remark 6. *The derivatives of f_{3d} are computed using various methods depending on the variables considered, to fasten their evaluation. First, note that the expression of f_{3d} does not depend on z and y . Moreover, its derivative w.r.t. m is simple and is computed analytically. Finally, finite difference is used to compute the derivatives w.r.t. the other variables.*

2.3.3 Normal acceleration, downrange and attitude

As it will be needed later in our work, we now focus on the acceleration component normal to the relative speed vector. The unsigned normal acceleration $a_{\text{nor}}^{u.s.}$ is defined as the norm of the part of the non-gravitational acceleration vector normal to the relative speed. Denoting $\mathbf{F} = \mathbf{T} + \mathbf{L} + \mathbf{D}$ yields

$$a_{\text{nor}}^{u.s.} = \frac{\|\mathbf{F} - (\mathbf{F} \cdot \mathbf{e}_{V_r})\mathbf{e}_{V_r}\|}{m}.$$

Compared to the 2D model, this expression is naturally unsigned. However, the norm in this expression may bring differentiation issues. Indeed, it will be needed to differentiate this term later when considering optimization problems (see e.g. (4.6g) in Chapter 4). Instead of $a_{\text{nor}}^{u.s.}$, we consider an alternate expression that remains signed (and thus differentiable), see below Equation (2.17).

First remark that, according to the aerodynamic model and as shown in Figure 2.6, \mathbf{T} , \mathbf{D} , \mathbf{L} , \mathbf{e}_{V_r} and \mathbf{e}_L all belong to the plane $(\mathbf{e}_{V_r}, \mathbf{e}_L)$, or equivalently to the plane $(\mathbf{e}_{O_r}, \mathbf{e}_{V_r})$, where \mathbf{e}_{O_r} and \mathbf{e}_{V_r} are orthogonal. Consequently, the term $\mathbf{F} - (\mathbf{F} \cdot \mathbf{e}_{V_r})\mathbf{e}_{V_r}$ only has a component along \mathbf{e}_{O_r} , which yields

$$\|\mathbf{F} - (\mathbf{F} \cdot \mathbf{e}_{V_r})\mathbf{e}_{V_r}\| = |\mathbf{F} \cdot \mathbf{e}_{O_r}|$$

Using the latter expression with the definitions of \mathbf{T} , \mathbf{D} , \mathbf{L} , \mathbf{e}_{V_r} , \mathbf{e}_L and \mathbf{C}_A yields

$$a_{\text{nor}} := \frac{L \cos \alpha - (T + D) \sin \alpha}{m}. \quad (2.17)$$

s.t. $a_{\text{nor}}^{u.s.} = |a_{\text{nor}}|$. Thanks to Equation (2.17), the 3D model also has a term a_{nor} describing the normal acceleration that is **signed**.

2.3.3.1 Downrange

The downrange, or distance of the vertical projection of the rocket to the landing site, equals $d := \sqrt{z^2 + y^2}$.

2.3.3.2 Attitude

The attitude θ is defined as the angle between \mathbf{e}_h and \mathbf{e}_A . It is **unsigned** (contrary to the planar rocket model). Its expression equals

$$\theta := \arcsin \sqrt{\sin^2 \xi + \cos^2 \xi \sin^2 \Psi}. \quad (2.18)$$

2.4 PDG as an Optimal Control Problem

First, we present the general PDG objectives and then write the main OCPs.

2.4.1 Mission goals and constraints

In this sub-section, we present all the constraints of our PDG problem.

2.4.1.1 Landing site target

It is desired to find a trajectory - i.e. states $x(t)$ and controls $u(t)$ - that steers the rocket from a given initial condition x^0 to the landing site in a time t_f . Assuming that the landing site is located at the origin of our coordinate system, the following end-point constraints are introduced

$$\begin{aligned} z(t_f) &= 0 \\ y(t_f) &= 0 \quad (\text{For the 3D model only}) \\ h(t_f) &= 0 \\ v_z(t_f) &= 0 \\ v_y(t_f) &= 0 \quad (\text{For the 3D model only}) \\ v_h(t_f) &= -\varepsilon_v^f \end{aligned}$$

Note that a small non-zero vertical speed at landing ($\varepsilon_v^f > 0$) is desired. Among others, this serves to avoid singularity of incidence at t_f and gives a margin regarding the thrust dominance assumption⁴. These conditions can be written as a linear equality constraint

$$A^f x(t_f) = b^f$$

where A^f is filled with zeros and ones only, and b^f contains zeros and $-\varepsilon_v^f$ only.

⁴If the rocket reaches $v_h = 0$ before $h = 0$, the engine must be stopped, since hovering is prevented by the thrust dominance assumption. Thus, seeking $v_h(t_f) = -\varepsilon_v^f$ leaves a small safety margin.

2.4.1.2 Mechanical bounds

As noted in Section 2.1.3, the rocket has several mechanical limitations. Its engine flow magnitude and rate of change are bounded. These impose that the real and controlled flows satisfy

$$q^- \leq q_r \leq q^+ \quad \text{and} \quad q^- \leq q_c \leq q^+, \quad (2.19)$$

where $q^+ > q^- > 0$.

Remark 7. *It should be noted that in the literature, the decision variables often considered (especially for non-atmospheric missions) are the components of the thrust vector \mathbf{T} itself. The bounded flow constraints are then taken into account through the constraint $T_{\min} \leq \|\mathbf{T}\| \leq T_{\max}$. In the latter, the lower bound defines an artificially non-convex constraint. Lossless convexification [2, 27], a method using an intermediate slack variable, is often used to overcome this problem. However, with our modeling choices, picking the engine flow as one of the decision variables makes the constraints described in Equation (2.19) convex, since the constraints of the shape “ $q^- \leq q \leq q^+$ ” are used for scalar values of q .*

2.4.1.3 Bounded mass

The fuel tank being finite, there are upper and more importantly lower bounds on the mass

$$m_{\text{dry}} \leq m \leq m_{\text{wet}}. \quad (2.20)$$

2.4.1.4 Safety bounds

For safety reasons, it is also desirable to remain within limited incidences. This constraint is one of the most important difference between planetary and atmospheric landing. It writes

$$|\alpha| \leq \alpha_{\max}.$$

This constraint has a straightforward interpretation for the 2D model, since it only implies a single control variable, α itself. However, it is more intricate for the 3D model, since α is defined through Equation (2.12) and depends non-linearly on state and control variables. We choose to impose the following constraints for the 3D model

$$|\alpha_z| \leq \alpha_{\max} \quad \text{and} \quad |\alpha_y| \leq \alpha_{\max},$$

which is not equivalent but conservative.

Moreover, since the guidance trajectory must be tracked by the underlying rocket control system, it is necessary that this trajectory does not exceed prescribed thresholds of normal accelerations. Thus, we impose

$$|a_{\text{nor}}| \leq a_{\text{nor}}^{\max}$$

where a_{nor} is defined in Equation (2.6) for the 2D rocket model, and in Equation (2.17) for the 3D one.

2.4.1.5 Constraints not considered

Other types of constraints can be found in the literature, such as⁵:

- Landing cone (a.k.a. glideslope) constraints [1, 35, 98],
- Pointing (a.k.a. attitude) constraints [35],
- Thermal flux (a.k.a. heating rate) constraints [22, 106, 107],
- Dynamic pressure constraints [22, 107].

However, as detailed below, these constraints are not considered here, though this would be possible as natural extensions.

Landing cone constraints are not considered due to the thrust dominance assumption. Indeed, the class of rockets studied in this thesis naturally performs landing trajectories with a high slenderness ratio. The same motivation rules out the pointing constraints.

Thermal flux constraints are critical mechanical requirements for re-entry problems at hypersonic speeds, when the vehicle directly relies on the atmosphere to brake [20]. Since the speeds involved for our landing scenarios are high (approximately between 0 and Mach 2) but not hypersonic⁶ and considering the unusual air flow around the rocket, as depicted in Figure 2.1, the thermal flux constraints are not needed. The same arguments apply for the dynamic pressure constraint.

2.4.2 Formulation as an optimal correction problem

The PDG problem is formulated as an OCP in free-final time, w.r.t. a reference trajectory.

Regarding the notations, whatever the chosen model is (2D or 3D), the dynamic function is noted f . The lower and upper control bounds are respectively denoted u^- and u^+ . The mixed state-control constraints are conveyed by a function c .

As mentioned in the Introduction, we consider that all of the above-mentioned constraints can be tuned by a parameter p that allows one to adjust their nominal value. For example, let us say that we parametrize the incidence and the normal acceleration bounds in the planar rocket model, i.e. $p = (p_1, p_2)^\top = (\Delta\alpha_{\max}, \Delta a_{\text{nor}}^{\max})^\top$. Then, the parameterized constraints become

$$|\alpha| \leq \alpha_{\max} + p_1 \quad \text{and} \quad |a_{\text{nor}}| \leq a_{\text{nor}}^{\max} + p_2.$$

From a general point of view, we say that all the constraints are parameterized by p . Therefore, the dependency on p of the right-hand side vector b^f , the control bounds u^- and u^+ , and the mixed state-control constraint c will be highlighted whenever necessary. It is assumed that $p = 0$ for a nominal landing. The exact choice of p varies depending on the mission needs, and will be discussed extensively in Chapter 5.

⁵Both pointing and landing cone constraints are well discussed in [58, Fig.19] for instance.

⁶Note that subsonic, supersonic and hypersonic speeds are usually defined as below Mach 1, between Mach 1 and Mach 5, and above Mach 5.

Let us consider a *reference trajectory*, which is described as a quadruplet $(\bar{x}, \bar{u}, \bar{\eta}, \bar{t}_f)$. Such a trajectory can be computed offline, for any given mission, using well-known and possibly time-consuming numerical methods [24]. It is assumed that this trajectory satisfies the constraints, i.e. that it satisfies the control constraints and $c(\bar{x}(t), \bar{u}(t), \bar{\eta}, p) \leq 0$ for all times, and that it is dynamically feasible, i.e. that it satisfies the Initial Value Problem (IVP)

$$\begin{cases} \bar{x}(0) = \bar{x}^0, \\ \dot{\bar{x}}(t) = f(\bar{x}(t), \bar{u}(t), \bar{\eta}), \quad \forall t \in [0, \bar{t}_f]. \end{cases} \quad (2.21)$$

At the beginning of the final burn, the gap between the current state and the current dynamics parameters and their reference values is denoted Δx^0 and $\Delta \eta$. They are conveyed by the input variable⁷ ξ .

Knowing ξ , and the value of p , the mathematical goal of PDG, when formulated w.r.t. this reference trajectory, is to find a control correction δu and a time-of-flight correction Δt_f making the rocket land while satisfying all the constraints and minimizing a certain performance index⁸ \mathcal{J} . For completeness, note that δu belongs to a functional space defined over $[0, \bar{t}_f + \Delta t_f]$. We will consider that this space⁹ equals $\mathcal{U}(\Delta t_f) := L^\infty([0, \bar{t}_f + \Delta t_f], \mathbb{R}^m)$ for the definition below, although we will restrict the problem to a much more specific class of control corrections in Chapter 4.

Definition 1 (Infinite dimensional problem, PDG (ξ, p)). *Given a reference described by its quadruplet $(\bar{x}, \bar{u}, \bar{\eta}, \bar{t}_f)$, where \bar{u} is defined over $[0, \bar{t}_f]$, find the optimal time-of-flight change Δt_f and the optimal control correction $\delta u \in \mathcal{U}(\Delta t_f)$ for the optimization problem PDG (ξ, p) defined by*

$$\min_{\delta u, \Delta t_f} \mathcal{J}(\delta u, \Delta t_f) \quad (2.22a)$$

$$s.t. \quad \dot{x}(t) = f(x(t), \bar{u}(t \cdot \bar{t}_f / (\bar{t}_f + \Delta t_f)) + \delta u(t), \bar{\eta} + \Delta \eta), \quad (2.22b)$$

$$x(0) = \bar{x}^0 + \Delta x^0 \quad (2.22c)$$

$$A^f x(\bar{t}_f + \Delta t_f) = b^f(p) \quad (2.22d)$$

$$u^-(p) \leq \bar{u}(t \cdot \bar{t}_f / (\bar{t}_f + \Delta t_f)) + \delta u(t) \leq u^+(p), \quad (2.22e)$$

$$c(x(t), \bar{u}(t \cdot \bar{t}_f / (\bar{t}_f + \Delta t_f)) + \delta u(t), \eta + \Delta \eta, p) \leq 0, \quad (2.22f)$$

where conditions (2.22b), (2.22e) and (2.22f) are meant for all $t \in [0, \bar{t}_f + \Delta t_f]$.

Remark 8. \mathcal{J} is assumed strictly convex. Moreover, for null inputs (i.e. $\Delta x^0 = 0$ and $\Delta \eta = 0$) it has a null minimum (i.e. $\delta u^* = 0$ and $\Delta t_f^* = 0$).

Remark 9. Written this way, the formulation of PDG (ξ, p) suggests that the bounded mass constraint is enforced over the whole interval $[0, t_f]$. However, since the engine flow is positive, the mass is decreasing, and thus it is only necessary to enforce the simpler condition $m(\bar{t}_f + \Delta t_f) \geq m_{\text{dry}}$ in practice.

⁷The exact definition of ξ will be detailed later, in Equation (4.4).

⁸The exact performance index used in this thesis is detailed in the examples of Chapter 4.

⁹ L^∞ denotes the set of essentially bounded measurable functions.

Written in this format, $\text{PDG}(\xi, p)$ falls into the field of perturbation methods for OCPs [11, 32], which is the ground base of Chapter 4. However, this problem is studied for any values of Δx^0 and $\Delta \eta$, because we are interested in the solutions of this problem even for non small values of these inputs.

Summary

In this chapter, we have defined dynamic models of the rocket, for 2D and 3D motions. The aerodynamic model is the main difference between the planetary landing problems studied in the literature and atmospheric landing problems. The general, infinite-dimensional version, of the PDG problem has been defined as finding the optimal correction $(\delta u, \Delta t_f)$ w.r.t. a reference trajectory, while satisfying all the problem constraints, for given values of ξ and p .

A simpler version of $\text{PDG}(\xi, p)$, considering only vertical motion, is considered in Chapter 3 (which can be skipped without loss of continuity), and Chapter 4 presents a method to compute an approximation of the solution of the general problem.

Chapter 3

Mathematical properties of the optimal vertical descent

Résumé

Ce chapitre se concentre sur un cas particulier du problème PDG présenté précédemment : l'optimisation de la consommation de carburant pour l'atterrissage vertical. Comme nous l'avons déjà vu, le rapport entre la translation latérale et la translation verticale est faible. En tant que cas limite, il est tentant d'étudier d'abord le problème purement vertical. Se concentrer sur le problème purement vertical permet plusieurs simplifications : il n'y a plus qu'une seule variable de décision (le débit du moteur) et le modèle aérodynamique est grandement simplifié. Cela rend l'analyse analytique du problème envisageable.

Le problème de l'atterrissage atmosphérique vertical optimal en termes de carburant est ici étudié en tant que problème de Commande Optimale en temps final libre. La principale contribution établit la nature de la loi de poussée optimale en consommation de carburant, en étendant les résultats de la littérature sur les problèmes sans atmosphère. Des conditions suffisantes et nécessaires sont fournies qui garantissent la nature Min-Max des extremums normaux. Il est également démontré que les extremales anormales - au sens du principe du maximum de Pontryagin (PMP) - sont soit Min, soit Max. Un sous-produit utile de cette étude est une caractérisation de l'ensemble atteignable pour les atterrissages verticaux. Cette notion sera réutilisée plus tard dans le chapitre 6 à des fins d'évaluation des performances.

This chapter focuses on a special case of the PDG problem presented previously: the fuel-optimal vertical landing. As already discussed, the ratio of lateral vs vertical translation is small. As a limit case, it is tempting to study the limit case of the purely vertical problem first. Focusing on the purely vertical problem enables several simplifications: there is only one decision variable left (the engine flow) and the aerodynamic model is greatly simplified. This makes the analytic analysis of the problem tractable.

The vertical fuel-optimal vertical atmospheric landing problem is here studied

as a free-final time OCP. The main contribution establishes the nature of the fuel-optimal thrust program, extending results from the literature on atmosphere-free problems. Sufficient and necessary conditions are provided that guarantee the Min-Max nature of the normal extremals. Abnormal extremals - in the sense of the Pontryagin Maximum Principle (PMP) - are also shown to be either Min or Max. A useful by-product of this study is a characterization of the reachable set for vertical landings. This notion will be re-used later in Chapter 6 for performance assessment purposes.

If necessary, the reader can skip directly to page 49 for a summary of the important results. Also, note that this chapter is a detailed version of [65].

3.1 Vertical descent

Historically, Meditch [64] and then Shi & Eckstein [88] have offered analytic solutions for the (atmosphere-free) vertical Moon landing problem. Since then, due to the spectacular development of reusable launcher technologies, powered landing strategies have been successfully addressed using numerical methods [16, 22, 54, 79, 98, 100]. Due to the non-negligible effects of the atmosphere, the analytic results derived for the Moon landing problem can not be directly adapted to the problem of Earth landing. Yet, analytical results on this problem would still represent valuable assets. On the one hand, analytic solutions are very useful to assess the quality of the numerical methods, by providing well-described reference solutions to standardized problems, see *e.g.* [20, 33, 42, 43, 77]. Further, when analytical investigations establish the switching structure of the solution, very efficient numerical methods can be employed, using a reduced number of unknown variables [33, 77, 99]. For complex dynamics and high-dimensional systems, obtaining such analytical results is usually considered as out-of-reach [84]. It is thus of importance to select only dominant factors while leaving out unnecessary details in the modeling. Following this *modus operandi*, we consider a simplified (but not simplistic) representation of the general powered landing problem and establish a non-trivial result.

The analysis presented in this chapter considers one key element: the effects of atmosphere. The model under study builds upon the variable-mass model of a rocket considered in [64] and incorporates atmospheric effects in the form of an altitude-dependent bias of the thrust only, as introduced in Equation (2.1) in Chapter 2. In this model of the final phase of the powered landing, the thrust generator is always turned on¹ and the thrust is upper and lower-bounded in a way that prevents hovering flight (according to the Thrust dominance assumption already presented).

Intuitively, one could expect that it is more efficient to wait until the last feasible moment to use maximal thrust, as early efforts trying to slow down the rocket are likely to be less effective due to the varying mass scaling of the dynamics. The contribution of this chapter is to establish conditions under which fuel-optimal vertical powered landing through the atmosphere is indeed of this expected Min-Max nature.

The arguments of proof are as follows. Under simple assumptions on the atmosphere pressure model (decreasingness, convexity), the optimal thrust program is first shown to have a Max-Min-Max structure, based on the PMP. Compared to [64],

¹Note that the ignition time optimization is a different topic.

some sharper differential inequalities on the adjoint states are necessary to obtain a conclusion. Also, both normal and abnormal extremals need to be tackled. Then, using additional inequality constraints derived from the Implicit Function Theorem (IFT), Min-Max structures are proven to be more fuel-optimal than Max-Min-Max structures. These conditions can be checked numerically, over a finite domain. It is also shown that these conditions hold for zero atmosphere (and scarce atmosphere, using a continuity argument), which makes a connection with [64].

The chapter is organized as follows. In Section 3.1, the dynamics and the powered landing problem are summarized, in harmony with Chapter 2. In Section 3.2, the flight envelope is described based on flow analysis and differential inequalities. In Section 3.3, the optimal thrust program is shown to be Min-Max using the PMP, the IFT and mild assumptions. Finally, we provide numerical details in Section 3.4, and concluding remarks in Section 3.5.

3.1.1 Single dimensional rocket model

Following Chapter 2, we describe the rocket having a purely vertical motion by its altitude h , speed v and total mass m . The dynamics write

$$\dot{h} = v, \quad \dot{v} = -g + \frac{T(h, q)}{m}, \quad \dot{m} = -q$$

where q is the engine flow, and the thrust T defined in Equation (2.1). Because the engine is firing, the sole effect of the atmosphere is through the atmospheric pressure in the expression of T . Recall that negative speed conveys descending movement. During the powered landing, the rocket engine is always firing and q is bounded.

In the problem setup under consideration, the engine flow bounds are s.t. the net thrust is always positive, *i.e.* q^- is s.t.

$$\underline{a}_{cc} := g \text{ISP} q^- - S_E \max_{h \geq 0} P_a(h) > 0. \quad (3.1)$$

3.1.1.1 Normalized dynamics

The following normalized variables are introduced

$$u := 2 \frac{q - q^-}{q^+ - q^-} - 1, \quad y_1 := \frac{h}{g \text{ISP}}, \quad y_2 := \frac{v}{g \text{ISP}}, \quad y_3 := \frac{2m}{q^+ - q^-}$$

where $y := (y_1, y_2, y_3)^\top$ denotes the normalized² states and where

$$\kappa := \frac{1}{\text{ISP}}, \quad r := \frac{q^+ + q^-}{q^+ - q^-}, \quad \pi(y_1) := P_a(g \text{ISP} y_1) \frac{2S_E}{g \text{ISP} (q^+ - q^-)}.$$

This yields the control-affine dynamics in \mathbb{R}^3

$$\dot{y} = f(y) + ug(y), \quad (3.2)$$

²In the numerical examples of Chapter 4, 5 and 6, the model is normalized for numerical stability, whereas, in this chapter, it aims at simplifying the writing of the dynamics.

where $|u| \leq 1$ and

$$\text{(Altitude)} \quad \dot{y}_1 = y_2 \quad (3.3a)$$

$$\text{(Speed)} \quad \dot{y}_2 = \frac{r + u - \pi(y_1)}{y_3} - \kappa \quad (3.3b)$$

$$\text{(Mass)} \quad \dot{y}_3 = -(r + u). \quad (3.3c)$$

Also, recall that the mass is bounded s.t. $m^- \leq y_3 \leq m^+$.

3.1.2 Assumptions specific to the vertical descent

The problem under study is also described by the two following assumptions.

Assumption 1 (Pressure model properties). *The normalized pressure function π , is of class C^2 , and $\pi > 0$, $\pi' < 0$, $\pi'' > 0$.*

This assumption is very general and holds for all reference Earth atmosphere models, such as [56]. Then, the thrust dominance assumption from Chapter 2 is re-written as follows.

Assumption 2 (Thrust dominance). $\dot{y}_2 \geq a_{cc} > 0$.

Assumption 2 implies condition (3.1), shows that the ratio r is greater than 1 and it also prevents hovering. Reaching null speed at a positive altitude is thus an undesired behavior and is not a steady state.

3.1.3 Optimal Control Problems

A natural goal for rocket landing is to maximize the final mass [64], or equivalently to minimize the fuel consumption. Landing is defined as final null altitude and (vertical) velocity. A constrained optimal control problem in free final time depending on an initial state y^0 can then be formulated.

Problem 1 (Fuel optimal landing with state inequality path constraints).

$$\min_{u(\cdot), t_f} \int_0^{t_f} r + u(s) ds \quad (3.4a)$$

$$\text{s.t.} \quad \dot{y} = f(y) + ug(y), \quad (3.4b)$$

$$|u| \leq 1 \quad (3.4c)$$

$$y(0) = y^0, y_1(t_f) = y_2(t_f) = 0 \quad (3.4d)$$

$$y_1(t) \geq 0, y_2(t) \leq 0, y_3(t) \in [m^-, m^+] \quad (3.4e)$$

State constraints (3.4e) are meant for any t in $[0, t_f]$. Additionally, we will consider another formulation where the state constraints (3.4e) have been removed, as they will be shown to be automatically satisfied.

Problem 2 (Fuel optimal landing).

$$\begin{aligned} \min_{u(\cdot), t_f} \quad & \int_0^{t_f} r + u(s) ds \\ \text{s.t.} \quad & \dot{y} = f(y) + ug(y), \\ & |u| \leq 1 \\ & y(0) = y^0, y_1(t_f) = y_2(t_f) = 0 \end{aligned}$$

Studying Problem (2) will help us describe the solutions of Problem 1.

Remark 10 (Terminology). *In the following proofs, maximal solutions of an ordinary differential equation are the solutions that cannot be extended in time.*

3.2 Preliminaries on the dynamics

This section aims at describing conditions under which state path inequalities (3.4e) can be ignored. A detailed study of the dynamics is conducted. First, the altitude and speed dynamics are studied using surfaces of \mathbb{R}^3 s.t. any trajectory that *lands* must start between these surfaces. This region is called the **flight envelope**. Then, the mass constraint is discussed.

Let us denote the domain $\mathcal{D} := \mathbb{R}^+ \times \mathbb{R}^- \times (0, m^+]$. Below, we say that a trajectory starting at some $y^0 \in \mathcal{D}$ *lands* applying the thrust $u(\cdot)$ if it reaches $y_1(t_f) = y_2(t_f) = 0$ for some $t_f > 0$. Note that the minimum mass constraint is not included in the first part of this discussion.

Beforehand, remark that there is a unique time T_u associated to a control $u(\cdot)$ s.t.

$$y_3^0 - \int_0^{T_u} r + u(s) ds = 0. \quad (3.5)$$

Since $r + u \geq r - 1 > 0$, the map $t \rightarrow 1/y_3(t)$ is not integrable near T_u because of (3.3c). Thus, the maximal solution of (3.2) starting at $y^0 \in \mathcal{D}$ is defined on the interval $[0, T_u)$. If $u \equiv \sigma$ is constant, then $T_\sigma = y_3^0 / (r + \sigma)$.

Lemma 1. *Let $\sigma \in [-1, 1]$ a constant parameter. For any y_2^0 and y_3^0 , there is a unique $y_1^0(\sigma, y_2^0, y_3^0)$ s.t. a trajectory starting at $(y_1^0(\sigma, y_2^0, y_3^0), y_2^0, y_3^0)^\top \in \mathcal{D}$ lands when applying the constant thrust $u \equiv \sigma$.*

Proof. The maximal solution y of (3.2) with $u \equiv \sigma$, starting at $y^0 \in \mathcal{D}$, is defined on $[0, T_\sigma)$. $y_2(\cdot)$ is continuous, increasing and diverges to $+\infty$ as t tends to T_σ . Thus, there is a unique time, denoted $t_*(y_1^0) \in [0, T_\sigma)$ s.t. $y_2(t_*(y_1^0)) = 0$. The IFT applied with Assumption 2, on equation³

$$\Phi_{f+\sigma g} \left(t_*(y_1^0), (y_1^0, y_2^0, y_3^0)^\top \right) \Big|_2 = 0 \quad (3.6)$$

shows that the application that maps y_1^0 into t_* is actually continuous, and differentiable, for all $y_1^0 \geq 0$. Then, define

$$\eta : z \in \mathbb{R}^+ \mapsto \Phi_{f+\sigma g} \left(t_*(z), (z, y_2^0, y_3^0)^\top \right) \Big|_1 \in \mathbb{R}. \quad (3.7)$$

³Here $\star|_i$ denotes the i^{th} component of \star .

From the regularity of $f + \sigma g$, the flow $\Phi_{f+\sigma g}$ is continuous and thus η is continuous. Necessarily, $\eta(0) < 0$. Moreover, since the acceleration is lower-bounded by $\underline{a_{cc}}$, it is possible to find an altitude $y_1^{crit} > 0$ large enough s.t. $\eta(y_1^{crit}) > 0$. Therefore, there is a $y_1^* \geq 0$ s.t. $\eta(y_1^*) = 0$. Using $t_f = t_*(y_1^*)$, one has $y_1(t_f) = y_2(t_f) = 0$ by construction of η . A comparison argument, as in the proof of Proposition 1, shows that η is actually increasing, proving the uniqueness of y_1^* . It yields $y_1^* = y_1^0(\sigma, y_2^0, y_3^0)$ using the above-mentioned variables. \square

Let us denote Σ_{\max} (respectively Σ_{\min}) the set of initial conditions s.t. landing is successful, at mass $y_3^f \in (0, m^+]$, when applying a constant maximum (resp. minimum) thrust. Denoting

$$\begin{aligned} y_1^{\max}(y_2, y_3) &:= y_1^0(1, y_2, y_3), \\ y_1^{\min}(y_2, y_3) &:= y_1^0(-1, y_2, y_3), \end{aligned}$$

yields

$$\begin{aligned} \Sigma_{\max} &:= \{(y_1^{\max}(y_2, y_3), y_2, y_3) : y_2 \leq 0, y_3 \in (0, m^+]\}, \\ \Sigma_{\min} &:= \{(y_1^{\min}(y_2, y_3), y_2, y_3) : y_2 \leq 0, y_3 \in (0, m^+]\}. \end{aligned}$$

Moreover, using flows of the backward-time dynamics, for $\sigma \in [-1, 1]$, define

$$\Sigma_{\sigma} := \left\{ \Phi_{-(f+\sigma g)} \left(t, (0, 0, y_3^f)^{\top} \right) : 0 \leq t \leq \frac{m^+ - y_3^f}{r + \sigma}, \quad 0 < y_3^f \leq m^+ \right\}$$

which provides the relations $\Sigma_{\max} = \Sigma_1$ and $\Sigma_{\min} = \Sigma_{-1}$. It is noteworthy that $y_1^{\max}(y_2, y_3) \leq y_1^{\min}(y_2, y_3)$, implying that Σ_{\max} is always “below” Σ_{\min} , as pictured in Figure 3.2.

Note that the applications y_1^{\min} and y_1^{\max} are continuous: this property stresses the continuity of the flows and the formal definition of Σ_{σ} . Continuity can also be proven using the IFT on function η from Equation (3.7), considering y_2^0 and y_3^0 as variables.

Proposition 1. *For any $y^0 \in \mathcal{D}$, if $y_1^{\min}(y_2^0, y_3^0) < y_1^0$ then for any control $u(\cdot)$ in $[-1, 1]$ the dynamics reaches null speed at a positive altitude.*

Proof. Consider $y^0 \in \mathcal{D}$ s.t. $y_1^{\min}(y_2^0, y_3^0) < y_1^0$ and denote

$$\tilde{y}^0 := (y_1^{\min}(y_2^0, y_3^0), y_2^0, y_3^0)^{\top}.$$

Let \tilde{y} be the maximal solution of (3.2) with $u \equiv 1$ and y be the maximal solution of (3.2) for some measurable function u satisfying $|u| \leq 1$ at all times. y starts at y^0 and \tilde{y} at \tilde{y}^0 . They are respectively defined on $[0, T_1)$ and $[0, T_u)$, where $T_u \leq T_1$. Using mass as a time-varying scaling, we get

$$\begin{pmatrix} \dot{y}_1(t) \\ \dot{y}_2(t) \end{pmatrix} \geq K \left(t, \begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix} \right) := \begin{pmatrix} y_2(t) \\ -\kappa + \frac{r-1-\pi(y_1(t))}{y_3^0 - t(r-1)} \end{pmatrix}$$

for any $t \in [0, T_u)$. By construction, \tilde{y} satisfies the equality version of this equation. Thus, comparison Lemma 13 (in Appendix) yields

$$\tilde{y}_1(t) \leq y_1(t), \quad \tilde{y}_2(t) \leq y_2(t), \quad \forall t \in [0, T_u).$$

Since y_2 is continuous, increasing and diverges as $t \rightarrow T_u$, there is a unique $t_* \in [0, T_u)$ s.t. $y_2(t_*) = 0$. Therefore, $y_1(t_*) \geq \tilde{y}_1(t_*) \geq 0$. Using a Taylor expansion on (3.3a) with the initial conditions shows that the last inequality is strict, whence the proposition. \square

Using a very similar proof, one shows the following result.

Proposition 2. *For any $y^0 \in \mathcal{D}$, if $y_1^{\max}(y_2^0, y_3^0) > y_1^0$ then for any control $u(\cdot)$ in $[-1, 1]$ the dynamics reaches null altitude at a negative speed.*

Proposition 1 defines the notion of being *too high*, meaning that if the rocket starts its powered descent above Σ_{\min} (in terms of altitude), then it will either lack fuel before reaching null speed, or go back up before touching the ground and then lack fuel at a non-zero altitude. In both cases, landing fails. Proposition 2 is the exact equivalent for the notion of being *too low*, meaning that the rocket will hit the ground at a non-zero speed if it starts below Σ_{\max} .

Further, note that if a trajectory lands s.t. the mass remains in $[m^-, m^+]$, then the acceleration is upper-bounded by $\bar{a}_{cc} := -\kappa + \frac{r+1}{m^-}$ for any positive time. Since the fuel flow is lower-bounded, the mass can remain in $[m^-, m^+]$ for at most $T_{\max} := \frac{m^+ - m^-}{r-1}$. Therefore, for any positive time, the speeds are lower-bounded by \underline{y}_2 and the altitudes are upper-bounded by \bar{y}_1 s.t.

$$\underline{y}_2 := -\bar{a}_{cc} T_{\max} \quad \text{and} \quad \bar{y}_1 := \bar{a}_{cc} \frac{T_{\max}^2}{2}. \quad (3.8)$$

Let us define $\mathcal{F} \subset \mathcal{D}$ the *flight envelope*, as the set of states y lying between Σ_{\max} and Σ_{\min} (in terms of altitude), and satisfying

$$y_1 \leq \bar{y}_1, \quad y_2 \geq \underline{y}_2, \quad \text{and} \quad m^- \leq y_3 \leq m^+. \quad (3.9)$$

Consequently, if $y^0 \in \mathcal{F}$ and Problem (2) has a solution, then altitude and speed constraints are enforced. If $y^0 \in \mathcal{D} \setminus \mathcal{F}$, then Problems 1 and 2 cannot have solutions.

Leaving out the limit cases of Σ_{\max} and Σ_{\min} , for which landing can be achieved by applying, respectively, the maximum and the minimum thrust, for the whole duration of the flight, we introduce

$$\mathcal{F}^* := \mathcal{F} \setminus (\Sigma_{\max} \cup \Sigma_{\min}). \quad (3.10)$$

The following result discusses feasibility of the landing. Optimality will be studied later on in Section 3.3.

Proposition 3. *If y^0 belongs to \mathcal{F}^* , then there is always a control u of structure Min-Max that lands.*

Proof. The Min-Max structure denotes a 2 step sequence starting with minimum value of the control and ending with maximum value. For such y^0 , denote $\tilde{y}^0 := (y_1^{\min}(y_2^0, y_3^0), y_2^0, y_3^0) \in \Sigma_{\min}$. Since $y^0 \in \mathcal{F}^*$, then

$$y_1^{\max}(y_2^0, y_3^0) < y_1^0 < y_1^{\min}(y_2^0, y_3^0).$$

Let us denote y and \tilde{y} the maximal solutions of Equation (3.2) with $u \equiv -1$, starting respectively at y^0 and \tilde{y}^0 . Then, using similar comparisons as in the previous proof, one obtains $y_1(t) < \tilde{y}_1(t)$ and $y_2(t) < \tilde{y}_2(t)$ for all positive times. Thus, one deduces that y_1 reaches zero at some time $t' > 0$, before y_2 does. Moreover, the map

$$\xi : t \in [0, t'] \rightarrow y_1(t) - y_1^{\max}(y_2(t), y_3(t)) \in \mathbb{R} \quad (3.11)$$

is continuous, and satisfies $\xi(0) > 0$ since the trajectory starts strictly above Σ_{\max} , and $\xi(t') < 0$ since $(0, y_2(t'), y_3(t'))$ is necessarily below Σ_{\max} in terms of altitude (recall that $y_2(t') < 0$). Thus, there exists a time $t'' \leq t'$ s.t. $y(t'') \in \Sigma_{\max}$. The desired Min-Max control law equals -1 on $[0, t'')$ and $+1$ for times $t \geq t''$. \square

As far as the mass is concerned, since it is a continuous decreasing function of time, enforcing the terminal constraint $y_3(t_f) \geq m^-$ is sufficient to guarantee the mass constraint (3.4e).

Therefore, *only the simplified Problem (2) needs to be solved*. If there is a solution that satisfies $y_3(t_f) \geq m^-$, then Problem (1) shares the same solution. Otherwise, if $y_3(t_f) < m^-$, then Problem (1) has no solutions. Indeed, since the solution is fuel-optimal, there is no other way to land with a greater final mass.

3.3 Optimal thrust programs

This section focuses on Problem (2) exclusively, which, according to the previous discussion gives an answer to Problem (1) or proves its infeasibility. We aim at proving that optimal controls are of Min-Max nature, where one of the min or max arcs may be absent. To establish this result (Theorem 1), we proceed as follows. First, stationary conditions are derived from the PMP. Then, using properties of the second adjoint state variable, the optimal thrust program is shown to be Max-Min-Max. Finally, the first maximum arc is shown to be absent under one (mild) additional assumption on the atmosphere model (Assumption 4).

3.3.1 Fuel Optimal Landing

Consider $y^0 \in \mathcal{F}$. Let u be an optimal thrust program for Problem (2) and y be the corresponding trajectory. Let t_f be the time-of-flight. It is assumed that $y_3(t_f) \geq m^-$. Thus, from the previous section, y lies in \mathcal{F} .

The Hamiltonian of Problem (2) is defined as

$$H := \lambda_0(r + u) + \lambda^\top (f(y) + ug(y)) \quad (3.12)$$

where $\lambda_0 \in \mathbb{R}$ and $\lambda : [0, t_f] \rightarrow \mathbb{R}^3$ denote the adjoint states. To study the control-affine Hamiltonian, consider the switching function

$$\Gamma(t) := \lambda_0 + \lambda(t)^\top g(y(t)) = \lambda_0 + \frac{\lambda_2(t)}{y_3(t)} - \lambda_3(t). \quad (3.13)$$

The PMP, as stated in [84, Thm. 2.2.1], yields

$$(\lambda_0, \lambda(t)) \neq 0_{\mathbb{R}^4}, \quad \forall t \in [0, t_f] \quad (3.14a)$$

$$\dot{\lambda}_1 = \lambda_2 \frac{\pi'(y_1)}{y_3} \quad (3.14b)$$

$$\dot{\lambda}_2 = -\lambda_1 \quad (3.14c)$$

$$\dot{\lambda}_3 = \frac{\lambda_2}{y_3^2} (r + u - \pi(y_1)) \quad (3.14d)$$

$$u = -\text{Sgn}(\Gamma(t)), \quad \text{when } \Gamma(t) \neq 0 \quad (3.14e)$$

$$\lambda(t_f) = \begin{pmatrix} \nu_1 & \nu_2 \end{pmatrix}^\top, \quad (\nu_1, \nu_2) \in \mathbb{R}^2 \quad (3.14f)$$

Equation (3.14a) states the non-triviality of the adjoint states. Following [24] and [102, Thm. 7.8.1], since the integral cost, the dynamics and the end-point constraints are time-invariant, the Hamiltonian is constant along the extremals and for such a free time, fixed endpoint problem, this constant is zero:

$$H(t) \equiv 0, \quad \forall t \in [0, t_f]. \quad (3.15)$$

The optimal pairs (y, u) are called *abnormal* extremals [6, 68] if $\lambda_0 = 0$, and *normal* extremals if $\lambda_0 \neq 0$. We now proceed to establish some intermediate results on the adjoint states.

Proposition 4. $(\lambda_1(t), \lambda_2(t)) \neq (0, 0)$ for all $t \in [0, t_f]$.

Proof. The linear time-varying dynamics of (λ_1, λ_2) is Lipschitz in (λ_1, λ_2) and continuous in time. Therefore, from the Cauchy-Lipschitz theorem, any maximal solution is unique. Thus, if there is a t_0 s.t. $(\lambda_1, \lambda_2)(t_0) = 0$, then $\lambda_2 \equiv 0$ over $[0, t_f]$, implying $\lambda_3 \equiv 0$ from (3.14d) and (3.14f) and then $\lambda_0 = 0$ from (3.15), violating (3.14a). \square

Now, remark that the sign of λ_2 and $\dot{\lambda}_2$ can be extrapolated from the following second-order equation

$$\ddot{\lambda}_2 = a(t)\lambda_2, \quad \text{where } a(t) := -\frac{\pi'(y_1(t))}{y_3(t)} > 0. \quad (3.16)$$

Indeed, the cones $\mathbb{R}^+ \times \mathbb{R}^+$ and $\mathbb{R}^- \times \mathbb{R}^-$ are both invariant through the dynamics (3.16). This shows that if λ_2 or $\dot{\lambda}_2$ is null at some $t_\lambda \in (0, t_f)$, then they will both remain in one of these cones after t_λ . Further, from Proposition 4 and (3.16), they will actually remain in interior subsets of these cones for times $t > t_\lambda$. Hence, by an exhaustive enumeration of possible cases we can state the following result.

Proposition 5. $(\lambda_2, \dot{\lambda}_2)$ necessarily match one of these conditions, as illustrated in Figure 3.1:

1. λ_2 and $\dot{\lambda}_2$ are never zero on $(0, t_f)$:
 - (a) $\lambda_2 > 0$ and $\dot{\lambda}_2 > 0$ on $(0, t_f)$,

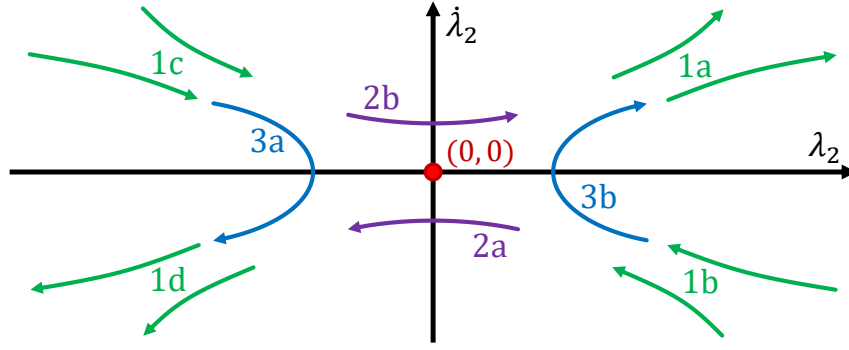


Figure 3.1: Possible scenarios for $(\lambda_2, \dot{\lambda}_2)$. The origin is prohibited due to Proposition 4.

- (b) $\lambda_2 > 0$ and $\dot{\lambda}_2 < 0$ on $(0, t_f)$,
 - (c) $\lambda_2 < 0$ and $\dot{\lambda}_2 > 0$ on $(0, t_f)$,
 - (d) $\lambda_2 < 0$ and $\dot{\lambda}_2 < 0$ on $(0, t_f)$,
2. There is a unique $t_\lambda \in (0, t_f)$ s.t. $\lambda_2(t_\lambda) = 0$ and $\dot{\lambda}_2 \neq 0$ on $[0, t_f]$:
 - (a) $\text{Sgn}(\lambda_2(t)) = -\text{Sgn}(t - t_\lambda)$ and $\dot{\lambda}_2 < 0$,
 - (b) $\text{Sgn}(\lambda_2(t)) = \text{Sgn}(t - t_\lambda)$ and $\dot{\lambda}_2 > 0$,
 3. There is a unique $t_\lambda \in (0, t_f)$ s.t. $\dot{\lambda}_2(t_\lambda) = 0$ and $\lambda_2 \neq 0$ on $[0, t_f]$:
 - (a) $\text{Sgn}(\dot{\lambda}_2(t)) = -\text{Sgn}(t - t_\lambda)$ and $\lambda_2 < 0$,
 - (b) $\text{Sgn}(\dot{\lambda}_2(t)) = \text{Sgn}(t - t_\lambda)$ and $\lambda_2 > 0$.

Note that for scenarios 1a and 1d (resp. scenarios 1b and 1c), either λ_2 or $\dot{\lambda}_2$ can be zero at $t = 0$ (resp. at $t = t_f$). Also, note that for scenario 2, λ_2 is necessarily non-zero at $t = 0$ and $t = t_f$ since $\dot{\lambda}_2$ is of constant sign. The same kind of remark applies to scenario 3 as well. The goal is to state whether these scenarios are consistent with conditions (3.14a)-(3.14f), and if so to what control structure they refer to.

Proposition 6. *Abnormal extremals are optimal programs of constant thrust.*

Proof. For abnormal extremals, $\lambda_0 = 0$. Using Equation (3.15) at $t = t_f$ yields $\nu_2 = 0 = \lambda_2(t_f)$. Thus, from Proposition 5, λ_2 has a constant non-zero sign over $[0, t_f]$. Moreover, from (3.14d) and (3.14f), one has

$$\text{Sgn}(\lambda_3(t)) = -\text{Sgn}(\lambda_2(t)), \quad \forall t \in [0, t_f]. \quad (3.17)$$

Therefore, for any $t \in [0, t_f]$: $\text{Sgn}(\Gamma(t)) = \text{Sgn}(\lambda_2)$. Hence, u has a constant value in $\{-1, +1\}$ over $[0, t_f]$. \square

This latest proposition shows that abnormal extremals require the initial state y^0 to be on constant thrust trajectories achieving landing, i.e. $y^0 \in \Sigma_{\max}$ or $y^0 \in \Sigma_{\min}$ must hold for these extremals.

From now on, we consider normal extremals only, and, without loss of generality⁴, we consider $\lambda_0 = 1$. Let us define

$$b(t) := \frac{\pi(y_1(t))}{y_3(t)}$$

Note that from Assumption 1 and the sign of y_2 , one can show that $a(\cdot)$ and $b(\cdot)$ are increasing from a study of their derivatives. Also, for all times in $[0, t_f]$, a and b are respectively lower and upper-bounded by

$$\underline{a} := -\frac{\pi'(\bar{y}_1)}{m^+} \text{ and } \bar{b} := \frac{\pi(0)}{m^-}. \quad (3.18)$$

Let us define $\gamma(t) := \dot{\lambda}_2(t) + \lambda_2(t)b(t)$, which satisfies

$$\frac{d\Gamma}{dt}(t) = \Gamma'(t) = \frac{\gamma(t)}{y_3(t)} \quad (3.19)$$

Since y_3 is positive, γ carries the sign of Γ' . From this point, Γ is the subject of our investigations.

Lemma 2. *If $\gamma < 0$ over $(0, t_f)$, then Γ is null at most on a single $t \in [0, t_f]$.*

Lemma 3. *$\Gamma < 0$ in the left-neighborhood of t_f .*

Proof. Equation (3.15) at t_f yields $\nu_2 = -(r + u(t_f))/\dot{y}_2(t_f)$. Thus, one gets

$$\Gamma(t_f) = -\frac{\kappa y_3(t_f) + \pi(0)}{y_3(t_f)\dot{y}_2(t_f)} < 0.$$

The conclusion follows from the continuity of $\Gamma(\cdot)$. □

λ_2 must be non-positive in a neighborhood of t_f . Indeed, let us assume that there is a time t' s.t. λ_2 is positive on $[t', t_f]$. Note that $\lambda_2(t_f)$ may be null. Then, using (3.14d) and (3.14f), λ_3 would necessarily be negative on $[t', t_f]$, leading to $\Gamma(t) > 0$ for t in $[t', t_f]$, which contradicts Lemma 3. This eliminates scenarios 1a, 1b, 2b and 3b.

Moreover, note that scenario 1d necessarily corresponds to Min-Max programs, where one arc may be absent, for it satisfies Lemma 2.

Then, the three remaining scenarios, namely 1c, 2a and 3a, require a refined sign study of λ_2 and $\dot{\lambda}_2$. Using differential equations bounding λ_2 , we can establish bounds on γ tight enough to derive valuable sign information.

Definition 2. *For a constant $c > 0$ and $t_0 \in (0, t_f)$, the C^2 scalar function x_c is defined over $[0, t_f]$ as the unique solution of the initial value problem*

$$\ddot{x}_c = cx_c \quad \text{with} \quad x_c(t_0) = \lambda_2(t_0) \quad \text{and} \quad \dot{x}_c(t_0) = \dot{\lambda}_2(t_0)$$

which yields

$$x_c(t) = \lambda_2(t_0) \cosh(\sqrt{c}(t - t_0)) + \frac{\dot{\lambda}_2(t_0)}{\sqrt{c}} \sinh(\sqrt{c}(t - t_0)).$$

⁴Equations being linear in λ , one can consider λ/λ_0 instead of λ .

Inspired from the definition of γ (from (3.19) and before), let us denote

$$\gamma_c(t) := \dot{x}_c(t) + x_c(t)b(t) \quad (3.20)$$

and introduce $z_\lambda := (\lambda_2, \dot{\lambda}_2)^\top$ and $z := (x_a, \dot{x}_a)^\top$ s.t.

$$\dot{z}_\lambda = F(t, z_\lambda) := \begin{pmatrix} 0 & 1 \\ a(t) & 0 \end{pmatrix} z_\lambda. \quad (3.21)$$

The next proofs require the following assumption.

Assumption 3. *The constants in (3.18) are s.t. $\bar{b} < \sqrt{a}$.*

Remark 11. *Assumption 3 depends on \bar{b} and a , which depend on the bounds on y_2 and y_1 . Though the estimates of y_2 and \bar{y}_1 provided in Equation (3.8) are coarse, they are sufficient for the numerical application discussed below. If needed, analytic bounds sharper than (3.8) could be computed.*

Proposition 7. *For scenario 2a, $\gamma(t) < 0, \forall t \in [0, t_f]$.*

Proof. Here, $\dot{\lambda}_2 < 0$ and $\text{Sgn}(\lambda_2(\cdot)) = -\text{Sgn}(\cdot - t_\lambda)$, where $t_\lambda \in (0, t_f)$. In this proof only, we consider the functions from Definition 2 with $t_0 = t_\lambda$. It leads to

$$\gamma_a(t) = \dot{\lambda}_2(t_\lambda) \left(\cosh(\sqrt{a}(t - t_\lambda)) + \frac{b(t)}{\sqrt{a}} \sinh(\sqrt{a}(t - t_\lambda)) \right).$$

For any $t > t_\lambda$, $\gamma_a(t) < 0$. Since a increases, scenario 2a yields $z_\lambda(t_\lambda) = z_a(t_\lambda)$ and for any $t \in [t_\lambda, t_f]$

$$\dot{z}_\lambda(t) = F(t, z_\lambda(t)) \text{ and } \dot{z}_a(t) \leq F(t, z_a(t)). \quad (3.22)$$

Comparison Lemma 13 yields $z_\lambda(t) \leq z_a(t)$ on $[t_\lambda, t_f]$. As a consequence

$$\gamma(t) \leq \gamma_a(t) < 0, \quad \forall t \in [t_\lambda, t_f]. \quad (3.23)$$

For any $t < t_\lambda$, $\gamma_a(t) < 0$ when Assumption 3 is satisfied. The same reasoning applies, except that $\lambda_2(t) > 0$ this time, and that the comparison Lemma 13 has to be applied in backward-time. It shows that $\gamma(t) \leq \gamma_a(t) < 0$ for any $t \in [0, t_\lambda]$, whence the desired property. \square

Lemma 3 and Proposition 7 imply that the sign of Γ changes at most once over $[0, t_f]$ for scenario 2a.

Lemma 4. *If $\lambda_2 < 0$ over $[0, t_f]$, if $\gamma(t_\gamma) = 0$ for some $t_\gamma \in (0, t_f)$ and if Assumption 3 holds, then: $\gamma(t) < 0, \forall t > t_\gamma$.*

Proof. By construction $\lambda_2(t_\gamma) = -\dot{\lambda}_2(t_\gamma)/b(t_\gamma)$. Necessarily, $\dot{\lambda}_2(t_\gamma) > 0$. In this proof only, we consider the functions from Definition 2 with $t_0 = t_\gamma$. It yields

$$\gamma_a(t) = \dot{\lambda}_2(t_\gamma) \left[\left(1 - \frac{b(t)}{b(t_\gamma)} \right) \cosh(\sqrt{a}(t - t_\gamma)) + \frac{b(t)b(t_\gamma) - a}{b(t_\gamma)\sqrt{a}} \sinh(\sqrt{a}(t - t_\gamma)) \right]$$

Since b increases, the factor associated to the cosh term is negative. Also, Assumption 3 yields

$$b(t)b(t_\gamma) - \underline{a} \leq (b(t) + \sqrt{\underline{a}})(b(t) - \sqrt{\underline{a}}) < 0 \quad (3.24)$$

Thus, $\gamma_{\underline{a}}(t) < 0$ for $t > t_\gamma$. Moreover, $z_\lambda(t_\gamma) = z_{\underline{a}}(t_\gamma)$ holds and since $\lambda_2 < 0$, for any $t \in [t_\gamma, t_f]$

$$\dot{z}_\lambda(t) = F(t, z_\lambda(t)) \text{ and } \dot{z}_{\underline{a}}(t) \leq F(t, z_{\underline{a}}(t)). \quad (3.25)$$

The conclusion stems from comparison Lemma 13. \square

Proposition 8. *Under the assumptions of Lemma 4, the sign of Γ changes at most twice on $[0, t_f]$.*

Proof. γ can be zero at most on an isolated point. Indeed, γ is continuous and if there is t_0 s.t. $\gamma(t_0) = 0$, then, from Lemma 4, it cannot be zero for greater times. Therefore, from (3.19), Γ can be zero at most on two isolated points. \square

Proposition 8 shows that the two remaining scenarios (1c and 3a) correspond to Max-Min-Max structures. It enables us to state the main result below.

Proposition 9. *Under Assumptions 1, 2 and 3, and for y^0 in \mathcal{F} , any solution of Problem (2) is necessarily a Max-Min-Max thrust program, where one or two arcs may be absent.*

3.3.2 Optimality of Min-Max Programs

We shall now discuss under which conditions Min-Max trajectories are always more fuel-optimal than Max-Min-Max trajectories, for some $y^0 \in \mathcal{F}^*$.

Let us consider a trajectory y starting at y^0 , with thrust structure Max-Min-Max. Denote t_1 its first time of switch (from max to min). The last max arc may be of null duration. Then, for every time $t'_1 \in [0, t_1]$, there is a trajectory with thrust structure Max-Min-Max, with first time of switch t'_1 , that lands, which is guaranteed by applying Proposition 3 at t'_1 . Below, we derive conditions under which the trajectory having the smallest first time of switch has the highest final mass, showing that the Min-Max trajectory starting from y^0 is fuel-optimal.

The second time of switch, denoted t_2 , and the final time t_f are implicitly imposed by t_1 so that the rocket lands. This relation will be given later. For the time being, note that the final mass, denoted y_3^f , satisfies

$$y_3^0 - y_3^f(t_1) = (r+1)t_1 + (r-1)(t_2(t_1) - t_1) + (r+1)(t_f(t_1) - t_2(t_1)). \quad (3.26)$$

The first two components of y are collected in $\mu(y)$, i.e. $\mu(y) := (y_1, y_2)^\top$. The landing condition is simply $\mu(y(t_f)) = 0$. Define

$$L(\tau_1, \tau_2, \tau_f) := \mu \left(\Phi_{f+g}(\tau_f - \tau_2, \Phi_{f-g}(\tau_2 - \tau_1, \Phi_{f+g}(\tau_1, y^0))) \right).$$

Then, the landing condition boils down to

$$L(t_1, t_2, t_f) = 0. \quad (3.27)$$

It describes the above-mentioned implicit dependence of (t_2, t_f) on t_1 . When applicable, the IFT used on (3.27) provides us with the differentiability and the value of the derivatives of t_2 and t_f w.r.t. t_1 , as

$$\begin{pmatrix} dt_2 & dt_f \\ dt_1 & dt_1 \end{pmatrix}^\top = - \left[\frac{\partial L}{\partial [t_2, t_f]} \right]^{-1} \cdot \frac{\partial L}{\partial t_1}. \quad (3.28)$$

To express these derivatives w.r.t. t_1 , intermediate quantities are introduced. The *transition* matrices $M(t_f)$ and $N(t_2)$ are respectively defined as the unique solutions to the matrix initial value problems

$$\dot{M}(t) = \frac{\partial(f+g)}{\partial y}(y(t)) \cdot M(t) \quad \text{and} \quad M(t_2) = \mathbb{I}_3, \quad (3.29)$$

$$\dot{N}(t) = \frac{\partial(f-g)}{\partial y}(y(t)) \cdot N(t) \quad \text{and} \quad N(t_1) = \mathbb{I}_3. \quad (3.30)$$

Let us define R_1 , R_2 , S_1 and S_2 by

$$\begin{pmatrix} R_1 & R_2 \end{pmatrix}^\top := \mu(M(t_f) \cdot (f-g)(y(t_2))), \quad (3.31)$$

$$\begin{pmatrix} S_1 & S_2 \end{pmatrix}^\top := \mu(M(t_f) \cdot N(t_2) \cdot (f+g)(y(t_1))). \quad (3.32)$$

Since Assumption 2 holds, the invertibility condition of $\frac{\partial L}{\partial [t_2, t_f]}$ needed to apply the IFT boils down to $R_1 \neq 0$. Then, one can provide a detailed version of (3.28)

$$\frac{dt_2}{dt_1} = 1 - \frac{S_1}{R_1}, \quad (3.33)$$

$$\frac{dt_f}{dt_1} = 1 - \frac{R_1 S_2 - R_2 S_1 + S_1 \dot{y}_2(t_f)}{\dot{y}_2(t_f) R_1}. \quad (3.34)$$

Using the previous terms with Equation (3.26) yields

$$\frac{dy_3^f}{dt_1}(t_1) = \frac{r+1}{\dot{y}_2(t_f) R_1} \left[R_1 (S_2 - \dot{y}_2(t_f)) + S_1 \left(\dot{y}_2(t_f) \frac{r-1}{r+1} - R_2 \right) \right]. \quad (3.35)$$

The conditions that enable us to state that Min-Max thrust programs are always more fuel-optimal than the Max-Min-Max ones, by allowing us to apply the IFT on L , are thus conveyed by the assumption below

Assumption 4. *The parameter defined in (3.31) is s.t. $R_1 \neq 0$, and one has $\frac{dy_3^f}{dt_1}(0) < 0$ for any $y^0 \in \mathcal{F}^*$ s.t. the rocket lands at $y_3(t_f) \geq m^-$.*

Note that, since it is formulated for any $y^0 \in \mathcal{F}^*$, it is sufficient to check these conditions for $t_1 = 0$ only. Moreover, these conditions can be either checked through (3.35), analytically - if the pressure model is known well enough and tractable - or numerically.

Remark 12. For illustration purposes only, let us check the validity of Assumption 4 when there is no atmosphere. When $\pi \equiv 0$, every term from (3.33) and (3.34) can be explicitly written using the fact that $\frac{r+u}{y_3} = -\frac{y_3}{y_3}$ for intermediate integrations, which yields

$$\begin{aligned} R_1 &= \frac{2}{r+1} \left(1 - \frac{y_3(t_f)}{y_3(t_2)} + \log \frac{y_3(t_f)}{y_3(t_2)} \right), & R_2 &= -\kappa + \frac{r-1}{y_3(t_f)}, \\ S_1 &= -\frac{2}{r-1} \log \frac{y_3(t_2)}{y_3(t_1)}, & S_2 &= -\kappa + \frac{r+1}{y_3(t_f)}. \end{aligned}$$

R_1 is negative since $y_3(t_2) > y_3(t_f)$. Thus, (3.35) becomes

$$\frac{dy_3^f}{dt_1}(t_1) = -\frac{4\kappa}{y_2(t_f)(r-1)} \frac{1}{R_1} \log \frac{y_3(t_2)}{y_3(t_1)} < 0. \quad (3.36)$$

The negativity of this quantity gives the desired conclusion. By continuity, the assumption also holds for scarce atmospheres. Further, an example based on a non-scarce tabulated pressure model is treated in Section 3.4.

3.3.3 Main result

Under Assumptions 1, 2, 3, and 4, if the final mass y_3^f of the landing Min-Max trajectory, starting from a y^0 in \mathcal{F} , satisfies $y_3^f \geq m^-$, then the optimal thrust program of Problem (1) is Min-Max, where one arc may be absent. Conversely, if $y_3^f < m^-$ or if $y^0 \notin \mathcal{F}$, then Problem (1) has no solution.

Henceforth, it is possible to describe the whole set of feasible initial conditions. Define

$$\begin{aligned} \Omega(y_3^f) &:= \{ \Phi_{-(f-g)}(\tau_1, \Phi_{-(f+g)}(\tau_2, (0, 0, y_3^f)^\top)) : \\ &\quad \tau_1 \geq 0, \quad \tau_2 \geq 0, \quad (r-1)\tau_1 + (r+1)\tau_2 \leq m^+ - y_3^f \} \end{aligned}$$

which denotes the set of states landing at final mass $y_3^f \leq m^+$ applying a Min-Max control. Minimum (resp. maximum) arcs last for τ_1 (resp. τ_2). Thus, the solution set \mathcal{F}^{sol} of the initial conditions y^0 s.t. Problem (1) has a solution is

$$\mathcal{F}^{sol} := \bigcup_{m^- \leq y_3^f \leq m^+} \Omega(y_3^f) \quad (3.37)$$

The following theorem summarizes this discussion.

Theorem 1 (Optimal thrust program of atmospheric vertical landing). *Under Assumptions 1, 2, 3 and 4, Problem (1) has a solution if and only if $y^0 \in \mathcal{F}^{sol}$. When $y^0 \in \mathcal{F}^{sol}$, the optimal thrust program is Min-Max, where one arc may be absent.*

Remark 13. Without Assumption 4, Max-Min-Max programs (where one or two arcs may be absent) are optimal.

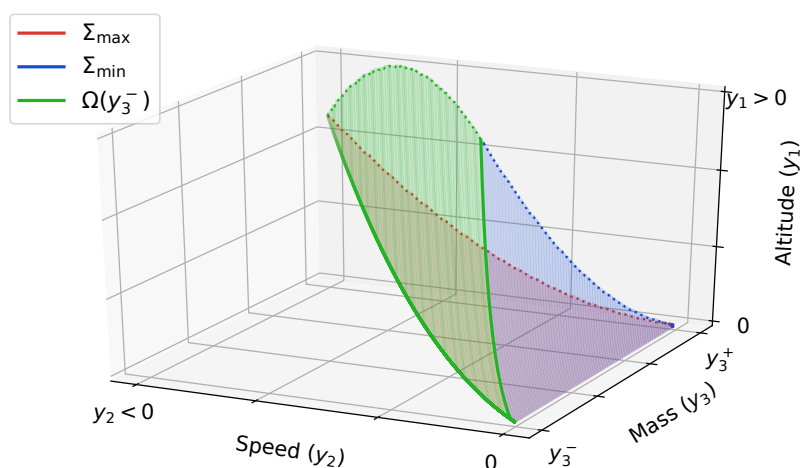


Figure 3.2: *Flight envelope*. \mathcal{F}^{sol} is delimited by Σ_{\max} , Σ_{\min} , $\Omega(y_3^-)$ and closed by the constraint $y_3 \leq y_3^+$ on the last side. For Σ_{\max} and Σ_{\min} , only the trajectories that land with a mass $y_3(t_f) \geq m^-$ are represented. The vertical axis conveys the altitude to ease the visualization.

3.4 Numerical illustrations

Let us consider the following (normalized) parameters

$$\kappa = 0.00285 \text{ s}^{-1}, \quad r = 4.0, \quad m^- = 458.3 \text{ s}, \quad m^+ = 520.3 \text{ s}.$$

In this example, the engine can be used at 60-100% of its maximum flowrate. Also, κ is taken close to the values of actual reusable launcher engines [72], such as the Merlin (Falcon 9) or the BE-4 (New Glenn). We consider a pressure model describing Earth's atmosphere from tabulated values, satisfying Assumption 1, s.t. $\pi(0) = 6.2 \times 10^{-1}$. Assumption 2 and 3 are satisfied since

$$\underline{a}_{cc} = 2.90 \times 10^{-3} > 0 \quad \text{and} \quad \bar{b}/\sqrt{\underline{a}} = 3.37 \times 10^{-1} < 1.$$

\mathcal{F}^{sol} is pictured in Figure 3.2 for these values. Assumption 4 is then checked numerically, by computing R_1 and $\frac{dy_3^f}{dt_1}(0)$ on a high density mesh covering \mathcal{F}^{sol} . The evaluation of (3.35) only requires to integrate ODEs, namely (3.2), (3.29) and (3.30), over fixed time intervals. Thus, it is vastly beneficial to use (3.35) instead of finite differences to check Assumption 4 in reasonable time.

To illustrate the optimality of the Min-Max structure, let us consider an example where three trajectories start from $y_1^0 = 1.25 \text{ s}$, $y_2^0 = -6.96 \times 10^{-2}$ and $y_3^0 = 441.2 \text{ s}$, as presented in Figure 3.3. The trajectory with the Min-Max structure has the greatest final mass. The other two trajectories have respectively a 2.2% and 5.7% lower final mass. Note that the associated time-of-flights t_f are respectively 25.6s, 31.7s and 41.1s.

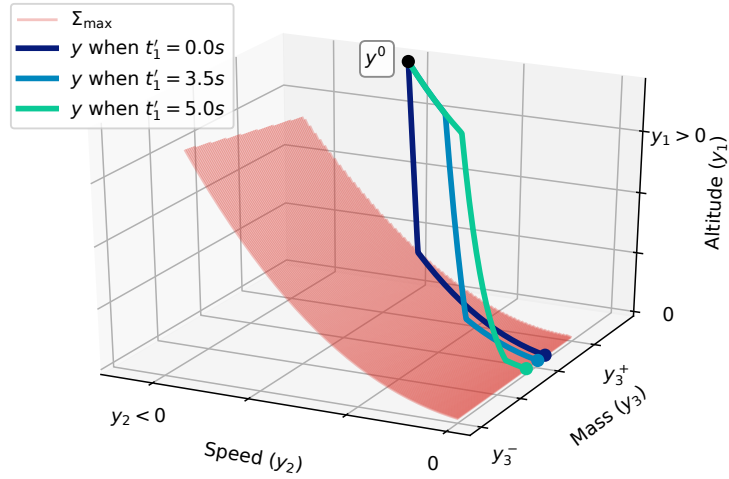


Figure 3.3: *Three Max-Min-Max trajectories*, with varying first time of switch t_1' . Maximum final mass is obtained for $t_1' = 0$, i.e. for a Min-Max thrust program.

3.5 Comments

The results presented above call for a few comments.

Vertical flight envelope applications A useful by-product of the proofs from above is the full characterization of the reachable set for the vertical motion (3.37), as shown in Figure 3.2. Using the notations from Figure 3.2, for any map $v(y) : \mathbb{R}^3 \rightarrow [-1, 1]$, every feedback control law $u : \mathbb{R}^3 \rightarrow \mathbb{R}$ s.t.

$$u(y) = \begin{cases} 1 & \text{if } y \in \Sigma_{\max}, \\ -1 & \text{if } y \in \Sigma_{\min} \cup \Omega(y_3^-), \\ v(y) & \text{otherwise,} \end{cases}$$

guarantees that a rocket starting inside the vertical flight envelope will land at null vertical speed. This could be used to design safe-set control laws [40, 105]. Also, in terms of software design, using the structure where a control “ u ” supervises another control “ v ” would help increase the Run-Time Assurance of the rocket G&C system [29, 86], but it brings us out-of-the-scope of this thesis.

Non robustness of the Min-Max trajectory The optimal thrust program being of a *bang-bang* nature, it is non-robust to some sources of uncertainty, due to the proximity of the ground. For instance, if the switch from the Min to the Max arc is delayed, the rocket necessarily comes out of the vertical flight envelope, which guarantees that there are no thrust programs that allow a proper landing. Given the uncertain and complex environment in which the actual rocket final burn occurs, it can be of high interest to consider more robust thrust programs, that are not too

close to the actuators limits, or, in other words, not too close to the vertical flight envelope boundary.

Conjecture regarding assumption 4 Theorem 1 relies on Assumption 4 to rule out the Max-Min-Max programs, and keep the Min-Max programs only. A greater number of numerical simulations, aiming at computing $\frac{dy_3^f}{dt_1}(0)$ while dispersing most of the involved parameters, make us *conjecture* that Assumption 4 could be made much weaker: it is sufficient but not necessary.

Recent results on the topic The results presented in this chapter correspond to the paper [65] published in 2021. Since then, Leparoux *et al.* published an article on a really close topic [55]. They explored the structure of optimal thrust programs of 3D rocket models for planetary landing. Among others, they showed that the optimal thrust programs were generally of the Max-Min-Max nature, and that it was not sensitive to several model changes. Among the model changes that they considered, they used an atmospheric model that resembles the thrust bias law $T = gISPq - S_E P(h)$ that we use in this manuscript, but with a *constant* pressure term instead.

Other approach for optimal thrust programs The optimal control problem studied in this chapter has a state of dimension 3, which makes the theoretical results from H. J. Sussmann and H. Schättler applicable as well [83,84,95,96]. They provided a variety of results based on the Lie brackets of the functions f and g involved in Equation (3.2), which can help characterize the optimal solution nature of Problem (2), but for a *minimum-time* criterion only.

Chapter 4

Nominal guidance via Quadratic Programming

Résumé

Le problème général de guidage par descente motorisée PDG (ξ, p) , défini dans l'équation (2.22), est un OCP, c'est-à-dire un problème d'optimisation de dimension infinie, en temps libre. Dans ce chapitre, ce problème est réécrit sous la forme d'un problème de dimension finie, aussi simple que possible, afin qu'il puisse être résolu rapidement et de manière fiable en vol. Cette approche est la méthode de guidage par descente motorisée que nous proposons.

Tout d'abord, PDG (ξ, p) est réécrit sous la forme d'un problème d'optimisation non-linéaire (NLP) paramétrique de faible dimension. Cette réécriture nécessite, entre autres, une description des variables d'état basée sur le flot d'équations différentielles ordinaires (ODE), une représentation à dimension finie de la variable de contrôle, et une remise à l'échelle de la variable temporelle pour tenir compte des variations du temps final.

Deuxièmement, la solution de ce dernier NLP est approximée par une expansion directionnelle du premier ordre, en utilisant les résultats classiques de l'analyse de sensibilité des NLP. Étant donné que des variations générales et non infiniment petites des paramètres ξ et p doivent être prises en compte dans l'application, il est nécessaire de traiter les changements dans l'ensemble des contraintes actives. Il s'agit là d'une caractéristique essentielle de l'analyse de sensibilité. Une méthode de calcul basée sur la programmation quadratique (QP) est décrite pour traiter cette question.

Enfin, des commentaires importants concernant l'utilisation *offline/online* de ce QP sont discutés et illustrés par trois exemples numériques. La méthode de guidage nominal décrite ici est générale et peut être appliquée aux problèmes 2D et 3D.

The general Powered Descent Guidance problem PDG (ξ, p) , defined in Equation (2.22), is an OCP, i.e. an infinite dimensional optimization problem, in free-final time. In this chapter, this problem is re-written as a finite dimensional problem, as simple as possible, so that it can be solved quickly and reliably in-flight. This approach is our proposed *nominal Powered Descent Guidance method*.

First, PDG (ξ, p) is re-written as a low dimensional parametric Non-Linear Program (NLP). Among others, this rewriting requires a description of the state variables based on the flow of Ordinary Differential Equations (ODEs), a finite-dimensional representation of the control variable, and a re-scaling of the time variable to account for the variations of the free-final time.

Second, the solution of the latter NLP is approximated by a directional first-order expansion, using classic sensitivity analysis results of NLPs. Because general and non-infinitely small variations of the parameters ξ and p must be considered in the application, it is necessary to deal with changes in the set of active constraints. This appears as a critical feature of the sensitivity analysis. A computational method based on Quadratic Programming (QP) is described to handle this.

Finally, important comments regarding the offline/online use of this QP are discussed, and illustrated on three numerical examples. The nominal guidance method described here is general, and can be applied to both the 2D and 3D problems.

This chapter is an updated version of [67], with enhanced examples and a new application to the 3D rocket model.

4.1 Non-Linear Programming formulation for PDG

The goal of this section is to explain how PDG (ξ, p) can be approximated using a NLP with few variables.

As recalled in Chapter 1, OCPs are commonly solved using either *direct* or *indirect* methods. On one hand, direct methods first discretize the optimization problem and then solve it using NLP techniques. On the other hand, indirect methods consist in formulating infinite dimensional stationary conditions first, and then discretizing them. The former is often more robust but less accurate than the latter. For both approaches, it is highly difficult to guarantee convergence times for complex non-linear problems.

We propose to approximate PDG (ξ, p) using a method in-between these two classic approaches. As used in certain direct methods [45, 51, 103], we discretize the control variable using an interpolation method. However, we do not use a coarse ¹ discretization scheme to convey the dynamic equation. The latter is here expressed exactly, as the flow of a certain ODE. A point worth special care is that the ODE is defined over a time domain whose endpoint is an unknown of the problem. This representation will help us form a finite dimensional problem denoted NLP (ξ) thereafter. The sensitivity analysis based method used to solve this latter problem will be the matter of the next section.

¹Here, *coarse* refers to discretization schemes such as Euler methods with few collocation points.

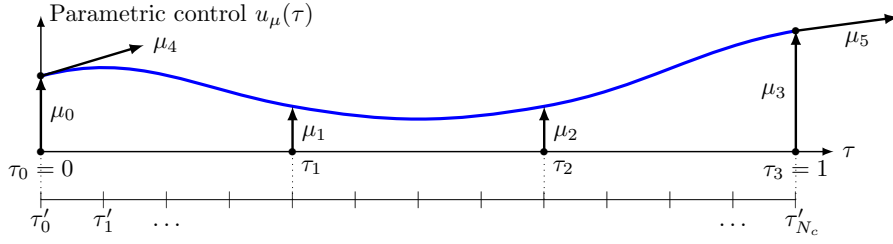


Figure 4.1: *Control discretization and time instances of the constraints for $N = 3$.* The correction is described by a parametric function $\tau \mapsto u_\mu(\tau)$. Here is represented a scalar Cubic Spline, described by its values μ_0, \dots, μ_3 at several time instances, and by its slopes μ_4 and μ_5 at the starting and end-points. The inequality constraints are enforced on the subdivision $\tau'_0, \dots, \tau'_{N_c}$.

4.1.1 Discretization of the decision variable

4.1.1.1 Free-final time

First, recall that in PDG (ξ, p) , the time-of-flight is an optimization variable implicitly defined by the constraints and the cost. Its change w.r.t. the reference time of flight \bar{t}_f is denoted Δt_f . We scale the time variable t by considering

$$\tau := \frac{t}{\bar{t}_f + \Delta t_f}$$

The final time is considered as an extra state of null dynamics: $\dot{t}_f = 0$. The augmented state equals $\tilde{x} := (x^\top, t_f)^\top$, and satisfies the dynamics

$$\dot{\tilde{x}} = \tilde{f}(\tilde{x}, u, \eta) := \begin{pmatrix} t_f f(x, u, \eta) \\ 0 \end{pmatrix}$$

where the variables are here defined for times $\tau \in [0, 1]$. The unknown Δt_f is now taken into account as an initial condition, s.t.

$$\tilde{x}(0) = \begin{pmatrix} \bar{x}^0 + \Delta x^0 \\ \bar{t}_f + \Delta t_f \end{pmatrix}$$

To alleviate the writing, the constraint $A^f x(1) = b^f(p)$ will be written $A^f \tilde{x}(1) = b^f(p)$ as well, where the latter matrix A^f is simply the former matrix A^f with an extra column of zeros on the right-hand side. Here, $b^f(p)$ remains unchanged. Likewise, the notation $c(\tilde{x}, u, \eta, p)$ will be used to refer to the former constraint $c(x, u, \eta, p) \leq 0$.

4.1.1.2 Parametric description of the control

We choose to describe the infinite dimensional variable δu using a smooth parametric description. Let us describe δu via a function

$$(\mu, \tau) \in \mathbb{R}^{N_\mu} \times [0, 1] \rightarrow u_\mu(\tau) \in \mathbb{R}^m$$

where $\mu \mapsto u_\mu(\tau)$ is *linear* for any fixed τ . There is a matrix valued function $M(\cdot)$ s.t. $u_\mu(\tau) = M(\tau)\mu$. This framework encompasses the use of many interpolation methods, from piecewise constant interpolation to Cubic Splines and Hermite polynomials. Our choice is detailed below.

For $N \geq 2$, consider a subdivision of the normalized time interval $[0, 1]$ denoted by the $N + 1$ time instances $\tau_0 = 0 < \tau_1 < \dots < \tau_N = 1$. We chose to describe u_μ as a Cubic Spline defined on the subdivision $(\tau_i)_{0 \leq i \leq N}$, as represented in Figure 4.1. These Splines are constructed using the classic method from [51, Sec. 3.3]. Thus, the vector μ embeds the values u^k of the corrections at τ_k and the slopes \dot{u}^0 and \dot{u}^N s.t.

$$\mu := \left((u^0)^\top, (u^1)^\top, \dots, (u^N)^\top, (\dot{u}^0)^\top, (\dot{u}^N)^\top \right)^\top \in \mathbb{R}^{N_\mu}. \quad (4.1)$$

where $N_\mu = m(N + 3)$.

The bounds on the engine flow must be discussed differently depending on the engine model considered. On the one hand, for the 2D rocket model, it was decided that the engine flow was directly controlled via q_r . Even though this simplifying choice was made for illustration purposes only, choosing a smooth parametric description for u_μ has a convenient by-product. Since the controlled flow can be expressed from the real flow s.t. $q_c = \tau_q \dot{q}_r + q_r$, then it will be possible to express exactly the controlled flow, as long as

- q_r remains differentiable,
- $\tau_q \dot{q}_r + q_r$ remains within $[q^-, q^+]$,
- the estimated value $\hat{q}_r(0)$ of the initial real flow is imposed, i.e.: $\bar{q}_r(0) + (q_r)_\mu(0) = \hat{q}_r(0)$.

On the other hand, the 3D model already takes into account the first-order dynamics on the engine flow. If $q^- \leq q_r^0 \leq q^+$ and that $q^- \leq q_c^0 \leq q^+$, then the real flow will remain within $[q^-, q^+]$. Therefore, imposing the flow bounds via the constraint $u^-(p) \leq \bar{u}(\tau) + u_\mu(\tau) \leq u^+(p)$ is sufficient to guarantee that q_c and q_r lie in the proper interval.

Moreover, it was previously mentioned that the engine flow dynamics was taken into account but not the orientation dynamics, since the time constant of the latter was significantly much smaller than the former. However, to remain as feasible as possible by the real system, it is of high interest to have a continuous and smooth control law description. Picking a parametric description such as the Cubic Splines ensures the smoothness. Moreover, imposing the initial value of the incidence - or the projected incidences for the 3D model - to be the same as the current estimate ensures the continuity. Therefore, we consider a constraint of the shape $\bar{u}(0) + u_\mu(0) = \hat{u}(0)$, which also writes

$$u_\mu(0) = \Delta u_{\text{init}} \quad (4.2)$$

where Δu_{init} is the gap between the reference control at time 0 and the current control.

Remark 14. *When converting OCPs to NLPs, Cubic Splines and other similar discretization methods are often used to approximate the control variable **and** the state itself [14, 36, 51, 103]. However, here, only the variable δu is described by finitely many values, and the state will be described exactly, using Equation (4.7) below. The state is not a Cubic Spline. This choice is motivated by the need to evaluate with a very high accuracy the states, especially at touchdown.*

4.1.1.3 Parametric problem

The control variable of the parametric description of our problem is

$$z := (\mu^\top, \Delta t_f)^\top. \quad (4.3)$$

It is of dimension $N_z = N_\mu + 1$. From the previous discussion, let us denote the whole input vector by

$$\xi := (\Delta x^{0\top}, \Delta \eta^\top, \Delta u_{\text{init}}^\top)^\top \in \mathbb{R}^{N_\xi}. \quad (4.4)$$

Describing the control correction by $u_\mu(\cdot)$ and imposing the control constraints

$$u^- \leq \bar{u} + u_\mu \leq u^+ \quad (4.5)$$

does not necessarily imply that μ is bounded. Indeed, the slopes of u_μ at $\tau = 0$ and $\tau = 1$ are not directly bounded by these constraints, especially if these constraints are enforced on a badly chosen subset of time instances. Likewise, neither the bounds from Equation (4.5) nor the other above-mentioned constraints necessarily imply that Δt_f is bounded. To guarantee that both μ and Δt_f remain bounded, additional constraints are imposed on the decision variable z s.t.

$$z_{\text{low}} \leq z \leq z_{\text{up}}.$$

By extending the mixed state-control constraints conveyed by the function c in Equation (2.22f), an approximation of PDG (ξ, p) is

$$\min_{z \in \mathbb{R}^{N_z}} J(z, \xi) \quad (4.6a)$$

$$\text{s.t. } \dot{\tilde{x}}(\tau) = \tilde{f}(\tilde{x}(\tau), \bar{u}(\tau) + u_\mu(\tau), \eta + \Delta \eta), \quad \forall \tau \in [0, 1] \quad (4.6b)$$

$$\tilde{x}(0) = \begin{pmatrix} x^0 + \Delta x^0 \\ \bar{t}_f + \Delta t_f \end{pmatrix} \quad (4.6c)$$

$$A^f x(1) = b^f(p) \quad (4.6d)$$

$$u_\mu(0) = \Delta u_{\text{init}} \quad (4.6e)$$

$$u^-(p) \leq \bar{u}(\tau) + u_\mu(\tau) \leq u^+(p), \quad \forall \tau \in [0, 1] \quad (4.6f)$$

$$c(\tilde{x}(\tau), \bar{u}(\tau) + u_\mu(\tau), \eta + \Delta \eta, p) \leq 0, \quad \forall \tau \in [0, 1] \quad (4.6g)$$

$$z_{\text{low}} \leq z \leq z_{\text{up}} \quad (4.6h)$$

Problem (4.6) has a finite-dimensional decision variable, but an infinite number of constraints.

4.1.2 Formulation of the finite dimensional guidance problem

4.1.2.1 Description of the state as the flow of an ODE

Our goal is to remove x from the description of (4.6). To this purpose, let us introduce a classic notation used for the flow of the ODE following e.g. [18, 90].

Definition 3 (Flow of f). *Consider the subsets $\mathcal{X} \subset \mathbb{R}^n$ (assumed open), $\mathcal{U} \subset \mathbb{R}^m$ (assumed compact) and $\Omega \subset \mathbb{R}^{n_\eta}$. Given a differentiable function $f : \mathcal{X} \times \mathcal{U} \times \Omega \rightarrow \mathbb{R}^n$, vectors $(x^0, \eta) \in \mathcal{X} \times \Omega$ and a control function² $u \in L^\infty([0, 1], \mathcal{U})$, the flow of the ODE defined by f is defined using the following Initial Value Problem (IVP)*

$$\forall t \in [0, 1], \quad x(t) = \Phi_f(t, x^0, \eta; u) \Leftrightarrow \begin{cases} x(0) = x^0 \\ \dot{x}(s) = f(x(s), u(s), \eta), \quad \forall s \in [0, t]. \end{cases}$$

Using this notation, let us describe the extended state $\tilde{x}[\tau, z, \xi] \in \mathbb{R}^{n+1}$ as

$$\tilde{x}[\tau, z, \xi] := \Phi_{\tilde{f}} \left(\tau, \begin{pmatrix} x^0 + \Delta x^0 \\ t_f + \Delta t_f \end{pmatrix}, \eta + \Delta \eta; \bar{u} + u_\mu \right), \quad \forall \tau \in [0, 1]. \quad (4.7)$$

The latter notation $\tilde{x}[\tau, z, \xi]$ will prove to be handy when writing the inequality and equality constraints (4.8) and (4.9) below. Note also that the hypothesis of Definition 3 guarantee that $\tilde{x}[\tau, z, \xi]$ is uniquely defined. For formal results on the existence and uniqueness of $\Phi_f(t, x^0, \eta; u)$, see e.g. [90, Appendix C.3].

Provided that \tilde{f} is continuously differentiable w.r.t. all of its inputs, since $\mu \mapsto u_\mu(t)$ is also assumed continuously differentiable for all times t , then $\tilde{x}[\tau, z, \xi]$ is continuously differentiable w.r.t. all of its inputs. For detailed properties of Φ , see Appendix A.2.2 recalling some useful classic results.

4.1.2.2 PDG as a NLP

To alleviate the writing, let us first consider that the constraint parameter p equals zero (the case with $p \neq 0$ will be handled afterwards).

The last ingredients that must be discretized in (4.6) are the inequality constraints. Indeed, constraints (4.6f) and (4.6g) are defined on an infinite number of points. We decide to enforce these constraints on a number of $N_c + 1$ times instance $\tau'_0 = 0 < \tau'_1 < \dots < \tau'_{N_c} = 1$ s.t. the subdivision $(\tau'_i)_{0 \leq i \leq N_c}$ is an uniform over-sampled version of $(\tau_i)_{0 \leq i \leq N}$, as shown in Figure 4.1. In other words, every interval $[\tau_i, \tau_{i+1}]$ is split into several sub-intervals, and the constraints (4.6f) and (4.6g) are enforced at their borders.

Thus, the discretized version of the inequality constraints can be re-written as

$$h(z, \xi) \leq 0$$

² $L^\infty([0, 1], \mathcal{U})$ denotes the set of essentially bounded measurable functions.

where

$$h(z, \xi) := \begin{pmatrix} \bar{u}(\tau'_0) + u_\mu(\tau'_0) - u^+(0) \\ u^-(0) - (\bar{u}(\tau'_0) + u_\mu(\tau'_0)) \\ c(\tilde{x}[\tau'_0, z, \xi], \bar{u}(\tau'_0) + u_\mu(\tau'_0), \eta + \Delta\eta, 0) \\ \vdots \\ \bar{u}(\tau'_{N_c}) + u_\mu(\tau'_{N_c}) - u^+(0) \\ u^-(0) - (\bar{u}(\tau'_{N_c}) + u_\mu(\tau'_{N_c})) \\ c(\tilde{x}[\tau'_{N_c}, z, \xi], \bar{u}(\tau'_{N_c}) + u_\mu(\tau'_{N_c}), \eta + \Delta\eta, 0) \\ z - z_{\text{up}} \\ z_{\text{low}} - z \end{pmatrix}. \quad (4.8)$$

Moreover, using the same notations, the equality constraints (4.6d) and (4.6e) are conveyed by the condition $g(z, \xi) = 0$ where

$$g(z, \xi) := \begin{pmatrix} A^f \tilde{x}[1, z, \xi] - b^f(0) \\ u_\mu(0) - \Delta u_{\text{init}} \end{pmatrix}. \quad (4.9)$$

Thus, we get an expression of the finite-dimensional constraints approximating PDG $(\xi, 0)$, i.e. for the special case where $p = 0$. The general case where $p \neq 0$ is then straightforward.

Indeed, there is no loss of generality in assuming that the constraint parameter p has a linear influence on the constraints of the original optimization problem, as explained below. For instance, for the 2D rocket model, the final horizontal position in the terminal constraint (4.6d) can be parametrized by a variable Δz^f , s.t.

$$A^f x(1) = b^f + \begin{pmatrix} 0 & 0 & \Delta z^f & 0 \end{pmatrix}^\top.$$

Likewise, to negotiate the incidence limit, the control bounds are changed by a variable $\Delta \alpha_{\text{max}}$ s.t.

$$u^- - (0, \Delta \alpha_{\text{max}})^\top \leq \bar{u}(t) + u_\mu(t) \leq u^+ + (0, \Delta \alpha_{\text{max}})^\top.$$

Therefore, using this assumption of linear influence of p , we assume that there are matrices H_p and B_p s.t. the original constraints $h(z, \xi) \leq 0$ and $g(z, \xi) = 0$ actually write

$$h(z, \xi) \leq H_p p \quad \text{and} \quad g(z, \xi) = B_p p$$

when $p \neq 0$. The matrices H_p and B_p are basically filled by zeros and a few ones.

To sum up all the preceding steps, PDG (ξ, p) is converted into a NLP as follows

- change the time variable from $t \in [0, t_f]$ to $\tau \in [0, 1]$,

- consider a parametric description $u_\mu(\cdot)$ of the infinite dimensional variable δu ,
- describe the dynamic equation and the initial condition constraints through Equation (4.7),
- enforce the inequality constraints on the time instances τ'_i for $i = 0, \dots, N_c$, instead of enforcing them for all $\tau \in [0, 1]$,
- enforce bounds on the decision variable z .

Definition 4. *The finite-dimensional approximation of the problem PDG (ξ, p) is defined by NLP (ξ, p) , which is the following non-linear optimization problem*

$$\min_z J(z, \xi) \quad (4.10a)$$

$$\text{s.t. } h(z, \xi) \leq H_p p, \quad (4.10b)$$

$$g(z, \xi) = B_p p. \quad (4.10c)$$

Remark 15 (State modeling choices). *Describing the state using Equation (4.7) is not usually recommended from a numerical point of view (e.g. [14]). Any iterative method aiming at solving NLP (ξ, p) (e.g. Successive Quadratic Programming [14]) requires the evaluation of $\tilde{x}[\tau, z, \xi]$ and its derivatives at each iteration, which means solving multiple ODEs at each iteration. However, as it will be detailed below, our goal is to provide an approximation of the solutions of NLP (ξ, p) using a (directional) first-order expansion. Therefore, it is only needed to evaluate these computationally expensive terms once and offline, making the use of $\tilde{x}[\tau, z, \xi]$ appropriate in this context.*

4.2 Sensitivity analysis for degenerate parametric NLP

Without loss of generality, to simplify the exposition, we consider in this section the special case where $p = 0$. The direct extension to the case $p \neq 0$ will be discussed in Section 4.3.

Let us study the standard problem (4.10), but where the constraint right-hand sides have been simplified s.t.

$$\text{NLP}(\xi) := \begin{cases} \min_z J(z, \xi) \\ \text{s.t. } h(z, \xi) \leq 0, \\ g(z, \xi) = 0. \end{cases}$$

In fact, NLP (ξ) is a parametric NLP formulated in a standard form [25, Eq. (1)]. When it is too hard to compute its exact solution, an alternative approach is to provide a *reasonable* approximation of it w.r.t. the parameter ξ , near a known solution. This section aims at answering the three following questions. First, if z^* denotes the optimal solution of NLP (ξ) , to what extent can one give a meaning to the expansion

$$“ z^*(\xi) = z^*(0) + \frac{dz^*}{d\xi}(0)\xi + o(\|\xi\|) ” ?$$

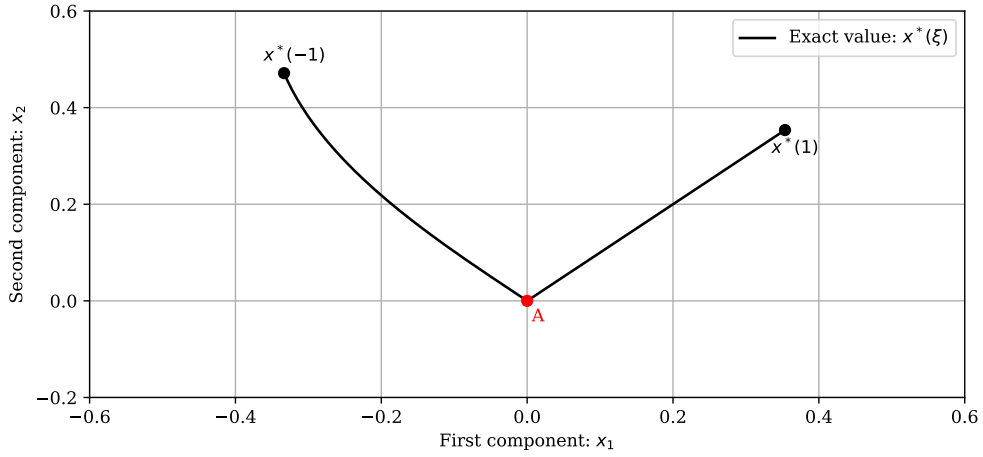


Figure 4.2: Representation of the optimal points of Basic Example 1 for $-1 \leq \xi \leq 1$, in the plane (x_1, x_2) . Point A is sometimes called a *cusp* in the literature [23].

Then, how can we compute this (local) expansion? Finally, how can this be applied to solve NLP (ξ) non-locally (at least approximately)?

4.2.1 An introductory toy example

Let us consider a low-dimensional example that will help us illustrate the challenges of NLP sensitivity. This toy problem has only two variables and one parameter ξ which appears non-linearly.

Basic Example 1. For a scalar parameter $\xi > 0$, consider the parametric NLP

$$\min_{x \in \mathbb{R}^2} \frac{1}{2}(x_1^2 + x_2^2) \quad (4.11)$$

$$s.t. \quad c_1(x, \xi) = -x_1 - x_2 + \frac{\xi}{\sqrt{1 + \xi^2}} \leq 0 \quad (4.12)$$

$$c_2(x, \xi) = x_1 - x_2\sqrt{1 + \xi^2} - \xi \leq 0 \quad (4.13)$$

This problem possesses a unique solution and can be solved analytically. The solution is illustrated in Figure 4.2. For $\xi > 0$, the constraint c_1 is active (with associated multiplier λ_1) and the optimum is

$$x_1^*(\xi) = \frac{\xi}{2\sqrt{1 + \xi^2}}, \quad x_2^*(\xi) = \frac{\xi}{2\sqrt{1 + \xi^2}}, \quad \lambda_1(\xi) = \frac{\xi}{2\sqrt{1 + \xi^2}}.$$

For $\xi < 0$, the constraint c_2 is active (with associated multiplier λ_2) and the optimum is

$$x_1^*(\xi) = \frac{\xi}{2 + \xi^2}, \quad x_2^*(\xi) = -\xi \frac{\sqrt{1 + \xi^2}}{2 + \xi^2}, \quad \lambda_2(\xi) = -\frac{\xi}{2 + \xi^2}.$$

Finally, for $\xi = 0$, since the point $(x_1, x_2) = (0, 0)$ globally minimizes the cost and satisfies the constraints, it is the optimum. Both constraints are active, and the associated multipliers equal 0.

A first remark is that for all values $\xi \neq 0$, the function $\xi \mapsto x^*(\xi)$ is continuously differentiable. This is pictured in Figure 4.2, and corresponds to the smooth parts of the black curve. Also, note that the multipliers of the active constraints are positive for $\xi \neq 0$. When $\xi = 0$, then $\lambda_1 = \lambda_2 = 0$ even though both constraints are active, which is what we will call a *degenerate* scenario following [48].

Then, the main point of interest is when $\xi = 0$, i.e. point *A* in Figure 4.2. The set of active constraints changes when the sign of ξ changes. The optimal solution x^* is not differentiable, but has a left-hand side and right-hand side derivatives. Therefore, the value of x^* can be inferred in the neighborhood of $\xi = 0$ using the following (directional) expansions, for $\varepsilon \geq 0$:

$$x^*(\varepsilon) = x^*(0) + \frac{dx^*}{d\xi}(0^+)\varepsilon + o(\varepsilon) \quad \text{and} \quad x^*(-\varepsilon) = x^*(0) - \frac{dx^*}{d\xi}(0^-)\varepsilon + o(\varepsilon).$$

The goal of the next sub-section is to highlight the conditions under which $x^*(\cdot)$ is at least directionally differentiable. The need to be able to handle degenerate scenarios for PDG - i.e. when some multipliers are zero when the input parameter is zero - will be demonstrated in Section 4.3, and is directly related to local changes in the active set of constraints.

4.2.2 Known results in parametric NLP sensitivity

In this section, we consider the problem NLP (ξ) from a general perspective, i.e. not necessarily expressed by the above-mentioned expressions for h and g . The goal is to present sufficient conditions that enable us to compute an expansion of the solutions of NLP (ξ). Without loss of generality, we seek an expansion in the neighborhood of $\xi = 0$.

First, after recalling necessary conditions for the existence of a minimizer, a classic theorem that gives conditions for the differentiability of the solution is recalled. A discussion on the key aspects of its proof stresses the need of a more general theorem, as its main assumption needs to be relaxed in view of application to PDG. With this view in mind, an alternative result from the literature, based on directional derivatives - a.k.a. Dini derivatives - is presented. Finally, computational aspects are discussed for the evaluation of the solution expansions.

4.2.2.1 Sensitivity analysis with Strict Complementary Slackness

Introduce the multipliers $\nu \in \mathbb{R}^{n_{\text{in}}}$ and $\lambda \in \mathbb{R}^{n_{\text{eq}}}$ and the Lagrangian

$$L(z, \nu, \lambda, \xi) := J(z, \xi) + \nu^\top h(z, \xi) + \lambda^\top g(z, \xi). \quad (4.14)$$

Classically, a tuple (z, ν, λ, ξ) is said to satisfy the Karush-Kuhn-Tucker (KKT) conditions³ if

$$L_z(z, \nu, \lambda, \xi) = 0, \quad (4.15a)$$

$$g(z, \xi) = 0, \quad (4.15b)$$

$$\nu^\top h(z, \xi) = 0 \quad \text{and} \quad \nu \geq 0. \quad (4.15c)$$

³Recall that the derivatives are denoted by putting the variable of differentiation as an index, e.g. $L_z = \partial L / \partial z$.

At $\xi = 0$, the multipliers are denoted ν_0 and λ_0 and are assumed to satisfy the KKT conditions.

The compact notation $L[0] = L(z_0, \nu_0, \lambda_0, 0)$ is used if necessary to alleviate the writing. Likewise, $h[0] = h(z_0, 0)$, $h_z[0] = h_z(z_0, 0)$, etc.

Lemma 5 (Second-Order Sufficient Conditions (SOSC), [37, Lemma 3.2.1]). *If the functions defining Problem NLP (0) are twice continuously differentiable in a neighborhood of z_0 , if there exist multipliers $\nu_0 \in \mathbb{R}^{n_{\text{in}}}$ and $\lambda_0 \in \mathbb{R}^{n_{\text{eq}}}$ s.t. the KKT conditions (4.15) hold and, further, if*

$$\kappa^\top L_{zz}(z_0, \nu_0, \lambda_0, 0)\kappa > 0 \quad (4.16)$$

for any non-zero vector $\kappa \in \mathbb{R}^{N_z}$ that satisfies

$$[h_i]_z(z_0, 0)\kappa \leq 0, \quad \forall i : h_i(z_0, 0) = 0, \quad (4.17a)$$

$$[h_i]_z(z_0, 0)\kappa = 0, \quad \forall i : (\nu_0)_i > 0, \quad (4.17b)$$

$$g_z(z_0, 0)\kappa = 0. \quad (4.17c)$$

then, z_0 is a strict local minimizing point of $\mathcal{P}(0)$.

Definition 5 (SCS). *For a pair (z, ν) , Strict Complementary Slackness (SCS) holds when*

$$\nu_i > 0, \quad \forall i : h_i(z, 0) = 0.$$

The conditions from Lemma 5 and the Strict Complementary Slackness condition allow one to present the following theorem, adapted from [37, Thm. 3.2.2], which states a well-known NLP sensitivity result.

Theorem 2 (Continuous differentiability, with SCS from [37]). *Assume that J , h and g are twice continuously differentiable in z and that their gradients w.r.t. z and the constraints are continuously differentiable in ξ in a neighborhood of $(z_0, 0)$. If*

(i) $[h_i]_z(z_0, 0)$ (for i s.t. $h_i(z_0, 0) = 0$) and $[g_j]_z(z_0, 0)$ (all j) are linearly independent,

(ii) the conditions of Lemma 5 are satisfied at z_0 for the multipliers ν_0 and λ_0 ,

(iii) SCS holds for (z_0, ν_0) ,

then

(a) z_0 is a local isolated minimizing point of problem NLP (0) and the associated Lagrange multipliers ν_0 and λ_0 are unique,

(b) for ξ in a neighborhood of 0, there exists a unique, once continuously differentiable vector function $y(\xi) = [z^*(\xi), \nu^*(\xi), \lambda^*(\xi)]^\top$ satisfying the second-order sufficient conditions for a local minimum of NLP (ξ) s.t. $y(0) = (z_0, \nu_0, \lambda_0)$, and hence $z^*(\xi)$ is a locally unique local minimum of NLP (ξ) with associated unique Lagrange multipliers $\nu^*(\xi)$ and $\lambda^*(\xi)$,

(c) for ξ near 0, the set of binding inequalities is unchanged, SCS holds, and the binding constraint gradients are linearly independent at $z^*(\xi)$.

The proof of this theorem conveys several key features helping to understand what makes the solution z^* continuously differentiable. Precisely, Equation (4.19) below is a highly useful by-product of the proof, providing an explicit formula for the derivative of z^* and the associated multipliers.

Remark 16. *The SCS assumption, combined with the conditions of Lemma 5, implies that $L_{zz}[0]$ is positive definite on the kernels of both matrices $h_z^a[0]$ and $g_z[0]$. Denoting by N_K the dimension of the intersection of these kernels, and by $Q \in \mathbb{R}^{N_z \times N_K}$ the matrix that generates this vector space, the conditions (4.16) and (4.17) can be re-written equivalently as $Q^\top L_{zz}[0]Q \succ 0$.*

Sketch of proof of Theorem 2, adapted from [37] and [25]. The idea of the proof is to use the Implicit Function Theorem (IFT) on a subset of the KKT conditions, which defines functions $z^*(\xi)$, $\nu^*(\xi)$, $\lambda^*(\xi)$ that are then shown to be locally unique minimizers of Problem NLP (ξ) using the conditions of Lemma 5.

At z_0 , the active inequality constraints are denoted with an exponent “ a ”. Their cardinal is denoted n_a . For instance, h^a denotes the rows of h that are active at z_0 , and ν_0^a are the corresponding multipliers. Without loss of generality, we assume that the active inequality constraints are the first n_a components of h and ν .

The main function of interest in this proof is⁴

$$K(z, \nu^a, \lambda, \xi) := \begin{pmatrix} L_z(z, \nu^a, \lambda, \xi) \\ \nu^a \bullet h^a(z, \xi) \\ g(z, \xi) \end{pmatrix} = \begin{pmatrix} J_z(z, \xi) + (\nu^a)^\top h_z^a(z, \xi) + \lambda^\top g_z(z, \xi) \\ \nu_1 h_1(z, \xi) \\ \vdots \\ \nu_{n_a} h_{n_a}(z, \xi) \\ g(z, \xi) \end{pmatrix}$$

and is often referred to as the *Kuhn-Tucker matrix* [25]. The conditions of Lemma 5 imply that the matrix

$$\frac{\partial K}{\partial [z, \nu^a, \lambda]}(z_0, \nu_0^a, \lambda_0, 0) = \begin{pmatrix} L_{zz} & [h_1]_z^\top & \dots & [h_{n_a}]_z^\top & g_z^\top \\ (\nu_0)_1 [h_1]_z & h_1 & & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ (\nu_0)_{n_a} [h_{n_a}]_z & 0 & & h_{n_a} & \\ g_z & 0 & \dots & & 0 \end{pmatrix} \quad (4.18)$$

is invertible. Note that all the elements on the right-hand side of the latter matrix are evaluated on $(z_0, \nu_0^a, \lambda_0, 0)$, but this has been omitted to alleviate the writing.

⁴ $x \bullet y$ denotes the component-wise product of vectors x and y , a.k.a. the Hadamard product.

Thus, the IFT applies on the condition $K(z, \nu^a, \lambda, \xi) = 0$ at $(z_0, \nu_0^a, \lambda_0, 0)$, which guarantees the existence of a differentiable function

$$y : \xi \mapsto (z^*(\xi), \nu^*(\xi), \lambda^*(\xi))$$

s.t. $z^*(0) = z_0$, $\nu^*(0) = \nu_0$, $\lambda^*(0) = \lambda_0$ and $K(z^*(\xi), (\nu^*)^a(\xi), \lambda^*(\xi), \xi) = 0$ in the neighborhood of $\xi = 0$.

Finally, the *strict* inequalities of SCS enables us to show that the set of active constraints does not change in the neighborhood of $\xi = 0$, which implies that $y(\xi)$ locally satisfies the sufficient conditions of Lemma 5, hence the theorem. \square

This proof has a useful by-product. Indeed, the construction of the triplet $(z^*(\xi), \nu^*(\xi), \lambda^*(\xi))$ via the use of the IFT directly provides the derivatives of these variables at $\xi = 0$

$$\begin{pmatrix} \frac{dz^*}{d\xi}(0) \\ \frac{d(\nu^*)^a}{d\xi}(0) \\ \frac{d\lambda^*}{d\xi}(0) \end{pmatrix} = - \left(\frac{\partial K}{\partial [z, \nu^a, \lambda]}(z_0, \nu_0^a, \lambda_0, 0) \right)^{-1} \left(\frac{\partial K}{\partial \xi}(z_0, \nu_0^a, \lambda_0, 0) \right). \quad (4.19)$$

However, as we have seen with Toy Example 1, SCS does not necessarily hold everywhere, which becomes a problem in the above-mentioned arguments. Indeed, if one of the multipliers of the active constraints becomes zero, then the corresponding line in Equation (4.18) becomes zero, leading to a singular Kuhn-tucker matrix. In this scenario, the IFT does not apply anymore.

4.2.2.2 Sensitivity analysis *without* Strict Complementary Slackness

To overcome the above-mentioned difficulty, another set of assumptions is needed.

Definition 6 (Strong SOSC). *There exists a scalar $a > 0$ s.t.*

$$\kappa^\top L_{zz}(z_0, \nu_0, \lambda_0, 0)\kappa \geq a\|\kappa\|^2 \quad (4.20)$$

for any $\kappa \in \mathbb{R}^{N_z}$ s.t.

$$\begin{aligned} g_z(z_0, 0)\kappa &= 0, \\ [h_j]_z(z_0, 0)\kappa &= 0, \quad \forall j \text{ s.t. } h_j(z_0, 0) = 0 \quad \text{and} \quad (\nu_0)_j > 0. \end{aligned}$$

The SOSC property is weaker than the SCS property. As shown by Jittorntrum [48], relaxing the SCS using Strong SOSC instead eventually leads to directional differentiability properties. To formulate this, let us introduce the following. For a function $\gamma : \mathbb{R}^p \rightarrow \mathbb{R}^q$, when it exists, the upper Dini derivative of γ at x in the direction d is a vector of \mathbb{R}^q and is denoted by

$$D_d^+ \gamma(x) := \lim_{\varepsilon \downarrow 0} \frac{\gamma(x + \varepsilon d) - \gamma(x)}{\varepsilon} \in \mathbb{R}^q.$$

For any given direction ξ , consider the *directional* problem NLP $(\varepsilon\xi)$, where $\varepsilon \geq 0$ is a scalar.

Theorem 3 (Directional differentiability, without SCS). *Let us assume that J , h and g are twice continuously differentiable in a neighborhood of z_0 . If*

(i) $[h_i]_z(z_0, 0)$ (for i s.t. $h_i(z_0, 0) = 0$) and $[g_j]_z(z_0, 0)$ (all j) are linearly independent,

(ii) Strong SOSC holds at z_0 ,

then there exists a unique continuous function $\varepsilon \rightarrow (z^*(\varepsilon\xi), \nu^*(\varepsilon\xi), \lambda^*(\varepsilon\xi))$ that locally minimizes $\text{NLP}(\varepsilon\xi)$, for $\varepsilon \geq 0$, s.t. $z^*(0) = z_0$, $\nu^*(0) = \nu_0$ and $\lambda^*(0) = \lambda_0$. Furthermore, its right-hand derivative at $\varepsilon = 0$ exists, i.e. the upper Dini derivatives $D_{\xi^+} z^*(0)$, $D_{\xi^+} \nu^*(0)$ and $D_{\xi^+} \lambda^*(0)$ exist.

For a proof of this theorem, see the discussions after the Theorems 3 and 4 in [48]. A direct consequence of Theorem 3 is that for any ξ , for $\varepsilon \geq 0$, the following expansion holds:

$$z^*(\varepsilon\xi) = z_0 + D_{\xi^+} z^*(0)\varepsilon + o(\varepsilon). \quad (4.21)$$

The same way Theorem 2 provided an expression for the derivatives of the solution tuple, the proof of Theorem 3 by Jittorntrum provides a way to compute the latter expansion, which will be used later in Equation (4.25). While Theorem 2 requires to inverse a linear system as shown in Equation (4.19), Theorem (3) needs to solve a QP, as indicated in the Proposition below.

Proposition 10 (Adapted from [48, Eq. 24]). *The vector $D_{\xi^+} z^*(0)$ in Equation (4.21) is the optimal solution of*

$$\begin{aligned} \min_{\Delta z \in \mathbb{R}^{N_z}} \quad & \frac{1}{2} \Delta z^\top L_{zz}[0] \Delta z + \xi^\top L_{\xi z}[0] \Delta z \\ \text{s.t.} \quad & h_z^a[0] \Delta z + h_\xi^a[0] \xi \leq 0, \\ & g_z[0] \Delta z + g_\xi[0] \xi = 0, \end{aligned}$$

whose uniqueness is guaranteed by Assumption (ii) in Theorem 3.

Proposition 10 provides a way to compute a local, directional, expansion of the optimal solution $z^*(\xi)$ using QP. Qualitatively, it boils down to linearizing the active inequality constraints, forgetting the inactive inequality constraints, linearizing the equality constraints, and taking into account the second-order approximation of the cost. Therefore, aside from the non-linear aspect of $\text{NLP}(\xi)$, the only missing features of Problem $\text{NLP}(\xi)$ in Proposition 10 are the inactive inequality constraints. This result can be improved and made more useful for practical applications.

To alleviate the writing, and without loss of generality, we assume for the rest of this section that $z_0 = 0$. If instead of considering only the active inequality constraints we consider all inequality constraints, i.e. that we solve

$$\text{QP}(\xi) := \min_{z \in \mathbb{R}^{N_z}} \frac{1}{2} z^\top L_{zz}[0] z + \xi^\top L_{\xi z}[0] z \quad (4.22a)$$

$$\text{s.t.} \quad h[0] + h_z[0] z + h_\xi[0] \xi \leq 0 \quad (4.22b)$$

$$g_z[0] z + g_\xi[0] \xi = 0 \quad (4.22c)$$

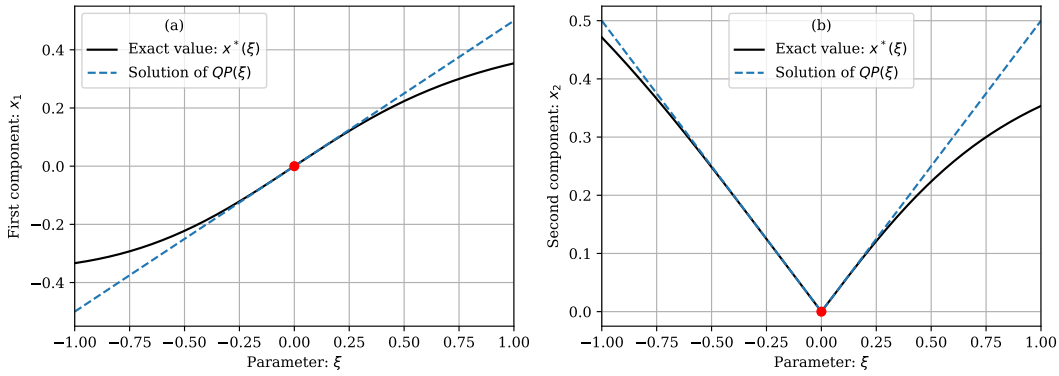


Figure 4.3: Comparison between the exact solution of Example 1 and the solution returned by QP (ξ), for $-1 \leq \xi \leq 1$.

then we have a more useful tool. On one hand, the result of Proposition 10 is preserved. Indeed, denote by i any (strictly) inactive inequality constraint at $\xi = 0$. Consider any arbitrary value of ξ . Since $\varepsilon \mapsto z^*(\varepsilon\xi)$ is continuous on a certain right-neighborhood of $\varepsilon = 0$ by Theorem 3, then $h_i(z^*(\varepsilon\xi), \varepsilon\xi) < 0$ holds in this neighborhood. By reducing this neighborhood if necessary, the condition

$$h_i[0] + [h_i]_z[0]z^*(\varepsilon\xi) + [h_i]_\xi[0]\varepsilon\xi < 0$$

holds as well in the vicinity of $\varepsilon = 0$ and will not affect the local property of Proposition 10. On the other hand, the linearized conditions (4.22b) provide a linear approximation of the inequality constraints, giving a non-local estimation of changes in the active set⁵.

As an intermediate summary, if a parametric NLP described by NLP (ξ) satisfies Strong SOSC and some other mild conditions described in Theorem 3, its solution point is locally uniquely defined, and admits a directional first-order expansion, that can be computed by solving QP (ξ) detailed in (4.22). This latter Quadratic Program consists simply in modifying NLP (ξ) by linearizing the constraints in both the parameter and the decision variable, and by taking a second-order expansion of the cost. Locally, the approximations provided by QP (ξ) are very accurate (in the sense of (4.21)), and can handle active set changes that depend on the direction ξ . Globally, QP (ξ) can keep providing an approximation of the solution, that will be as good as the approximations made in (4.22b) and (4.22c) are.

Remark 17. A direct application of QP (ξ) to the Basic Example 1 is shown in Figure 4.3.

Remark 18. Similar methods aiming at computing an expansion of $z^*(\xi)$ can be found in more recent work. See for example the work of Bonnans & Shapiro [19, Sec. 5.2].

Remark 19. From a high-level point of view, this Chapter describes how to convert an infinite dimensional problem into a finite one, which in turns is solved using

⁵See [25, Sec. 4.3] for a complete discussion on that aspect.

parametric sensitivity analysis. However, the converse approach which consists in performing parametric sensitivity analysis on the infinite dimensional problem directly is an alternative that has been explored in the literature. See e.g. [32], that describes how directional differentiability can be used on the stationary conditions of OCPs with multiple types of constraints.

4.3 Fast nominal guidance method

Up to here, we have presented a method to re-formulate the original problem PDG (ξ, p) into the finite-dimensional problem NLP (ξ, p) . For $p = 0$, the solutions of NLP $(\xi, 0)$ are approximated by solving QP (ξ) .

Let us extend this sensitivity-based method to the case where $p \neq 0$, and summarize how the newly formed problem QP (ξ, p) is intended to be used in practice to provide nominal guidance.

4.3.1 An offline/online approach for nominal guidance

First, note that the role of p in the definition of NLP (ξ, p) is equivalent to the one of ξ , in the sense that both are given parameters. Thus, by introducing convenient intermediate matrices and vectors, we can extend the QP from Equation (4.22) to the general case under the form

$$\text{QP}(\xi, p) := \begin{cases} \min_{z \in \mathbb{R}^{N_z}} & \frac{1}{2} z^\top P z + \xi^\top Q z \\ \text{s.t.} & G z \leq h_0 + H_\xi \xi + H_p p \\ & A z = b_0 + B_\xi \xi + B_p p \end{cases} \quad (4.23)$$

where H_p and B_p are the same as the ones used in the definition of the constraints of NLP (ξ, p) and where

$$\begin{aligned} P &:= J_{zz}(0, 0) & Q &:= J_{\xi z}(0, 0) \\ G &:= h_z(0, 0) & A &:= g_z(0, 0) \\ h_0 &:= -h(0, 0) & H_\xi &:= -h_\xi(0, 0) \\ b_0 &:= -g(0, 0) & B_\xi &:= -g_\xi(0, 0) \end{aligned}$$

It is this expression of QP (ξ, p) that will be used from now on.

Remark 20. Note that the reason why p does not appear in the cost of QP (ξ, p) is that the cost J in Equation (4.10a) does not depend on p , but only on z and ξ .

Remark 21 (Shortcuts). In the remainder of this thesis, the dependency of the linear part of the cost and the constraint right-hand side on ξ in QP (ξ) may be omitted to alleviate the writing, by using the vectors q , h and b s.t.

$$q := J_{\xi z}(0, 0)^\top \xi, \quad h := h_0 + H_\xi \xi, \quad \text{and} \quad b := b_0 + B_\xi \xi.$$

To maximize computational efficiency, the constant matrices of QP (ξ, p) are computed *offline*, for a reference trajectory, using the transition matrices formulas from the Appendix. As described in Section 4.1, such a reference trajectory

$(\bar{x}, \bar{u}, \bar{\eta}, \bar{t}_f)$ must satisfy the constraints of NLP (ξ, p) , and its design depends critically on the mission goals. The reference may be defined as the solution of a more complex optimization problem, out-of-the-scope of the thesis, solved using indirect methods [24]. Its computation can take anything from several seconds to hours, depending on the desired accuracy, the optimization solver or the computer used for it. On the other hand, computing the transition matrices, even to a good accuracy, only needs a few seconds, even with a non efficiency-optimized code.

4.3.2 Guidance law

As introduced in (4.3), the optimal values μ^* and Δt_f^* returned by QP (ξ, p) via z^* enable us to describe the guidance control law as a continuous time function. Indeed, interpreting \bar{u} and u_μ as functions defined on $[0, 1]$, the guidance law becomes

$$u^*(t) = \bar{u} \left(\frac{t}{\bar{t}_f + \Delta t_f^*} \right) + u_{\mu^*} \left(\frac{t}{\bar{t}_f + \Delta t_f^*} \right), \quad \forall t \in [0, \bar{t}_f + \Delta t_f^*]. \quad (4.24)$$

4.3.3 Directional first-order estimate of waypoints

Let us denote by $z_{\text{nlp}}(\xi, p)$ the hypothetical solution of NLP (ξ, p) and by $z_{\text{qp}}(\xi, p)$ the solution of QP (ξ, p) . Then, from Section 4.2, the following directional expansion holds

$$z_{\text{nlp}}(\varepsilon\xi, \varepsilon p) = z_{\text{qp}}(\varepsilon\xi, \varepsilon p) + o(\varepsilon).$$

Moreover, using the definition of the augmented state $\tilde{x}[\tau, z, \xi]$ from Equation (4.7), for any $\tau \in [0, 1]$ we have

$$\tilde{x}[\tau, z, \xi] = \begin{pmatrix} \bar{x}[\tau] \\ \bar{t}_f \end{pmatrix} + \frac{\partial \tilde{x}}{\partial z}[\tau, 0, 0]z + \frac{\partial \tilde{x}}{\partial \xi}[\tau, 0, 0]\xi + o(\|(z, \xi)\|).$$

Let us introduce \tilde{x}_{lin} as

$$\tilde{x}_{\text{lin}}(\tau, \xi, p) := \begin{pmatrix} \bar{x}[\tau] \\ \bar{t}_f \end{pmatrix} + \frac{\partial \tilde{x}}{\partial z}[\tau, 0, 0]z_{\text{qp}}(\xi, p) + \frac{\partial \tilde{x}}{\partial \xi}[\tau, 0, 0]\xi \in \mathbb{R}^{n+1}. \quad (4.25)$$

Composing the previous expressions yields the directional first-order expansion of the state

$$\tilde{x}[\tau, z_{\text{nlp}}(\varepsilon\xi, \varepsilon p), \varepsilon\xi] = \tilde{x}_{\text{lin}}(\tau, \varepsilon\xi, \varepsilon p) + o(\varepsilon).$$

Since \tilde{x}_{lin} is an approximation of the *augmented* state, it can be split in half: the first n components form the state approximation x_{lin} , and the last component form the time-of-flight approximation t_f^{lin}

$$\tilde{x}_{\text{lin}}(\tau, \xi, p) = \begin{pmatrix} x_{\text{lin}}(\tau, \xi, p) \\ t_f^{\text{lin}}(\tau, \xi, p) \end{pmatrix}.$$

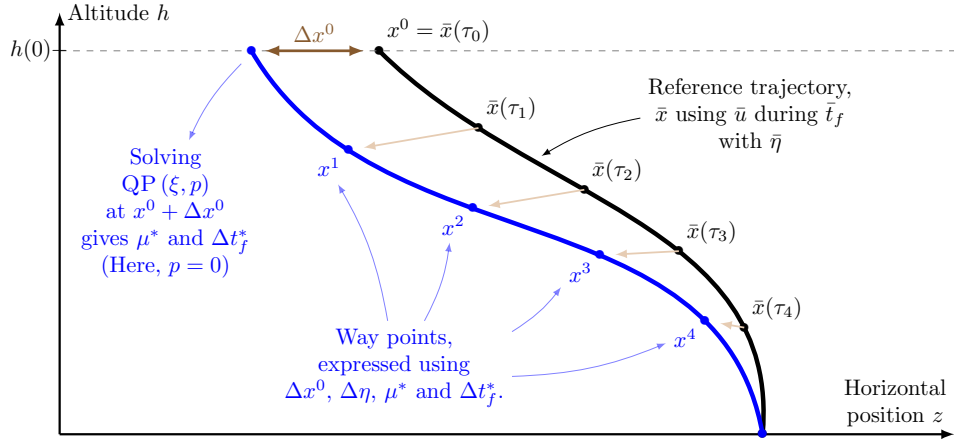


Figure 4.4: Summary of the nominal guidance method, as presented in Chapter 4.

Note that t_f^{lin} does not actually depend on τ by construction of the augmented state, and Equation (4.25) applied to t_f^{lin} simply yields $t_f^{\text{lin}}(\xi, p) = \bar{t}_f + \Delta t_f^*(\xi, p)$.

In practice, the solution z^* of QP (ξ, p) can be used either to form the guidance law from Equation (4.24), or by computing *waypoints* x^k using the state approximation provided by x_{lin} s.t.

$$x^k := x_{\text{lin}}(\tau_k, \xi, p) \quad (4.26)$$

where $(\tau_k)_k$ is a prescribed subdivision of $[0, 1]$ of normalized time instance. The values τ_k are equivalent to the non-normalized time instances t_k s.t.

$$t_k = \tau_k \cdot (\bar{t}_f + \Delta t_f^*(\xi, p)).$$

4.4 Numerical examples

Let us discuss the three following examples, each highlighting a distinct feature of the solutions of QP (ξ, p) . Since the case where $p \neq 0$ is the matter of Chapter 5, these examples focus exclusively on nominal descents, i.e. where $p = 0$.

For all the examples, the cost J that we consider is defined directly by giving its matrices P and Q . We consider that $Q = 0$, and that the matrix P is a diagonal matrix of positive weights, which favors early corrections (i.e. the weight associated to $u(\tau_0)$ is smaller than the one of $u(\tau_N)$).

The reference trajectory chosen for each example is represented by a black line. Moreover, for the examples using the 3D model, the reference trajectory is assumed to lie in the plane $(\mathbf{e}_z, \mathbf{e}_h)$ (though corrections implying out of plane trajectories are discussed in the last example).

The data have been normalized for all the examples.

4.4.1 Effectiveness of calculated guidance

The first example aims at showing that in some neighborhood of the reference parameters, the expansion (4.25) provides accurate corrections enforcing the terminal constraints, even when using the corrections in *open-loop* only. It is directly illustrated on the 3D rocket model.

Figures 4.5 illustrate a dispersion of the input variable ξ along a single component, namely Δv_h^0 , for both positive and negative values. The inputs are voluntarily dispersed over a small range of values.

Due to the terminal constraints, the six states corresponding to the positions and speeds are expected to be null at the final time (except $v_h(t_f) = -\varepsilon_{v_h}^f$). If the reference guidance law (\bar{u}, \bar{t}_f) is applied directly to a scenario where $\xi \neq 0$ - i.e. no corrections are applied - then strong constraint violation errors will appear. However, if the change in parameter is corrected using u^* from (4.24) (i.e. using QP (ξ, p)), the terminal constraints are supposed to be approximately satisfied up to the first-order. These two behaviors are well observed in the sub-figure (a) of Figure 4.5, which represents the terminal horizontal position $z(t_f)$. The other terminal constraints (on y, h, v_z, v_y and v_h) have voluntarily been omitted, as their correction curve is much flatter than for z . In other words, it means that even though this terminal constraint component has the worst open-loop correction curve, it still demonstrates that the first-order corrections brought by QP (ξ, p) work well in a non-trivial neighborhood of $\xi = 0$ in practice.

4.4.2 Changes in the active set

The second example aims at showing that the optimal solutions of QP (ξ, p) are indeed only Dini-differentiable and not smooth, even in a standard landing scenario. Let us use the 2D rocket model, and consider that ξ only varies in horizontal position $\Delta z^0 = z^0 - \bar{z}^0$.

The directional-derivatives of α at the middle point t_2 in the two directions $\Delta z^0 = 1$ and $\Delta z^0 = -1$ differ, as shown in Figure 4.6-(d). This behavior has a real-world interpretation. When $\Delta z^0 > 0$, using more incidence on $\alpha(t_2)$ is not a possible option as the constraint is already active and becomes strictly active (the associated multiplier becomes positive when $\Delta z^0 > 0$). However, when $\Delta z^0 < 0$, lowering $\alpha(t_2)$ is possible, allowing the presented corrections.

4.4.3 Non-local constraint satisfaction

This third example aims at showing the behavior of QP (ξ, p) for large values of ξ . Let us consider the 3D model, with a reference trajectory contained in the plane $(\mathbf{e}_z, \mathbf{e}_h)$.

As illustrated in Figure 4.7, we are here interested in the behavior of QP (ξ, p) for values of ξ covering a grid in $(\Delta z^0, \Delta y^0)$, the change in initial horizontal positions. Considering a such choice of inputs allows us to stress that QP (ξ, p) is able to compute out-of-plane trajectories.

As pointed out in the sub-Figures 4.7-(c) and (d), non-local constraints are activated for sufficiently large values of the input, as demonstrated by the trajectories that reach the incidence bounds.

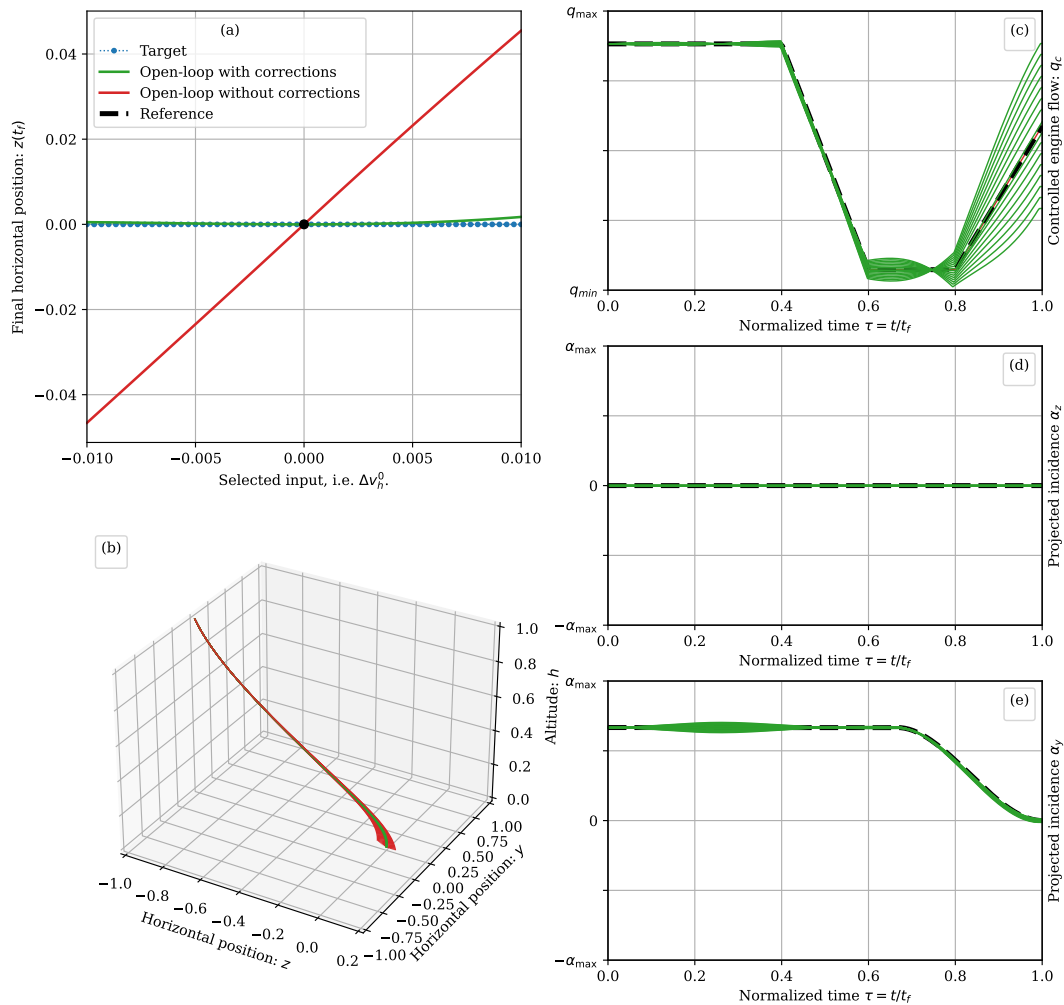


Figure 4.5: *First-order correction, for Δv_h^0 varying in the vicinity of 0.* The sub-Figure (a) is the main purpose of this example, showing that the green curve is a second order residual w.r.t. to the input Δv_h^0 . Note that sub-Figure (c) shows very late corrections in the engine flow. The weighting matrix P has been chosen to favor *early* corrections. Since QP (ξ, p) returned *late* corrections anyway, it means that earlier flow corrections would have required even higher incidence corrections (which is partially explained by the high dynamic pressure at the beginning of the descent).

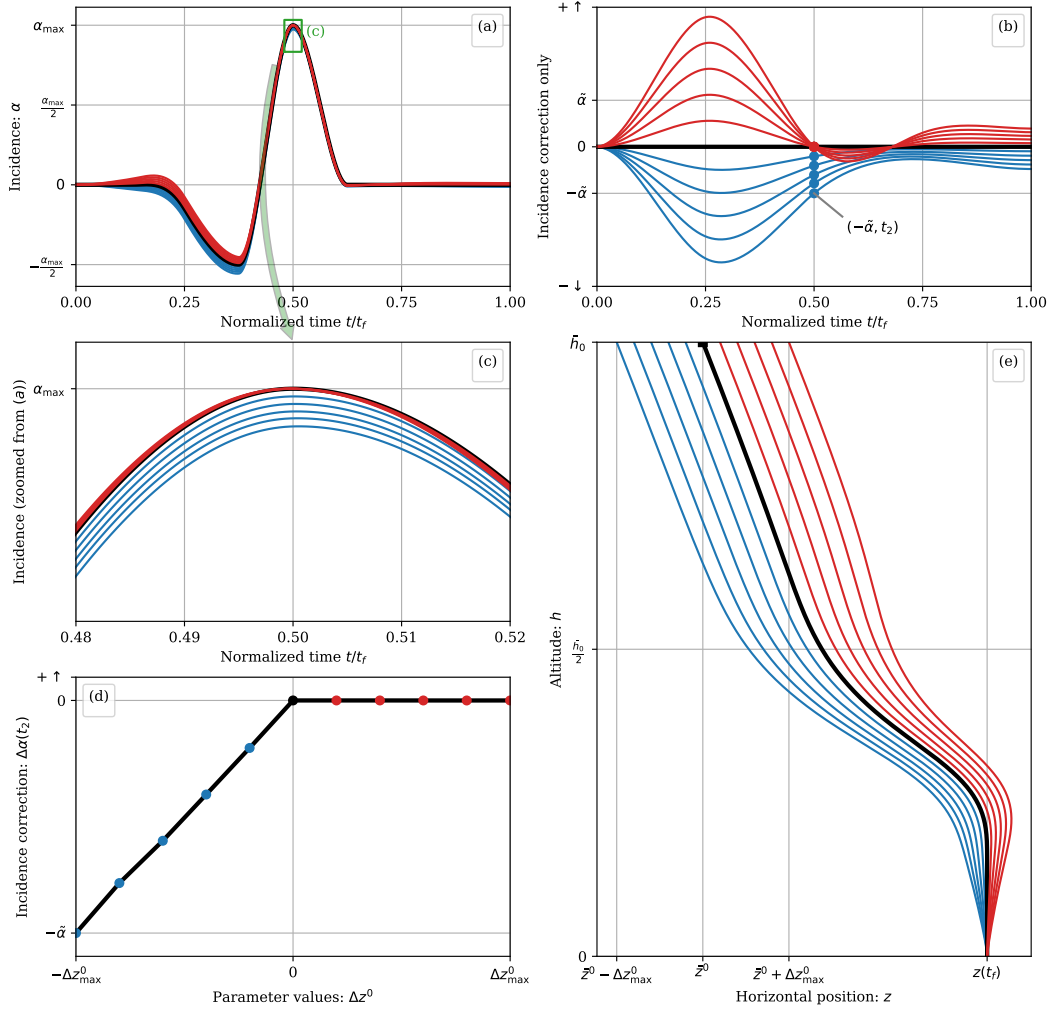


Figure 4.6: Changes in the input variable leading to local changes in the active set. The input variable used in this example is Δz^0 .

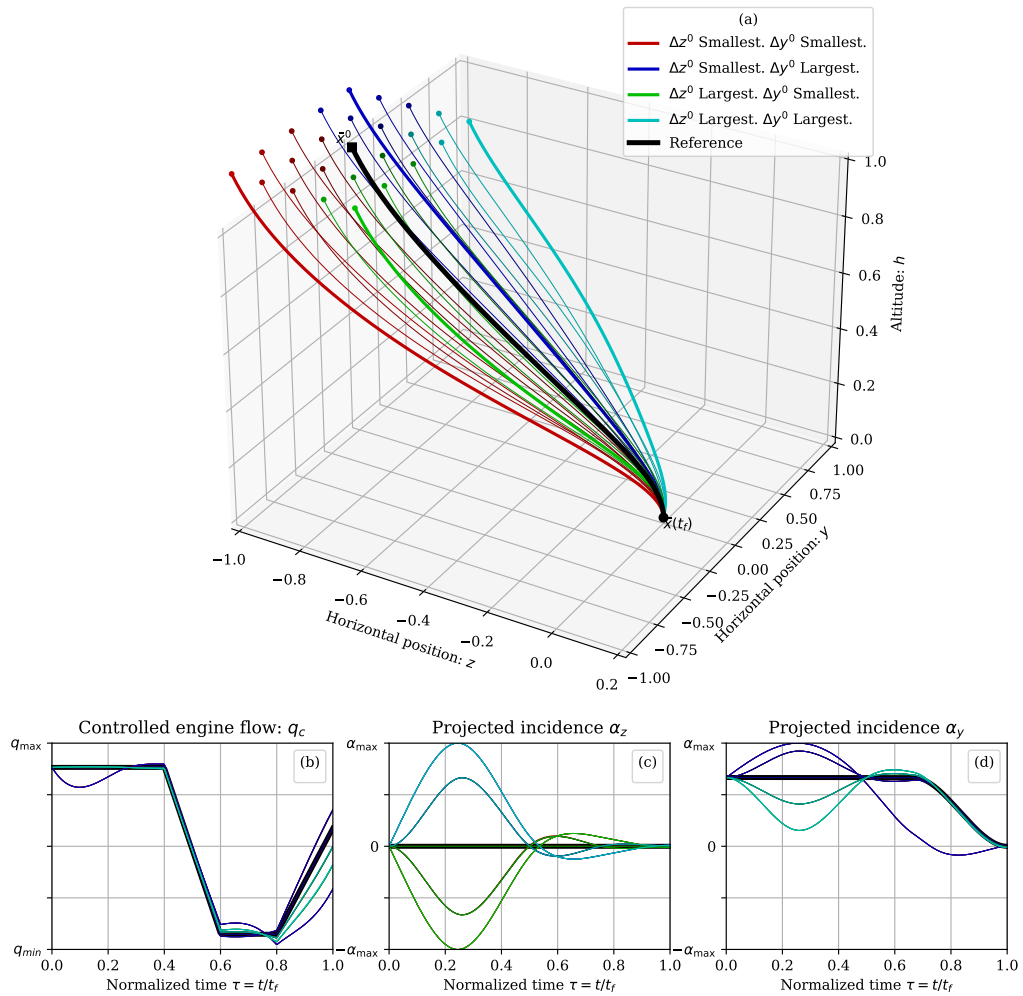


Figure 4.7: *Non-local behavior of QP (ξ, p), when Δz^0 and Δy^0 change. For large values of Δz^0 and Δy^0 , several constraints start to be triggered, such as the incidence bounds (upper and/or lower bounds).*

If one considers even larger values of ξ in this example, it comes a point where the constraints of $\text{QP}(\xi, p)$ are infeasible, at least as long as we keep $p = 0$. Hence the need for an algorithm that computes the proper value for p according to a hierarchy of objectives, which will be the matter of the next chapter.

Chapter 5

Emergency guidance via Linear and Quadratic Programming

Résumé

Le problème QP $(\xi, 0)$ a été le sujet principal jusqu'à présent. À un moment donné, si ξ est trop grand, alors QP $(\xi, 0)$ peut être infaisable. Par exemple, si la position horizontale initiale est trop éloignée du site d'atterrissage, il n'est pas possible de concevoir une trajectoire qui satisfasse toutes les contraintes en même temps : ces dernières sont incompatibles. La question centrale de cette thèse est donc la suivante : que faire dans cette situation ?

Lorsque QP (ξ, p) est infaisable à $p = 0$ et pour une valeur donnée de ξ , le problème de l'atterrissage doit être modifié pour retrouver la faisabilité, dans une certaine mesure. Cela peut se faire en relâchant les contraintes, c'est-à-dire en modifiant p . Une stratégie de relaxation doit être définie.

Tout d'abord, nous identifions les principaux *paramètres négociables* qui peuvent être relâchés, dans la Section 5.1, et nous soulignons leur importance relative. Cette liste ordonnée découle des connaissances préalables et de la compréhension commune des ingénieurs chargés de la réussite de la mission.

Ensuite, à partir de cette liste ordonnée, une suite de problèmes d'optimisation visant à minimiser l'amplitude des paramètres négociables est présentée dans la Section 5.2 et nommée HEGO pour *Optimisation Hiérarchique pour le Guidage d'Urgence*. L'utilisation de cette suite produit une trajectoire de guidage, tout en imposant une hiérarchie prescrite entre les paramètres négociables sélectionnés, au sens de l'ordre extended colexicographic order présenté au Chapitre 1. La formulation des problèmes sous-jacents repose uniquement sur la programmation linéaire et quadratique. De plus, des garanties théoriques sur le caractère bien posé et la régularité de la fonction qui à ξ associe la trajectoire optimale sont fournies. Comme il est d'un grand intérêt pour les applications pratiques, où la régularité des méthodes numériques est toujours souhaitable, il est démontré que cette dernière fonction est Lipschitz-continue, évitant ainsi les sauts dans la trajectoire de guidage pour des entrées arbitrairement proches. D'autres propriétés peuvent être établies, et la monotonie

directionnelle de l'amplitude du paramètre de négociation est examinée dans un cas particulier.

L'Algorithme HEGO est généralisé dans la Section 5.6. Cette généralisation permet de distinguer ce qui est lié à la méthode de guidage nominal sous-jacente de ce qui est propre aux problèmes de négociation eux-mêmes. On montre notamment comment HEGO pourrait être utilisé avec un autre choix de méthode de guidage nominal (i.e. au lieu de $QP(\xi, p)$), parmi les méthodes directes pour les OCP.

Enfin, des résultats numériques sont présentés dans la Section 5.7. Ils mettent en avant divers aspects de la méthode de guidage d'urgence, à la fois sur les modèles de fusée 2D et 3D. Plusieurs ensembles de paramètres négociables sont utilisés pour illustrer les différentes stratégies d'urgence accessibles par la méthodologie HEGO ainsi proposée.

The problem $QP(\xi, 0)$ has been the main topic so far. At some point, if ξ is too large, then $QP(\xi, 0)$ may be infeasible. For instance, if the initial horizontal position is too far from the landing site, then it is not possible to design a trajectory that satisfies all the constraints at the same time: they are inconsistent. Hence, the central question of this thesis is: what should one do in this case?

When $QP(\xi, p)$ is infeasible at $p = 0$ for a given value of ξ , the landing problem has to be modified to recover feasibility to some extent. This can be done by revising the constraints, i.e. by modifying p . Some revision strategy has to be defined.

First, we identify the main *negotiable parameters* that can be relaxed, in Section 5.1, and emphasize their relative importance. This ordered list stems from prior knowledge and common understanding between engineers in charge of the mission success.

Then, from this ordered list, a sequence of optimization problems aiming at minimizing the magnitude of the negotiable parameters is introduced in Section 5.2 and denoted HEGO for *Hierarchical Emergency Guidance Optimization*. Using the sequence produces a guidance trajectory, while enforcing a prescribed hierarchy between the selected negotiable parameters, in the sense of the extended colexicographic order introduced in Chapter 1. Formulating the underlying problems relies only on Linear and Quadratic Programming. Moreover, theoretical guarantees on the well-posedness and the smoothness of the mapping from ξ to the optimal trajectory are provided. As it is of high interest for a practical application where smoothness of numerical calculation procedures is always desirable, the latter map is shown to be Lipschitz-continuous, preventing *jumps* in the guidance trajectory for arbitrarily similar inputs. Some further properties can be established, e.g. the directional monotonicity of the negotiation parameter magnitude is discussed for a special case.

The HEGO algorithm is generalized in Section 5.6. It distinguishes what is linked to the underlying nominal guidance method, from what is fundamental to the negotiation problems themselves. In details, it is shown how HEGO could be used with another choice of nominal guidance method (i.e. instead of $QP(\xi, p)$), among direct methods for OCPs.

Finally, numerical results are presented in Section 5.7. They highlight various aspects of the emergency guidance method, on both the 2D and the 3D rocket

models. Several sets of negotiable parameters are employed, to illustrate the various emergency policies achievable by the proposed HEGO methodology.

The first sections of this chapter are a detailed version of [66]. However, the last sections, including the theoretical proofs and the numerical simulations, are new (unpublished) elements.

5.1 Negotiable parameter choices

For a given input ξ , when landing has been declared infeasible by QP (ξ, p) , it is necessary to loosen some of the constraints. First, the parameters that can be negotiated are listed and it is shown how they modify the constraints of QP (ξ, p) . Then, a model describing their relative importance is proposed.

5.1.1 Negotiated constraints

The parameters that can be negotiated are the ones describing the goals of the landing. The physics-based equations of motion are not negotiable. However, the location of the landing site is, at least partially, negotiable. Indeed, if the landing site is located in a wide and flat area, it is of interest to allow touchdowns in a neighborhood of the ideal landing site¹. Some of the other parameters defining the constraints can be partially loosened. For example, the incidence limit should be seen more as a safety constraint - and a way to limit long-term fatigue of the rocket - and could be slightly widened if necessary, whereas the engine flow limitations are non-negotiable mechanical constraints.

The *negotiable parameters* are already conveniently conveyed by p , the variable introduced in Chapter 2. The purpose of this chapter is to discuss extensively how these negotiable parameters are mathematically modeled, what are the possible choices for its components, and how its value is chosen.

Let us recall that p has a linear influence on the Right-Hand Side (RHS) of the constraints of QP (ξ, p) , s.t. the *nominal constraints*

$$\begin{cases} Gz \leq h_0 + H_\xi \xi \\ Az = b_0 + B_\xi \xi \end{cases}$$

are transformed into the *negotiated constraints* by the action of p

$$\begin{cases} Gz \leq h_0 + H_\xi \xi + H_p p \\ Az = b_0 + B_\xi \xi + B_p p \end{cases} \quad (5.1)$$

For any such p , the matrices H_p and B_p are basically filled with zeros and a few ones, which makes them sparse.

These negotiated constraints come with an extra condition: it is assumed that all negotiable parameters are negotiable within prescribed limits, i.e. that p is bounded

¹Note that this would not apply to landings on offshore platforms, for obvious reasons.

Parameter name	Variables	Negotiable?	Comment
Final horizontal positions	$\Delta z^f, \Delta y^f$	✓	If landing area is solid and flat
Final horizontal speeds	$\Delta v_z^f, \Delta v_y^f$	✗	Otherwise the rocket would tilt at landing.
Final altitude	Δh^f	✗	See Example 2 for a discussion.
Final vertical speed	Δv_h^f	≈	Tiny margin, imposed by landing gear design
Incidence bound (2D) Projected incidence bound (3D)	$\Delta \alpha_{\max}$	✓	Safety and flight quality bound.
Engine flow bounds	$\Delta q_{\min}, \Delta q_{\max}$	✗	Physical constraint.
Normal acceleration	$\Delta a_{\text{nor}}^{\max}$	✓	Safety and flight quality bound.

Table 5.1: List of negotiable parameters in PDG (ξ, p) .

under the form

$$p_{\text{low}} \leq p \leq p_{\text{up}}. \quad (5.2)$$

If necessary, this cube-like constraint could be modeled as a bounded polytope, described by an inequality $Dp \leq d$ for some pair (D, d) , without changing any of the reasoning below.

For example, as it will be discussed in more details in Example 1 at the end of this chapter, a possible choice for the variable p is the pair $(\Delta \alpha_{\max}, \Delta z^f)^\top$ for the 2D rocket model. From a more exhaustive point of view, the list of all parameters that can or cannot be loosened in practice is presented in Table 5.1.

5.1.2 On the relative importance of the parameters

Adapting Orwell's Animal Farm quote, *all parameters are important but some are more important than others* (in (5.1)). It is necessary to enforce a hierarchy of importance between the negotiation parameters. Thus, let us separate the negotiation parameters in R different sub-parameters $p^{(j)}$ of ranked (increasing) importance, as

$$p = \left((p^{(1)})^\top \quad \dots \quad (p^{(R)})^\top \right)^\top \in \mathbb{R}^{n_{\text{neg}}}.$$

The higher the index j , the more critical $p^{(j)}$ is. The dimension of $p^{(j)}$ is noted n_j .

Mathematically speaking, the negotiable parameters are compared using the 1-norm of their sub-parameters, by comparing their most critical sub-parameters first. As first introduced in (1.1) in Chapter 1, a vector p_a is said to be more negotiated

negotiable parameters that recovers feasibility. There are two salient expectations regarding the method that computes these negotiable parameters and the associated trajectory that will be called *emergency trajectory*.

On the one hand, this method must pick the smallest negotiable parameter as possible, in the sense of the emergency order. This aims at using as little negotiable parameters as possible while enforcing the relative importance of the parameters.

On the other hand, the map $\xi \mapsto z^*$ giving the optimal emergency trajectory should be as smooth as possible (continuity is a minimum), in order to prevent *jumps* in the trajectory between arbitrarily close values of ξ . Such jumps could be very detrimental and cause serious issues to the control algorithms (out-of-the-scope of the thesis).

With these objectives in mind, we introduce the method HEGO (i.e. Algorithm 2 below), composed of a finite sequence of negotiation problems labeled LP_j and a refinement problem labeled Refine, which aims at fulfilling the two latter goals. HEGO is then tested on a low-dimensional example, helping to understand some of the methodology design choices. Finally, several aspects of this algorithm are put into perspective. The theoretical proofs showing that this algorithm behaves as expected will be discussed in Section 5.3.

5.2.1 Algorithmic principle of HEGO

From a high-level perspective, finding the smallest negotiation parameters that satisfy the constraints could take the form of a single optimization problem s.t.

$$\begin{aligned} z^*, p^* &\longleftarrow \min_{z,p} \text{Penalty}(p) \\ \text{s.t.} & \text{Negotiated constraints for } (z, p), \text{ from Eq. (5.1).} \end{aligned}$$

However, there are several limitations to this strategy. First, nothing guarantees that neither z^* nor p^* are unique when coming out of such a procedure. Moreover, since the cost of the latter problem differs from the cost of the nominal guidance problem $QP(\xi)$, it is highly likely that the map $\xi \mapsto z^*$ encounters a discontinuity when the emergency is *triggered*, i.e. when going from the most inner set to the intermediate set of Figure 5.1. Finally, and most importantly, the hierarchy is ignored with this kind of problem description. At best, the parameters can be non-homogeneously weighted to reflect their importance, but not their strict ranking.

Instead, we propose to successively minimize the magnitude of the negotiable parameters while enforcing the existence of at least one feasible trajectory at each step. This is a penalty-free approach. As required by the emergency order, this minimization procedure will focus sequentially on the sub-parameters, starting by the last one (i.e. $p^{(R)}$), down to the first one (i.e. $p^{(1)}$). This means that the most critical negotiable parameters are minimized first. At each step, only the proper sub-parameter $p^{(j)}$ is minimized, s.t. the result be $p^{(j)} = 0$ if it is *not necessary* to use this parameter to recover feasibility. Also, to enforce the desired hierarchy, a kind of memory effect is needed so that each step takes into account the results of the previous steps, by preserving the negotiation levels. This will be the role of condition (5.4e) below. As will appear, Linear Programming (LP) will play a key role to implement these negotiation steps.

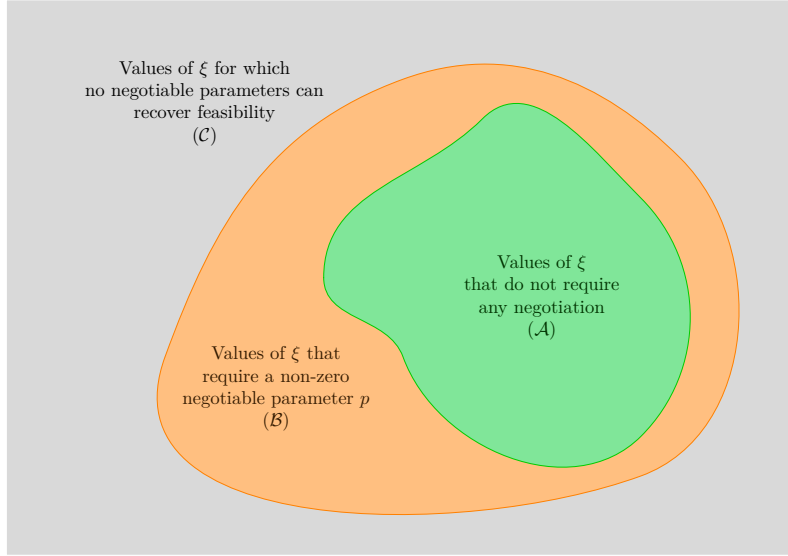


Figure 5.1: Pictorial representation of the possible values for the input ξ .

Quantitatively, let us first introduce the *negotiation problems* LP_j as

$$\text{LP}_j := \min_{z,p} \|p^{(j)}\|_1 \quad (5.4a)$$

$$\text{s.t. } Gz \leq h_0 + H_\xi \xi + H_p p \quad (5.4b)$$

$$Az = b_0 + B_\xi \xi + B_p p \quad (5.4c)$$

$$p_{\text{low}} \leq p \leq p_{\text{up}} \quad (5.4d)$$

$$\|p^{(i)}\|_1 = \mathcal{P}_i^*, \quad i = j+1, \dots, R \quad (5.4e)$$

where \mathcal{P}_i^* denotes the optimal value of LP_i . To make this problem definition well-posed, note that the constraint (5.4e) does not exist when $j = R$. Moreover, note that the inputs of each LP_j are ξ and \mathcal{P}_i^* for $i = j+1, \dots, R$. To alleviate the writing, these inputs are omitted wherever the context is clear enough. Finally, it is important to highlight the fact that z and p are both optimization variables in LP_j , even though only a few coefficients of p are involved in the cost (5.4a).

The role of LP_j is to minimize a cost on the j -th negotiable sub-parameters, while making sure that there are still feasible trajectories z , and that the already determined levels of relaxation of the previous negotiation problems are unchanged in the process. The reason why constraint (5.4e) must be satisfied instead of a constraint of the type “ $p^{(j)} = p^{(j)*}$ ” is that LP_j does not necessarily have a unique solution. Imposing $\|p^{(j)}\|_1 = \mathcal{P}_j^*$ encompasses all the solutions of LP_j and makes sure that the level of negotiation reached at step i remains satisfied in the follow-up negotiations. Descending from $j = R$ to $j = 1$ and imposing this latter constraint guarantees that the emergency order is enforced.

Thus, solving successively LP_j for $j = R, \dots, 1$ (decreasing indices) provides a way to recover feasibility, while hierarchically minimizing what needs to be sacrificed

from the original guidance problem, by building the sequence $\mathcal{P}_1^*, \dots, \mathcal{P}_R^*$ starting from the end. Among others, a noteworthy property of this process is that if landing is feasible without any negotiation, then solving LP_j will return $\|p^{(j)}\|_1 = 0$ at each step, implying that the overall vector p equals zero. Once this negotiation sequence has been computed, there may be many possible values for p , and for z as well. It is thus necessary to pick the best trajectory among these ones, by solving

$$\text{Refine} \quad := \quad \min_{z,p} \quad \frac{1}{2} z^\top P z + \xi^\top Q z \quad (5.5a)$$

$$\text{s.t.} \quad G z \leq h_0 + H_\xi \xi + H_p p \quad (5.5b)$$

$$A z = b_0 + B_\xi \xi + B_p p \quad (5.5c)$$

$$p_{\text{low}} \leq p \leq p_{\text{up}} \quad (5.5d)$$

$$\|p^{(i)}\|_1 = \mathcal{P}_i^*, \quad i = 1, \dots, R \quad (5.5e)$$

Like LP_j , the problem *Refine* takes as inputs ξ and \mathcal{P}_i^* for $i = 1, \dots, R$. These two optimization problems are the basic bricks of the central algorithm of this thesis, defined below.

Algorithm 2 Hierarchical Emergency Guidance Optimization (HEGO)

Require: Difference w.r.t. reference values: $\xi = (\Delta x^0, \Delta \eta, \Delta u_{\text{init}})$.

for $j = R, \dots, 1$ (decreasing indices) **do**

$$\mathcal{P}_j^* \leftarrow \min \text{LP}_j(\xi, \mathcal{P}_{j+1}^*, \dots, \mathcal{P}_R^*) \quad // \text{ From definition (5.4)}$$

end for

$$z^* \leftarrow \text{argmin Refine}(\xi, \mathcal{P}_1^*, \dots, \mathcal{P}_R^*) \quad // \text{ From definition (5.5)}$$

return z^*

For the *argmin* operation, the value of p is voluntarily ignored, since it is not needed nor unique.

It is important to observe that this guidance procedure provides, depending on the situation, either a nominal guidance or an emergency guidance, as stated in the following proposition.

Proposition 11. *If ξ is s.t. the constraints of the nominal guidance problem $\text{QP}(\xi, 0)$ are feasible, then HEGO and $\text{QP}(\xi, 0)$ return the same value z^* .*

Proof. If the constraints of $\text{QP}(\xi, 0)$ are feasible, then there exists at least one pair (z, p) with $p = 0$ that satisfies the negotiated constraints (5.1). Thus, all the problems LP_j will necessarily return $\mathcal{P}_j^* = 0$, and constraint (5.5e) will subsequently impose $p = 0$, making *Refine* and $\text{QP}(\xi, 0)$ coincide. Uniqueness of the solutions yield the conclusion. \square

Qualitatively, the nominal guidance method based solely on $\text{QP}(\xi, 0)$ can provide landing trajectories on the set \mathcal{A} from Figure 5.1 whereas HEGO is able to do it on the sets \mathcal{A} and \mathcal{B} . HEGO therefore provides landing guidance for a wider range of input values ξ . It also meets the requirements regarding the negotiable parameter minimization and the hierarchy enforcement. These features make it an “universal” guidance algorithm.

The analysis of the map $\xi \mapsto z^*$ defined by this algorithm is presented later in Section 5.3. Among others, Proposition 12 will show the well posedness of the

algorithm, and Theorem 7 proves the Lipschitz-continuity of $\xi \mapsto z^*(\xi)$ for HEGO. Before getting into these technical details, let us discuss the following example.

5.2.2 An illustrative toy example

Consider a low-dimensional example that resembles the landing problem, illustrating how HEGO works and why the negotiation problem hierarchy matters. Although this example could seem greatly over-simplified at first glance, its similarities with the actual landing problem are noteworthy, especially when comparing the curves from Figure 5.3 (left) and Figure 5.5 (d) reporting the negotiation of parameters.

The problem takes as input a scalar $\xi \geq 0$, and aims at minimizing the norm of $z = (z_1, z_2)^\top \in \mathbb{R}^2$, under some constraints

$$\begin{aligned} \min_{z, p} \quad & z_1^2 + z_2^2 \\ \text{s.t.} \quad & \text{(Ineq}_1) \quad z_1 \geq 0 \\ & \text{(Ineq}_2) \quad z_2 \geq 0 \\ & \text{(Eq)} \quad z_2 = 1 - \xi - z_1 \end{aligned}$$

This problem is represented in Figure 5.2 (0). By analogy, let us imagine that z_1 conveys the incidence, z_2 the engine flow and ξ the initial horizontal position error. Therefore,

- (Ineq₁) conveys the incidence bound, that may be negotiated by a variable p_1 , s.t. $z_1 \geq -p_1$.
- (Ineq₂) conveys the mechanical limits of the engine flow, which are non-negotiable.
- (Eq) represents the terminal condition on the horizontal position and is directly influenced by ξ . It may be negotiated by a parameter p_2 if necessary, s.t. $z_2 = 1 - \xi - z_1 + 2p_2$.

Moreover, note that the negotiable parameters $p = (p_1, p_2)^\top \in \mathbb{R}^2$ can be negotiated up to some extent, so we decide to bound them between 0 and 1. More precisely, imposing $0 \leq p_2 \leq 1$ means that we allow the rocket to eventually land farther of the landing site, but on one side only and up to a certain limit.

Finally, as in the actual landing problem, we consider that negotiating p_2 is more critical than p_1 (i.e. $R = 2$). Let us now review the possible scenarios depending on the input value.

5.2.2.1 Nominal scenario

When ξ remains low, i.e. $0 \leq \xi \leq 1$, then there is no need for parameter negotiation, because the problem is feasible when $p_1 = p_2 = 0$. Running Algorithm 2 will give null values for the negotiation penalties. In this case, the optimal solution is $z_1^*(\xi) = z_2^*(\xi) = (1 - \xi)/2$. This scenario is represented in Figure 5.2 (A).

5.2.2.2 First negotiation scenario

When $1 < \xi \leq 2$, the initial constraints are not compatible anymore and must be negotiated, whence the need for Algorithm 2. The first step of the latter is

$$\begin{aligned} \min_{z,p} \quad & |p_2| \\ \text{s.t.} \quad & z_1 \geq -p_1 \\ & z_2 \geq 0 \\ & z_2 = 1 - \xi - z_1 + 2p_2 \\ & 0 \leq p_1 \leq 1 \\ & 0 \leq p_2 \leq 1 \end{aligned}$$

and will result in a null optimal negotiation, i.e. $\mathcal{P}_2^*(\xi) = 0$, since it is possible to recover feasibility without using p_2 . The second step will be the problem

$$\begin{aligned} \min_{z,p} \quad & |p_1| \\ \text{s.t.} \quad & z_1 \geq -p_1 \\ & z_2 \geq 0 \\ & z_2 = 1 - \xi - z_1 + 2p_2 \\ & 0 \leq p_1 \leq 1 \\ & 0 \leq p_2 \leq 1 \\ & 0 = |p_2| \end{aligned}$$

where “ $0 = |p_2|$ ” is here to enforce the result from the former negotiation problem. The latter will return the optimal negotiation $\mathcal{P}_1^*(\xi) = \xi - 1$. Thus, the optimization variable z can be re-optimized over the newly negotiated set - which is actually a singleton. In the end, it yields

$$z_1^*(\xi) = 1 - \xi \quad \text{and} \quad z_2^*(\xi) = 0.$$

This scenario is represented in Figure 5.2 (B).

5.2.2.3 Mild negotiation scenario

Then, let us consider a scenario that requires even more negotiations, i.e. when $2 < \xi \leq 4$. In this case, the first step of Algorithm 2 gives a non-zero optimal negotiation s.t. $\mathcal{P}_2^*(\xi) = \frac{\xi-2}{2}$, and the second step gives the result $\mathcal{P}_1^*(\xi) = 1$.

Qualitatively, this must be interpreted as: “the parameter p_2 must be modified just enough so that there is a value of p_1 that provides a non-empty feasible set for z ”. In this scenario, $z_1^*(\xi) = -1$ and $z_2^*(\xi) = 0$, which are plotted in Figure 5.2 (C).

5.2.2.4 Infeasible scenario

Finally, for $\xi > 4$, there are no possible solutions for the first negotiation problem. Therefore, Algorithm 2 does not return anything, because there are no solutions to this over-constrained problem. Note that the limit $p_2 \leq 1$ is the ultimate constraint that makes the negotiation problem infeasible.

5.2.2.5 The importance of the parameters hierarchy

The values of the variables z and p are represented in Figure 5.3 (left), w.r.t. the input ξ .

The chosen parameters hierarchy is crucial. Indeed, if instead of considering that “ p_2 is more critical than p_1 ” it was considered that the whole vector (p_1, p_2) could be negotiated at once, the results would have been completely different, as shown in Figure 5.3 (right). In the latter case, there would be only one negotiation problem

$$\begin{aligned} \min_{z,p} \quad & |p_1| + |p_2| \\ \text{s.t.} \quad & z_1 \geq -p_1, \\ & z_2 \geq 0, \\ & z_2 = 1 - \xi - z_1 + 2p_2, \\ & 0 \leq p_1 \leq 1, \\ & 0 \leq p_2 \leq 1. \end{aligned}$$

Doing this would imply that the variable p_2 would be used to recover feasibility before p_1 .

5.2.3 Noteworthy remarks

Remark 23 (Linear Programs). *Using standard material from the literature, such as [14, Example 1.13], decomposing $p = \rho_+ - \rho_-$ with $\rho_+, \rho_- \geq 0$ makes it possible to solve LP_j using Linear Programming, whence the name LP_j . Indeed, this decomposition yields the convenient re-writing $\|p\|_1 = \mathbb{1}^\top(\rho_+ + \rho_-)$. See also the re-writing method in the Lipschitz-continuity proofs below.*

Remark 24 (In the literature). *From a very general mathematical programming point of view, recovering feasibility in Linear Programming has been discussed extensively by Chinneck [28] for instance. Problem (5.4) builds upon right-hand side constraint “alteration” methods, by exploiting the available levers conveyed through the parameter p , the matrices H_p and B_p , and the need to enforce the parameter hierarchy.*

Remark 25 (Hierarchy does not impact feasibility). *Though the hierarchy notion is important, it does not change the set of inputs s.t. there exists at least one value of the negotiable parameters that make the constraints (5.1) feasible. Mathematically, it means that the set of $\xi \in \mathbb{R}^{N_\xi}$ s.t.*

$$\exists(z, p) \in \mathbb{R}^{N_z} \times \mathbb{R}^{n_{\text{neg}}} : \begin{cases} Gz \leq h_0 + H_\xi \xi + H_p p \\ Az = b_0 + B_\xi \xi + B_p p \\ p_{\text{low}} \leq p \leq p_{\text{up}} \end{cases}$$

does not depend on \succeq_e by construction. The latter order will only determine which values of p will be used for a given ξ . This fact can be observed in Figure 5.3, by remarking that the regions (D) and (C') correspond to the same sets for ξ .

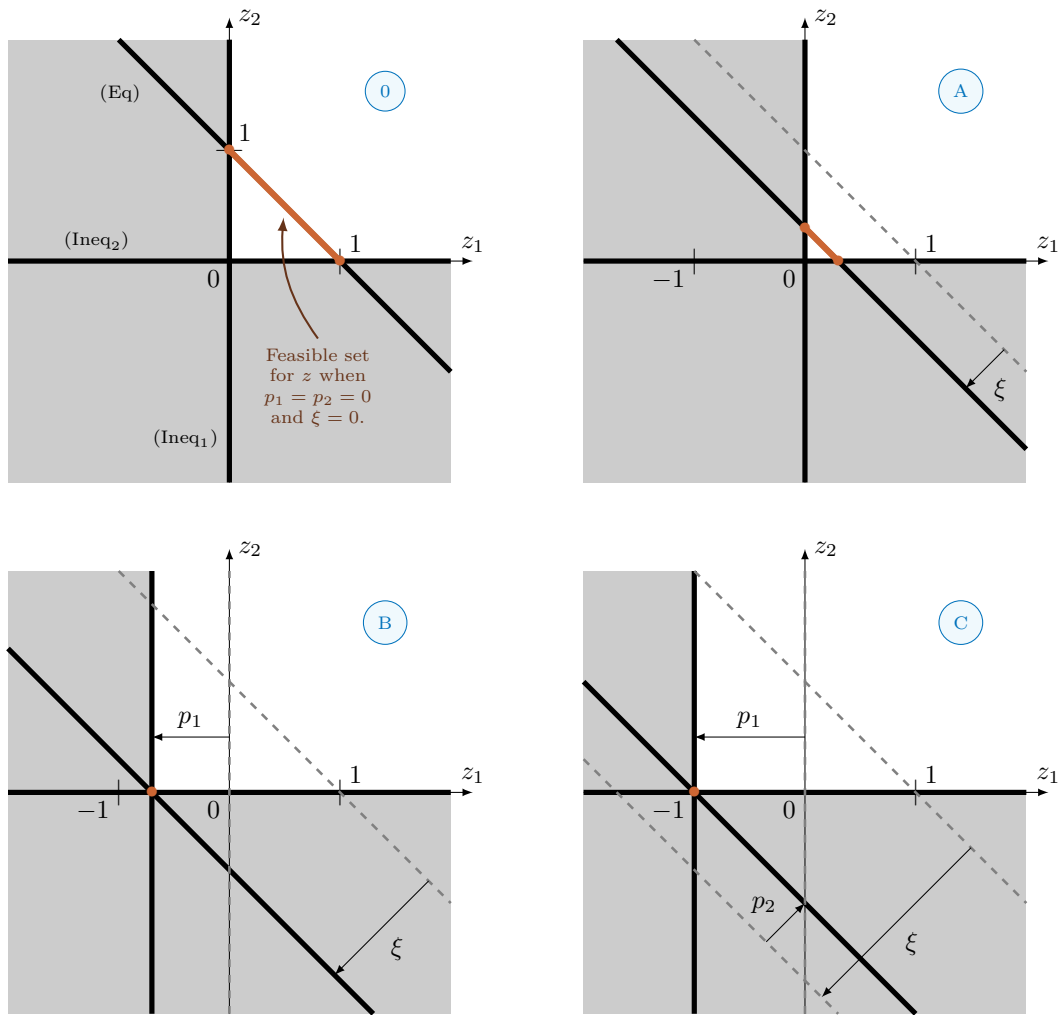


Figure 5.2: Illustration of the constraint for the problem of Section 5.2.2 w.r.t. the input values, using HEGO.

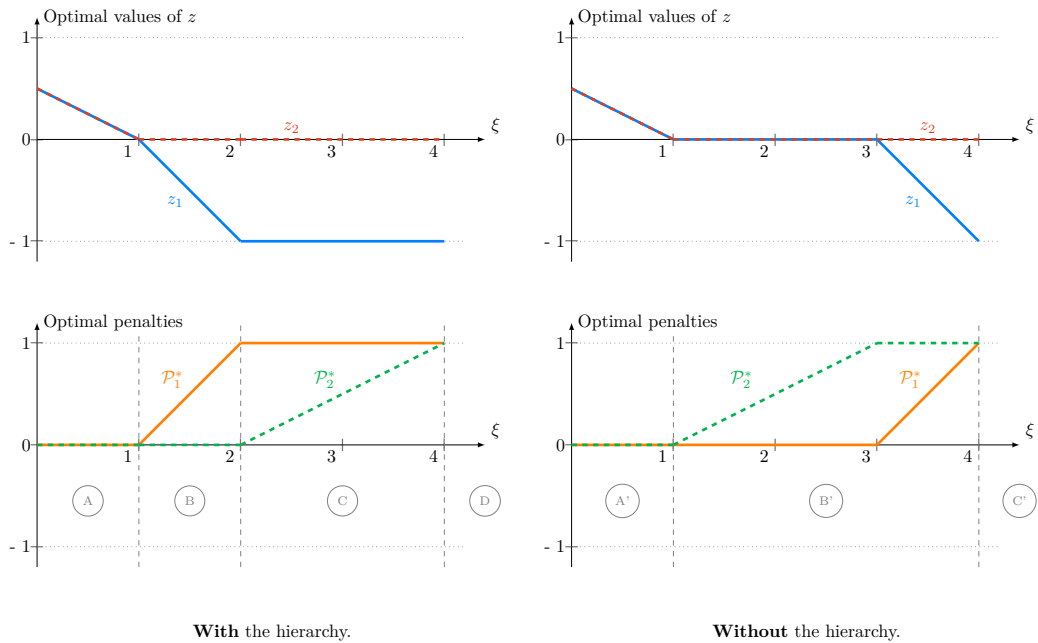


Figure 5.3: *Curves associated to the example of Section 5.2.2.* (Left) The first remark is that all the quantities displayed are indeed continuous w.r.t. the input ξ . On the *bottom* chart, there are four distinguishable areas. (A) corresponds to the nominal scenario, when no negotiation is needed. (B) and (C) corresponds to scenarios that require respectively 1 and 2 non-zero negotiation parameter values to recover feasibility. Finally, scenario (D) is when there are no options left., and no allowed values of p_1 or p_2 can help recover feasibility for z . (Right) Without enforcing any hierarchy, the previous zones (B), (C) merge into a single zone (B'), where both variables p_1 and p_2 are negotiated at the same time. The penalties differ from the previous case, and so do the optimal values z_1 and z_2 .

Remark 26 (Limits of the hierarchy). *HEGO guarantees that the most critical parameters are minimized first. However, since the negotiable parameters can have a very different influence on the constraints, it is still possible to have scenarios where a critical parameter is non-zero, but a less critical parameter is zero.*

For instance, let us consider the following trivial example where the negotiated constraints take the form

$$\begin{aligned} z_1 &= \xi_1 + p_1, \\ z_2 &= \xi_2 + p_2, \\ -1 &\leq z_1 \leq 1, \\ -1 &\leq z_2 \leq 1. \end{aligned}$$

Since there are no links between the variables indexed by 1 and the ones indexed by 2, even imposing that p_2 be more critical than p_1 does not guarantee that p_1 will ever be used when the magnitude of ξ_2 increases.

To summarize, the least critical negotiable parameters will be used in lieu of the most critical ones only if the fundamental nature of the problem makes it possible.

Remark 27. *(Post-defense note) As noted by one of the jury members ³ after the thesis defense, trying to minimize the value of a parameter with respect to a lexicographic or co-lexicographic order is similar to some methods used in multi-objective optimization. For instance, see [30] for more details on the topic.*

5.3 Smoothness of the HEGO algorithm

For the reasons mentioned previously, the fact that the outputs of Algorithm 2 - i.e. HEGO- do not change *too fast* when its inputs vary is of high interest.

The goal of this section is to prove that the map $\xi \mapsto z^*$ defined by Algorithm 2 is globally Lipschitz on its definition domain. Proving this property relates to QP sensitivity analysis w.r.t. the constraints RHS and the linear part of the cost. When the cost is defined with a positive definite matrix, this property holds and is a well-known result [59]. However, this has to be adapted to our framework, where only a part of the optimization variable is unique (i.e. z is unique, but p is not necessarily unique).

To alleviate the writing, and *without loss of generality*, we make the assumption that the cost of Refine only has a quadratic term in z , i.e. that $Q = 0$ in Equation (5.5a). Indeed, extending the constraints RHS sensitivity results to perturbations in the linear part of the cost is obtained using well-known dualization methods of QPs [21, 41, 59].

First, a re-writing of the problems LP_j and Refine is introduced, to show that the result z^* from Algorithm 2 is well-defined and unique. Then, we show that the optimal negotiation maps giving \mathcal{P}_i^* and the optimal solution maps giving z^* are Lipschitz continuous functions of the RHS of their constraints. This is proved using a series of well-known results and by adapting a theorem by Mangasarian & Shiau [59] to our framework. Finally, we conclude on the properties of Algorithm 2.

³I would like to thank Laurent Pfeiffer for its attention to detail and its rich feedback.

5.3.1 Problem re-writing

The constraints of LP_j and Refine can be rewritten into a unified, standard and linear framework. Denote by \mathcal{C}_j these constraints

$$\mathcal{C}_j := \begin{cases} Gz \leq h_0 + H_\xi \xi + H_p p, \\ Az = b_0 + B_\xi \xi + B_p p, \\ p_{\text{low}} \leq p \leq p_{\text{up}}, \\ \|p^{(i)}\|_1 = \mathcal{P}_i^*, \quad i = j+1, \dots, R. \end{cases}$$

For $j = 1, \dots, R$, the constraints \mathcal{C}_j convey those of LP_j , and \mathcal{C}_0 convey those of Refine. Slack variables, denoted $s_G, s_{\text{up}}, s_{\text{low}}$, can be used to transform the inequalities of \mathcal{C}_j into the constraints

$$\begin{aligned} Gz + s_G &= h_0 + H_\xi \xi + H_p p, \\ p + s_{\text{up}} &= p_{\text{up}}, \\ p - s_{\text{low}} &= p_{\text{low}}, \\ s_G, s_{\text{up}}, s_{\text{low}} &\geq 0 \end{aligned}$$

Let us define the variable $x = (z_+, z_-, \rho_+, \rho_-, s_G, s_{\text{up}}, s_{\text{low}})$ where

$$z = z_+ - z_-, \quad (5.6a)$$

$$p = \rho_+ - \rho_-, \quad (5.6b)$$

$$x \geq 0. \quad (5.6c)$$

and introduce the matrices $\bar{\mathcal{A}}^j$ and \bar{r}^j as

$$\bar{\mathcal{A}}^j := \begin{pmatrix} G & -G & -H_p & H_p & \mathbb{1}_{\dim s_G} & 0 & 0 \\ A & -A & -B_p & B_p & 0 & 0 & 0 \\ 0 & 0 & \mathbb{1}_{n_{\text{neg}}} & -\mathbb{1}_{n_{\text{neg}}} & 0 & \mathbb{1}_{n_{\text{neg}}} & 0 \\ 0 & 0 & \mathbb{1}_{n_{\text{neg}}} & -\mathbb{1}_{n_{\text{neg}}} & 0 & 0 & -\mathbb{1}_{n_{\text{neg}}} \\ 0 & 0 & \mathcal{I}_{j+1} & \mathcal{I}_{j+1} & 0 & 0 & 0 \end{pmatrix}, \quad \bar{r}^j := \begin{pmatrix} h_0 + H_\xi \xi \\ b_0 + B_\xi \xi \\ p_{\text{up}} \\ p_{\text{low}} \\ \mathcal{P}_{j+1}^R \end{pmatrix}$$

where \mathcal{I}_{j+1} and \mathcal{P}_{j+1}^R denote $\mathcal{P}_{j+1}^R := \left(\mathcal{P}_R^* \dots \mathcal{P}_{j+1}^* \right)^\top \in \mathbb{R}^{R-j}$ and

$$\mathcal{I}_{j+1} = \begin{pmatrix} 0 & \dots & 0 & \mathbb{1}_{n_R}^\top \\ & & \mathbb{1}_{n_{R-1}}^\top & 0 \\ \vdots & \ddots & & \vdots \\ 0 & \mathbb{1}_{n_{j+1}}^\top & \dots & 0 \end{pmatrix} \in \mathbb{R}^{(R-j) \times n_{\text{neg}}}.$$

Note that for $j = R$, the last line of $\bar{\mathcal{A}}^R$ and \bar{r}^R is absent. Also, for $j = 0$, the first column of zeros is absent for \mathcal{I}_1 . Recall for the rest of the proofs that \bar{r}^j is the vector

bearing all the dependency of the problems LP_j and Refine on ξ . Finally, note \mathcal{A}^j and r^j the following matrices

$$\mathcal{A}^j := \begin{pmatrix} \bar{\mathcal{A}}^j \\ -\bar{\mathcal{A}}^j \end{pmatrix} \quad \text{and} \quad r^j := \begin{pmatrix} \bar{r}^j \\ -\bar{r}^j \end{pmatrix}.$$

Using the new non-negative variable x , and the notations above, the constraints \mathcal{C}_j can be re-written under the two equivalent forms

$$\begin{cases} \bar{\mathcal{A}}^j x = \bar{r}^j \\ x \geq 0 \end{cases} \quad \text{or} \quad \begin{cases} \mathcal{A}^j x \geq r^j \\ x \geq 0 \end{cases} \quad (5.7)$$

Further, for $j = 1, \dots, R-1$, by construction of the constraints \mathcal{C}_j , the following property holds

$$\{x \mid \bar{\mathcal{A}}^j x = \bar{r}^j, x \geq 0\} = \{x \mid \bar{\mathcal{A}}^{j+1} x = \bar{r}^{j+1}, x \geq 0, \mathbf{c}_{j+1}^\top x = \mathcal{P}_{j+1}^*\} \quad (5.8)$$

and will be used later to prove Proposition 12. Regarding the cost, define the vector \mathbf{c}_j of the same dimension as x s.t.

$$\mathbf{c}_j := \begin{cases} (\dots, 0, \mathbb{1}_{n_j}^\top, \mathbb{0}_{1 \times (n_{\text{neg}} - n_j)}, \mathbb{1}_{n_j}^\top, 0, \dots)^\top & \text{if } j = 1, \dots, R \\ 0 & \text{if } j = 0 \end{cases}$$

where the terms $\mathbb{1}_{n_j}^\top$ correspond to the position of the j^{th} negotiable variable $p^{(j)}$, here conveyed by the corresponding parts of ρ_+ and ρ_- from Equation (5.6). Moreover, let us define

$$\mathcal{D} := \left(\begin{array}{cc|c} P & -P & \mathbb{0} \\ -P & P & \mathbb{0} \\ \hline & \mathbb{0} & \mathbb{0} \end{array} \right) \in \mathbb{R}^{\dim x \times \dim x}.$$

It is straightforward to verify that, for all P positive definite, the matrix \mathcal{D} is positive semidefinite.

Thus, the optimization problems LP_j and Refine can be re-written in a more standard form than their original definition, respectively (5.4) and (5.5). For any $j = 1, \dots, R$, LP_j can be described as a Linear Program of the form

$$(\text{Primal LP})_j \quad \begin{cases} \min_x & \mathbf{c}_j^\top x \\ \text{s.t.} & \bar{\mathcal{A}}^j x = \bar{r}^j \\ & x \geq 0 \end{cases} \quad (5.9)$$

and Refine as a Quadratic Program which is

$$(\text{Primal QP}) \quad \begin{cases} \min_x & \frac{1}{2} x^\top \mathcal{D} x + \mathbf{c}_0^\top x \\ \text{s.t.} & \bar{\mathcal{A}}^0 x = \bar{r}^0, \\ & x \geq 0. \end{cases} \quad (5.10)$$

It is also possible to re-write Problem (5.10) into the following canonical QP

$$\text{(Canonical QP)} \quad \begin{cases} \min_x & \frac{1}{2}x^\top \mathcal{D}x + \mathbf{c}_0^\top x \\ \text{s.t.} & \mathcal{A}^0 x \geq r^0, \\ & x \geq 0. \end{cases} \quad (5.11)$$

Certainly, in view of numerical implementation, there are more memory-efficient ways to translate Refine into these formats, but the formulations above are handy in the theoretical proof below. Also, note that we keep the same definition for \mathcal{P}_j^* as in Problem (5.4), i.e. \mathcal{P}_j^* denotes the optimal value of the cost of Problem (5.9) (its well-posedness will be established in Proposition 12).

5.3.2 Uniqueness of the optimal trajectory

Recalling that $\mathbf{c}_0 = 0$, let us introduce the dual⁴ of Problem (5.10) s.t.

$$\text{(Dual QP)} \quad \begin{cases} \max_{x, \mu, \lambda} & -\frac{1}{2}x^\top \mathcal{D}x - \lambda^\top \bar{r}^0 \\ \text{s.t.} & \mathcal{D}x - \mu + \bar{\mathcal{A}}^{0\top} \lambda = 0, \\ & \mu \geq 0. \end{cases} \quad (5.12)$$

Using the Strong Duality Theorem⁵ we get the following property.

Lemma 6. *If $\{x \mid \bar{\mathcal{A}}^0 x = \bar{r}^0, x \geq 0\}$ is not empty, then Problem (5.10) has no duality gap, i.e. Problem (5.10) and Problem (5.12) have optimal values and they are equal.*

Lemma 7. *Under the assumption of Lemma 6, let (x^*, μ^*, λ^*) and $(\tilde{x}, \tilde{\mu}, \tilde{\lambda})$ be optimal solutions of both Problems 5.10 and 5.12. Then, $\mathcal{D}x^* = \mathcal{D}\tilde{x}$.*

Proof, adapted from Lemma 2.1 in [12]. Since the tuples from the statement are optimal and since there is no duality gap, then the primal and dual cost are equal s.t.

$$\frac{1}{2}(x^*)^\top \mathcal{D}x^* = -\frac{1}{2}(\tilde{x})^\top \mathcal{D}\tilde{x} - \lambda^\top \bar{r}^0$$

Since the tuples are optimal solutions, *complementary slackness* holds, i.e. $\mu^\top \tilde{x} = 0$ and $\mu^\top x^* = 0$ (see e.g. Proposition 5.1.5 in [13]). Applying the equality constraints from (5.12) at \tilde{x} , and multiplying it by $(x^*)^\top$ to the left gives

$$(x^*)^\top \mathcal{D}\tilde{x} = (x^*)^\top \mu - (x^*)^\top \bar{\mathcal{A}}^{0\top} \lambda.$$

Thus

$$\frac{1}{2}(x^* - \tilde{x})^\top \mathcal{D}(x^* - \tilde{x}) = -\lambda^\top \bar{r}^0 + (x^*)^\top \bar{\mathcal{A}}^{0\top} \lambda = \lambda^\top (\bar{\mathcal{A}}^0 x^* - \bar{r}^0) = 0$$

which gives $\mathcal{D}(x^* - \tilde{x}) = 0$ since $\mathcal{D} \succeq 0$, hence the desired result. \square

⁴See e.g. [21, Section 5.2].

⁵Theorem 11, recalled in the Appendix.

Let us now introduce the set

$$\chi(\xi) := \left\{ x \mid \bar{\mathcal{A}}^R x = \bar{r}^R, x \geq 0 \right\}.$$

Proposition 12. *If $\chi(\xi)$ is non-empty, then the value of $z^* = z_+ - z_-$ computed by Algorithm 2, obtained by successively solving Problems 5.9 and 5.10, exists and is unique.*

Proof. Let us prove by induction for $j = R, \dots, 1$ (with decreasing indices), that “Problem (5.9) has a finite optimal value at index j , and constraints (5.7) are feasible at index $j - 1$ ”. Beforehand, note that the cost function of all Problems 5.9 is lower-bounded by 0.

For $j = R$, since $\chi(\xi)$ is assumed non-empty, and since the cost is lower-bounded by 0, then it has a finite optimal value, denoted \mathcal{P}_R^* . Moreover, this minimum is attained, as guaranteed by Lemma 12 in the Appendix, at a point denoted x^R . Thanks to the Equation (5.8), we get that x^R is a feasible point for the constraints (5.7) at index $R - 1$.

To prove the induction, let us assume that Problem (5.9) has a finite optimal value at index j , and constraints (5.7) are feasible at index $j - 1$, for some $j \geq 2$. Using the relation from Equation (5.8), and by induction, the set $\left\{ x \mid \bar{\mathcal{A}}^{j-1} x = \bar{r}^{j-1}, x \geq 0 \right\}$ is non-empty. Thus, since its cost is lower-bounded by 0, Problem (5.9) at index $j - 1$ has a finite optimal value \mathcal{P}_{j-1}^* , also attained at some point denoted x^{j-1} . The latter being a feasible point for constraints (5.7) at index $j - 2$, one concludes the induction proof.

This shows that the optimal penalties $\mathcal{P}_1^*, \dots, \mathcal{P}_R^*$ are well defined, and that the constraints (5.7) are feasible at index 0. Consequently, Problem (5.10) also has a minimum. By Lemma 7, and using the expression of \mathcal{D} , we get that any solution of Problem (5.10) has a unique value for z_+ and z_- , showing that $z^* = z_+ - z_-$ exists and is unique. This concludes the proof. \square

Among others, this last proposition points out that as long as the first-to-be-computed negotiation problem is feasible - i.e. LP_R - then Algorithm 2 will necessarily terminate. This is summarized as follows, using the set $\Lambda := \{\xi' \mid \chi(\xi') \neq \emptyset\}$.

Corollary 1. *Algorithm 2 returns a solution for the input ξ if and only if $\xi \in \Lambda$.*

Proof. If $\chi(\xi) \neq \emptyset$, Proposition 12 shows that Algorithm 2 has a solution. Otherwise, $\chi(\xi) = \emptyset$, implies that LP_R is infeasible, and then Algorithm 2 fails. \square

The set Λ is not empty, since 0 belongs to Λ by definition of QP (ξ, p) .

Proposition 13. *Λ is convex.*

Proof. Using the definition of \bar{r}^R , there is a constant vector \bar{b} and a constant matrix \bar{B} s.t. $\bar{r}^R = \bar{b} + \bar{B}\xi$, where \bar{b} and \bar{B} directly depend on h_0, H_ξ, b_0 and B_ξ appearing in the formulation of QP (ξ, p) defined in (4.23). Let us assume that ξ^1 and ξ^2 are s.t. $\chi(\xi^1) \neq \emptyset$ and $\chi(\xi^2) \neq \emptyset$. For $x^1 \in \chi(\xi^1)$ and $x^2 \in \chi(\xi^2)$, the condition

$$(1-t)x^1 + tx^2 \in \chi\left((1-t)\xi^1 + t\xi^2\right), \quad \forall t \in [0, 1],$$

holds due to the linearity of (5.7) w.r.t. x and ξ , which shows the desired result. \square

It is noteworthy that Λ is not necessarily bounded. Indeed, if the intersection of the kernels of matrices B_ξ and H_ξ (defined in Problem (4.23)) is wider than the singleton $\{0\}$, then the set Λ is even guaranteed to be unbounded.

5.3.3 Regularity w.r.t. the right-hand side of the constraints

The proof that z^* is a Lipschitz-continuous map of its inputs can be split in two steps. First, we need to show that the optimal penalties \mathcal{P}_j^* are Lipschitz-continuous maps of their inputs, and then that z^* is also a Lipschitz-continuous map of ξ and \mathcal{P}_i^* for $i = 1, \dots, R$. The former result is rather straightforward and will be dealt with in Lemma 8. However, the latter result requires a bit more attention, and will be detailed in Theorem 6.

Remark 28. *The results used below are expressed with both the Euclidean and the maximum norms, respecting their original formulation, whereas the main result is given in an homogeneous form - i.e. using only the Euclidean norm - in Theorem 7.*

5.3.3.1 RHS regularity of the negotiation maps

The following theorem is adapted from [59], and applies to the standard LP

$$\min_x p^\top x \quad (5.13a)$$

$$\text{s.t. } Ax \leq b, \quad (5.13b)$$

$$Cx = d. \quad (5.13c)$$

A *feasible* point for Problem (5.13) is a point x that satisfies (5.13b) and (5.13c). A *solution* point for Problem (5.13) is a feasible point that is minimal for (5.13a).

Theorem 4 (Adapted from [59, Thm. 2.4]). *Let the Linear Program (5.13) have non-empty solution sets S^1 and S^2 for right-hand sides (b^1, d^1) and (b^2, d^2) , respectively. There exists a constant $L > 0$ s.t. for each $x_*^1 \in S^1$, there exists an $x_*^2 \in S^2$ s.t.*

$$\|x_*^1 - x_*^2\|_\infty \leq L \left\| \begin{pmatrix} b^1 - b^2 \\ d^1 - d^2 \end{pmatrix} \right\|_2 \quad (5.14)$$

A direct corollary is the following.

Corollary 2 (Lipschitz continuity of the optimal value function of LPs w.r.t. RHS perturbations). *Let the Linear Program (5.13) have non-empty solution sets S^1 and S^2 for right-hand sides (b^1, d^1) and (b^2, d^2) , respectively, with associated optimal values p_1^* and p_2^* . Then, there exists a constant $K > 0$ s.t.*

$$|p_1^* - p_2^*| \leq K \left\| \begin{pmatrix} b^1 - b^2 \\ d^1 - d^2 \end{pmatrix} \right\|_2$$

Lemma 8. *For any $j = 1, \dots, R$, the maps $\xi \in \Lambda \mapsto \mathcal{P}_j^*$ are Lipschitz continuous.*

Proof. In order for these maps to be well-defined, recall that each value \mathcal{P}_j^* depends on ξ and the preceding values $\mathcal{P}_{j+1}^*, \dots, \mathcal{P}_R^*$, (except for \mathcal{P}_R^* that only depends on ξ). Thus, for all $j = 1, \dots, R$, we are precisely interested in the maps Γ_j :

$$\Gamma_j : \begin{array}{l} \Lambda \rightarrow \mathbb{R}_+ \\ \xi \mapsto \mathcal{P}_j^*(\xi, \Gamma_{j+1}(\xi, \dots), \dots, \Gamma_R(\xi)). \end{array}$$

where $\Gamma_R(\xi) = \mathcal{P}_R^*(\xi)$. Consider $j = R$. By composition, since the RHS of Problem (5.9) is affine dependent on ξ , and since the optimal value of Problem (5.9) is Lipschitz-continuous w.r.t. its RHS (due to Corollary 2), then Γ_R is Lipschitz continuous. Then, by induction, let us assume that at each step $j = 1, \dots, R$, the previous functions $\Gamma_{j+1}, \dots, \Gamma_R$ are Lipschitz continuous. Using the same composition argument, we obtain the desired result. \square

5.3.3.2 RHS regularity of the Linear Complementary Problem

To show that the map $(\xi, \mathcal{P}_1^*, \dots, \mathcal{P}_R^*) \mapsto z^*$ is Lipschitz continuous, we proceed in three steps. First, we recall that polytopes satisfy a Lipschitz continuity-like property (Theorem 5) w.r.t. their RHS. Then, we show how the former map is related to a certain Linear Complementary Problem (LCP, in Lemma 9). Finally, the Lipschitz-continuity of the uniquely defined components of the LCP is established (Theorem 6).

Theorem 5 (Adapted from [59, Thm. 2.2], Lipschitz-continuity of feasible points of linear inequalities and equalities). *Let the conditions $Ax = b$ and $Cx \leq d$ have non-empty feasible sets F^1 and F^2 for the right-hand sides (b^1, d^1) and (b^2, d^2) , respectively. There exists a constant μ , that depends only on A and C , s.t. for each $x^1 \in F^1$, there exists an $x^2 \in F^2$ closest to x^1 in the ∞ -norm s.t.*

$$\|x^1 - x^2\|_\infty \leq \mu \left\| \begin{pmatrix} b^1 - b^2 \\ d^1 - d^2 \end{pmatrix} \right\|_2$$

Proof of the following lemma can be found in [69, Sec. 16.4.4].

Lemma 9 (Linear Complementary Problem, [59, 69]). *Assume that \mathcal{D} is positive semidefinite. Then, x is a solution of Problem (5.11) if and only if there exists a vector η s.t.*

$$\hat{x} := \begin{pmatrix} x \\ \eta \end{pmatrix} \tag{5.15}$$

is a solution to the following LCP

$$M\hat{x} + q \geq 0, \quad \hat{x} \geq 0, \quad (M\hat{x} + q)^\top \hat{x} = 0 \tag{5.16}$$

where $M := \begin{pmatrix} \mathcal{D} & -\mathcal{A}^{0\top} \\ \mathcal{A}^0 & 0 \end{pmatrix}$ and $q := \begin{pmatrix} c_0 \\ -r^0 \end{pmatrix}$.

Given a subset $J \subset \{1, \dots, \dim \hat{x}\}$, any solution of the following linear system⁶

$$M_j \hat{x} + q_j \geq 0, \quad \hat{x}_j = 0, \quad j \in J, \quad (5.17a)$$

$$M_j \hat{x} + q_j = 0, \quad \hat{x}_j \geq 0, \quad j \notin J, \quad (5.17b)$$

is a solution to the LCP (5.16) for (M, q) . For such sets J , denote $Q(J)$ the set of all q vectors for which (5.17) has a solution⁷.

Lemma 10 (Active set partitions [59, Lemma 3.1]). *Let q^1 and q^2 be two distinct vectors and let $q(t) := (1-t)q^1 + tq^2$ for every $t \in [0, 1]$. Assume that the LCP (5.16) for $(M, q(t))$ is solvable for $t \in [0, 1]$. Then, there exists a partition $0 = t_0 < \dots < t_N = 1$ s.t. for $i = 1, \dots, N$*

$$q(t_{i-1}) \in Q(J_i), \quad q(t_i) \in Q(J_i), \quad \text{for some } J_i \subset \{1, \dots, n\}.$$

The constructive proof of this result can be found in [59, p.592]. Its main purpose is to provide a characterization of the active set changes along $[0, 1]$.

When D is positive definite, then Lemma 10 is instrumental to show that the solutions of the LCP (5.16) are Lipschitz w.r.t. q [59]. Among others, the latter reasoning relies on the fact that the positive definiteness of D uniquely defines the solution. However, in our framework, D is only positive semidefinite, and the result can not be applied directly. Instead of using much more abstract results of the literature (see e.g. [9, 53]), we decided to adapt the proofs of Mangasarian & Shiau [59] and to focus only on the part of the optimization variable that is uniquely defined.

More precisely, the variable \hat{x} defined in (5.15) equals

$$\hat{x} = (z_+, z_-, \rho_+, \rho_-, s_G, s_{\text{up}}, s_{\text{low}}, \eta),$$

where the first part “ z_+, z_- ” is uniquely defined, as pointed out in Proposition 12. Therefore, we consider that \hat{x} can be decomposed in two parts x_u and x_m , s.t. x_u is uniquely defined and $\hat{x} = (x_u, x_m)$.

Let us introduce the following theorem, which is a generalized version of [59, Thm. 3.2].

Theorem 6 (Lipschitz continuity of the uniquely defined components of the LCP). *Let q^1 and q^2 be points s.t. the LCP (5.16) for $(M, q(t))$ with $q(t) = (1-t)q^1 + tq^2$ has a solution $\hat{x}(t) = (x_u(t), x_m(t))$ s.t. $x_u(t)$ is unique, for every $t \in [0, 1]$. Then, there exists a constant $\sigma > 0$, depending only on M , s.t. any solutions $\hat{x}^1 = (x_u^1, x_m^1)$ and $\hat{x}^2 = (x_u^2, x_m^2)$ of (5.16) with respective vectors q^1 and q^2 satisfy*

$$\|x_u^1 - x_u^2\|_\infty \leq \sigma \|q^1 - q^2\|_2 \quad (5.18)$$

Proof. There exists a subdivision $0 = t_0 < t_1 < \dots < t_N = 1$ satisfying the properties of Lemma 10. For $i = 0, \dots, N$, let $\hat{x}(t_i) = (x_u(t_i), x_m(t_i))$ be a solution of (5.16) for $(M, q(t_i))$. Note that the $x_u(t_i)$ are unique, but the $x_m(t_i)$ are not necessarily unique.

⁶ M_j denotes the j^{th} row of M .

⁷If necessary, more details on the role of $Q(J)$ are provided in [59, Sec. 3].

For any set of indices $J \subset \{1, \dots, \dim \hat{x}\}$ and any matrix A , denote by A_J (resp. $A_{\bar{J}}$) the matrix composed of the rows of A whose indices are in J (resp. in $\{1, \dots, \dim \hat{x}\} \setminus J$). With this notation, for any $1 \leq i \leq N$ and for any $t \in [t_{i-1}, t_i]$, the LCP (5.16) reduces to the linear problem

$$\begin{pmatrix} M_{J_i} \\ \mathbb{1}_{\bar{J}_i} \end{pmatrix} \hat{x}(t) + \begin{pmatrix} q(t)_{J_i} \\ \mathbb{0}_{\bar{J}_i} \end{pmatrix} \geq 0 \quad \text{and} \quad \begin{pmatrix} M_{\bar{J}_i} \\ \mathbb{1}_{J_i} \end{pmatrix} \hat{x}(t) + \begin{pmatrix} q(t)_{\bar{J}_i} \\ \mathbb{0}_{J_i} \end{pmatrix} = 0 \quad (5.19)$$

by construction of J_i and $Q(J_i)$. Then, according to Theorem 5, stating the Lipschitz-continuity of feasible points of linear constraints, there exists a solution $\hat{y}(t_{i-1}) := (y_u(t_{i-1}), y_m(t_{i-1}))$ of (5.16) for $(M, q(t_{i-1}))$, i.e. a feasible point of (5.19) at $t = t_{i-1}$, s.t.

$$\|\hat{x}(t_i) - \hat{y}(t_{i-1})\|_\infty \leq \mu^i \|q(t_i) - q(t_{i-1})\|_2$$

for some $\mu^i > 0$. Let us define $\sigma := \max\{\mu^i \mid 1 \leq i \leq N\}$. Since the first part of $\hat{x}(t_{i-1})$ and $\hat{y}(t_{i-1})$ is uniquely defined, i.e. $x_u(t_i) = y_u(t_i)$, the following inequality holds

$$\begin{aligned} \|x_u^1 - x_u^2\|_\infty &\leq \sum_{i=1}^N \|x_u(t_i) - x_u(t_{i-1})\|_\infty \\ &= \sum_{i=1}^N \|x_u(t_i) - y_u(t_{i-1})\|_\infty \\ &\leq \sum_{i=1}^N \|\hat{x}(t_i) - \hat{y}(t_{i-1})\|_\infty \\ &\leq \sum_{i=1}^N \mu^i \|q(t_i) - q(t_{i-1})\|_2 \\ &\leq \sigma \sum_{i=1}^N \|(t_i - t_{i-1})(q^1 - q^2)\|_2 \\ &= \sigma \|q^1 - q^2\|_2. \end{aligned}$$

Hence the desired result. \square

Remark 29. The constants of Theorem 4 and Theorem 6 are actually defined via a constructive approach, which is detailed in [59].

5.3.4 Conclusion on the Lipschitz-continuity of HEGO

Theorem 7 (Lipschitz-continuity of Algorithm 2). *There exists a constant $L > 0$ s.t. for any inputs ξ^1 and ξ^2 in Λ , the unique solutions $(z^*)^1$ and $(z^*)^2$ returned by Algorithm 2 satisfy*

$$\|(z^*)^1 - (z^*)^2\|_2 \leq L \|\xi^1 - \xi^2\|_2.$$

Proof. This theorem links all the previously stated results. The optimal function z^* is well defined on Λ due to Proposition 12. Its value is the minimum of Problem (5.11), whose RHS is affinely dependent on the optimal penalties $\mathcal{P}_1^*, \dots, \mathcal{P}_R^*$. The latter are Lipschitz-continuous maps of ξ , as shown in Lemma 8. Moreover, the solutions of Problem (5.11) are solutions of the LCP (5.16), as recalled in Lemma 9. The uniquely defined components of the solutions of the LCP - i.e. the variables z_+ and z_- in \hat{x} - are Lipschitz-continuous functions of the vector q from (5.16), in the sense of Theorem 6. Also, the latter vector q is affinely dependent on the vector r^0 (see Lemma 9), which is affinely dependent on ξ and the optimal penalties $\mathcal{P}_1^*, \dots, \mathcal{P}_R^*$. Thus, by composition, z_+ and z_- are Lipschitz-continuous maps of ξ , in the sense of Equation (5.18). The desired result, i.e. with the 2-norm on both sides of the Lipschitz inequality, stems from the equivalence of the norms in finite dimension. \square

A direct corollary of Theorem 7 is the following.

Corollary 3. *Let $z^*(\xi)$ denote the value returned by Algorithm 2 with input ξ . Then, the optimal value function $\xi \mapsto J(z^*(\xi))$ is Lipschitz-continuous on Λ .*

5.4 Monotonicity of the optimal negotiations

This section aims at giving a mathematical meaning to the sentence “*the farther from the reference trajectory, the higher the negotiation*”, for the special case $R = 1$, i.e. when there is no hierarchy involved. Getting farther from the reference is modeled *directionally* by considering the map

$$t \mapsto t\xi$$

where $t \geq 0$ and ξ is an arbitrary input direction. Among others, we aim at showing that the map $t \mapsto \mathcal{P}_1^*(t\xi)$ is non-decreasing, for non-negative values of t , which is what we call the *negotiation monotonicity*.

Formally, when $R = 1$, there is only a single negotiation problem, that writes

$$\mathcal{P}^* \longleftarrow \min_{z,p} \|p\|_1 \tag{5.20a}$$

$$Gz \leq h_0 + H_\xi \xi + H_p p \tag{5.20b}$$

$$Az = b_0 + B_\xi \xi + B_p p \tag{5.20c}$$

$$p_{\text{low}} \leq p \leq p_{\text{up}} \tag{5.20d}$$

where \mathcal{P}^* is the optimal value. Using the same kind of re-writing technique as in Section 5.3.1, one can show that there are matrices M and Q , and a vector r , s.t. Problem (5.20) is equivalent to the following LP in its standard primal form

$$V_P^*(r) \longleftarrow \min_{x,y} \mathbb{1}^\top x \tag{5.21a}$$

$$\text{s.t. } Mx + Qy = r, \tag{5.21b}$$

$$x, y \geq 0. \tag{5.21c}$$

where $V_P^*(r)$ denotes its optimal value function, and $r = r_0 + K\xi \in \mathbb{R}^{nr}$ for some matrix K and some vector r_0 . The problems are equivalent in the sense that

$$\mathcal{P}^*(\xi) = V_P^*(r_0 + K\xi). \quad (5.22)$$

Following Proposition 15 in the Appendix, the dual associated to Problem (5.21) is

$$V_D^*(r) \longleftarrow \max_{\mu} \mu^\top r \quad (5.23a)$$

$$\text{s.t. } M^\top \mu \leq \mathbf{1}, \quad (5.23b)$$

$$Q^\top \mu \leq 0. \quad (5.23c)$$

Also, we will consider a RHS change in an arbitrary direction d . Let us define

$$\mathcal{W} := \{t \in \mathbb{R} : V_P^*(r + td) \text{ is feasible and finite}\} = \{t \in \mathbb{R} : V_D^*(r + td) \text{ is finite}\}.$$

Lemma 11. *For any r , $V_P^*(r + td)$ is feasible and finite if and only if $V_D^*(r + td)$ is finite.*

Proof. If $V_P^*(r + td)$ is feasible and finite, then $V_D^*(r + td)$ is finite by the Strong Duality Theorem⁸. For the converse case, note that $V_D^*(r + td)$ is always feasible since $\mu = 0$ satisfies (5.23b) and 5.23c. Also recall that the Dual of $V_D^*(r + td)$ is $V_P^*(r + td)$. Therefore, using once more the Strong Duality Theorem, if $V_D^*(r + td)$ is finite then $V_P^*(r + td)$ is feasible and finite. This gives the conclusion. \square

For $t \in \mathcal{W}$, let us define the function v s.t.

$$v(t) := V_P^*(r + td) = V_D^*(r + td) \quad (5.24)$$

which highlights the absence of duality gap.

Now that the directional RHS sensitivity function is well-defined, thanks to Theorem 11, we can use standard results from the literature (as recalled by Proposition 16 in Appendix) to formulate the following theorem.

Theorem 8 (Direct application of [5, 70]). *\mathcal{W} is a closed interval (possibly unbounded). v is a continuous convex function, which is piecewise affine on a finite number of sub-intervals of \mathcal{W} .*

Theorem 9. *Consider any ξ and define $d := K\xi$. Assume that*

$$(i) \ V_P^*(r_0) = 0,$$

$$(ii) \ \xi \text{ is s.t. } V_P^*(r_0 + td) \text{ is defined for } t > 0, \text{ on a (small) non-trivial interval.}$$

Then, the negotiation map $t \mapsto \mathcal{P}^(t\xi) := V_P^*(r_0 + td)$ is non-decreasing on $\mathcal{W} \cap \mathbb{R}^+$.*

Proof. We proceed by combining the local affine description of $t \mapsto V_P^*(r_0 + td)$ with the above mentioned results. Let us introduce the quantity $D_d(r)$ s.t.

$$D_d(r) \longleftarrow \max_{\lambda} \lambda^\top d$$

$$\text{s.t. } M^\top \lambda \leq \mathbf{1},$$

$$Q^\top \lambda \leq 0,$$

$$\lambda^\top r = V_P^*(r).$$

⁸Recalled as Theorem 11 in Appendix.

Thanks to the absence of duality gap over \mathcal{W} , as shown in Equation (5.24), Theorem 12 used with assumption (ii) states that there exists a $t' > 0$ s.t. for any scalar $t \in [0, t']$ we have

$$V_P^*(r_0 + td) = V_P^*(r_0) + tD_d(r_0)$$

where $D_d(r_0)$ is finite. This formula is what we call *Gauvin's formula* in the Appendix.

Since $V_P^*(r_0) = 0$ by assumption (i), then 0 is a feasible vector for $D_d(r_0)$, and consequently $D_d(r_0) \geq 0$. Finally, since $t \mapsto \mathcal{P}^*(t\xi)$ is convex (from Theorem 8) and has a non-negative slope at $t = 0$, it is necessarily non-decreasing on $\mathcal{W} \cap \mathbb{R}^+$. \square

Remark 30. Assumption (i) in Theorem 9 is guaranteed by the assumption stating that the reference trajectory must satisfy the constraints of NLP (ξ, p) : for null inputs - i.e. RHS equals r_0 in Problem (5.21) - there is no need for negotiation (i.e. $V_P^*(r_0) = 0$).

Remark 31. Assumption (ii) in Theorem 9 is not over restrictive. It means that we consider only inputs ξ s.t. the negotiation problem remains feasible when exploring the inputs in this direction.

Remark 32. For the general emergency problem - i.e. when there are $R \geq 2$ negotiable sub-parameters - the negotiation maps $t \mapsto \mathcal{P}_j^*(t\xi)$ behave differently. Showing that they are all continuous and piecewise affine can be achieved with little effort, but they are generally not convex, as the introductory example of Section 5.2.2 shows. This latter example also shows that even $t \mapsto \sum_{j=1}^R \mathcal{P}_j^*(t\xi)$ is not necessarily convex.

However, it has been conjectured that the negotiation maps $t \mapsto \mathcal{P}_j^*(t\xi)$ remain non-decreasing functions of t . This conjecture is currently under investigation, at the time of writing this manuscript.

5.5 Non-monotonicity of the optimal trajectories

Contrary to the negotiation maps, the directional optimal solution maps $t \mapsto z^*(t\xi)$ defined by the outputs of HEGO are not necessarily component-wise monotonous. In the same fashion as in Section 5.2.2, consider the following example which illustrates this property. It is one of the reason why the heatmaps of the optimal time-of-flight variation, i.e. Figures B.9 and B.20, and the curve of Δt_f^* , in Figure B.23, exhibit such complex patterns.

Basic Example 4. Consider an optimization problem with $z = (z_1, z_2)^\top \in \mathbb{R}^2$, with a scalar input $\xi \geq 0$, and with a scalar negotiable parameter $p \geq 0$. Consider four constraints s.t.

$$\begin{aligned} (\text{Const. 1}) \quad & z_2 \geq \xi \\ (\text{Const. 2}) \quad & z_2 \geq z_1 \\ (\text{Const. 3}) \quad & z_2 \leq 2 + p - z_1 \\ (\text{Const. 4}) \quad & z_2 \leq 2 + z_1 \end{aligned}$$

To alleviate the writing, we denote by a function “ c ” the latter constraints s.t. these are satisfied if and only if $c(\xi, z, p) \leq 0$. In this case, the (single) negotiation problem associated to these constraints is simply

$$\begin{aligned} \mathcal{P}^* \quad \leftarrow \quad & \min_{z_1, z_2, p} p \\ & \text{s.t. } c(\xi, z, p) \leq 0, \\ & p \geq 0. \end{aligned}$$

Moreover, given the arbitrary cost $J(z) := \frac{1}{2}(z_1 - 2)^2 + \frac{1}{2}(z_2 + 2)^2$, the refine problem writes

$$\begin{aligned} \min_{z_1, z_2, p} \quad & J(z) \\ \text{s.t.} \quad & c(\xi, z, p) \leq 0, \\ & p \geq 0, \\ & p \leq \mathcal{P}^*. \end{aligned}$$

Using HEGO with the two latter problems, we observe that the first component of the optimal solution, i.e. z_1^* , has a non-monotonous behavior w.r.t. ξ , as illustrated in Figure 5.4.

5.6 Emergency guidance method generalization

HEGO, as presented in Algorithm 2, is an emergency guidance method built upon a nominal guidance method that relies on Quadratic Programming. This convenient framework, with its linear constraints, allows the use of mature LP and QP solvers for the implementation of the problems LP_j and Refine. However, it is possible to take some distance from this presentation, as other kinds of mathematical programming are often used to provide an approximation of PDG (ξ, p) , as discussed in Chapter 1. Thus, let us describe the link between nominal and emergency guidance methods from a high-level perspective.

This section can be skipped with no loss of continuity.

5.6.1 Generalized notations

Let us denote by $J(z, \xi)$ the cost that must be minimized, and by $\mathcal{F}_{eas}(\xi, p) \subset \mathbb{R}^{N_z}$ the feasible set, which conveys the various control constraints, the dynamic model, etc. We insist that this set can convey much more general constraints than Equation (5.1). For example, it could stem from any direct collocation method, as presented in well-known references [14, 45, 104], or other state-of-the-art methods [58]. Also, it is desired that $p \in \Omega$ for some set $\Omega \subset \mathbb{R}^{n_{neg}}$.

5.6.2 Generalized emergency order

The goal is still to provide a trajectory, even when $\mathcal{F}_{eas}(\xi, 0)$ is empty. To that purpose, instead of the 1-norm, let us introduce more general negotiation functions

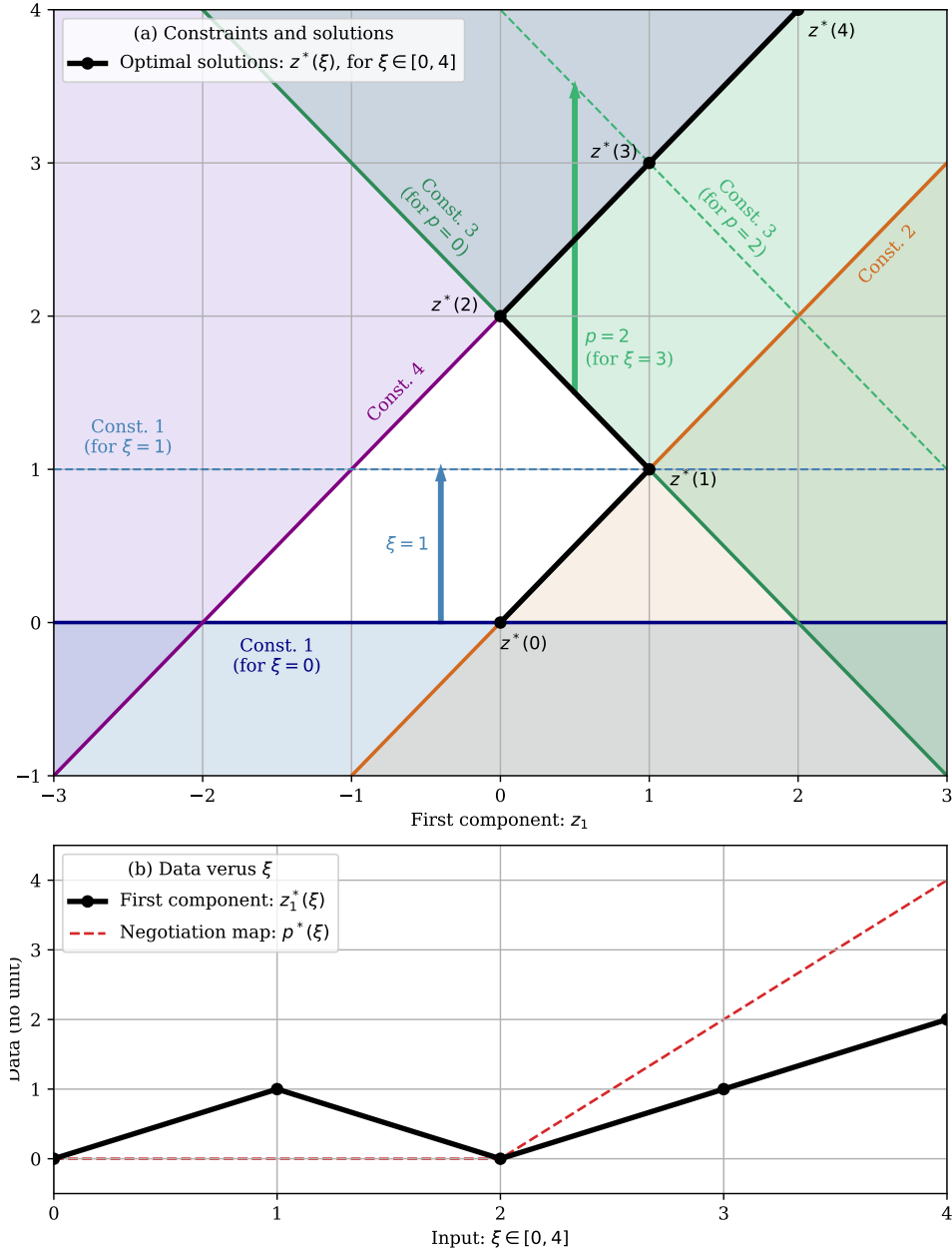


Figure 5.4: *Illustration of Basic Example 4.* This example shows that there can be components z_i^* among the outputs of HEGO that are non-monotonous.

$\gamma_j : \mathbb{R}^{n_j} \rightarrow \mathbb{R}^+$, for $j = R, \dots, 1$, assumed convex. It helps us define a generalized emergency order denoted \succeq_γ . The latter is also co-lexical, as \succeq_e , but in the sense of the negotiation functions γ_j . Like in (5.3), a vector p_a is said to be more negotiated than another vector p_b s.t. $p_a \succeq_\gamma p_b$, if and only if

$$\begin{aligned} & \gamma_R(p_a^{(R)}) > \gamma_R(p_b^{(R)}) \\ \text{or } & \gamma_R(p_a^{(R)}) = \gamma_R(p_b^{(R)}) \text{ and } \gamma_{R-1}(p_a^{(R-1)}) > \gamma_{R-1}(p_b^{(R-1)}), \\ \text{or } & \dots \\ \text{or } & \gamma_R(p_a^{(R)}) = \gamma_R(p_b^{(R)}) \text{ and } \dots \text{ and } \gamma_1(p_a^{(1)}) > \gamma_1(p_b^{(1)}), \\ \text{or } & \gamma_R(p_a^{(R)}) = \gamma_R(p_b^{(R)}) \text{ and } \dots \text{ and } \gamma_1(p_a^{(1)}) = \gamma_1(p_b^{(1)}). \end{aligned}$$

Therefore, the general meaning of *maximizing the launcher's integrity* is to find the smallest p in the sense of \succeq_γ .

5.6.3 Generalized sequence of optimization problems

Let us here define a more general version of LP_j and Refine. With the above-defined notations, the *generalized* nominal guidance problem is the following optimization problem

$$\begin{aligned} (\text{Nominal}) \quad & \min_z J(z, \xi), \\ & \text{s.t. } z \in \mathcal{F}_{eas}(\xi, 0). \end{aligned}$$

As before, what is of interest here is what happens when ξ is s.t. $\mathcal{F}_{eas}(\xi, 0)$ is empty. Thus, we will seek to minimize a cost γ_i for each parameter $p^{(i)}$.

Let us introduce the first negotiation problem, dealing with the most critical negotiation parameter $p^{(R)}$, s.t.

$$\begin{aligned} \Gamma_R^* \quad & \longleftarrow \min_{z, p} \gamma_R(p^{(R)}), \\ & \text{s.t. } z \in \mathcal{F}_{eas}(\xi, p), \\ & p \in \Omega, \end{aligned}$$

where Γ_R^* denotes the optimal value of this problem. Let S_R be the set of points p s.t. there is a $z \in \mathcal{F}_{eas}(\xi, p)$ and s.t. (z, p) minimizes γ_R . In other words, S_R is the set of minimizers of the latter problem, but projected on the negotiable parameters set. By successively negotiating the other parameters, in a similar fashion as LP_j , we define the follow-up negotiation problems as

$$\begin{aligned} \Gamma_j^* \quad & \longleftarrow \min_{z, p} \gamma_j(p^{(j)}), \\ & \text{s.t. } z \in \mathcal{F}_{eas}(\xi, p), \\ & p \in \Omega, \\ & p^{(j+1)} \in S_{j+1}, \end{aligned}$$

where $j = 1, \dots, R-1$, and where S_i denotes the set of minimizers of the i^{th} problem, projected onto the negotiable parameters set (which is, by its recursive definition, a

subset of the previous sets S_i for $i = j + 1, \dots, R$). Consequently, the *generalized* version of the Refine problem becomes

$$\begin{aligned} z^*, \star &\longleftarrow \operatorname{argmin}_{z, p} J(z, \xi), \\ \text{s.t. } & z \in \mathcal{F}_{eas}(\xi, p), \\ & p \in \Omega, \\ & p^{(1)} \in S_1 \end{aligned}$$

where the notation z^*, \star means that the value of p is ignored, since it is not necessarily unique. The two latter problems can be re-written in a more convenient form.

Since the conditions $z \in \mathcal{F}_{eas}(\xi, p)$ and $p \in \Omega$ do not change between these problems, it is possible to simplify the above-mentioned notations by switching the abstract condition $p^{(j+1)} \in S_{j+1}$ into

$$\gamma_i(p^{(i)}) = \Gamma_i^*, \quad \forall i = j + 1, \dots, R. \quad (5.25)$$

However, even for a convex function γ_i , the equality constraint $\gamma_i(p^{(i)}) = \Gamma_i^*$ is numerically ill-posed, since it defines a non-convex level-set in general (e.g. when $\gamma_i(\cdot) = \|\cdot\|_2$). Thankfully, it can be relaxed without losing generality, by using only the inequality

$$\gamma_i(p^{(i)}) \leq \Gamma_i^*. \quad (5.26)$$

Indeed, for any $j = 1, \dots, R - 1$, if there is a pair (z, p) s.t.

$$\begin{aligned} z &\in \mathcal{F}_{eas}(\xi, p), \\ p &\in \Omega, \\ \gamma_i(p^{(i)}) &\leq \Gamma_i^*, \quad \forall i = j + 1, \dots, R \end{aligned}$$

then by construction of Γ_i^* as the minimum feasible point $p^{(i)}$, we necessarily have $\gamma_i(p^{(i)}) \geq \Gamma_i^*$, showing that $\gamma_i(p^{(i)}) = \Gamma_i^*$ still holds for each $i = j + 1, \dots, R$.

Therefore, the generalized negotiation and refine operations boil down to the following problems

$$\Gamma_j^* \longleftarrow \min_{z, p} \gamma_j(p^{(j)}), \quad (5.27a)$$

$$\text{s.t. } z \in \mathcal{F}_{eas}(\xi, p), \quad (5.27b)$$

$$p \in \Omega, \quad (5.27c)$$

$$\gamma_i(p^{(i)}) \leq \Gamma_i^*, \quad \forall i = j + 1, \dots, R \quad (5.27d)$$

and

$$z^*, \star \longleftarrow \operatorname{argmin}_{z, p} J(z, \xi), \quad (5.28a)$$

$$\text{s.t. } z \in \mathcal{F}_{eas}(\xi, p), \quad (5.28b)$$

$$p \in \Omega, \quad (5.28c)$$

$$\gamma_i(p^{(i)}) \leq \Gamma_i^*, \quad \forall i = 1, \dots, R \quad (5.28d)$$

5.6.4 High-level description of safety margins

Enforcing the condition $z \in \mathcal{F}_{eas}(\xi, p)$ for some value ξ may sometimes bring the variables of the problem to the frontier of what is feasible for a given p . Therefore, given a set \mathcal{M} that contains 0, we would like to impose that if z is feasible for a value of p , then for every $\Delta p \in \mathcal{M}$ there is another z' feasible for $p + \Delta p$. Consequently, the condition $z \in \mathcal{F}_{eas}(\xi, p)$ from the previous problems (5.27) and (5.28) must be modified into

$$z \in \mathcal{F}_{eas}(\xi, p), \quad (5.29a)$$

$$\mathcal{F}_{eas}(\xi, p + \Delta p) \neq \emptyset, \quad \forall \Delta p \in \mathcal{M} \quad (5.29b)$$

The latter modeling falls into the context of robust optimization (see e.g. [10, Ch.1]). It cannot be used as-is, since it conveys an infinite number of constraints. Let us make two further assumptions in order to simplify (5.29)

1. The set \mathcal{F}_{eas} is convex in z , and linearly influenced by p . For example, we assume the existence of a (possibly non-linear) convex function g_ξ and (possibly non-linear) matrix-valued maps $H(\cdot)$, $A(\cdot)$, $b(\cdot)$ and $B(\cdot)$ s.t.

$$\mathcal{F}_{eas}(\xi, p) = \{z \in \mathbb{R}^{N_z} : g_\xi(z) \leq H(\xi) \cdot p \text{ and } A(\xi) \cdot z = b(\xi) + B(\xi) \cdot p\}$$

2. The set \mathcal{M} is convex, and assumed to have a finite number of extreme point. In other words, there are K vectors $\Delta p^i \in \mathbb{R}^{n_{neg}}$ s.t.

$$\mathcal{M} = \text{ConvexHull}(\Delta p^1, \dots, \Delta p^K).$$

Proposition 14. *Under the two latter assumptions, there are elements z^k s.t.*

$$z^k \in \mathcal{F}_{eas}(\xi, p + \Delta p^k), \quad \forall k = 1, \dots, K$$

if and only if conditions (5.29a) and (5.29b) are satisfied.

Proof. The return implication of the equivalence is guaranteed by construction. To prove the direct implication, let us consider any arbitrary vector $\Delta \bar{p} \in \mathcal{M}$. By construction of \mathcal{M} , there are K non-negative scalars σ_k s.t. $\sum \sigma_k = 1$ and

$$\Delta \bar{p} = \sum_{k=1}^K \sigma_k \Delta p^k.$$

Let us denote by z^k an element of $\mathcal{F}_{eas}(\xi, p + \Delta p^k)$ and consider the vector $\bar{z} := \sum_{k=1}^K \sigma_k \Delta p^k$. Using the convexity of g_ξ , one gets

$$g_\xi(\bar{z}) \leq \sum_{k=1}^K \sigma_k g_\xi(z^k) \leq \sum_{k=1}^K \sigma_k H(\xi) \cdot (p + \Delta p^k) = H(\xi) \cdot (p + \Delta \bar{p})$$

and likewise: $A(\xi) \bar{z} = b(\xi) + B(\xi) \cdot (p + \Delta \bar{p})$. Therefore, \bar{z} belongs to $\mathcal{F}_{eas}(\xi, p + \Delta \bar{p})$, whence (5.29b). Constraint (5.29a) also holds, since it corresponds to the sub-case $\Delta \bar{p} = 0$, hence the conclusion. \square

Therefore, incorporating Proposition 14 into Problems 5.27 and 5.28 is a way to enforce safety margins while performing nominal an emergency guidance, using only a finite number of constraints.

5.7 Illustrations

In this section, four numerical examples are proposed along with qualitative discussions. Example 1 illustrates the basic principles of HEGO. Example 2 shows that a wide selection of negotiable parameters can be used together. Example 3 demonstrates the modeling capabilities offered by HEGO. Finally, Example 4 shows how emergency guidance scales to the 3D model. A quantitative analysis of this last example is proposed in Section 6.2 of Chapter 6.

Note that all of these examples have the same number of discretization points, i.e. $N = 4$, where N is defined in p. 56 right before Equation (4.1). For the 2D model (resp. the 3D model), it means that the size of z is $N_z = 15$ (resp. $N_z = 22$).

All the data presented in the examples below is normalized.

5.7.1 With the 2D model

Example 1 (Basic 2D scenario). *Let us consider a simple choice of negotiable parameters, consisting in the incidence limit $\Delta\alpha_{\max}$ and the final horizontal position Δz^f s.t.*

$$p = \left(\Delta\alpha_{\max}, \Delta z^f \right)^\top.$$

In terms of hierarchy, we impose

- $p^{(1)} = \Delta\alpha_{\max}$ (least critical),
- $p^{(2)} = \Delta z^f$ (most critical).

This example is illustrated in Figures 5.5 and 5.6, where the input ξ is dispersed for positive values of the change in initial horizontal position Δz^0 .

Example 2 (Advanced 2D scenario). *Let us consider a more advanced version of Example 1, where the negotiable parameter is now*

$$p = \left(\Delta\alpha_{\max}, \Delta a_{\text{nor}}^{\max}, \Delta z^f, \Delta h^f \right)^\top$$

In terms of hierarchy, we impose

- $p^{(1)} = \Delta\alpha_{\max}$ (least critical),
- $p^{(2)} = \Delta a_{\text{nor}}^{\max}$,
- $p^{(3)} = \Delta z^f$,
- $p^{(4)} = \Delta h^f$ (most critical).

This example is illustrated in Figure 5.7.

Recall that Δh^f denotes the final altitude, which can seem surprising at first site. Why do not we use the final vertical speed Δv_h^f instead? The reason is linked to the linearization used when we define QP (ξ). Indeed, Δv_h^f appears, in practice, to be a

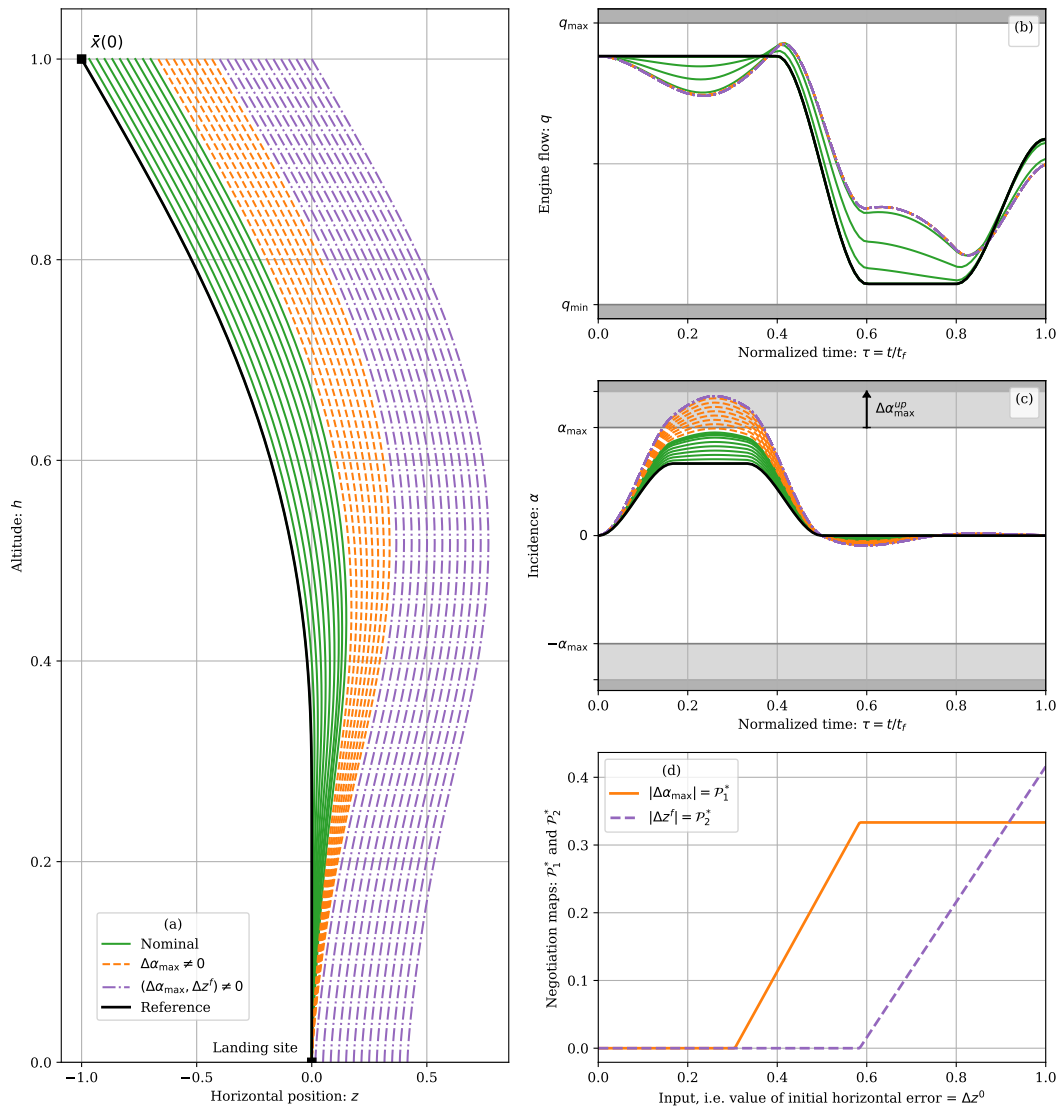


Figure 5.5: Dispersion of Δz^0 over $[0, 1]$, for the 2D rocket model from Example 1. The curves of sub-Figures (a), (b) and (c) are plotted for 30 values of Δz^0 . The similarity between the introductory example of Section 5.2.2 and this actual example is clear in the sub-Figure (d).

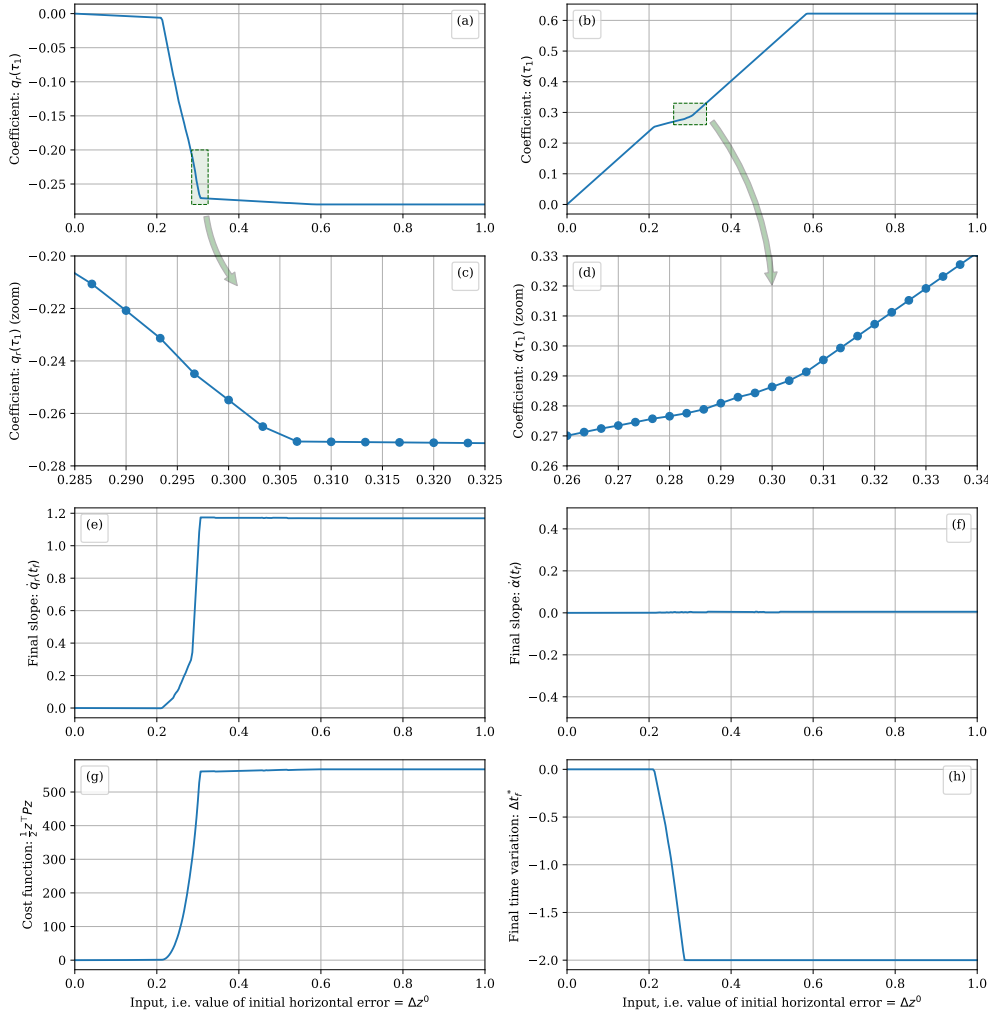


Figure 5.6: Dispersion of Δz^0 over $[0, 1]$, for the 2D rocket model from Example 1. All these charts have very different y -scales. The charts (c) and (d) are zoomed views of (a) and (b). Except for the charts (c) and (d), where the blue dots represent the computed values, all the curves have been plotted for 301 values of Δz^0 . The charts (a) to (f) shows a subset of μ , of size $2(N + 3)$. Precisely, using the nomenclature from Equation (4.1), $q_r(\tau_1) = \mu_3$, $\alpha(\tau_1) = \mu_4$, $\dot{q}_r(t_f) = \mu_{2(N+3)-1}$ and $\dot{\alpha}(t_f) = \mu_{2(N+3)}$. The Lipschitz-continuity stated in Theorem 7 can be observed on all the charts, except (g) whose Lipschitz-continuity is related to Corollary 3. Note that nominal guidance is performed up to $\Delta z^0 \approx 0.3$ (see Figure 5.5-(d)), highlighting the fact that significant active set changes can occur even with nominal guidance. Also, as one can observe in (h) between $\Delta z^0 = 0.0$ and 0.2 , Δt_f^* has a non-zero though very small slope. This is mostly due to the fact that changing Δt_f^* has a strong influence on simultaneously the vertical and the horizontal components of the trajectory, meanwhile Δz^0 influences (almost only) the horizontal one.

less useful lever than Δh^f . Mathematically speaking, it means that the image of the matrix⁹ $B_{\Delta v_h^f}$ does not describe the same vector space as the one of $B_{\Delta h^f}$, and thus will not have the same ability to recover the changes in ξ . This remark also applies to the matrices $B_{\Delta v_h^f}$ and $B_{\Delta h^f}$.

In practice, negotiating Δh^f is blindly allowed in the optimization problems of HEGO. However, when HEGO says that $\Delta h^f < 0$ is necessary, the optimal trajectory z^* will reach the ground before reaching the new final altitude Δh^f . The state x at which the altitude of this trajectory reaches null altitude may be defined as the actual negotiated landing state.

Note that in Figure 5.7, dispersing the inputs w.r.t. Δz^0 does not trigger the use of Δh^f , whose negotiation curve \mathcal{P}_4^* remain flat in the sub-Figure (d). However, this example is the ground base of the numerical assessment provided in Chapter 6, where sufficiently rich scenarios are considered, and show how Δh^f is used.

Example 3 (2D scenario with repeated negotiable parameters). To demonstrate the modeling capabilities that Algorithm 2 offers, let us consider an example where several negotiable parameter are “repeated”:

$$p = \left(\Delta\alpha_{\max}^1, \Delta z_1^f, \Delta\alpha_{\max}^2, \Delta z_2^f \right)^\top$$

In terms of hierarchy, we impose

- $p^{(1)} = \Delta\alpha_{\max}^1$ (least critical),
- $p^{(2)} = \Delta z_1^f$,
- $p^{(3)} = \Delta\alpha_{\max}^2$,
- $p^{(4)} = \Delta z_2^f$ (most critical).

This example is illustrated in Figure 5.8.

This choice of negotiation parameters allows to negotiate the incidence and the final horizontal position alternatively. This may be helpful when the nature of the area neighboring the landing site becomes increasingly worse when moving away. To some extent, it can be applied to the terrain presented in Figure 1.3 from Chapter 1: the order of priority is to first negotiate the incidence, then the horizontal position (as long as it remains within the crops or the beach), then the incidence again, and finally trying to land in the forest or in the ocean.

5.7.2 With the 3D model

Example 4 (3D scenario). Consider the direct transcription of Example 2 to the 3D rocket model, s.t.

$$p = \left(\Delta\alpha_{\max}, \Delta a_{\text{nor}}^{\max}, \Delta z^f, \Delta y^f, \Delta h^f \right)^\top$$

In terms of hierarchy, we impose

⁹For instance, $B_{\Delta h^f}$ means the column of B_p corresponding to Δh^f .

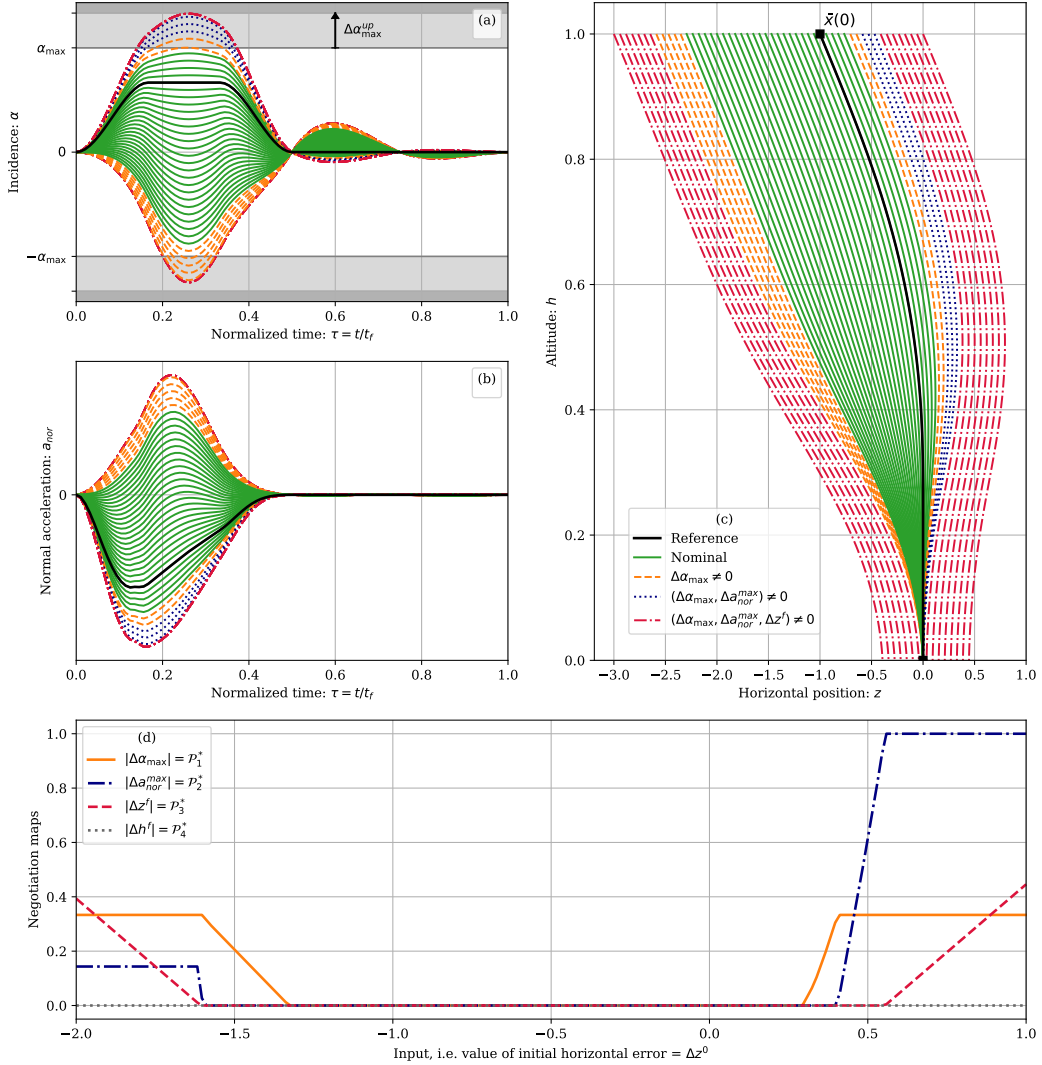


Figure 5.7: Dispersion of Δz^0 over $[-2, 1]$, for the 2D rocket model from Example 2. In sub-Figure (c), a thinner mesh (not required for the purpose of this example) would reveal the presence of blue-dot trajectories between the orange and the red ones, on the left of the reference trajectory. The asymmetry in the negotiation maps (sub-Figure (d)) comes from the reference trajectory itself, which is a non-trivial curve in the plane (z, h) .

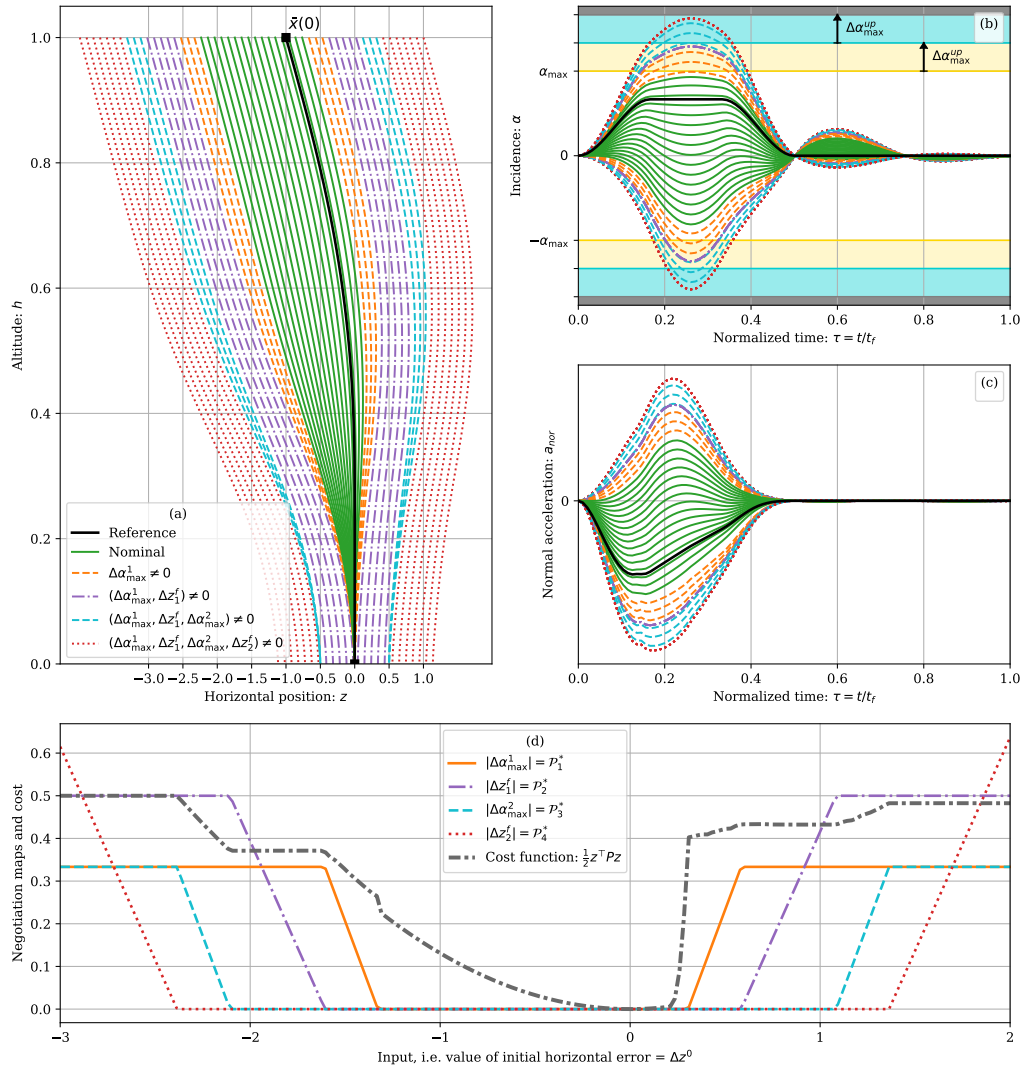


Figure 5.8: Dispersion of Δz^0 over $[-4, 2]$, for the 2D rocket model from Example 3. Note that the negotiation maps from sub-Figure (d) are exactly the kind of negotiation structure that support the conjecture presented in Remark 32. The Lipschitz-continuity of the optimal cost function, stated in Corollary 3, can be observed on the same sub-Figure. Also, remark that the cost function variations convey several active set changes within the nominal guidance mode, that can be visualized through the slope changes. The reason why the cost function remains constant when the horizontal final positions are being negotiated is that the optimal trajectory is shifted purely horizontally in these cases.

- $p^{(1)} = \Delta\alpha_{\max}$ (least critical),
- $p^{(2)} = \Delta a_{\text{nor}}^{\max}$,
- $p^{(3)} = (\Delta z^f, \Delta y^f)^\top$,
- $p^{(4)} = \Delta h^f$ (most critical).

This example is illustrated in Figure 5.9. To illustrate behaviors that are not shared with the 2D rocket model, the input are dispersed w.r.t. Δy^0 , the initial horizontal position component that is out-of-plane compared to the reference trajectory.

These negotiation parameters are used for the quantitative analysis in Chapter 6.

Summary

In this chapter, we exposed a method to provide emergency guidance. It boils down to a sequential minimization of the amplitude of negotiable parameters, enforcing a prescribed hierarchy between these parameters, and is implemented using a finite number of LPs and a single QP. The method HEGO is capable of producing both nominal (when possible) and emergency guidance solutions thanks to a unified formulation.

Theoretical guarantees prove that the outputs of HEGO are Lipschitz-continuous w.r.t. its inputs, preventing the solutions from varying *too fast* for small changes in the inputs.

Numerical simulations have demonstrated how HEGO behaves on 2D and 3D examples, and that its outputs are consistent with the above-mentioned theoretical guarantees.

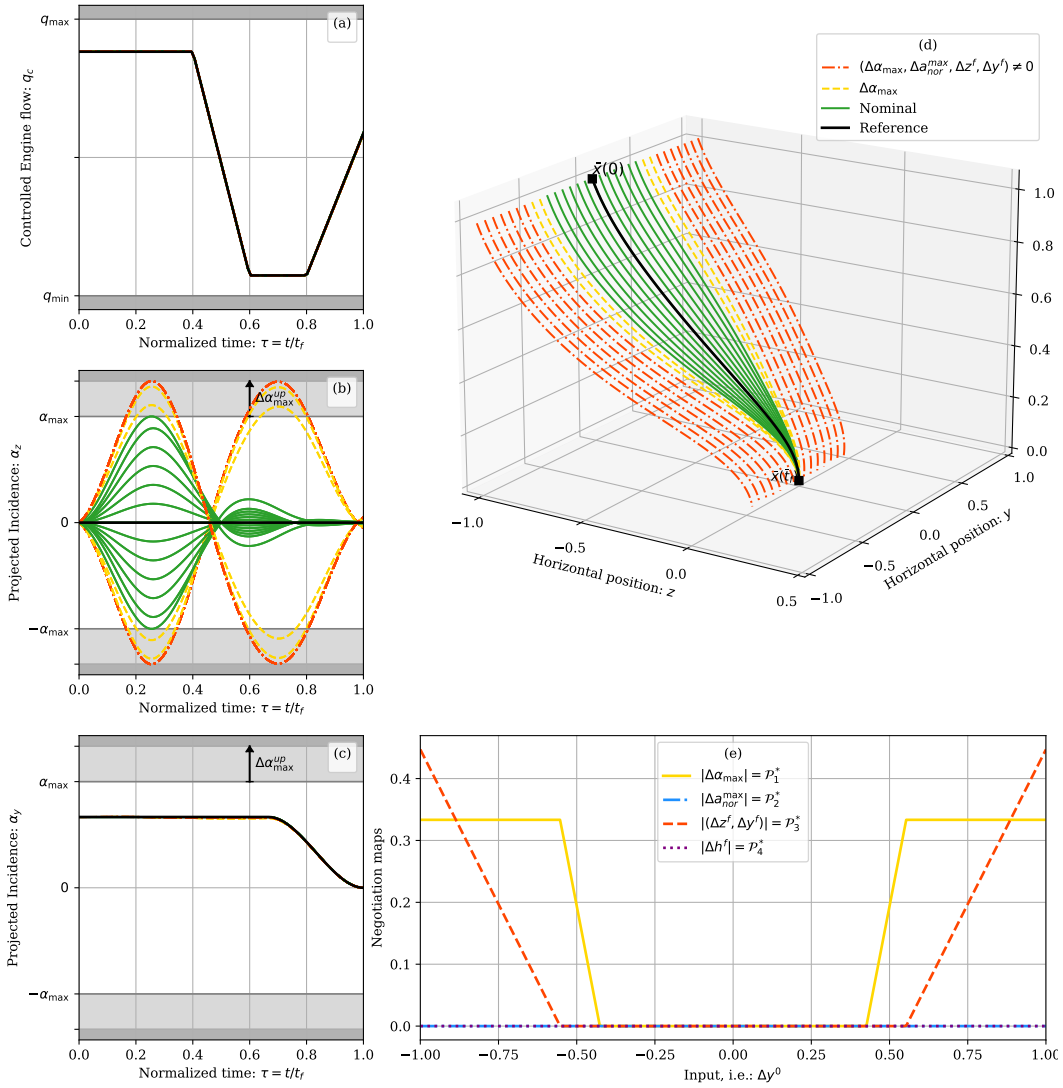


Figure 5.9: Dispersion of Δy^0 over $[-1, 1]$, for the 3D rocket model. Dispersing the inputs along Δy^0 leads to the computation of out-of-plane trajectories. The dispersion of the inputs of HEGO will be analyzed quantitatively on this exact same scenario in Chapter 6.

Chapter 6

Performance evaluation

Résumé

L'objectif du Chapitre 5 était de présenter l'Algorithme 2 (aussi appelé HEGO), qui assure la fonction de guidage même lorsque l'entrée ξ rend les contraintes nominales infaisables. Son comportement a été illustré qualitativement dans plusieurs exemples à la fin du Chapitre 5.

Nous proposons ici des évaluations quantitatives pour HEGO. Tout d'abord, des commentaires de haut niveau concernant l'implémentation d'HEGO sont proposés. Ensuite, nous présentons une analyse quantitative de l'Exemple 4 du Chapitre 5, en calculant les dispersions bivariées des entrées sur le modèle de fusée 3D. En outre, comme HEGO vise à traiter des scénarios d'atterrissage infaisables, ces résultats sont comparés aux enveloppes de vol vertical introduites dans le Chapitre 3.

The purpose of Chapter 5 was to introduce Algorithm 2 (also denoted HEGO), which provides guidance even when the input ξ makes the nominal constraints infeasible. Its qualitative behavior has been illustrated in several examples at the end of Chapter 5.

Here, we propose quantitative assessments for HEGO. First, we provide high-level comments regarding the implementation of HEGO. Then, we present a quantitative analysis of Example 4 from Chapter 5, by computing bivariate dispersions of the inputs on the 3D rocket model. Also, since HEGO aims at dealing with infeasible landing scenarios, its outcomes are compared with the vertical flight envelopes introduced in Chapter 3.

To improve readability, the figures of this chapter have been moved at its end.

6.1 General comments

The time required to run HEGO depends directly on the design choices presented in Chapters 4 and 5.

HEGO has been implemented in `python` and tested with `cvxopt` [8], `mosek` [7], `glpk` [38] and `qpSWIFT` [74] as underlying solvers for the LP and QP solvers. The benchmarks have been forced to run on a single CPU (in practice, tested on an

Intel® Core™ i9-9900K, at 3.60GHz, and on an Intel® Core™ i7-8550U CPU at 1.80GHz). Its run time typically ranges between

- a few milliseconds for the 2D rocket model with 2 negotiable parameters,
- and ≈ 60 ms for the 3D rocket model with 5 negotiable parameters.¹

Major influence on run time

The parameters that naturally affect the computation time are the dimensions N_z (i.e. main decision variable z) and n_{neg} (i.e. negotiation parameter p). Let us denote by $N_{\text{opt}} := N_z + n_{\text{neg}}$ the dimension of the optimization variable involved in the sub-problems of HEGO. Recall that m is the dimension of the control variable, and that N is the number of discretization points (as shown in Figure 4.1). Since Cubic Splines are used for the description of the control corrections in Chapter 4, we get

$$N_{\text{opt}} = (m + 3)N + 1 + n_{\text{neg}}$$

showing the relative importance of the above-mentioned sizes. Also, the computation time may be influenced by the number of constraints, which is directly proportional to N_c , as defined in Equation (4.8), and by the number of partitions of the negotiable parameter (i.e. R defined in Chapter 5).

Minor or no influence on run time

It is noteworthy that some parameters do not influence, or in a negligible way, the computation time. Among others, the dimension of ξ has a negligible impact on the run time of HEGO. Indeed, it only matters when computing the constraints right-hand side

$$\begin{aligned} Gz &\leq h_0 + \boxed{H_\xi \xi} + H_p p, \\ Az &= b_0 + \boxed{B_\xi \xi} + B_p p. \end{aligned}$$

A useful application would be to handle thinner descriptions of the atmosphere parameters, such as the wind for instance. Let us assume that one is able to measure the horizontal wind component at a high resolution for the low atmosphere layers (let us say 1 point of measure per 10 m, up to 10 km, for illustration purposes). Then, ξ would resemble:

$$\xi = \left((\Delta x^0)^\top, w_{0\text{m}}, w_{10\text{m}}, w_{20\text{m}}, \dots, w_{10000\text{m}} \right)^\top$$

The dimension of ξ now equals 1009 in this case (when Δx^0 is of dimension 8 for the 3D rocket model of Chapter 2), though this has a non-significant impact on HEGO run-time.

¹These performances have been demonstrated live during the thesis defense, with on-demand initial condition changes.

6.2 Input dispersion on 3D rocket model

Let us consider the same reference trajectory as in Example 4, for the 3D rocket model. We pick the same choices of negotiable parameters as in Example 4, i.e.

$$p = \left(\Delta\alpha_{\max}, \Delta a_{\text{nor}}^{\max}, \Delta z^f, \Delta y^f, \Delta h^f \right)^\top$$

with $R = 4$ s.t.

- $p^{(1)} = \Delta\alpha_{\max}$ (least critical),
- $p^{(2)} = \Delta a_{\text{nor}}^{\max}$,
- $p^{(3)} = \left(\Delta z^f, \Delta y^f \right)^\top$,
- $p^{(4)} = \Delta h^f$ (most critical).

The purpose of this section is to quantify and assess the quality of the outputs of HEGO, over bivariate dispersions of the inputs. We consider the 15 inputs presented in Table 6.2. From the 105 pairs of inputs that can be formed from this list, we report results obtained with a selection of 20 pairs, enumerated in Table 6.3. These pairs have been selected for their representativeness, and because they demonstrate a wide variety of behaviors.

To categorize the outputs of HEGO, we use the notion of *emergency mode*. It is defined as the index of the most critical negotiable sub-parameter that has a non-zero 1-norm, thus taking its values within $\{0, \dots, R + 1\}$. Mathematically, using handy notations, it is defined s.t.

$$\text{EmergencyMode}(\xi) := \begin{cases} 0 & \text{if } \mathcal{P}_1^*(\xi) = \dots = \mathcal{P}_R^*(\xi) = 0, \\ R + 1 & \text{if } \text{LP}_R(\xi) \text{ is not feasible,} \\ \arg \max_{i=1, \dots, R} \{ i \mid \mathcal{P}_i^*(\xi) \neq 0 \} & \text{otherwise.} \end{cases} \quad (6.1)$$

where the values $\mathcal{P}_i^*(\xi)$ come from negotiation problems of HEGO. The color coding associated with the emergency modes is presented in Table 6.1.

6.2.1 Selection of figures

First, two pairs are detailed in Figures 6.1 and 6.2. They present the outputs of HEGO for two pairs, respectively $(\Delta z^0, \Delta y^0)$ and $(\Delta q_r^0, \Delta q_c^0)$. The first pair yields a typical collection of inputs that require the negotiation capabilities of HEGO, whereas the second pair shows that in some cases no negotiation is needed.

To ease the visualization, the pairs from Table 6.3 have been split in two batches, labeled A and B. For each batch, the following charts are provided:

- Emergency mode map (Figures 6.3 and 6.4),
- Projected incidence α_y (Figures 6.5 and 6.6),







Color	Value	Label	Comment
	0	Nominal	Negotiation not needed.
	1	$\Delta\alpha_{\max}$	is the most critical negotiated sub-parameter.
	2	$\Delta a_{\text{nor}}^{\max}$...
	3	$(\Delta z^f, \Delta y^f)^\top$...
	4	Δh^f	...
	5	\times	HEGO has no solutions.

Table 6.1: Color coding for the emergency modes.

- Projected incidence α_z (Figures 6.7 and 6.8),
- Controlled engine flow q_c (Figures 6.9 and 6.10),
- Heatmap of the first negotiation map: \mathcal{P}_1^* (Figures 6.11 and 6.12),
- Heatmap of the third negotiation map: \mathcal{P}_3^* (Figures 6.13 and 6.14).

For the figures having the input pairs as their x and y -axis, the black dot \bullet conveys the origin $(0, 0)$ (for example the emergency mode maps). Also, the reference trajectory \bar{x} and the reference control \bar{u} are represented in **plain black**.

Additional data is provided in Appendix, regarding the various states for these dispersions, the heatmap of the optimal time-of-flight change Δt_f^* , and the other negotiation maps.

6.2.2 Observations and comments

Variable combinations. Some variables combinations have a natural *constructive* or *destructive* behavior. The pair $(\Delta z^0, \Delta v_z^0)$ is the clearest one. Starting farther away from the landing site but with a greater horizontal speed (or closer but with a lower horizontal speed) is a typical constructive behavior that explains the green strip in Figure 6.3-(a).

Continuity. The negotiation maps $\xi \mapsto \mathcal{P}_i^*(\xi)$ discussed in Chapter 5 are Lipschitz-continuous. The reader might think that some of the maps below would suggest the contrary, due to abrupt changes. It is only a matter of zoom and Lipschitz constants. The detailed example pictured in Figure 6.15 provides various zoom levels to see how the negotiation maps can vary.

Normal acceleration negotiation. As indicated by the black line in Figure 6.16, the ratio between the maximum normal acceleration of the reference trajectory and the normal acceleration bound is close to 1, leaving little room for negotiations. However, it is not always necessary to negotiate $\Delta a_{\text{nor}}^{\max}$, as pictured in Figure 6.1. For a large number of values of the pair $(\Delta z^0, \Delta y^0)$, the guidance trajectories provided

Part of ξ	Variable	Points	Dispersion interval	Comment
Δx^0	Δz^0	81	$[-1, 1]$	$\pm 100\%$ of z^0
	Δy^0	81	$[-1, 1]$	
	Δh^0	81	$[-0.2, 0.2]$	$\pm 20\%$ of h^0
	Δv_z^0	61	$[-2, 2]$	$\pm 100\%$ of v_z^0
	Δv_y^0	61	$[-2, 2]$	
	Δv_h^0	81	$[-0.2, 0.2]$	$\pm 20\%$ of v_h^0
	Δm^0	41	$[-0.2, 0.2]$	$\pm 20\%$ of remaining fuel
	Δq_r^0	41	$[0, 2]$	Largest possible interval
Δu_{init}	Δq_c^0	41	$[-0.8, 0.22]$	Largest possible interval
	$\Delta \alpha_y^0$	41	$[-1, 1]$	Largest possible interval
	$\Delta \alpha_z^0$	41	$[-0.83, 0.15]$	Largest possible interval
$\Delta \eta$	ΔISP	41	$[-2, 2]$	
	$\Delta \tau_q$	41	$[-0.8, 1]$	
	$w_{z,0}$	61	$[-1, 1]$	
	$w_{y,0}$	61	$[-1, 1]$	

Table 6.2: List of dispersed input variables (subset of the components of ξ). The dispersion intervals are given in normalized units. Beware, one normalized unit of z differs from one normalized unit of y .

Label	Component 1	Component 2	Batch
a	Δz^0	Δv_z^0	
b	Δz^0	Δv_y^0	
c	Δz^0	Δy^0	
d	Δz^0	Δh^0	A
e	Δz^0	Δv_h^0	
f	Δz^0	$w_{z,0}$	
g	Δz^0	$w_{y,0}$	
h	Δv_z^0	Δv_y^0	
i	Δy^0	Δv_h^0	
j	Δy^0	Δh^0	
k	Δy^0	Δv_z^0	
l	Δy^0	Δv_y^0	
m	Δm^0	Δh^0	
n	Δm^0	Δv_h^0	B
o	$\Delta \alpha_z^0$	$\Delta \alpha_y^0$	
p	ΔISP	$\Delta \tau_q$	
q	Δq_c^0	Δq_r^0	
r	ΔISP	Δh^0	
s	ΔISP	Δv_h^0	
t	$\Delta \tau_q$	Δh^0	

Table 6.3: List of pairs of input components dispersed in the charts below. For example, using the data from Table 6.2, there are $81 \times 81 = 6561$ pairs of values of $(\Delta z^0, \Delta v_z^0)$ that have been considered.

Negotiable parameter	Usefulness		
	Overall	Horizontal	Vertical
$\Delta\alpha_{\max}$	+++	+++	0
$\Delta a_{\text{nor}}^{\max}$	++	++	0
$(\Delta z^f, \Delta y^f)$	+++	+++	0
Δh^f	+	0	+

Table 6.4: *Performance summary.* The qualitative scale goes from 0 (useless) to +++ (extremely useful).

by HEGO correspond to sharper turns, but does not require to negotiate $\Delta a_{\text{nor}}^{\max}$ by a lot. The corresponding heatmaps (i.e. the heatmaps representing \mathcal{P}_2^*) are in the Appendix.

Several complementary comments are provided along with the Figures below.

6.2.3 Conclusion on the example

This example demonstrated that the negotiable parameters have very different impacts on how they help solve the emergency problem.

- $\Delta\alpha_{\max}$ and $(\Delta z^f, \Delta y^f)$ appear to be extremely relevant to recover feasible trajectories when inputs influencing the horizontal behavior of the landing change. $\Delta a_{\text{nor}}^{\max}$ has a non-negligible though small influence on these changes.
- Δh^f has an influence on the vertical part of the landing, but the trajectories relaxed using Δh^f are not doable in practice, for the margin it offers is too thin.

These remarks are summarized in Table 6.4.

6.3 Comparison with vertical flight envelopes

As detailed in Chapter 5, Algorithm HEGO provide landing guidance for nominal and emergency problems. Thus, the set of inputs that do not require emergency guidance define the “nominal reachable set”, in the sense of HEGO. Due to the approximations made in Chapter 4 (such as the discretization and the linearization), the set of inputs that are considered feasible without negotiation differ from the actual reachable set.

The full characterization of the reachable set of the “complete” rocket model (i.e. the 2D or 3D rocket model) is not computationally tractable, due to the nonlinearities and the number of states involved. However, as detailed at the end of Chapter 3, it is possible to fully describe the reachable set of the rocket for purely

Label	Figure	Trajectory type	Engine flow structure	Comment
A	6.18	Vertical	Near Min-Max	
B	6.19	Vertical	Max-Min-Middle	
C	6.20	Planar	Max-Min-Middle	Reference trajectory of Section 6.2.
D	6.21	Fully 3D	Max-Min-Max	Out-of-plane reference.

Table 6.5: *Reference trajectories considered for the benchmark w.r.t. the vertical flight envelope.* All four trajectories share the same reference time-of-flight \bar{t}_f . Note that all of these trajectories have been computed using the 3D rocket model, even if it has been used with a purely vertical motion. These trajectories are represented in Figure 6.17.

vertical motion. Due to the high slenderness ratio of the rocket final-burn, this latter set is a coarse approximation of the actual reachable set of the complete model.

Thus, we propose a comparison between the classification offered by HEGO in terms of emergency modes, and the vertical flight envelope defined in Chapter 3. We consider four different reference trajectories, which are listed in Table 6.5 and displayed in Figure 6.17.

For all four trajectories, a slice of the reachable set is considered, and HEGO is computed over a grid of inputs on each of these slices. In the chart (a) of their respective, the Max, Min and Min-Max surfaces defining the vertical flight envelope are defined at the end of Chapter 3.

The main observation is that HEGO is *conservative*. Indeed, on one hand, it declares some inputs “non-nominal” when they would be expected to be feasible without negotiating the constraints, as shown analytically in Chapter 3. On the other hand, there are no “false-positive”, i.e. all the inputs declared “nominal” by HEGO are indeed inside the vertical flight envelope. Also, it is noteworthy that the reference trajectory significantly influences the outcomes of HEGO, though it does not change the latter conservative property.

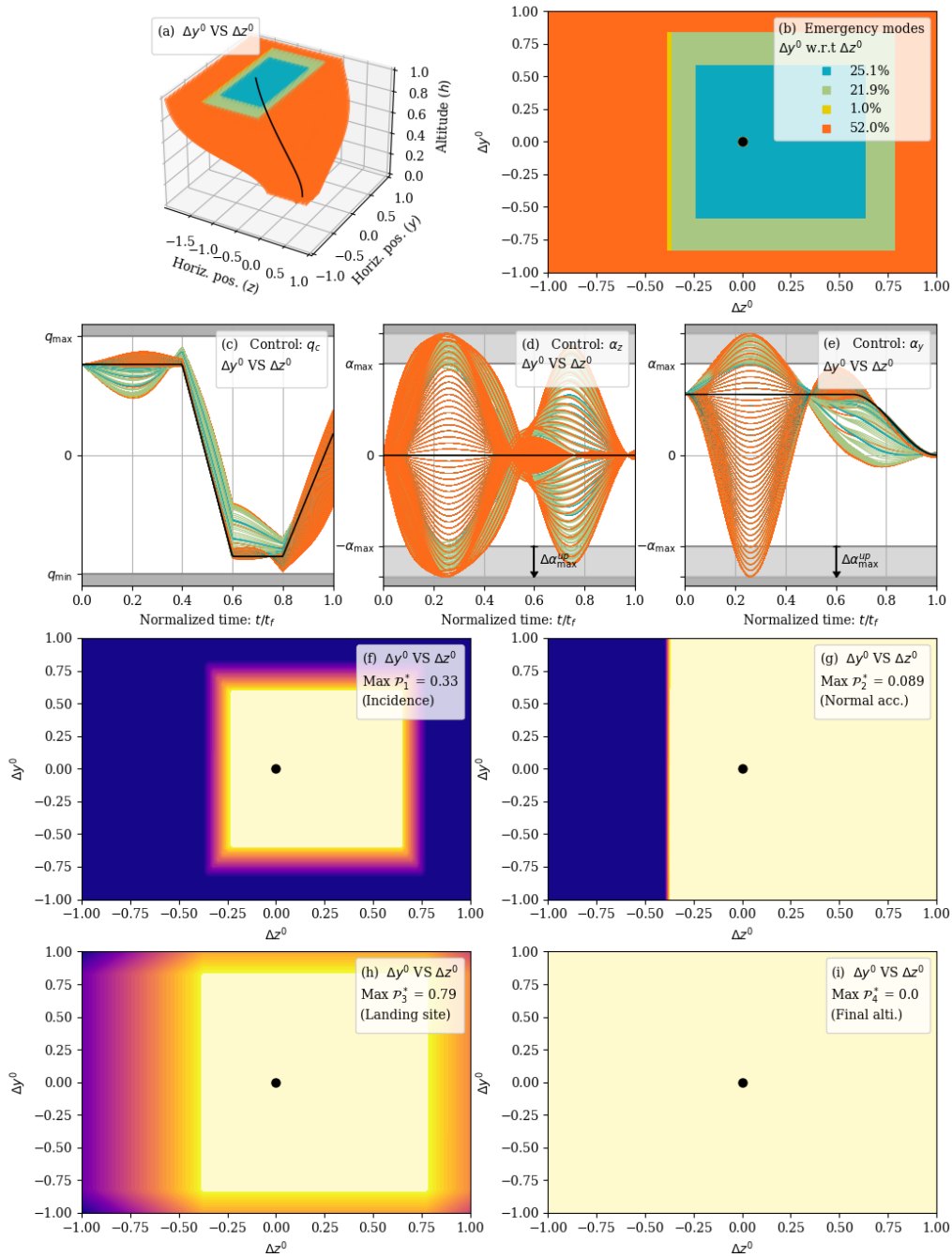


Figure 6.1: *Dispersion of $(\Delta z^0, \Delta y^0)$.* Dispersing this pair leads to in-plane and out-of-plane trajectories. The negotiation maps (charts (f) to (i)) have a structure deeply linked to the partition described in the emergency mode map from chart (b).

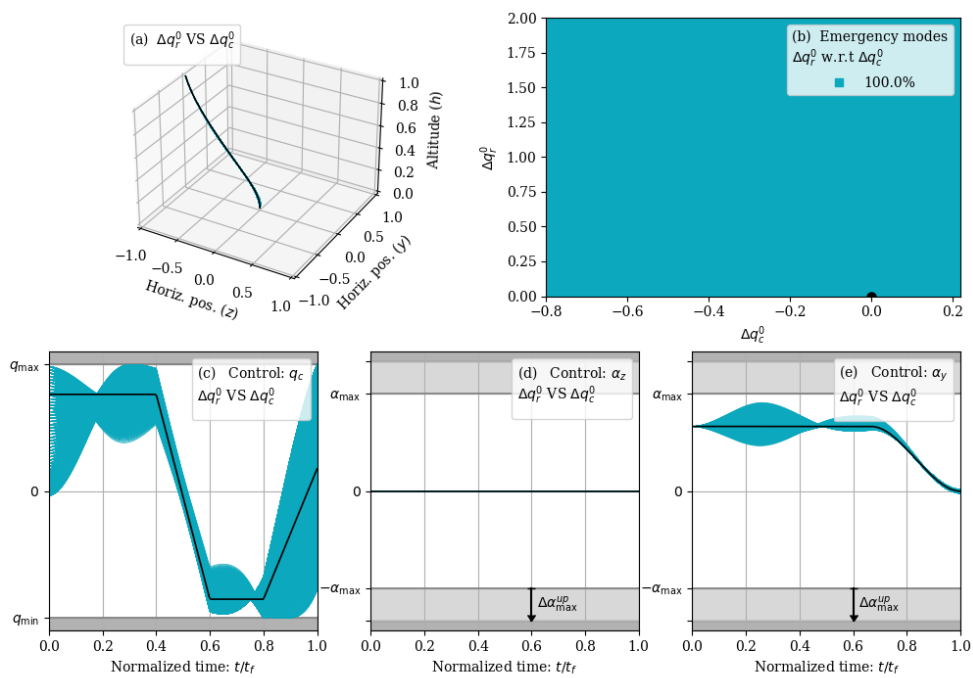


Figure 6.2: Dispersion of $(\Delta q_c^0, \Delta q_r^0)$. The inputs correspond to purely nominal scenarios. The negotiation maps are not displayed, since they are all constant and equal to zero.

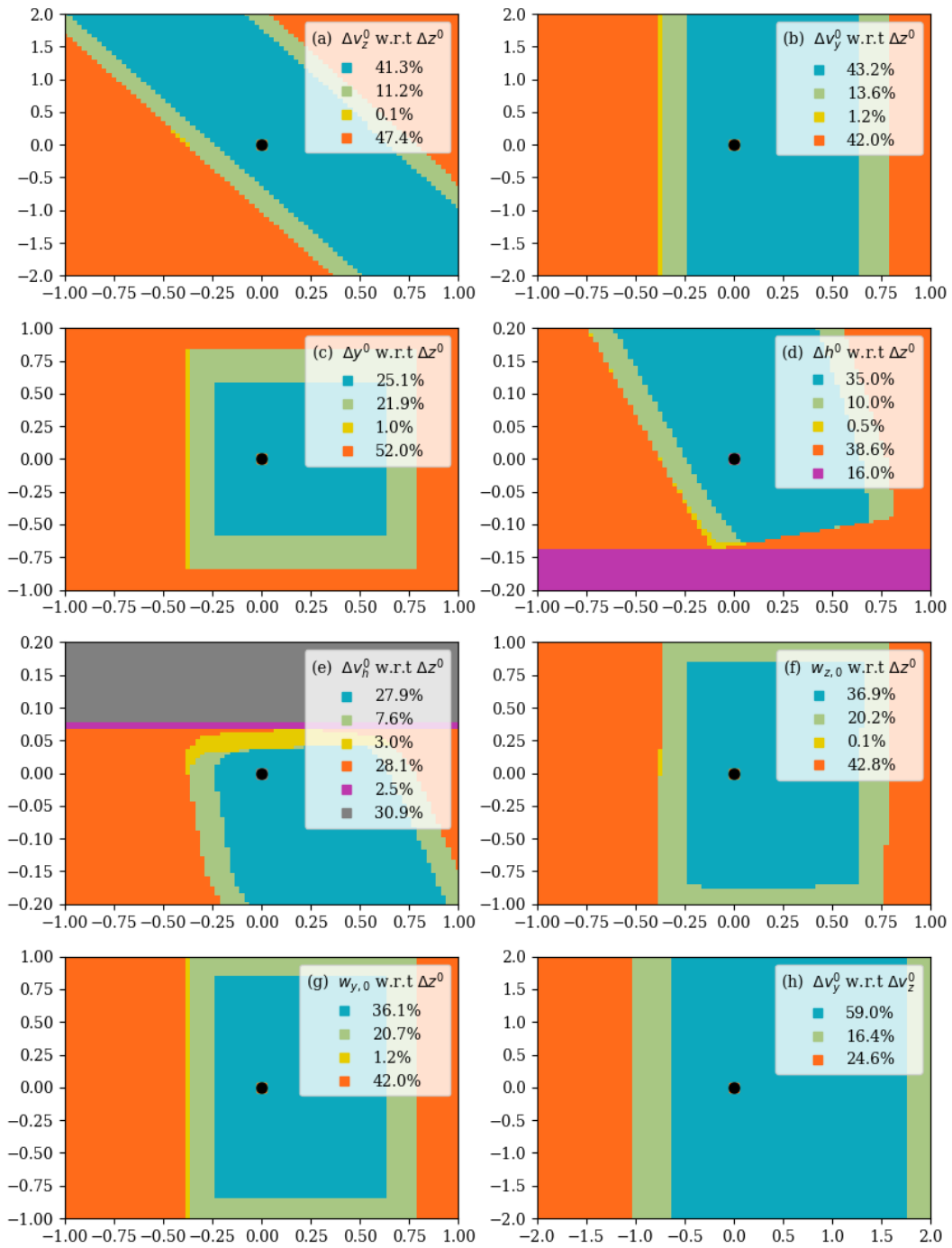


Figure 6.3: *Emergency modes. Batch A.* Each chart represents the emergency modes obtained by dispersing a given pair of inputs. For instance, in (a), Δv_z^0 w.r.t Δz^0 means that Δz^0 is in the x -axis, and Δv_z^0 is in the y -axis. Since the role of the out-of-plane variables (such as Δy^0 , Δv_y^0 or $w_{y,0}$) is symmetric in this problem having a planar reference trajectory, their associated charts have an axis of symmetry.

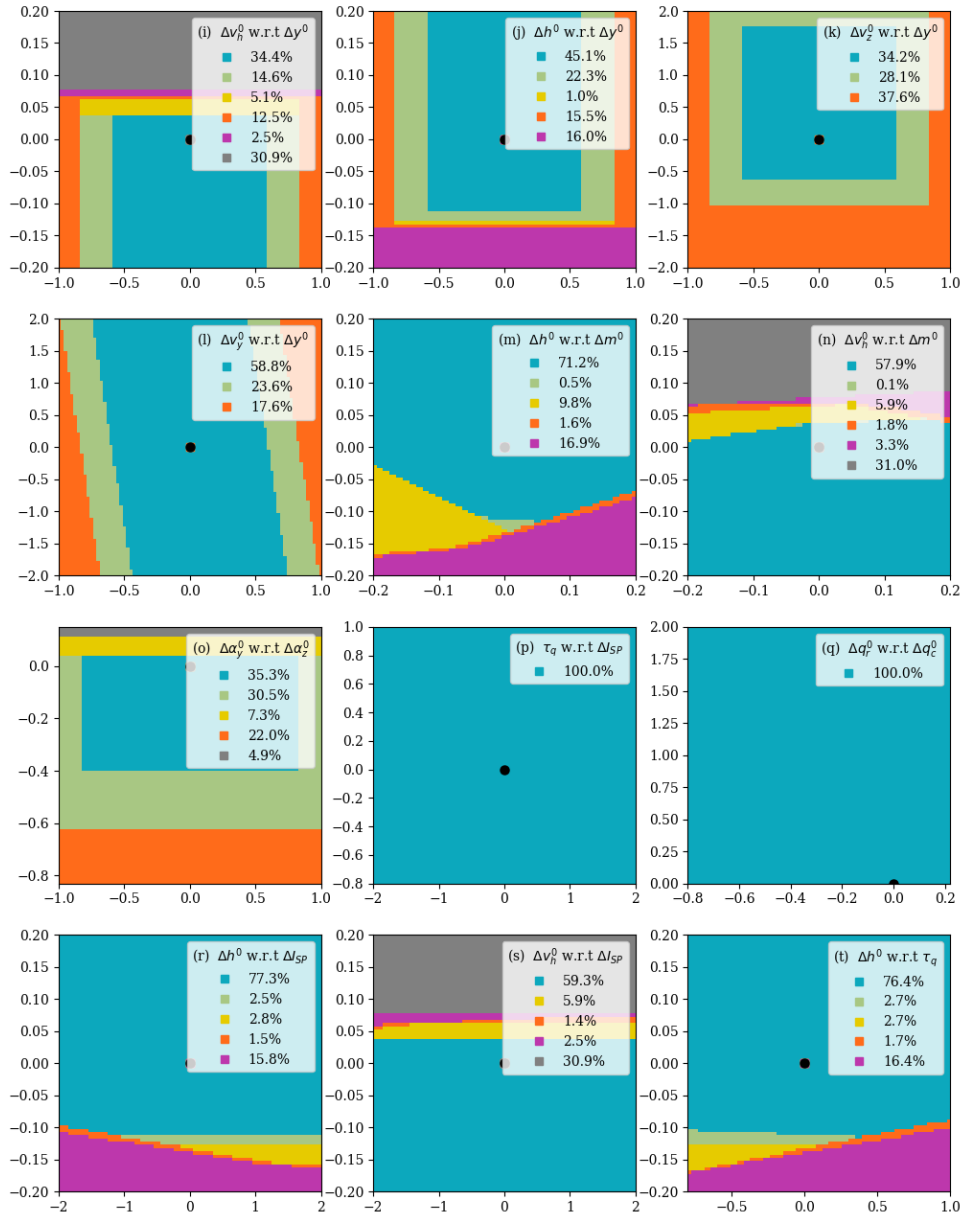


Figure 6.4: *Emergency modes. Batch B.* Finding purely horizontal or purely vertical lines separating different emergency modes for some specific pairs reveals that the associated input variables are uncorrelated. Among others, the variables conveying the vertical motion (h, v_h, m, I_{SP}) are mostly uncorrelated with the ones conveying the horizontal motion (z, v_z, y, v_y), especially when considering the emergency mode of Δh^f . The pairs (i) and (m) are typical supporting examples. Moreover, contrary to what appears to be, charts (p) and (q) are not feature-less. The specific case of the pair (q) is detailed above in Figure 6.2.

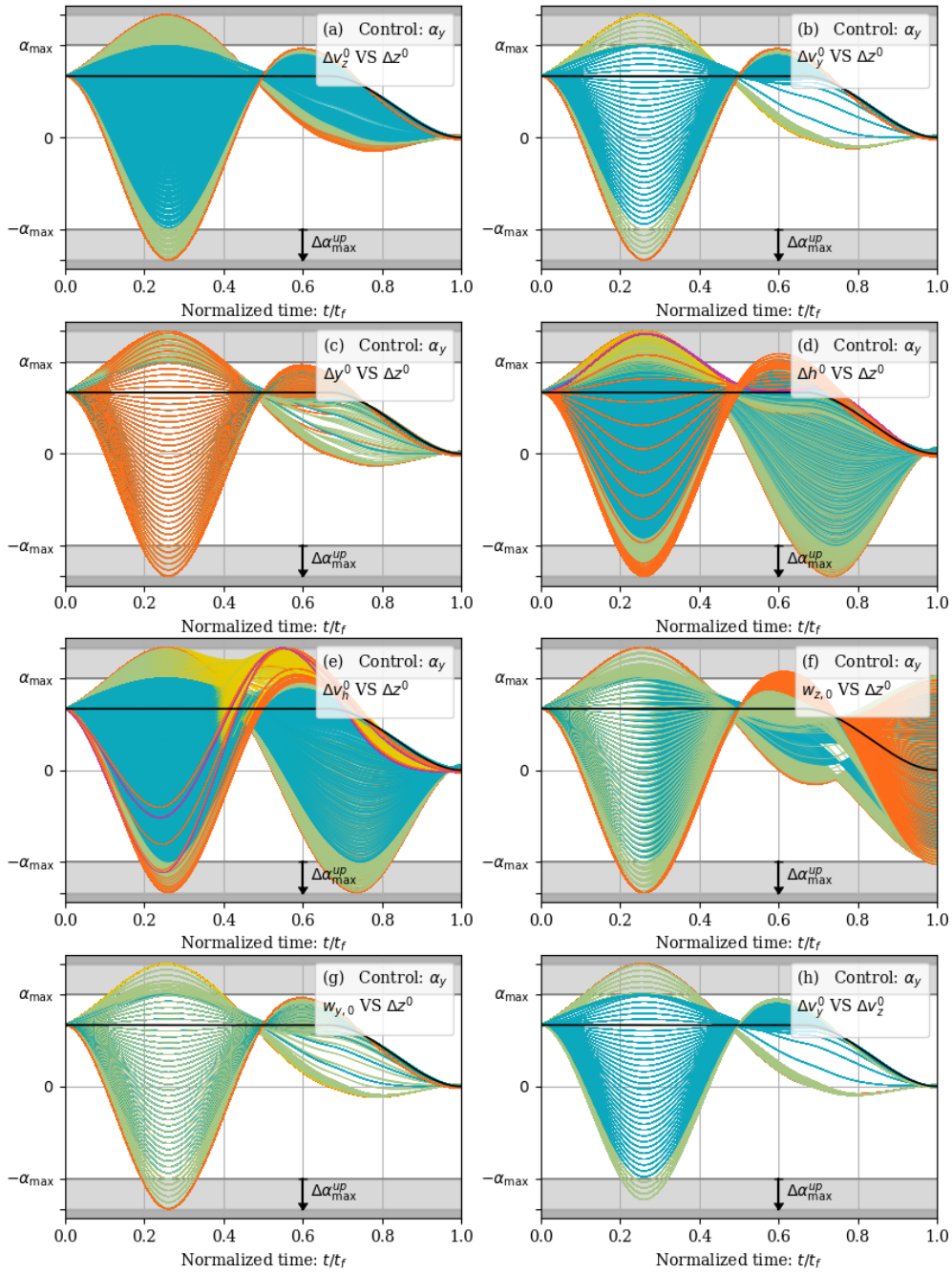


Figure 6.5: *Projected incidence α_y (in-plane) w.r.t. normalized time. Batch A.* In most charts, the curves describe a sort of *pivot* around a the normalized time $\tau = 0.5$. This behavior is directly linked to the Cubic Spline description of the correction, detailed in Chapter 4. Note that for some pairs - e.g. pair (e) - the incidence is negotiated but not always up to its maximum value, even when more critical parameters are negotiated first. The main reason is due to the way the active set changes w.r.t. to the input, which is illustrated in detail in Figure B.23.

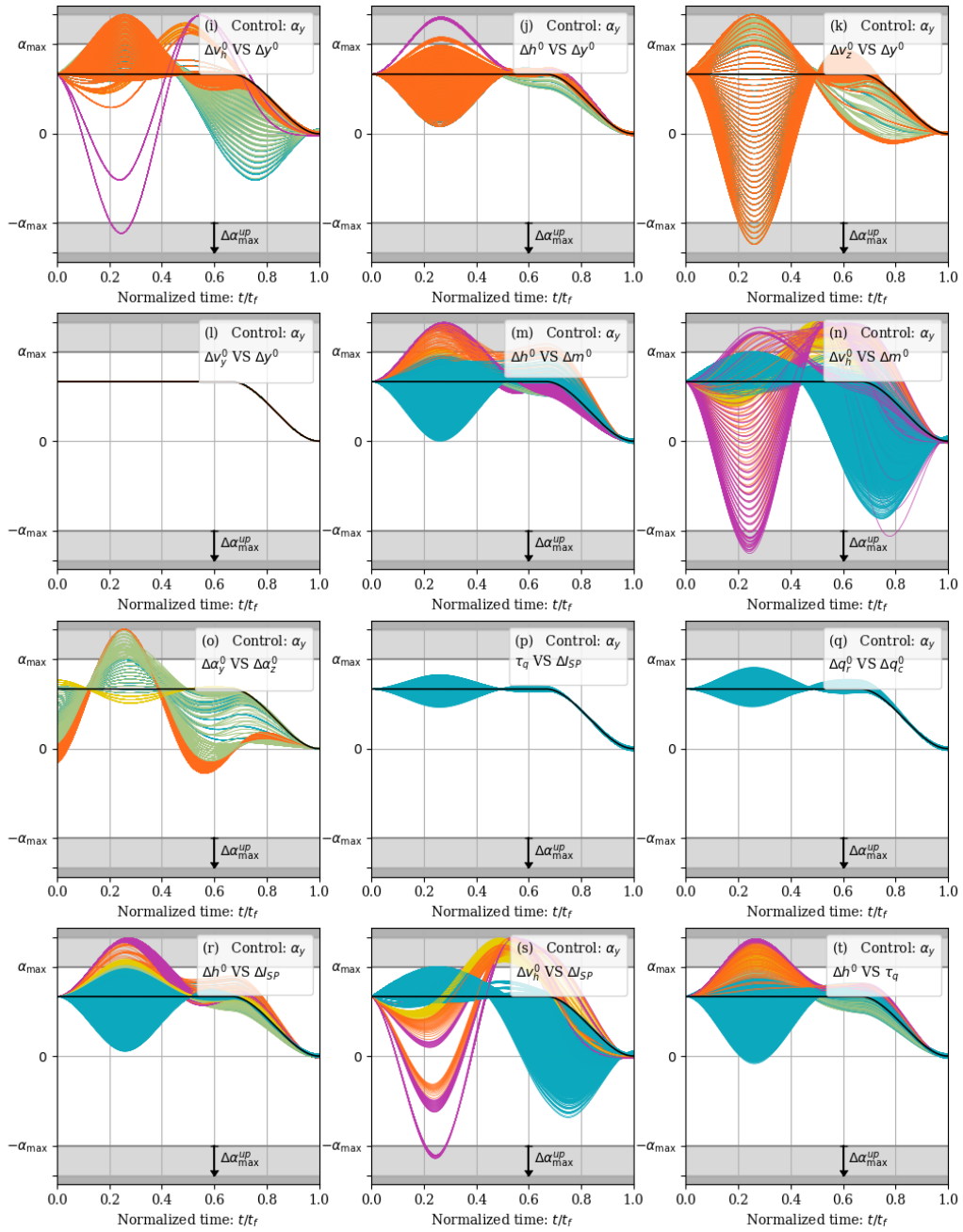


Figure 6.6: Projected incidence α_y (in-plane) w.r.t. normalized time. Batch B. It is normal that the pair (l), with purely out-of-plane variables, has nearly zero impact on α_y , since the latter conveys the in-plane projected incidence.

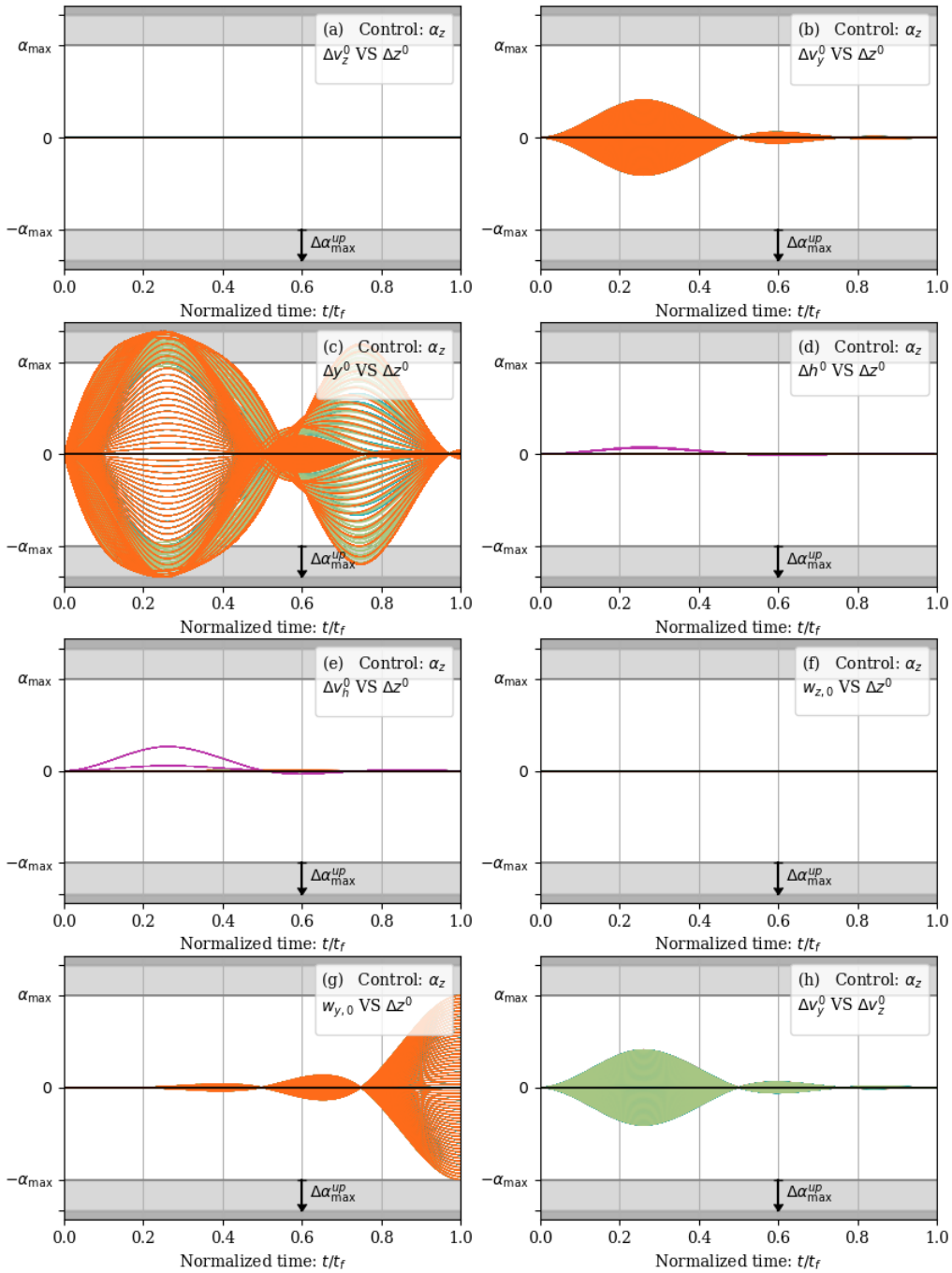


Figure 6.7: *Projected incidence α_z (out-of-plane) w.r.t. normalized time. Batch A.* Contrary to what the chart (e) suggests, there is no discontinuity between the various scenarios. The fact that one can see a “gap” in the curves is only a matter of mesh refinement: a thinner mesh would show that this gap is continuously filled with curves. However, this gap shows that on some input subsets, the algorithm may have stiff output variations, as discussed in Figure 6.15.

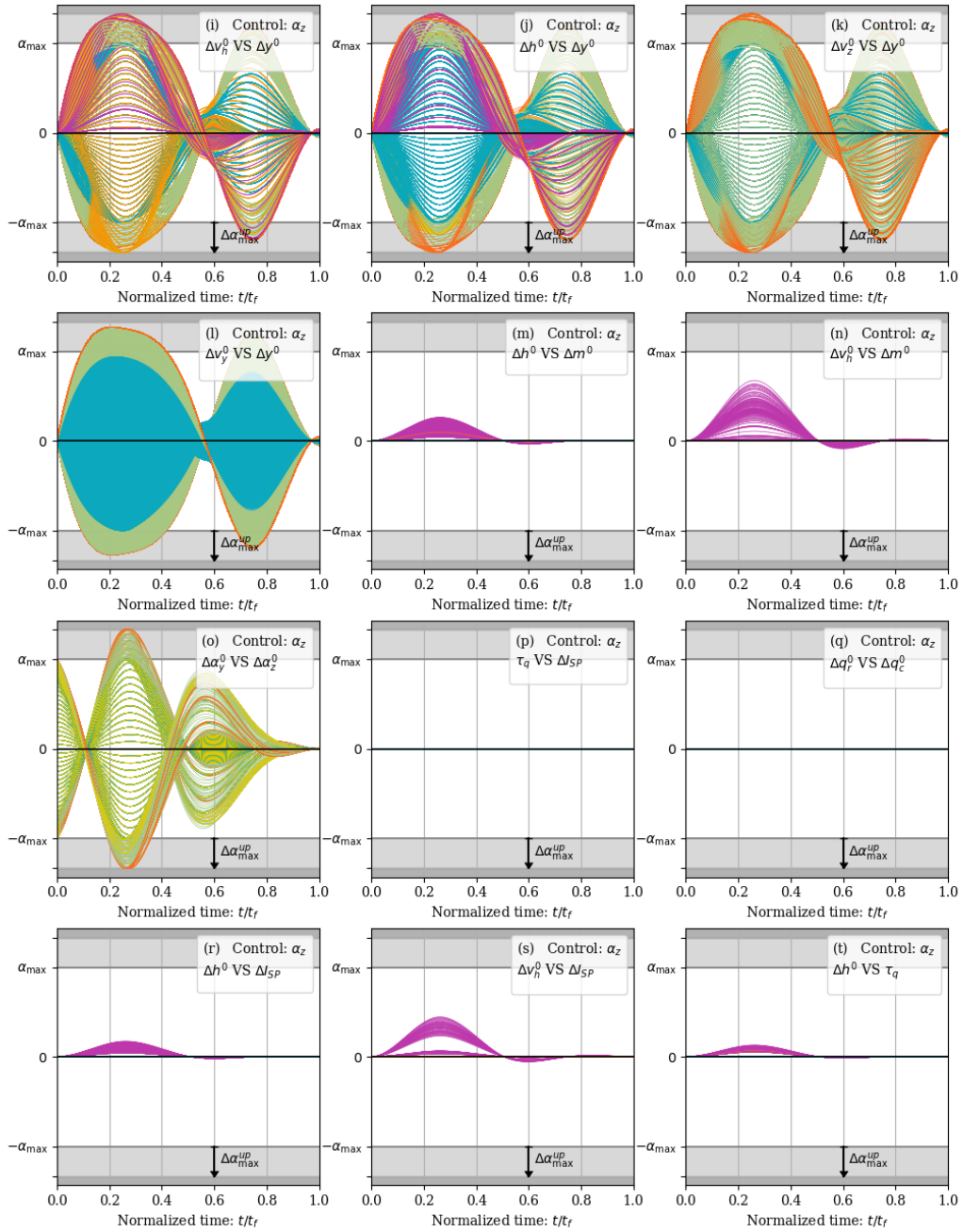


Figure 6.8: Projected incidence α_z (out-of-plane) w.r.t. normalized time. Batch B.

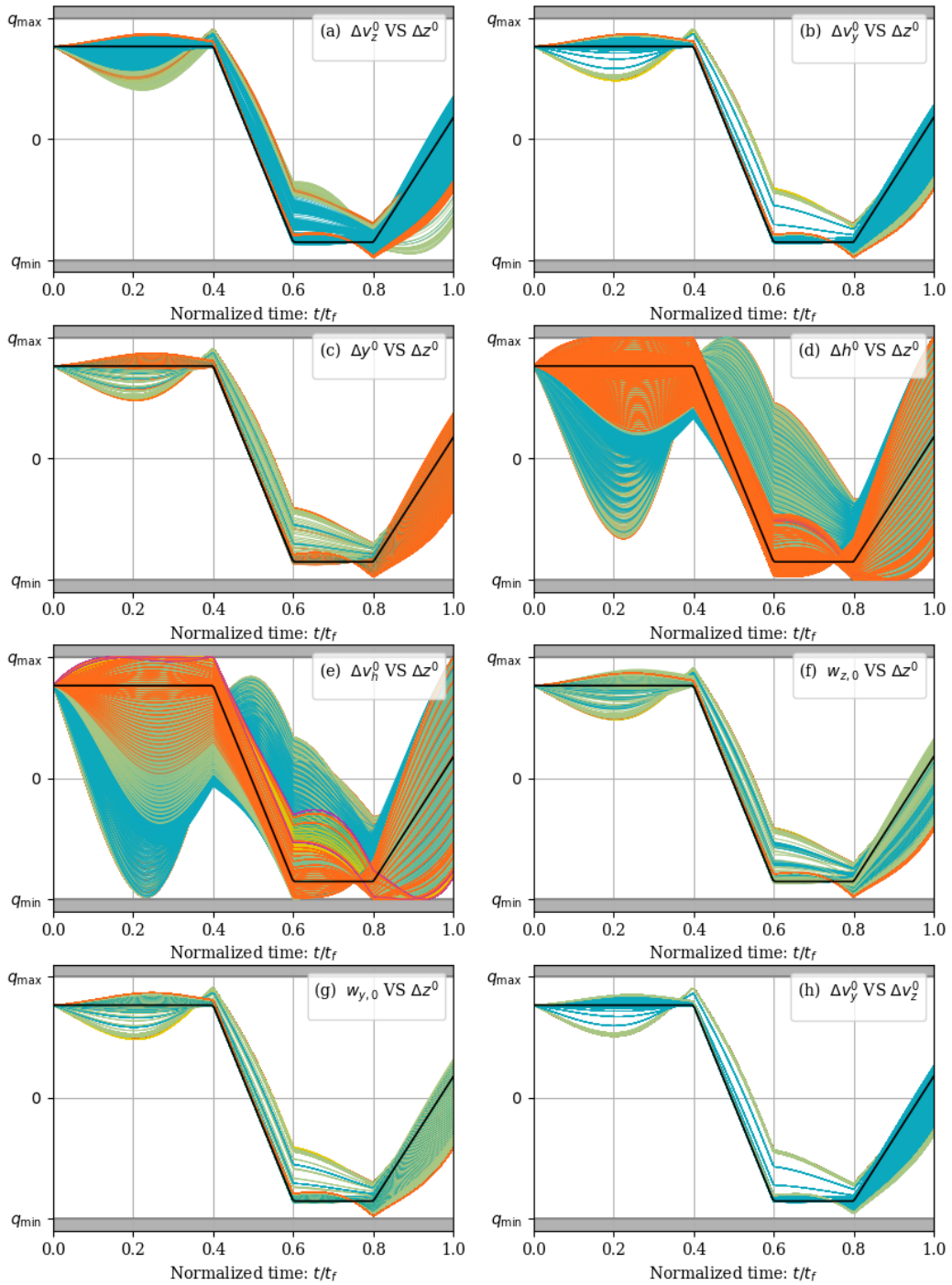


Figure 6.9: *Controlled engine flow q_c w.r.t. normalized time. Batch A.* The highly different influence of the horizontal variables and the vertical ones on the engine flow is clearly pictured by these charts. A change on the input pair (c) only implies little changes in the controlled engine flow. However, the pair (e) has a high impact on the engine flow, especially due to the presence of Δv_h^0 .

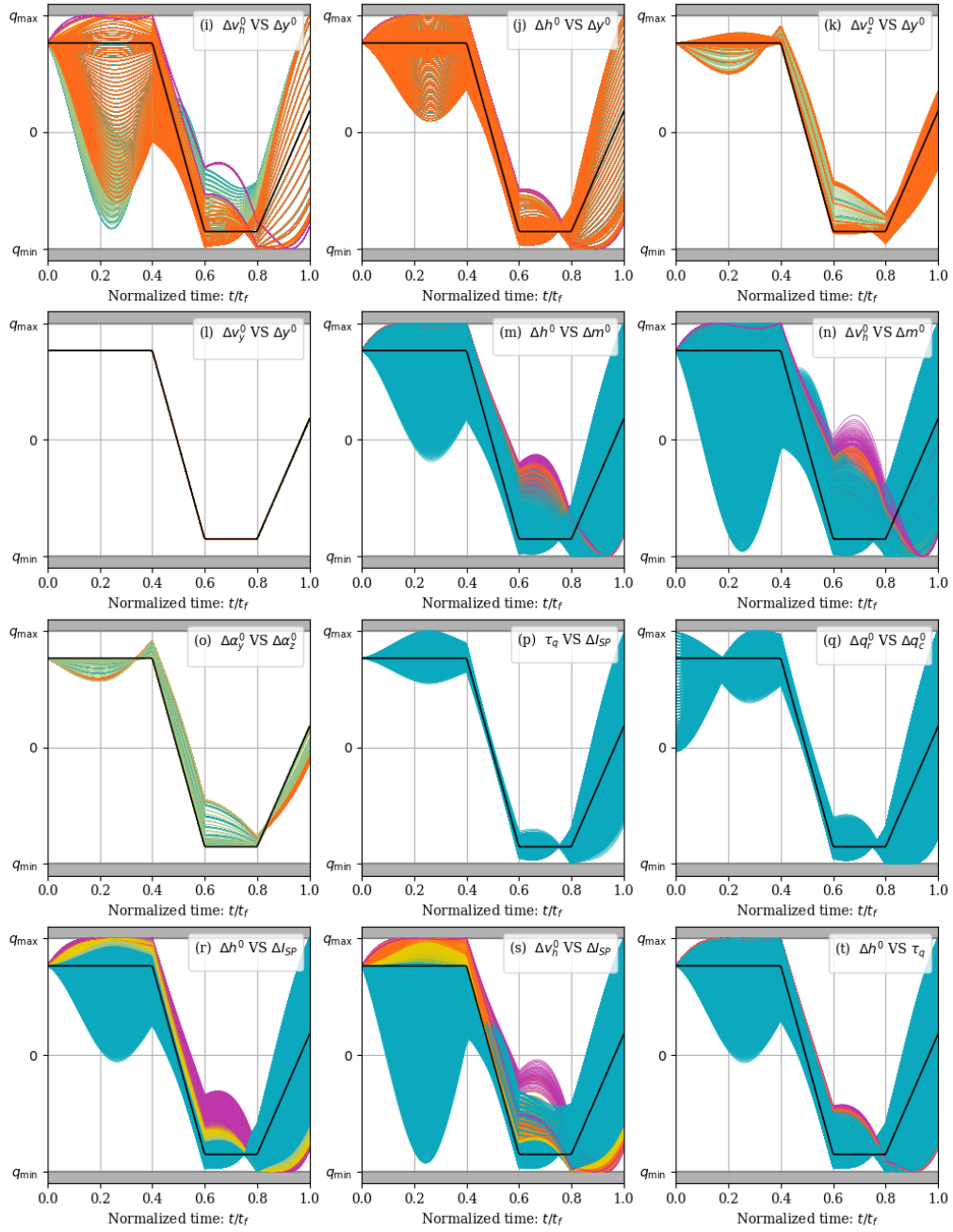


Figure 6.10: Controlled engine flow q_c w.r.t. normalized time. Batch B.

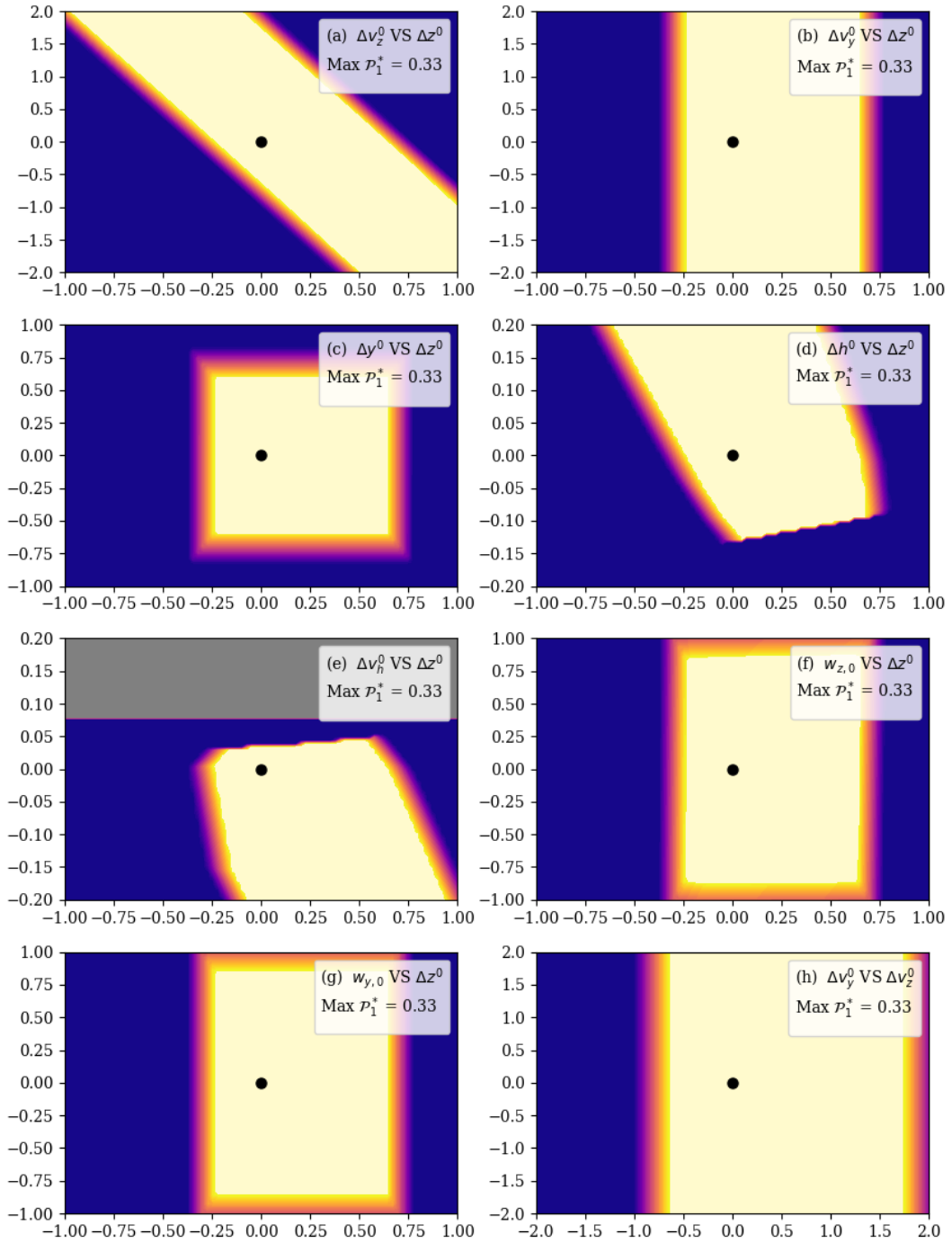


Figure 6.11: Heatmap of the first negotiation, i.e. $\mathcal{P}_1^* = |\Delta\alpha_{\max}|$. Batch A.

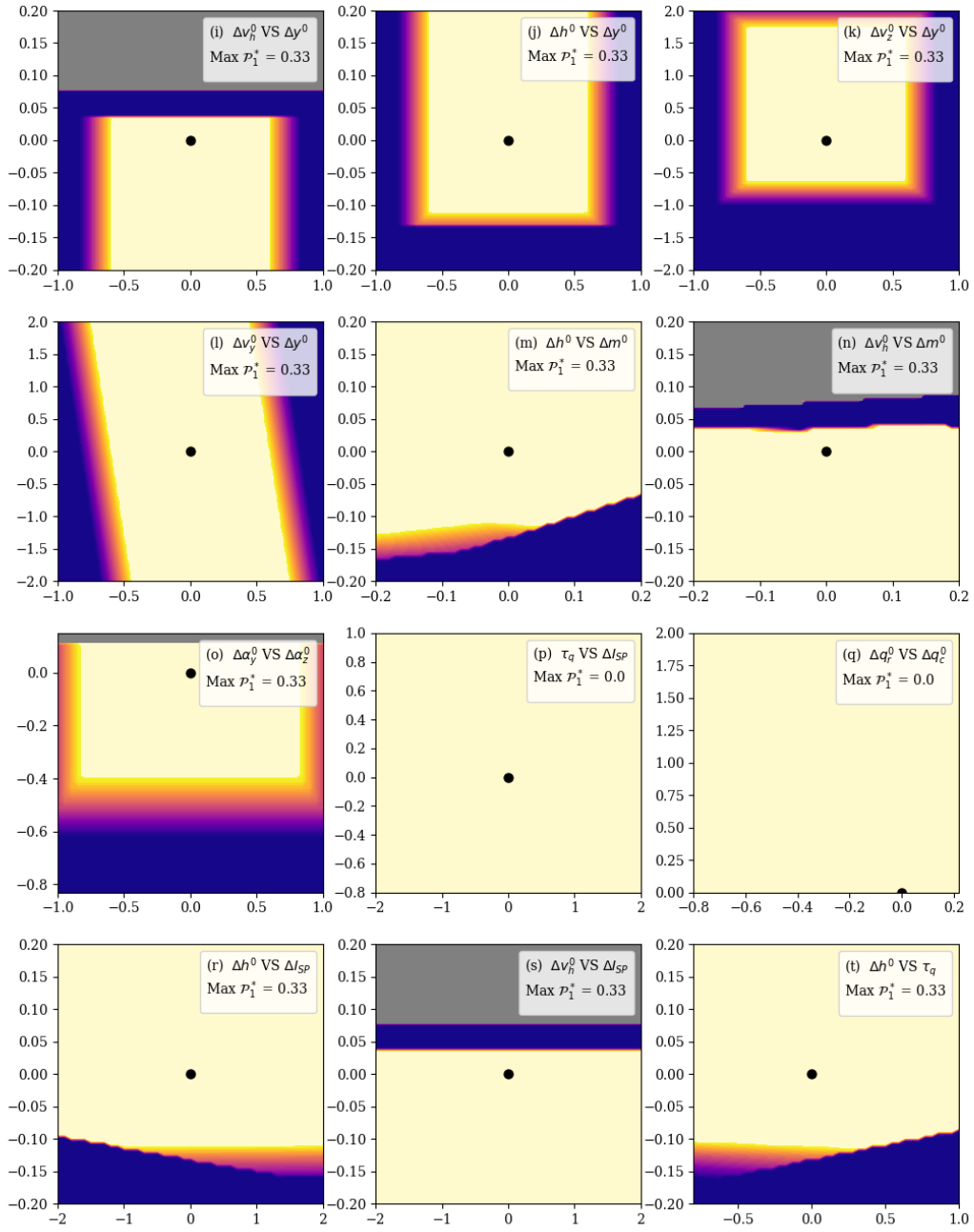


Figure 6.12: Heatmap of the first negotiation, i.e. $\mathcal{P}_1^* = |\Delta\alpha_{\max}|$. Batch B.

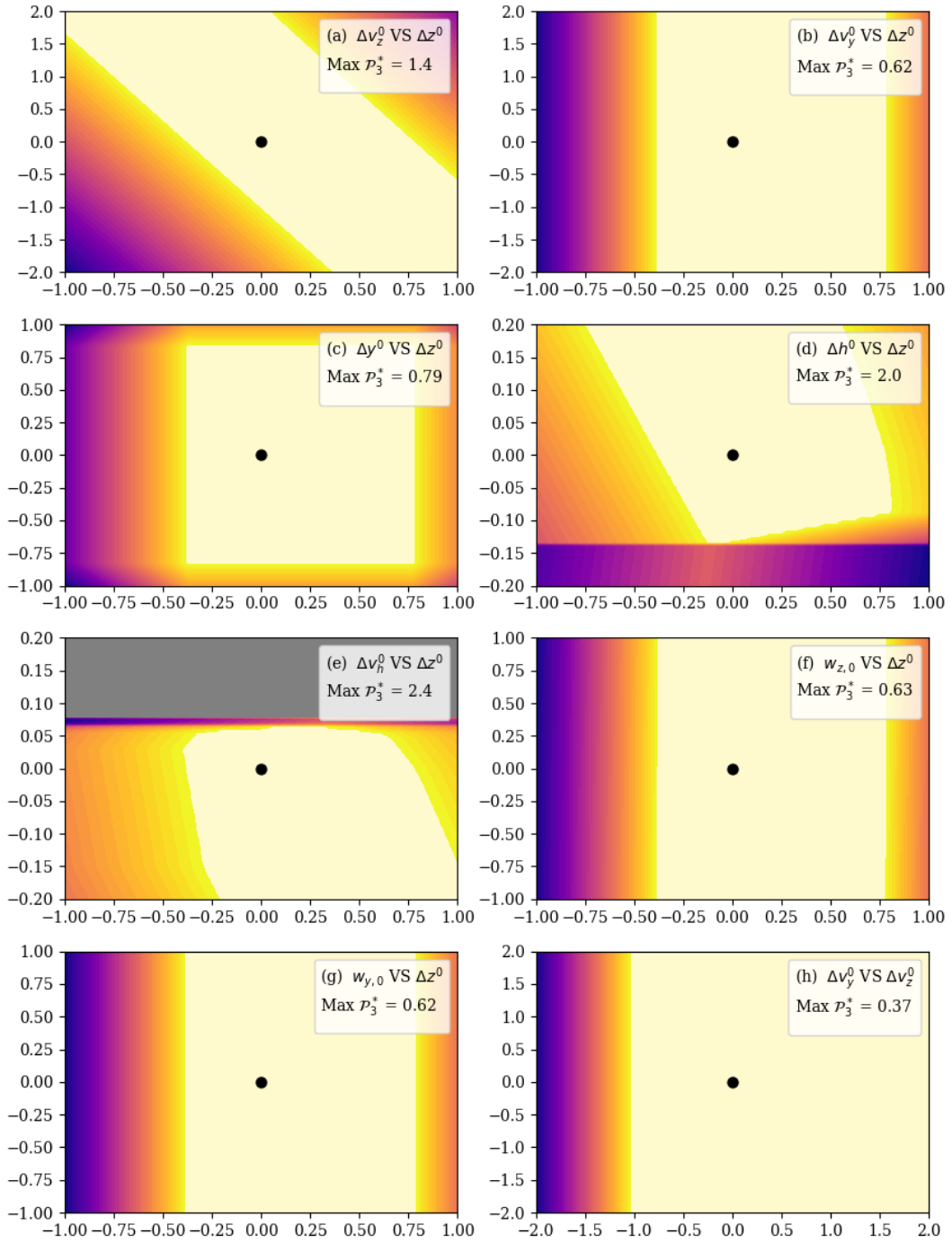


Figure 6.13: Heatmap of the third negotiation, i.e. $\mathcal{P}_3^* = \|(\Delta z^f, \Delta y^f)^\top\|_1$. Batch A. These negotiation maps have a structure deeply linked to the partition described in the emergency mode maps, as shown in Figure 6.3.

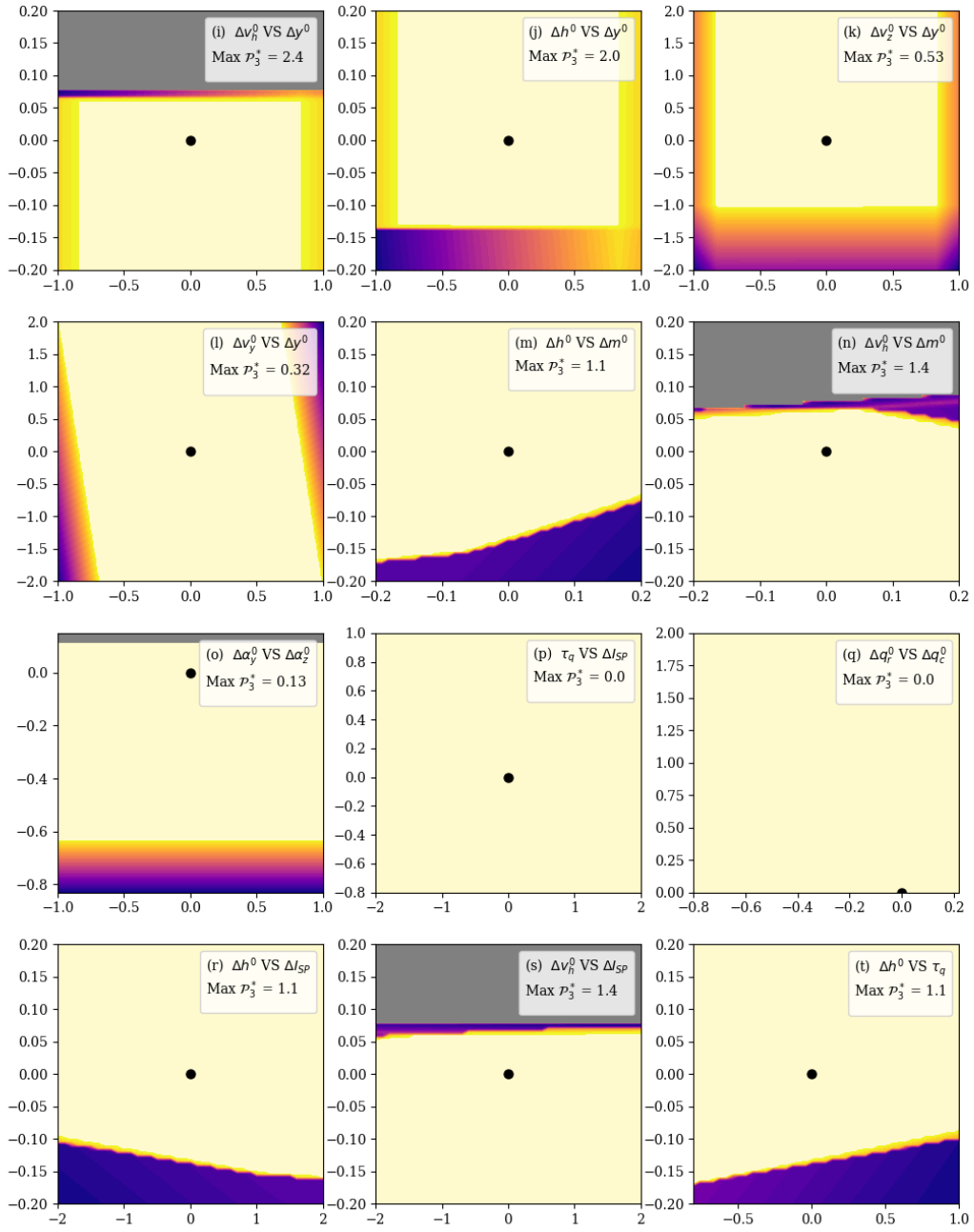


Figure 6.14: Heatmap of the third negotiation, i.e. $\mathcal{P}_3^* = \|(\Delta z^f, \Delta y^f)^\top\|_1$. Batch B.

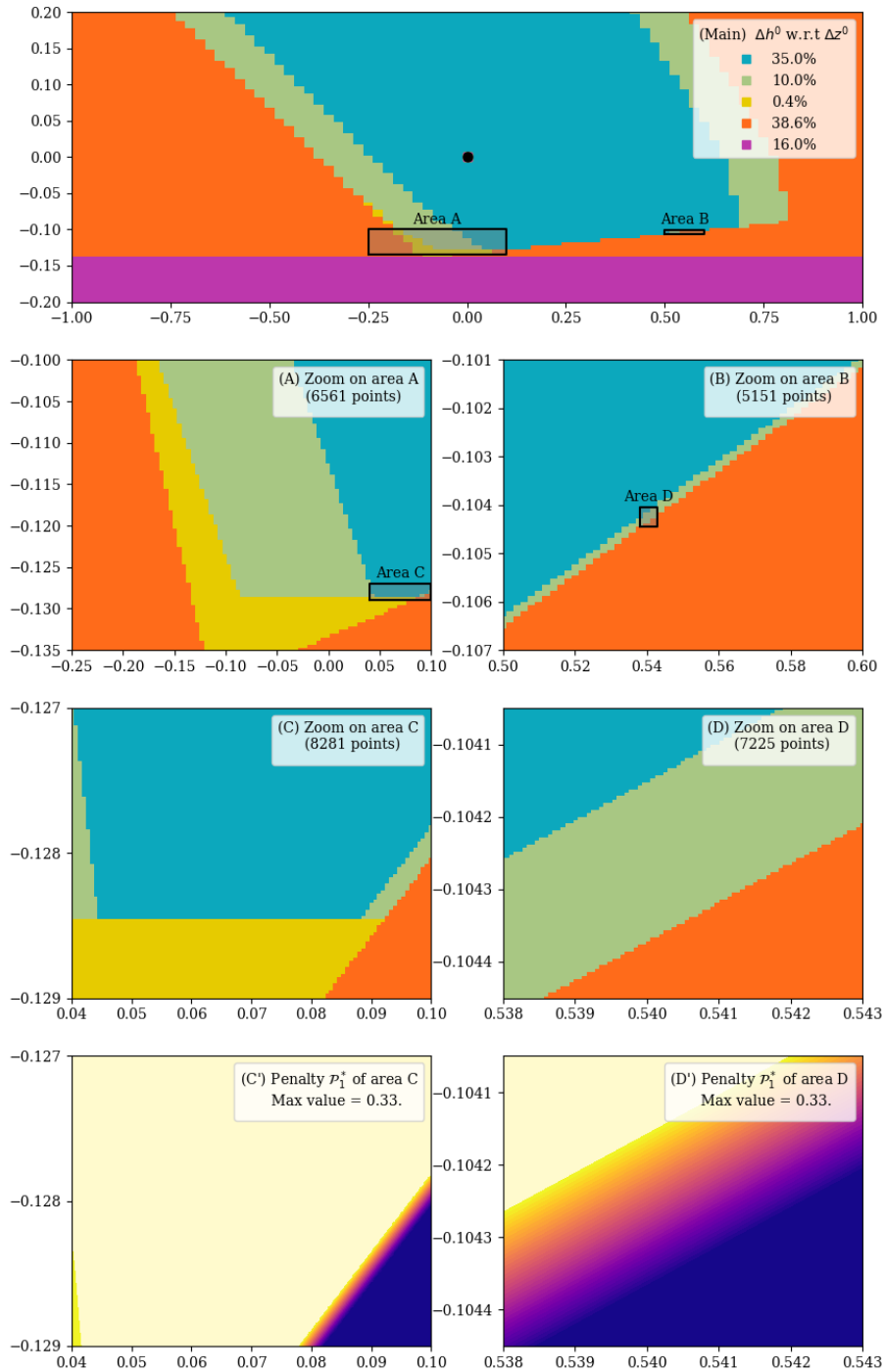
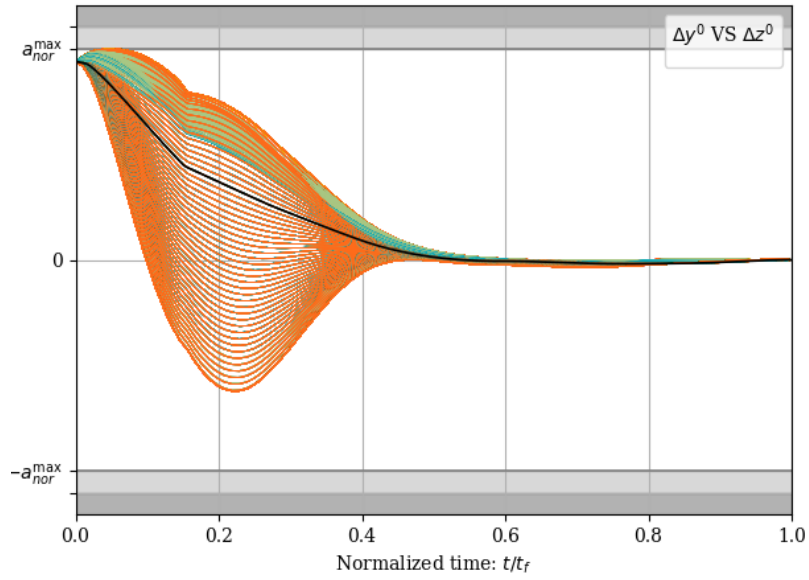
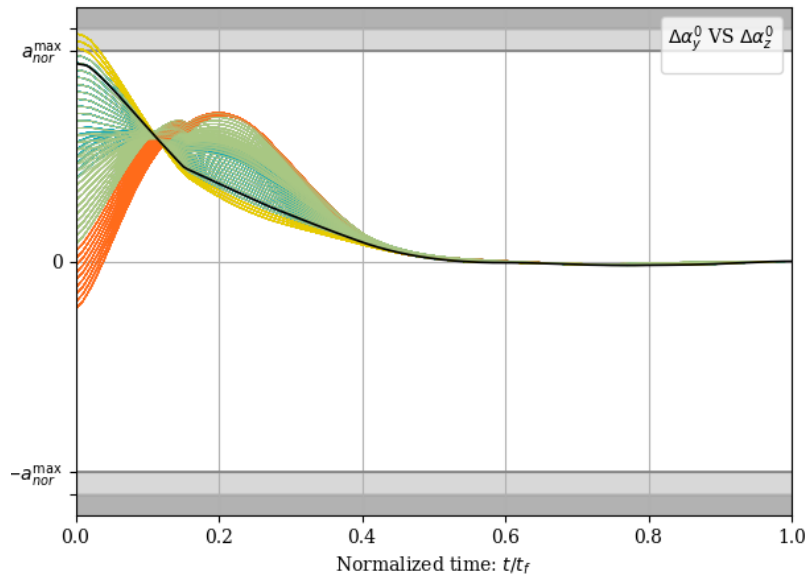


Figure 6.15: Several zooms on stiff parts of the pair $(\Delta z^0, \Delta h^0)$, illustrating that the negotiation maps are indeed Lipschitz-continuous in practice. All the x -axis convey Δz^0 , and all the y -axis convey Δh^0 .



(Top) Dispersion of the pair $(\Delta z^0, \Delta y^0)$, labeled (c).



(Bottom) Dispersion of the pair $(\Delta \alpha_z^0, \Delta \alpha_y^0)$, labeled (o).

Figure 6.16: Normal load a_{nor} on two different input pairs.

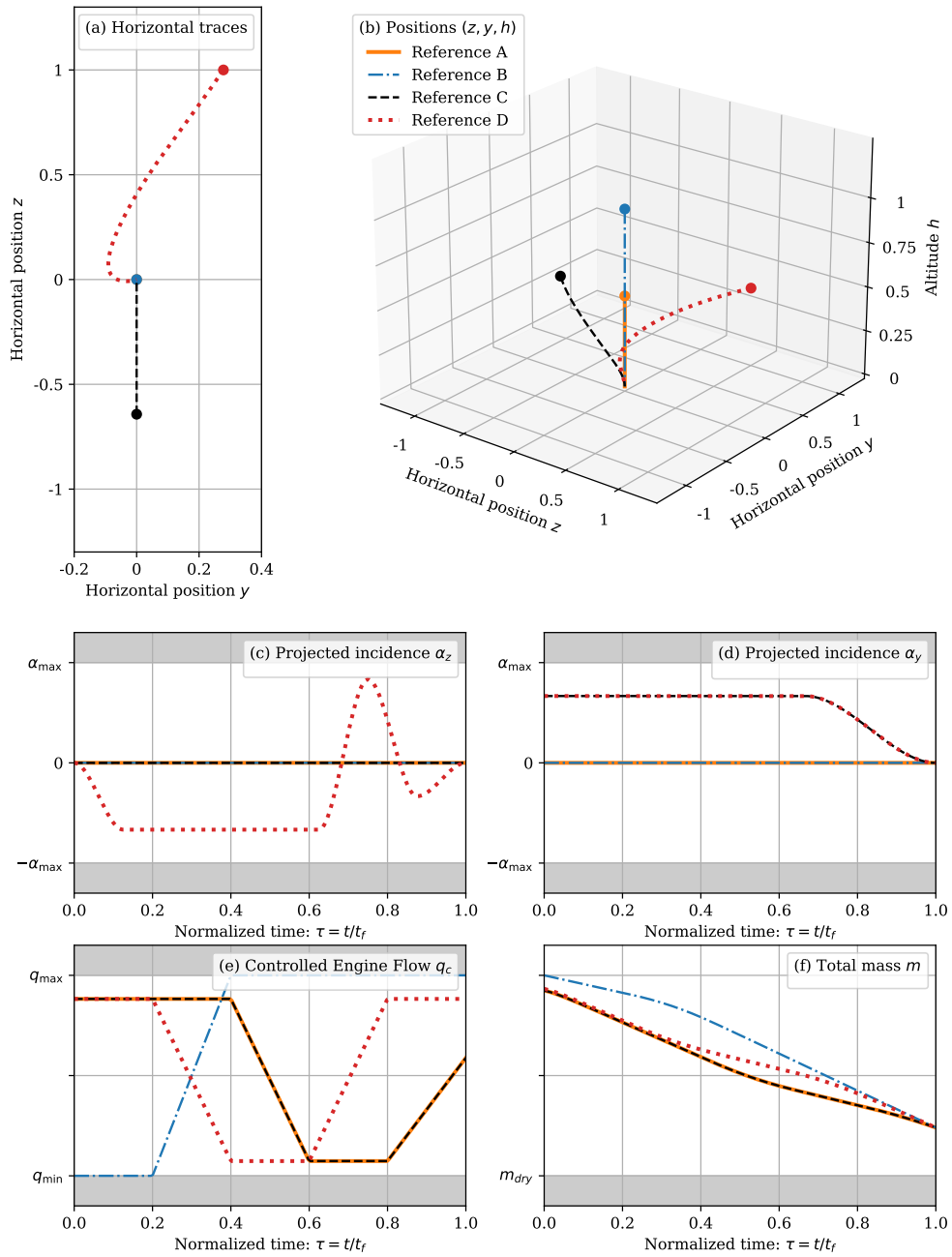


Figure 6.17: Reference trajectories selected for the benchmark against the vertical flight envelope, as listed in Table 6.5.

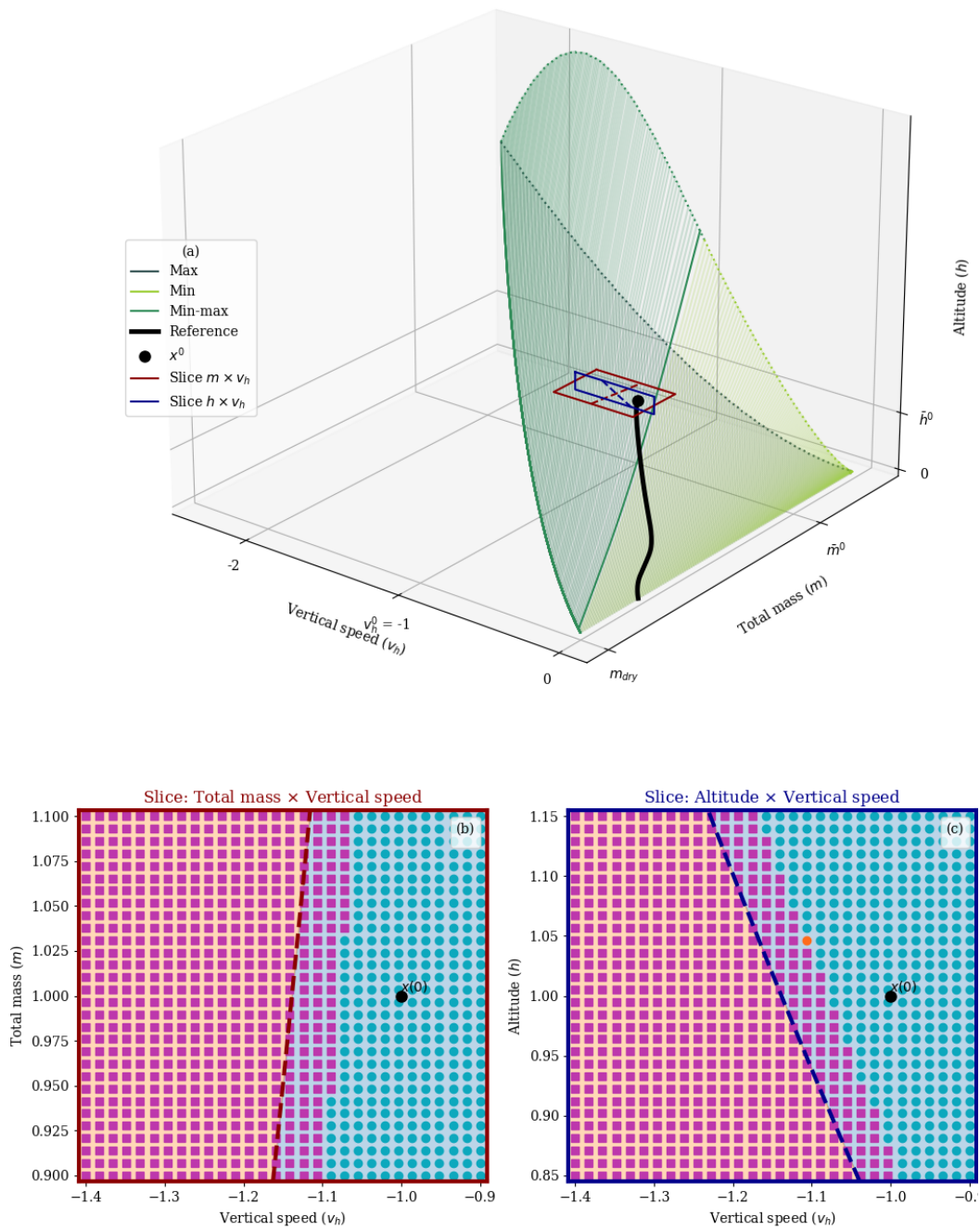


Figure 6.18: *Vertical envelope and associated slices (Reference A)*. As shown in the chart (b), the red slice shows what happens when dispersing the total mass m w.r.t. the vertical speed v_h . The light yellow area conveys the area which is out of the flight envelope. On the contrary, the blue area is the area inside it. The dashed line conveys the separation between these areas.

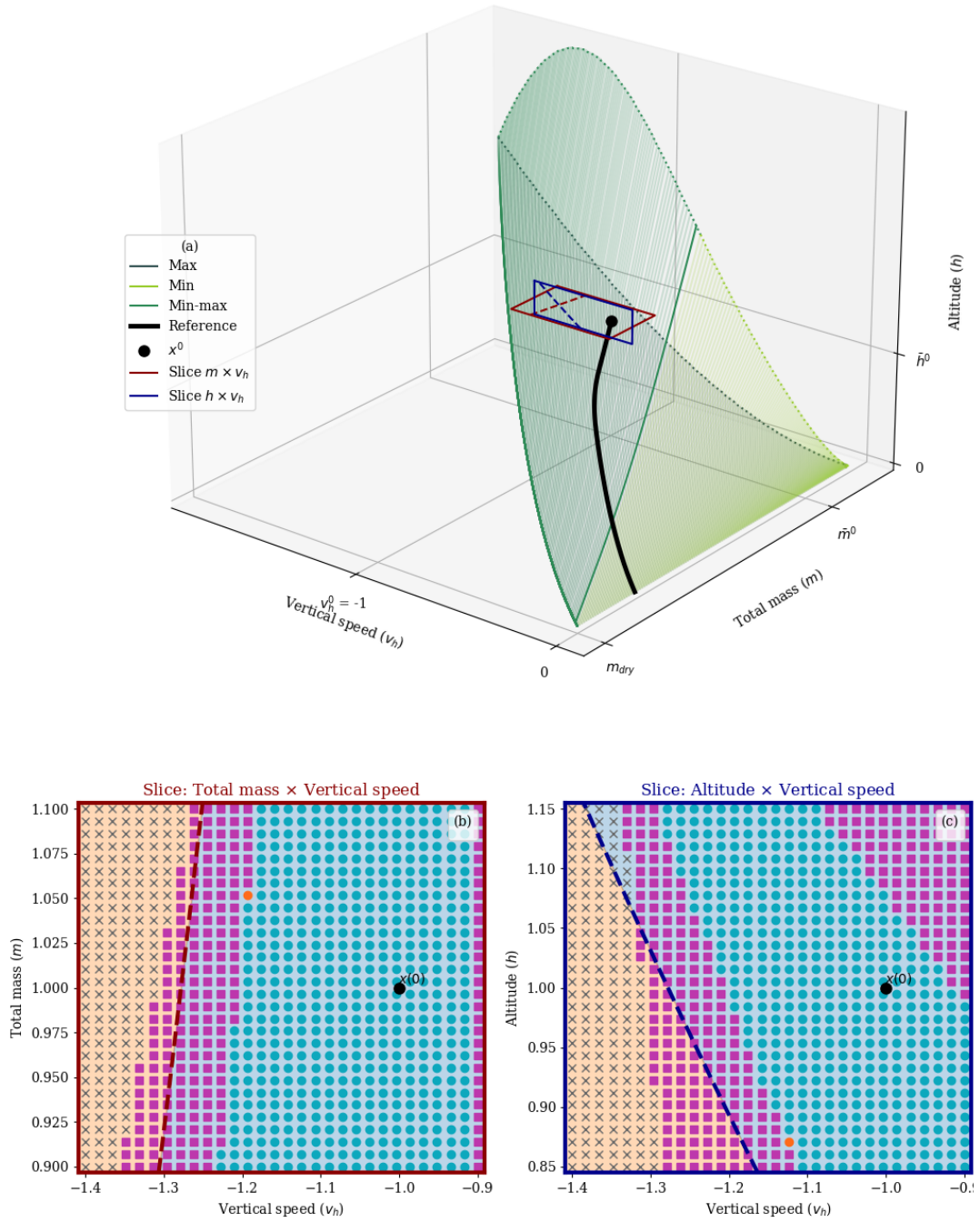


Figure 6.19: Vertical envelope and associated slices (Reference B).

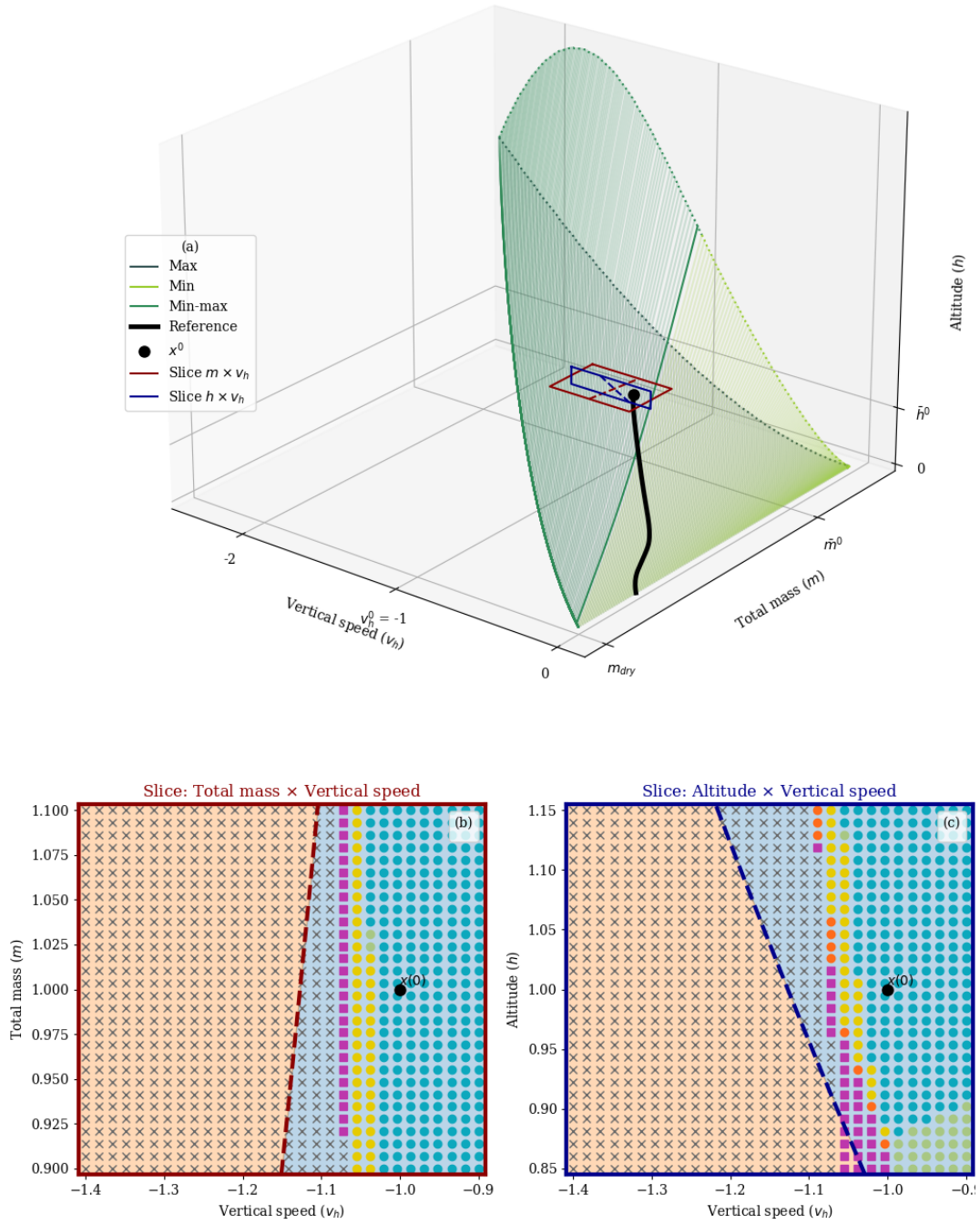


Figure 6.20: Vertical envelope and associated slices (Reference C).

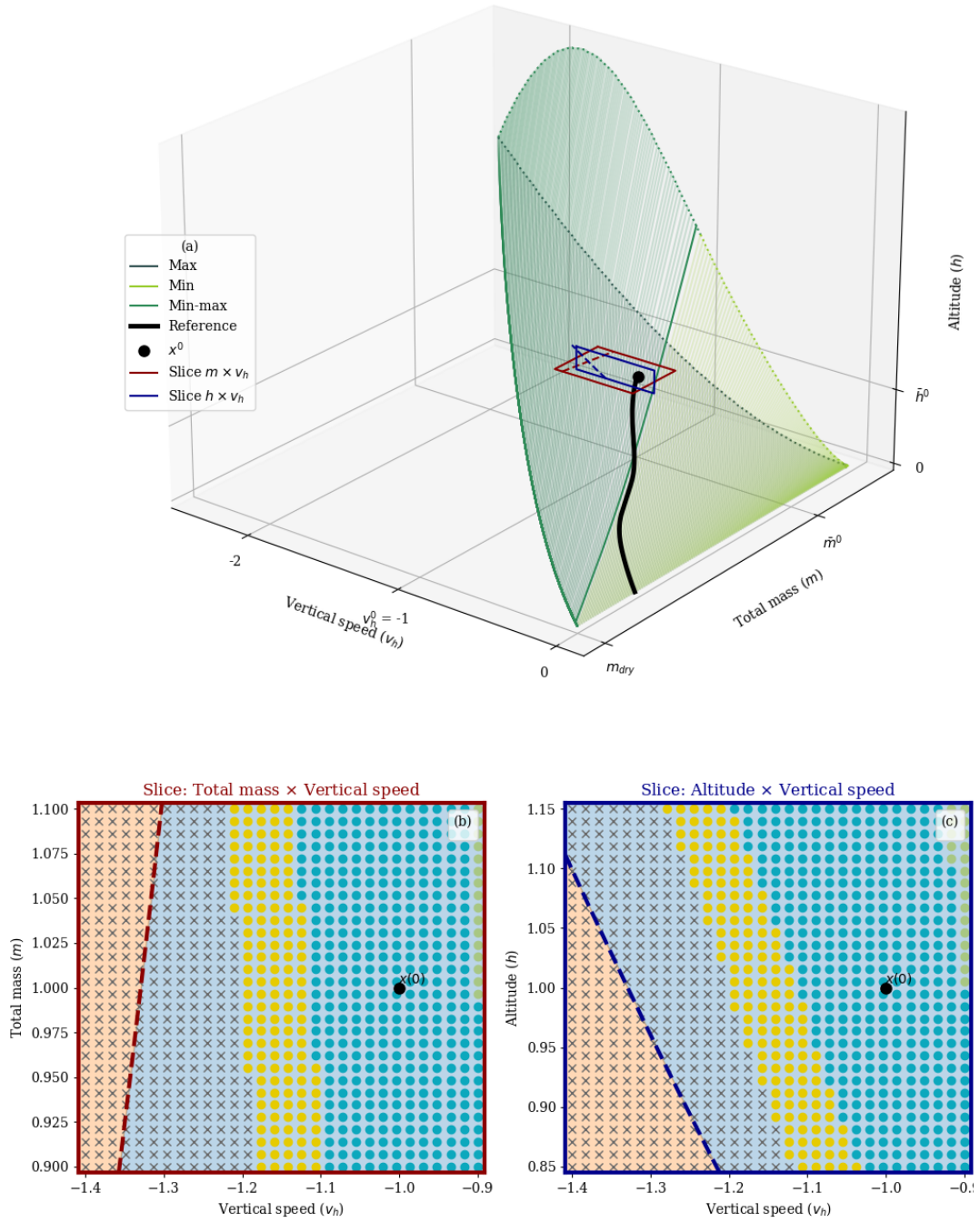


Figure 6.21: Vertical envelope and associated slices (Reference D).

Conclusion

Résumé

Ce chapitre propose un résumé succinct de ce manuscrit, quelques commentaires sur les sujets qui ont été volontairement omis, et des pistes de recherche pour d'éventuels travaux futurs.

The main focus of this thesis is on computing a landing trajectory for a reusable tossback vehicle in response to changes in the flight parameters. When this guidance problem is infeasible, one faces an emergency situation. In such cases it is acceptable to sacrifice some of the constraints originally formulated in the nominal (non emergency) situation.

A methodology (Algorithm HEGO) has been developed in the manuscript to solve this emergency guidance problem. It is applicable in the -relative vast- vicinity of a reference trajectory having a reasonably high slenderness ratio. It is capable of handling the sacrifice of constraints according to predefined extended colexicographic order. As illustration, results have been presented and have served to compute performance charts in Chapter 6. For actual missions, aerospace engineers would be interested in the quantitative version of Table 6.4.

A significant advantage of the proposed method, i.e. Algorithm HEGO, is that it is deterministic, and its implementation relies on mature technologies (LP and QP) for which off-the-shelf solvers are available. There are nearly no heuristic used to make the method work, apart from the tuning of the termination condition tolerances of the numerical solvers.

As announced in the introduction, several topics have been voluntarily considered out-of-scope in this thesis. Some of them represent possible future research directions, and are sketched here. The modeling choices of Chapter 4 could be tailored to other frameworks, to be able to handle singular arcs in the parametric control description for instance.

As far as the overall G&C system is concerned, the control part has been considered out-of-scope for this manuscript. The interplay between the control and the guidance algorithms has been swiftly discussed in [66]. Simulations gathering both parts of the G&C system would provide a more complete performance assessment.

Negotiating vertical components (Δv_h^f or Δh^f) has proved to be ill-posed, as shown in Chapter 6. To obtain robustness regarding the vertical motion, it is needed to go back to the mission design itself, and eventually consider other landing strategies than the classic tossback trajectory structure, pictured in Figure 1.2. More

generally, the choice of the reference trajectory has a strong impact on the final performances.

The sensitivity-based PDG method presented in Chapter 4 has a sufficient accuracy for our application (i.e. final burn guidance of a tossback vehicle with high slenderness ratio) but depends significantly on the chosen reference trajectory and its generalization to more complex maneuvers is limited. Its application to large diverts of high agility vehicles would require a dedicated study with extensive numerical benchmarks. However, it may be of interest to apply this sensitivity-based guidance method to multi-phase problems. Indeed, combining the re-entry glide and the final burn phases in a single guidance problem would be a relevant problem, for which the sensitivity based approach would have a great potential.

Our implementation of HEGO conveniently builds upon the linear description of the constraints. However, following the generalization discussed in Section 5.6, it would be interesting to develop a version of HEGO with other underlying guidance methods, such as successive convexification or pseudo-spectral methods, which could be applied to large-divert problems.

Finally, the emergency problem that we formulated is only one way to describe the problem of landing guidance relaxation. Our modeling is not sufficient to handle disjunctive scenarios, where one would need to choose between two separate landing sites for instance. This represents an important research direction, that could be worth exploring.

Appendix A

Technical tools

Résumé

Cette annexe présente des outils mathématiques nécessaires à la compréhension du manuscrit. La première partie se concentre sur des résultats classiques d'optimisation convexe, et la deuxième rappelle des résultats fort utiles sur le comportement des équations différentielles ordinaires.

A.1 Optimization results

A.1.1 Duality gap

The results presented here are taken from D. P. Bertsekas' book [13], which contains all the proofs of the theorems and lemmas presented below.

Let f , g and h be functions defined over \mathbb{R}^n , where f has scalar values, g has m components, and h has r components. Let us define the primal optimization problem

$$f^* \leftarrow \min_{x \in \mathbb{R}^n} f(x) \quad (\text{A.1a})$$

$$\text{s.t. } g(x) \leq 0, \quad (\text{A.1b})$$

$$h(x) = 0 \quad (\text{A.1c})$$

where f^* denotes its optimal value.

An optimization problem is said to be *feasible* if there exists at least one element satisfying its constraints. An optimization problem is said to be *finite* if its optimal value exists and does not equal $\pm\infty$.

The *Lagrangian* of Problem (A.1) is defined by

$$L(x, \mu, \lambda) := f(x) + \mu^\top g(x) + \lambda^\top h(x)$$

where $\mu \in \mathbb{R}^m$ and $\lambda \in \mathbb{R}^r$.

Definition 7. Vectors μ^* and λ^* are said to be **Lagrange multipliers** for the primal Problem (A.1) if $\mu^* \geq 0$ and $f^* = \inf_{x \in \mathbb{R}^n} L(x, \mu^*, \lambda^*)$.

The dual of Problem (A.1) is the following problem

$$\sup_{\mu \in (\mathbb{R}^+)^m, \lambda \in \mathbb{R}^r} \inf_{x \in \mathbb{R}^n} L(x, \mu, \lambda). \quad (\text{A.2})$$

Let us denote by q^* the optimal value of Problem (A.2). Here, $q^* \in \mathbb{R} \cup \{\pm\infty\}$.

Theorem 10 (Weak Duality Theorem [13, Prop. 5.1.3]). *Assume that the primal Problem (A.1) is feasible, but not necessarily finite. Then: $q^* \leq f^*$.*

The *duality gap* is defined as the following non-negative quantity

$$\Delta_{\text{gap}} := f^* - q^*. \quad (\text{A.3})$$

We say that there is no duality gap if $\Delta_{\text{gap}} = 0$, and that there is a duality gap if $\Delta_{\text{gap}} > 0$.

Let us introduce the matrices and vectors E, d, A, b of proper dimensions, assume that f is convex, and define the problem

$$\min_{x \in \mathbb{R}^n} f(x) \quad (\text{A.4a})$$

$$\text{s.t. } Ex = d, \quad (\text{A.4b})$$

$$Ax \leq b. \quad (\text{A.4c})$$

Theorem 11 (Strong Duality Theorem - Linear constraints [13, Prop. 5.2.1]). *Assume that the primal Problem (A.4) is feasible and its optimal value f^* is finite. Let also f be convex over \mathbb{R}^n . Then, there is no duality gap and there exists at least one Lagrange multiplier.*

Lemma 12 (Existence of Primal Optimal Solutions of LPs [13, Lemma 5.2.1]). *Assume that Problem (A.4) is feasible and its optimal value f^* is finite. Let also f be linear. Then, Problem (A.4) has at least one optimal solution.*

A.1.2 Right-hand side sensitivity of Linear Programs

Consider the following primal LP in its standard form and its optimal value function v (viewed as a function of its right-hand side b) s.t.

$$v(b) := \min_x c^\top x \quad (\text{A.5a})$$

$$\text{s.t. } Ax = b, \quad (\text{A.5b})$$

$$x \geq 0. \quad (\text{A.5c})$$

Proposition 15. *If $v(b)$ has a finite value, then the dual of (A.5) is*

$$\max_{\mu} \mu^\top b \quad (\text{A.6a})$$

$$\text{s.t. } A^\top \mu \leq c \quad (\text{A.6b})$$

By convention, if $A^\top \mu \leq c$ is infeasible, then the optimal value of the latter problem is $-\infty$.

The proof of this property is standard material that can be found in most optimization text-books (see e.g. [21, Ch. 5]). It is provided here for completeness and for its tutorial aspect. Also, beware of the fact that the role of μ and λ is inverted in the previous proposition and its proof compared to Equation (A.2).

Proof. Introduce the Lagrangian

$$L(x, \mu, \lambda) := c^\top x + \mu^\top (b - Ax) - \lambda^\top x.$$

Let us form the function g s.t.

$$g(\mu, \lambda) := \inf_x L(x, \mu, \lambda)$$

Since $g(\mu, \lambda) = \mu^\top b + \inf_x (c - A^\top \mu - \lambda)^\top x$, then

$$g(\mu, \lambda) = \begin{cases} \mu^\top b & \text{if } c - A^\top \mu - \lambda = 0 \\ -\infty & \text{otherwise.} \end{cases}$$

Therefore, the dual of (A.5) is

$$\begin{aligned} & \sup_{\mu, \lambda} g(\mu, \lambda), \\ & \text{s.t. } \lambda \geq 0. \end{aligned}$$

Using the Strong Duality Theorem (Theorem 11 above) with the fact that $v(b)$ is finite and $x \mapsto c^\top x$ is convex, makes the sup of the latter problem a max. Finally, after simplification of $\lambda = c - A^\top \mu \geq 0$, ones gets (A.6).

Note that if the condition $c - A^\top \mu \geq 0$ is not feasible - or equivalently that $A^\top \mu \leq c$ is not feasible - then g equals $-\infty$ for any value of μ and λ , and therefore the optimal value of the dual is $-\infty$. \square

A direct consequence of Proposition 15 and the Strong Duality Theorem presented above is that

$$v(b) = \min\{c^\top x \mid Ax = b, x \geq 0\} = \max\{\mu^\top b \mid A^\top \mu \leq c\}.$$

whenever the primal problem is feasible and finite. By convention, when the primal problem is infeasible, v takes the optimal value of the dual problem (possibly $\pm\infty$).

Theorem 12 (From [41, Thm. 1]). *Under the assumption of Proposition 15, for any direction d and for any scalar $t > 0$ sufficiently small, we always have*

$$v(b + td) = v(b) + t \sup\{\lambda^\top d \mid A^\top \lambda \leq c, \lambda^\top b = v(b)\}. \quad (\text{A.7})$$

Remark 33. *Theorem 12 must be understood as follows:*

- *If it is known that $t \mapsto v(b + td)$ exists on some interval $[0, t']$ for $t' > 0$, then the “sup” term in Equation (A.7) is a “max”, and Equation (A.7) holds (at least) on a non-trivial sub-interval of $[0, t']$.*

- On the contrary, if the “sup” term equals $+\infty$, then the primal problem with right-hand side $b + td$ is infeasible for any $t > 0$.

Equation (A.7), that we also refer to as *Gawwin’s formula* in this manuscript, is a powerful tool to analyze the RHS sensitivity of LPs. It is important to remark that the maximization problem in (A.7) is deeply linked to the dual LP from (A.6), s.t.

$$\begin{aligned} \sup_{\lambda} \lambda^\top d &= \sup_{\lambda} \lambda^\top d \\ \text{s.t. } A^\top \lambda \leq c &\quad \text{s.t. } \lambda \in \text{Maximizers of (A.6).} \\ \lambda^\top b = v(b) & \end{aligned}$$

Proposition 16 (Adapted from [5, Prop. 2.3]). *There exists a closed interval $[\alpha, \beta]$ (possibly empty) s.t.*

- (i) $v(b + td) = \infty$ for all $t \in [\alpha, \beta]$,
- (ii) Either $v(b + td) = -\infty$ for all $t \in [\alpha, \beta]$ or $v(b + td) \in \mathbb{R}$ for any $t \in [\alpha, \beta]$ and in this case $t \mapsto v(b + td)$ is a continuous convex piecewise affine function.

The latter proposition is stated in [5, Prop. 2.3], and its detailed proof can be found in [70, Sec. 8.11]. It is important to clarify that the number of sub-intervals on which $t \mapsto v(b + td)$ is affine is finite, as stated in [70, Sec. 8.9, Item 4].

A.2 Differential Equations

A.2.1 Comparison theorem

For $n \geq 1$, a function $F : I \times \mathcal{X} \subset \mathbb{R} \times \mathbb{R}^n \rightarrow \mathbb{R}^n$ is said to be *quasi-monotone increasing* if, for every pair (t, x) and (t, v) in $I \times \mathcal{X}$ and every $i = 1, \dots, n$, one gets $F_i(t, x) \geq F_i(t, v)$ whenever $x_i = v_i$ and $x \geq v$.

Lemma 13 (Adapted from [80, IX.2.6]). *Let F be a continuously-differentiable, quasi-monotone increasing function and $x : [t_0, \tau] \rightarrow \mathbb{R}^n$ the maximal solution of $\dot{x}(t) = F(t, x(t))$ through some point $(t_0, x^0) \in I \times \mathcal{X}$. Assume $v : [t_0, \tau'] \rightarrow \mathbb{R}^n$, $\tau' \leq \tau$ is a differentiable function s.t. $(t, v(t)) \in I \times \mathcal{X}$ and*

- (i) $v(t_0) \leq x^0$,
- (ii) $\dot{v}(t) \leq F(t, v(t)), \quad \forall t \in (t_0, \tau')$.

Then, $v(t) \leq x(t)$ for any t in $[t_0, \tau']$. If \leq is replaced by \geq in (i) and (ii), then $v(t) \geq x(t)$ for any t in $[t_0, \tau']$.

A.2.2 Flow of Ordinary Differential Equations

The sensitivity computations described below are standard material in the literature. The results recalled below focus mainly on the computational aspects of the different derivatives of the flow. For further references, complementary approaches can be found in [18, Sec 3.2] and [75, Ch. 2] for abstract presentations, and in [49, Sec.

2.4], [90, Eq. 4.13], [31, XI.1.3, XI.1.4], [24, Eq. 2.3.18] and [73] for more applied methods. Useful material about the flow properties can also be found in [84, Sec. 4.5] (properties of the flow linked to Lie brackets).

Let us consider a dynamic function $f : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^{n_\eta} \rightarrow \mathbb{R}^n$ with state x , control u and parameter η , which defines the ODE

$$\dot{x} = f(x, u, \eta).$$

Definition 8. *Let us consider $x^0 \in \mathbb{R}^n$, $u : [0, 1] \rightarrow \mathbb{R}^m$ and $\eta \in \mathbb{R}^{n_\eta}$. For $t \geq 0$, the flow Φ_f of f is defined s.t. $\Phi_f(t, x^0, \eta; u) = x(t)$ where x is defined over $[0, 1]$ by the following IVP*

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t), \eta), \quad \forall t \in [0, 1], \\ x(0) &= x^0. \end{aligned}$$

The non-relevant inputs of Φ_f may be omitted to alleviate the writing. For instance, if f depends only on its state, its flow will be denoted $\Phi_f(t, x^0)$.

The propositions below present how to compute the derivatives of Φ_f w.r.t. each of its variables. Basically, these derivatives are the results of Initial Value Problems (IVPs) involving the derivatives of f . Note, first, that the time derivative of Φ_f is a direct consequence of its definition and, for any $t \in [0, 1]$, it equals

$$\frac{\partial \Phi_f}{\partial t}(t, x^0, \eta; u) = f(\Phi_f(t, x^0, \eta; u), u(t), \eta).$$

A.2.2.1 Sensitivity w.r.t. the initial condition

The derivative of Φ_f w.r.t. the initial condition is referred to as the *state transition matrix* (STM), or simply the *transition matrix* [24].

Proposition 17. *The derivative of Φ_f w.r.t. the initial condition x^0 is $\frac{\partial \Phi_f}{\partial x^0}(T, x^0) = M(T)$ where the transition matrix M satisfies the matrix-IVP*

$$\dot{M}(t) = \frac{\partial f}{\partial x}(\Phi_f(t, x^0)) M(t), \quad \forall t \in [0, T], \quad (\text{A.8})$$

$$M(0) = \mathbb{I}_n. \quad (\text{A.9})$$

When the context is clear enough, the following notation is sometimes used (e.g. in Chapter 3, or in [24]):

$$\frac{\partial x(T)}{\partial x(0)} = \frac{\partial \Phi_f}{\partial x^0}(T, x(0)).$$

Proposition 18 (Adapted from [90, Lemma 4.2.2]). *Pick any $k = 1, 2, \dots, \infty$. If f is of class C^k , then $(t, x^0) \mapsto \Phi_f(t, x^0)$ is of class C^k .*

A.2.2.2 Sensitivity w.r.t. to a parameter

Proposition 19. *Let $x^0 \in \mathbb{R}^n$ be an initial condition. Define $\mathcal{Y}(\eta, t) := \Phi_f(t, x^0, \eta)$. For any $t \in [0, 1]$, the derivative of \mathcal{Y} w.r.t. η is*

$$\frac{\partial \mathcal{Y}}{\partial \eta}(\eta, t) = y(t)$$

where y satisfies the following matrix-IVP

$$\begin{aligned} \dot{y}(t) &= \frac{\partial f}{\partial x}(x(t), \eta) y(t) + \frac{\partial f}{\partial \eta}(x(t), \eta), \\ y(0) &= \mathbb{O}_{n \times n_\eta}. \end{aligned}$$

where $x(t) = \Phi_f(t, x^0, \eta)$.

The IVP of Proposition 19 can be derived from Proposition 17, by considering an extended state composed of the original state, and one state having null dynamics for each component of the parameter η . For further details on this, see e.g. [31]. Using this remark, the smoothness of the flow w.r.t. its parameters can be derived from Proposition 18, as stated in the corollary below.

Corollary 4. *Pick any $k = 1, 2, \dots, \infty$ and $x^0 \in \mathbb{R}^n$. If f is of class C^k , then $(t, \eta) \mapsto \Phi_f(t, x^0, \eta)$ is of class C^k .*

A.2.2.3 Sensitivity w.r.t. the control

Proposition 20 (Adapted from [18, Sec 3.2]). *Consider $v : [0, 1] \rightarrow \mathbb{R}^m$. For any $t \in [0, 1]$, the differential of Φ_f w.r.t. u , denoted $d_u \Phi_f$, equals*

$$d_u \Phi_f(t, x^0; u) \cdot v = z_{u,v}(t)$$

where $z_{u,v} : [0, 1] \rightarrow \mathbb{R}^n$ is defined by the IVP

$$\begin{cases} \dot{z}_{u,v}(t) = \frac{\partial f}{\partial x}(x(t), u(t)) z_{u,v}(t) + \frac{\partial f}{\partial u}(x(t), u(t)) v(t), & \forall t \in [0, 1], \\ z_{u,v}(0) = \mathbb{O}_{n \times 1}, \end{cases} \quad (\text{A.10})$$

where $x(t) = \Phi_f(t, x^0; u)$.

When the control depends on a vector parameter, the following corollary holds.

Corollary 5. *Consider $u : (\theta, t) \in \mathbb{R}^d \times [0, 1] \rightarrow u(\theta, t) \in \mathbb{R}^m$ and $\mathcal{Z}(\theta, t) := \Phi_f(t, x^0; u(\theta, \cdot))$. For any $t \in [0, 1]$, the derivative of \mathcal{Z} w.r.t. θ is*

$$\frac{\partial \mathcal{Z}}{\partial \theta}(\theta, t) = z(t)$$

where z is defined by the matrix-IVP

$$\begin{aligned} \dot{z}(t) &= \frac{\partial f}{\partial x}(x(t), u(\theta, t)) z(t) + \frac{\partial f}{\partial u}(x(t), u(\theta, t)) \frac{\partial u}{\partial \theta}(\theta, t), \\ z(0) &= \mathbb{O}_{n \times d}. \end{aligned}$$

where $x(t) = \Phi_f(t, x^0; u(\theta, \cdot))$.

Since the variable u belongs to a functional space, it is more technical to state how smooth is the flow w.r.t. to the control. Hence the need for the following proposition, which formally describes to what extent the function $z_{u,v}$ from Equation (A.10) defines the first-order expansion of Φ_f w.r.t. u .

Proposition 21 (Adapted from [75, Prop. 1.3]). *Consider a bounded set $\Omega \subset \mathbb{R}^m$ and a vector $x^0 \in \mathbb{R}^n$. Let us assume that $f : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$ is differentiable, that f , $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial u}$ are continuous and that there exists $K < \infty$ s.t.*

$$\left\| \frac{\partial f}{\partial x}(x, u) \right\| \leq K, \quad \forall (x, u) \in \mathbb{R}^n \times \Omega.$$

Then there exists a function $\varepsilon : \mathbb{R}_+^* \rightarrow \mathbb{R}_+^*$ s.t. $\varepsilon(s)/s \xrightarrow{s \downarrow 0} 0$ and

$$\left\| \Phi_f(\cdot, x^0; u + v) - \left(\Phi_f(\cdot, x^0; u) + z_{u,v}(\cdot) \right) \right\|_{L^\infty} \leq \varepsilon(\|v\|_{L^\infty})$$

where¹

- $u \in \{w \in L^1([0, 1], \mathbb{R}^m) : w(t) \in \Omega \text{ a.e. on } [0, 1]\}$,
- $v \in L^\infty([0, 1], \mathbb{R}^m)$,
- $z_{u,v}$ is defined in Equation (A.10).

A.2.2.4 Technical summary

Consider a parametric control $t \mapsto u_\mu(t)$, continuously differentiable in t and μ . The following expansion holds

$$\begin{aligned} \Phi_f\left(t, x^0 + \Delta x^0, \eta + \Delta\eta; u_\mu\right) &= \Phi_f\left(t, x^0, \eta; u_0\right) + A(t)\Delta x^0 + B(t)\Delta\eta + C(t)\mu \\ &\quad + \varepsilon(\Delta x^0, \Delta\eta, \mu) \end{aligned}$$

where ε is a function s.t. $\varepsilon(\star)/\|\star\|_2$ tends to zero when $\star = (\Delta x^0, \Delta\eta, \mu)$ tends to zero, and where A , B and C are matrix valued functions defined by the following IVPs

$$\begin{aligned} \dot{A}(t) &= \frac{\partial f}{\partial x}(x(t), u_0(t), \eta) A(t), & A(0) &= \mathbb{I}_n, \\ \dot{B}(t) &= \frac{\partial f}{\partial x}(x(t), u_0(t), \eta) B(t) + \frac{\partial f}{\partial \eta}(x(t), u_0(t), \eta), & B(0) &= \mathbb{O}_{n \times n_\eta}, \\ \dot{C}(t) &= \frac{\partial f}{\partial x}(x(t), u_0(t), \eta) C(t) + \frac{\partial f}{\partial u}(x(t), u_0(t), \eta) \left. \frac{\partial u_\mu}{\partial \mu} \right|_{\mu=0}(t), & C(0) &= \mathbb{O}_{n \times n_\mu}, \end{aligned}$$

where $x(t) = \Phi_f(t, x^0, \eta; u_0)$.

¹Here, $L^\infty([0, 1], \mathbb{R}^m)$ denotes the vector space of essentially bounded measurable functions, for the essential supremum norm: $\|u\|_{L^\infty} := \inf\{C \geq 0 : \|u(t)\|_2 \leq C \text{ a.e. on } [0, 1]\}$. Similarly, $L^1([0, 1], \mathbb{R}^m)$ denotes the vector space of essentially bounded measurable functions, for the 1-norm: $\|v\|_{L^1} := \int_0^1 \|v(t)\|_2 dt$.

Appendix B

Additional data

Résumé

Ce chapitre contient des données complémentaires concernant l'exemple détaillé dans la Section 6.2 du Chapitre 6.

This chapter contains complementary data regarding the example detailed in Section 6.2 of Chapter 6. Are displayed in these extra charts:

- the height states with respect to the normalized time,
- the heatmap of the optimal time-of-flight change Δt_f^* ,
- and the heatmap of the second and fourth negotiations (\mathcal{P}_2^* and \mathcal{P}_4^*).

These charts are presented for each pairs of Table 6.3.

Additionally, Figure B.23 displays a cross-section view of a specific pair, to help visualize the variations of the optimal time-of-flight change w.r.t. to a specific input direction.

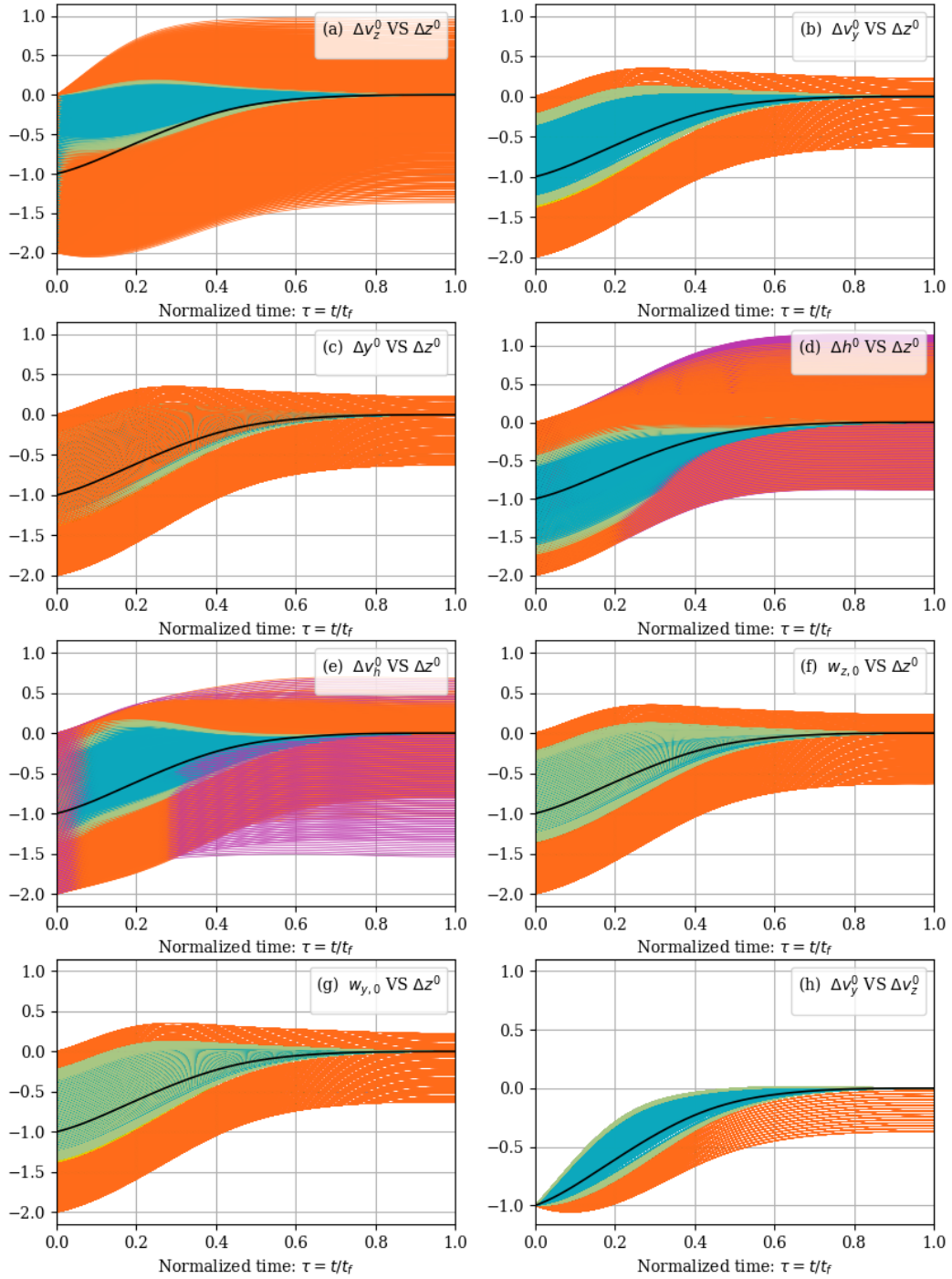


Figure B.1: Horizontal position z (in-plane) w.r.t normalized time. Batch A.

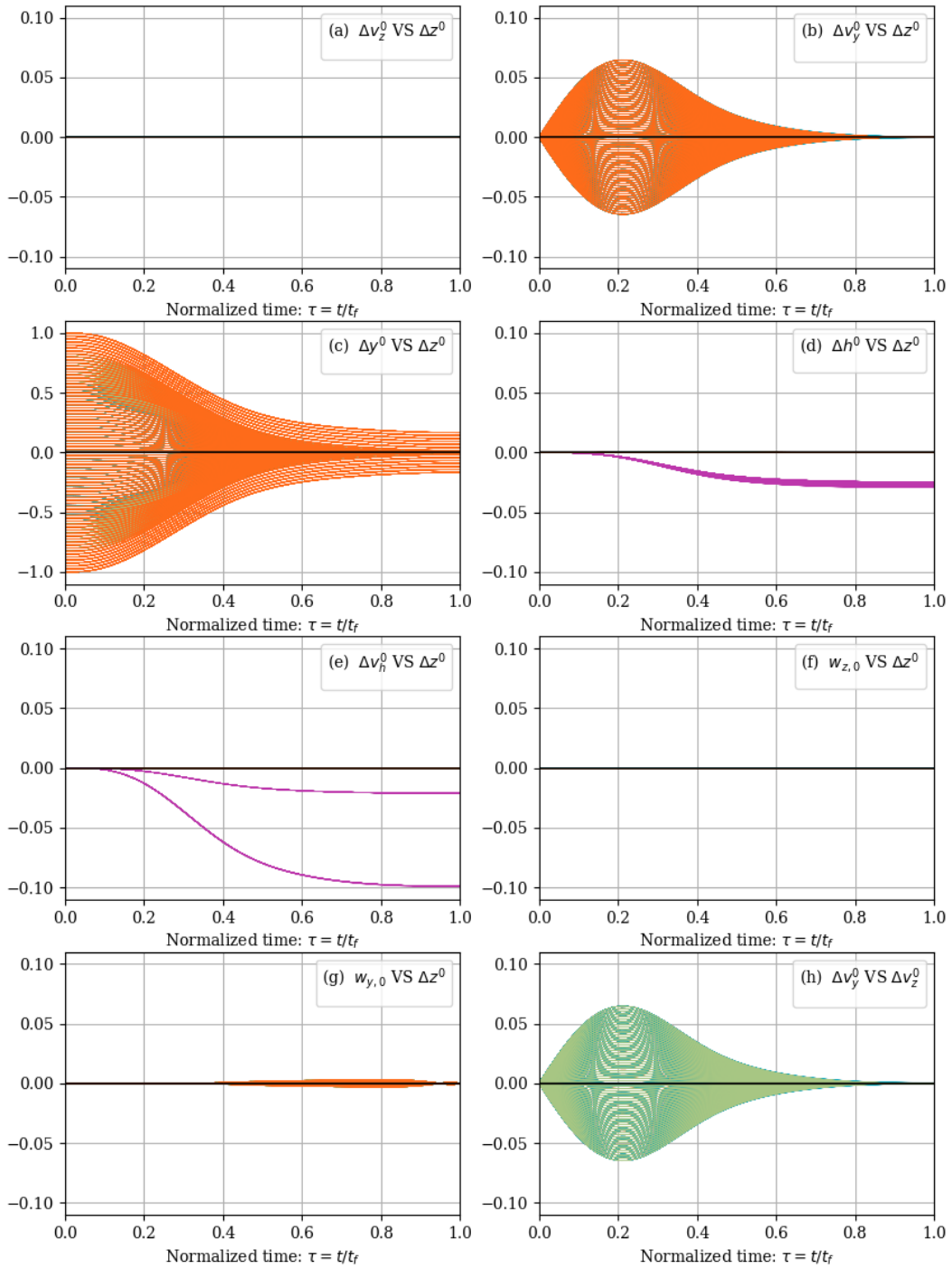


Figure B.2: Horizontal position y (out-of-plane) w.r.t normalized time. Batch A.

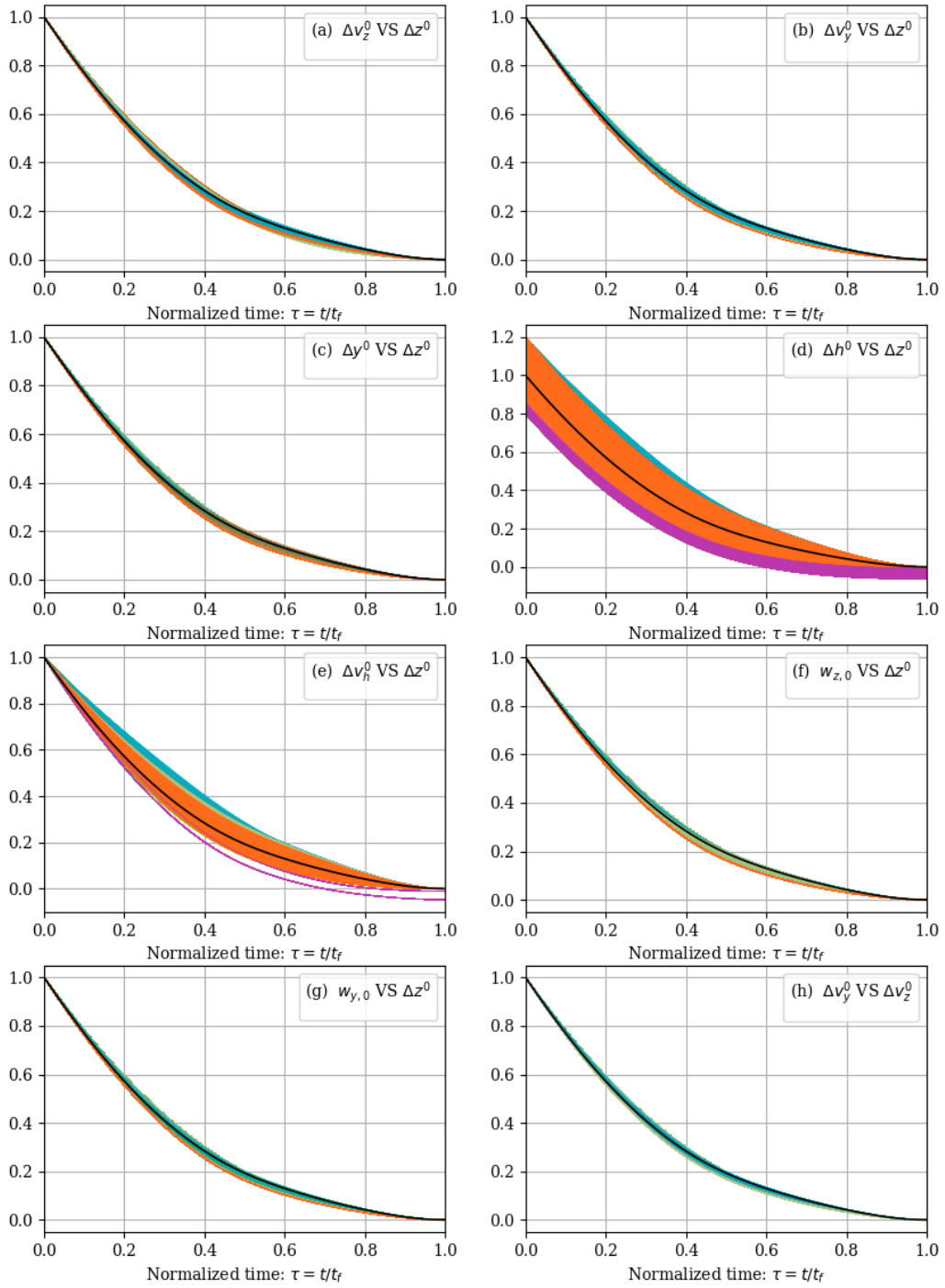


Figure B.3: Altitude h w.r.t normalized time. Batch A.

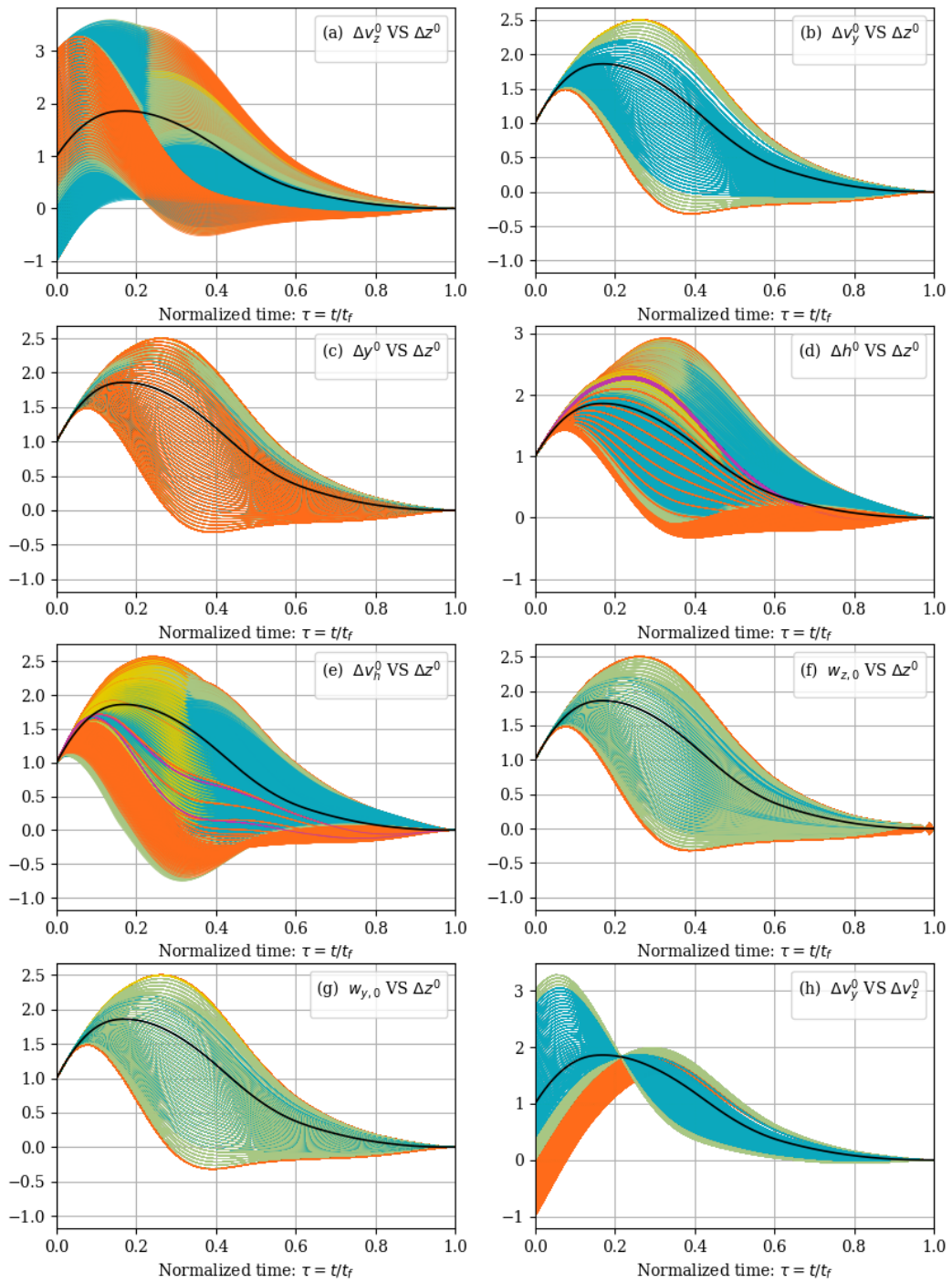


Figure B.4: Horizontal speed v_z (in-plane) w.r.t normalized time. Batch A.

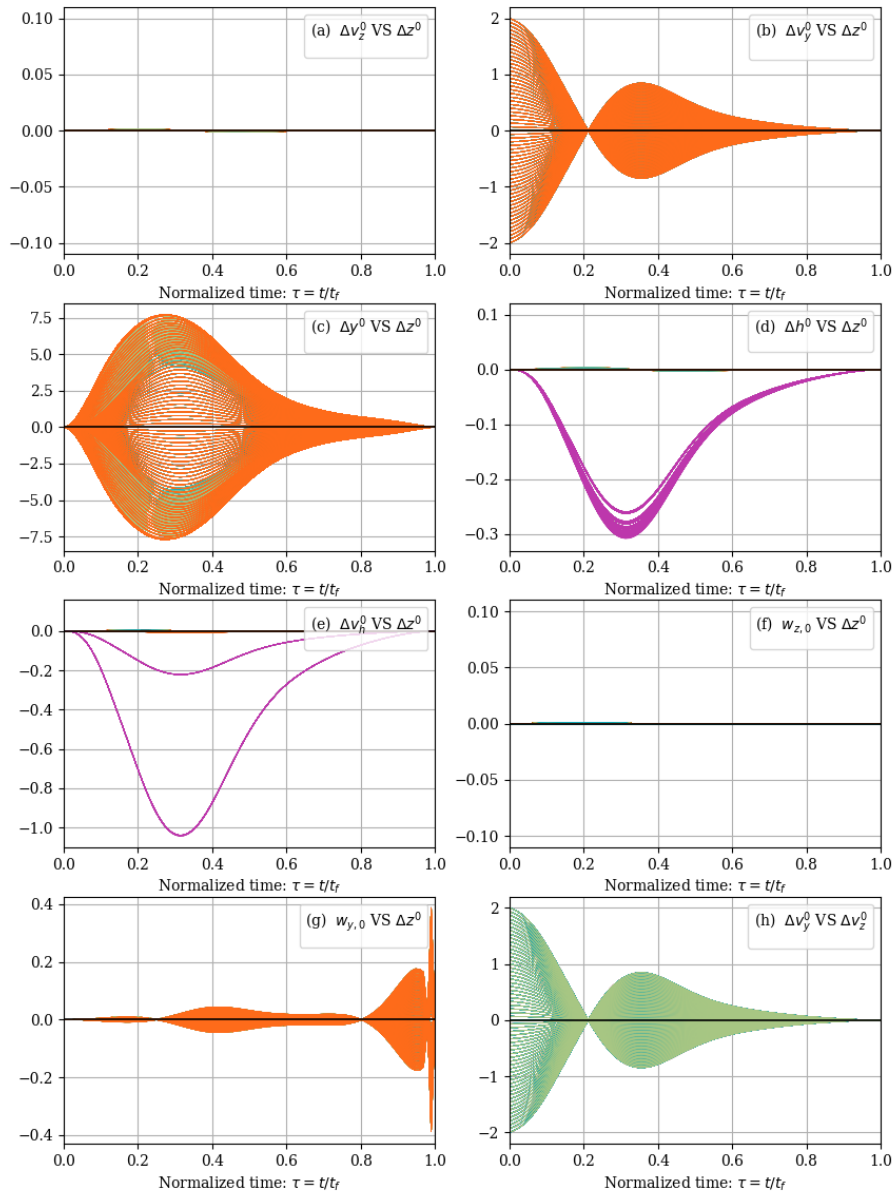


Figure B.5: *Horizontal speed v_y (out-of-plane) w.r.t normalized time. Batch A.* For the pair (g), the final values of the horizontal speed v_y seem erratic. On the one hand, the amplitude of the horizontal speed v_y is not critical, as the vertical scale suggests (see e.g. Figure (c), with a scale 20 times larger), and as the corresponding value of y in Figure B.2-(e) shows. On the other hand, this response to non-zero values of $w_{y,0}$ is not ideal, since it means that the rocket keeps a non-zero horizontal speed until the very last moment. This is partly due to the different ways of describing the wind profile (piecewise-affine) and the control corrections (Cubic Splines). A thinner parametric control correction would bring different results. Another way to reduce this oscillating behavior would be to impose the slope of the control correction to be exactly zero at $\tau = 1$, instead of penalizing it in the cost.

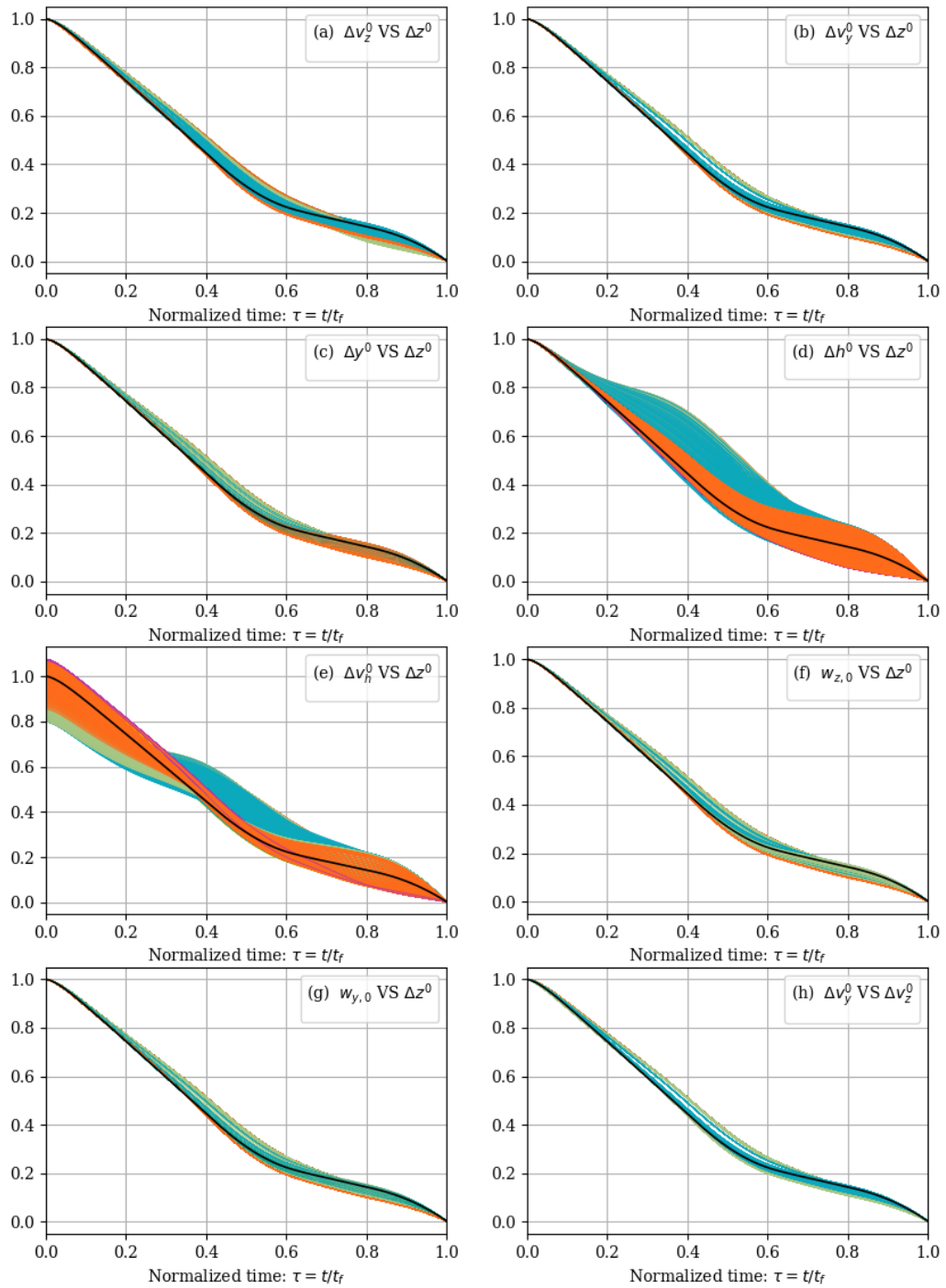


Figure B.6: Vertical speed v_h w.r.t normalized time. Batch A.

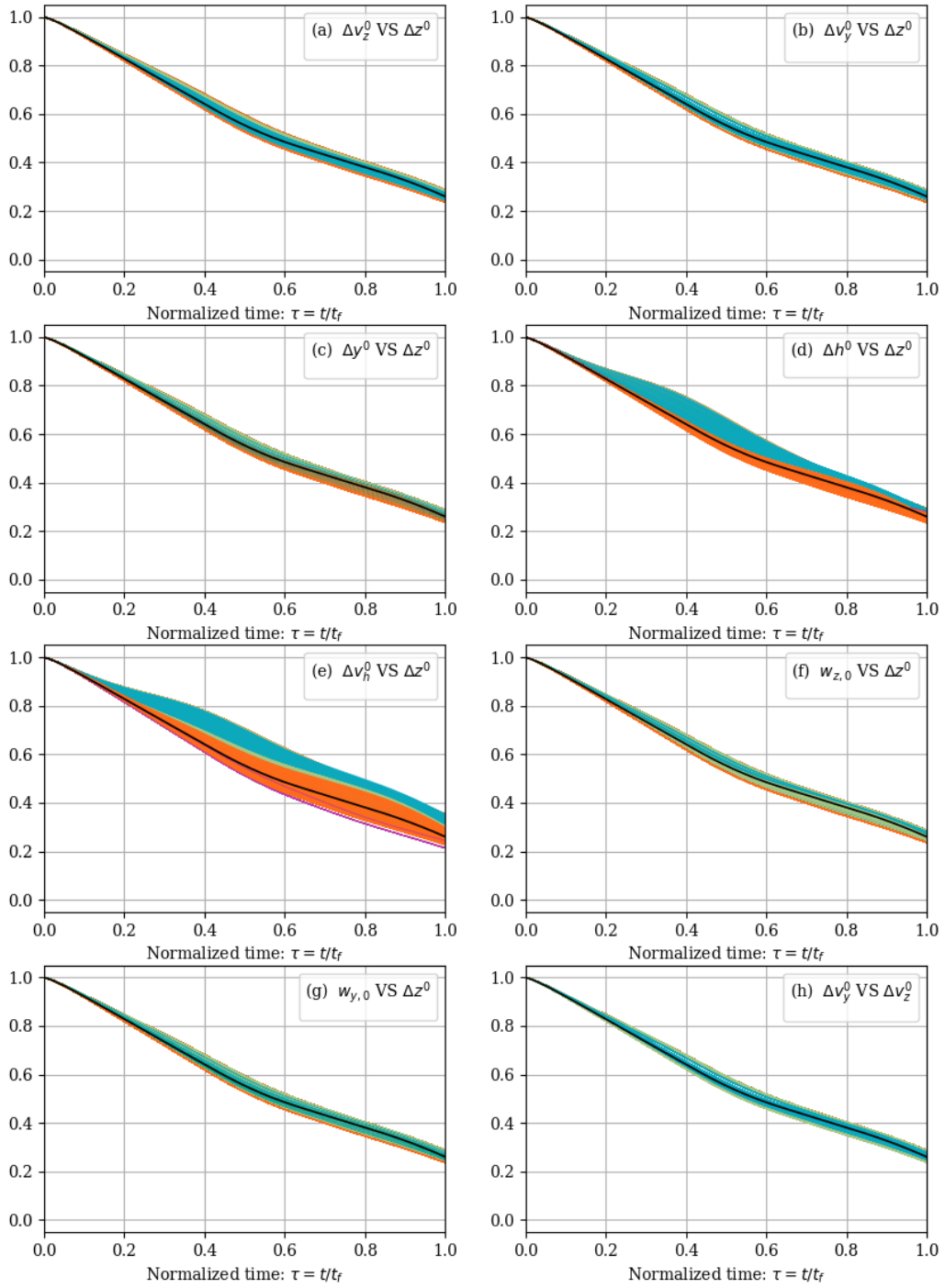


Figure B.7: Mass m w.r.t normalized time. Batch A.

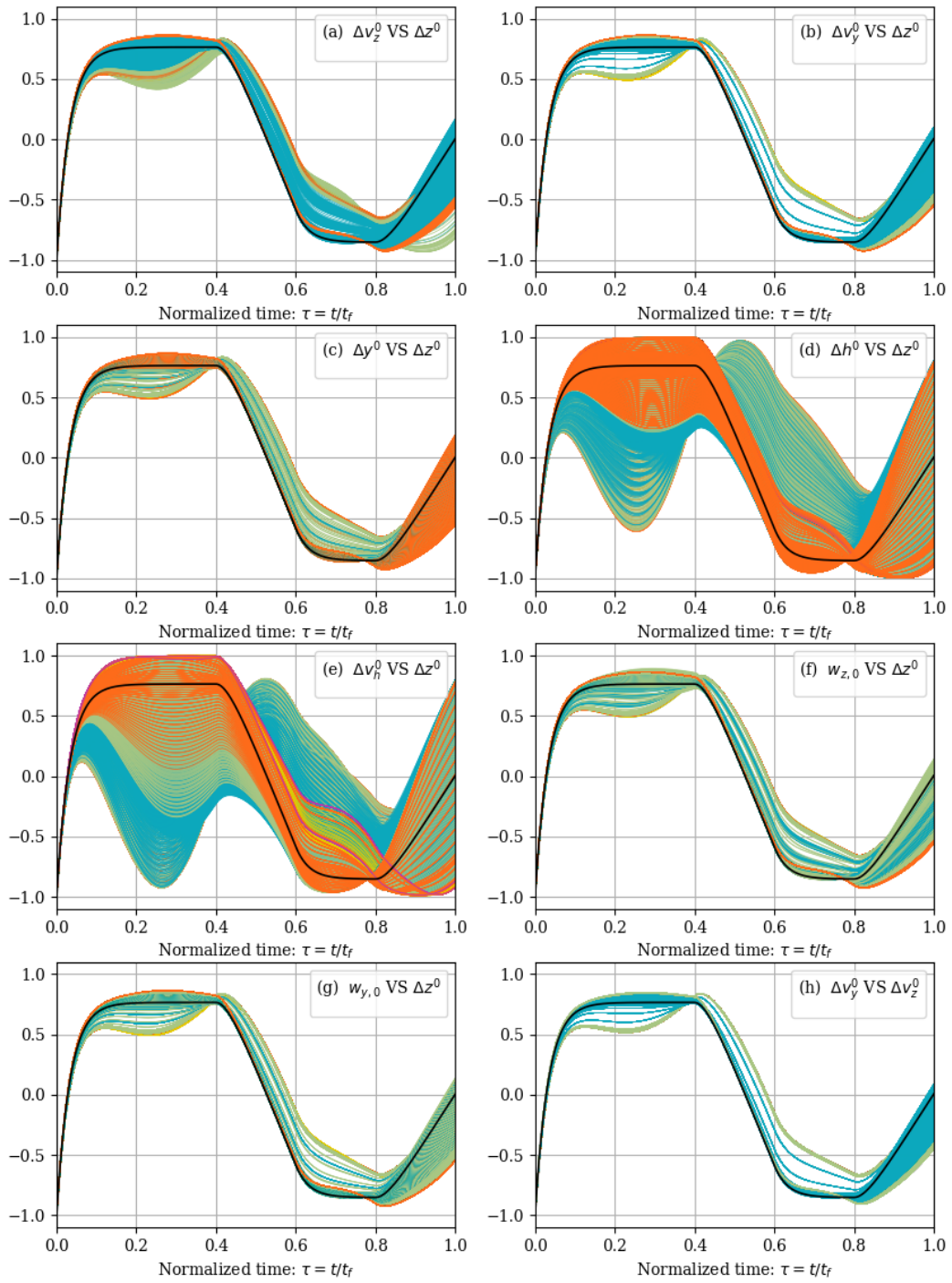


Figure B.8: Real engine flow q_r w.r.t normalized time. Batch A.

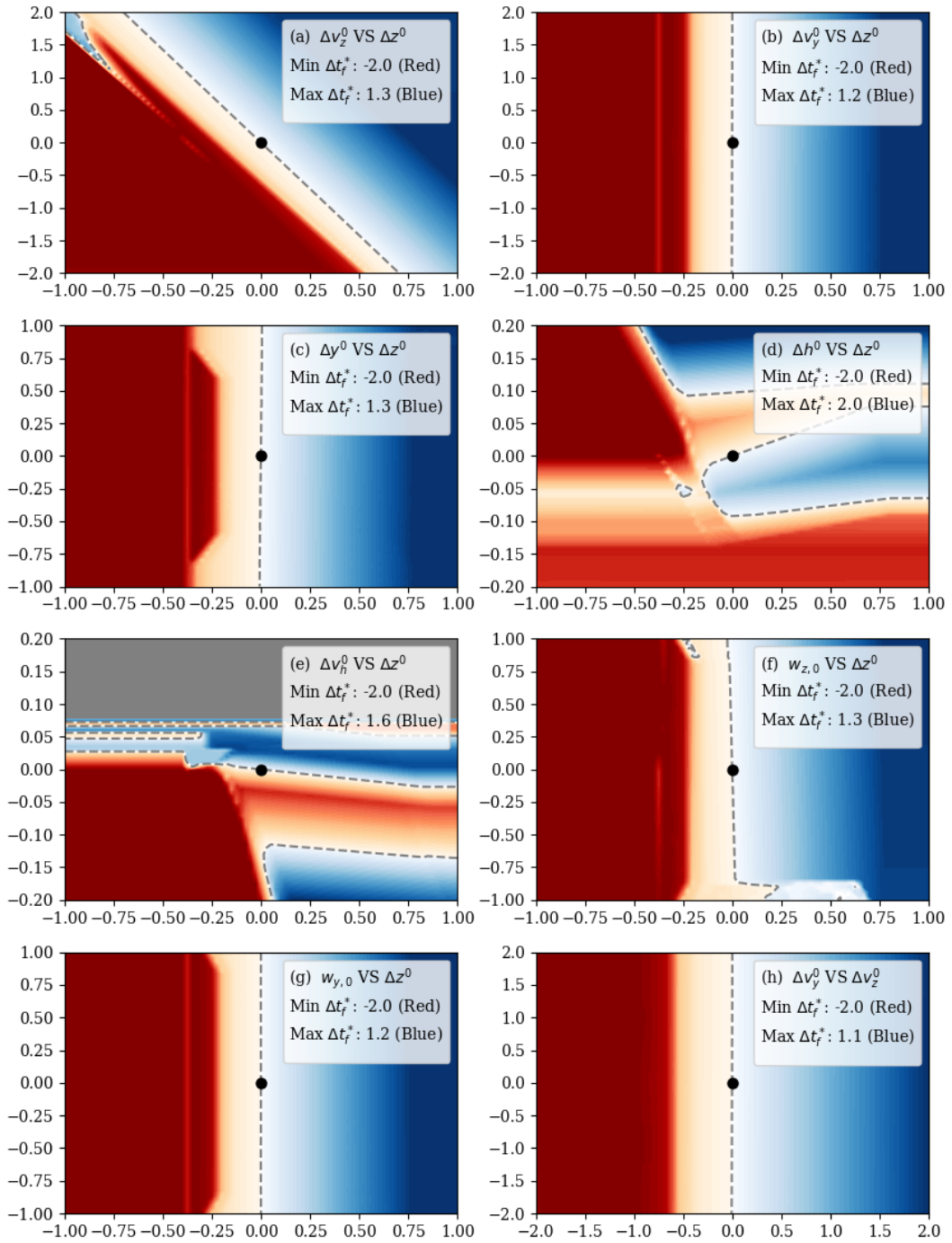


Figure B.9: Optimal time-of-flight change Δt_f^* . Batch A.

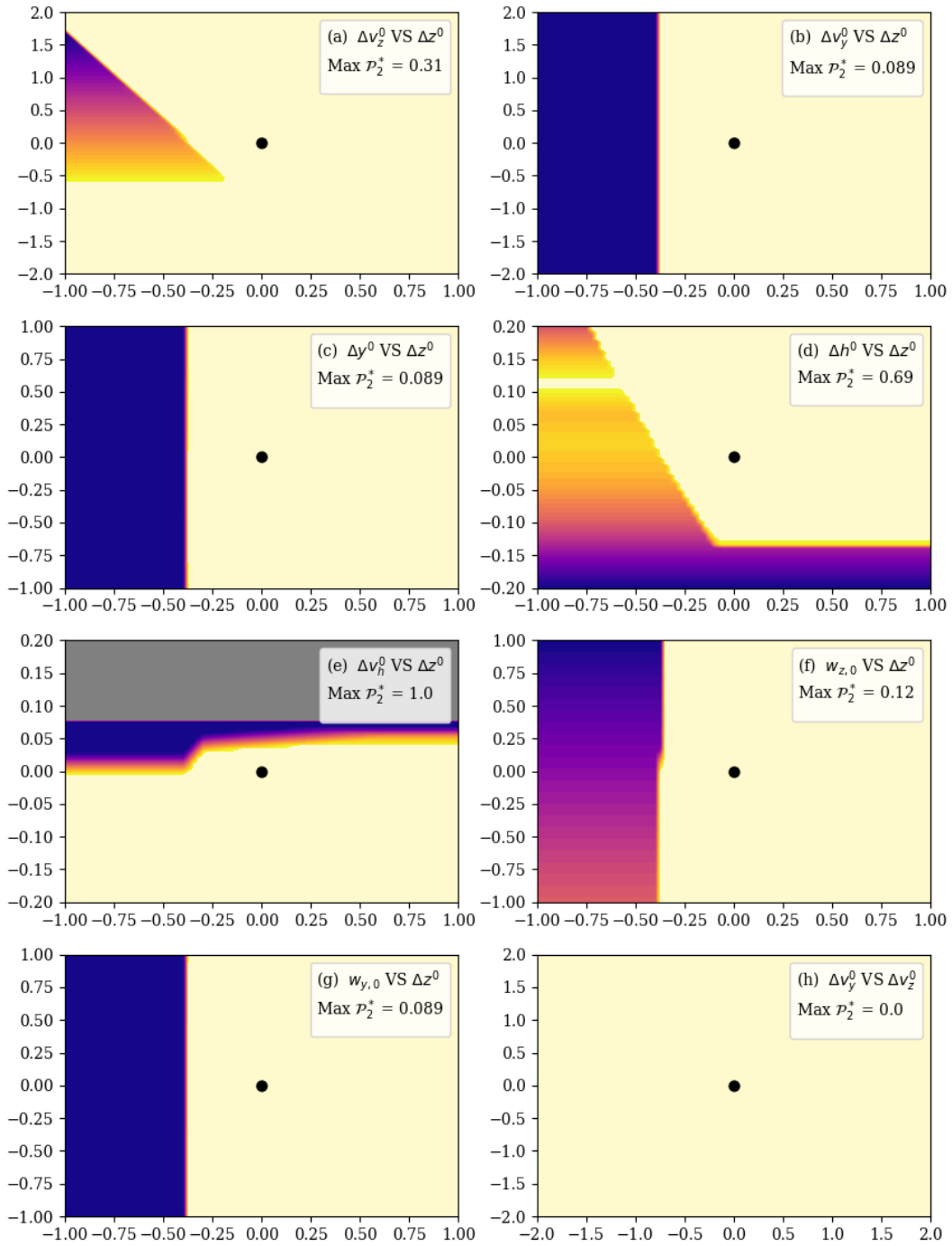


Figure B.10: Heatmap of second negotiation $\mathcal{P}_2^* = |\Delta a_{\text{nor}}^{\text{max}}|$. Batch A.

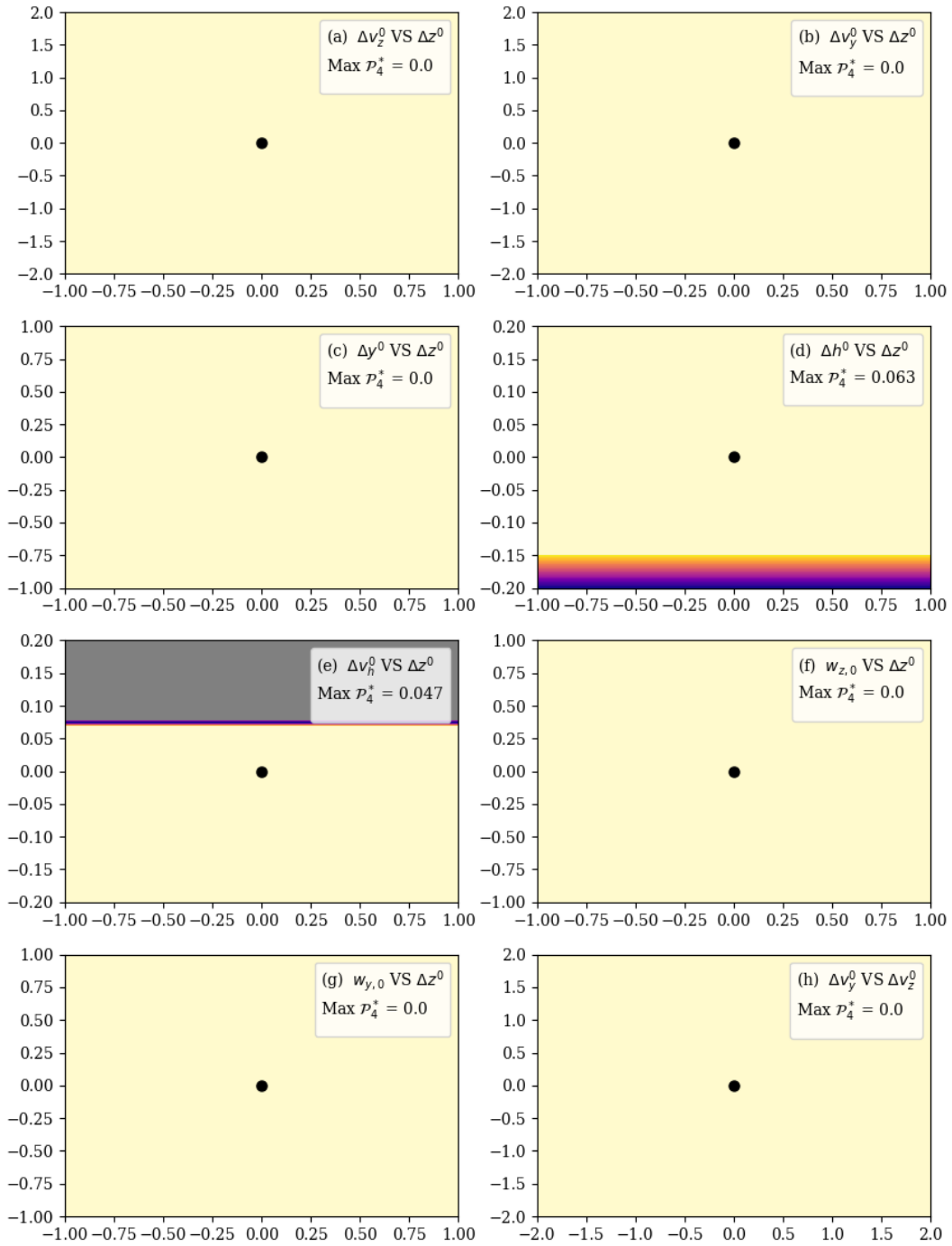


Figure B.11: Heatmap of fourth negotiation $\mathcal{P}_4^* = |\Delta h^f|$. Batch A.

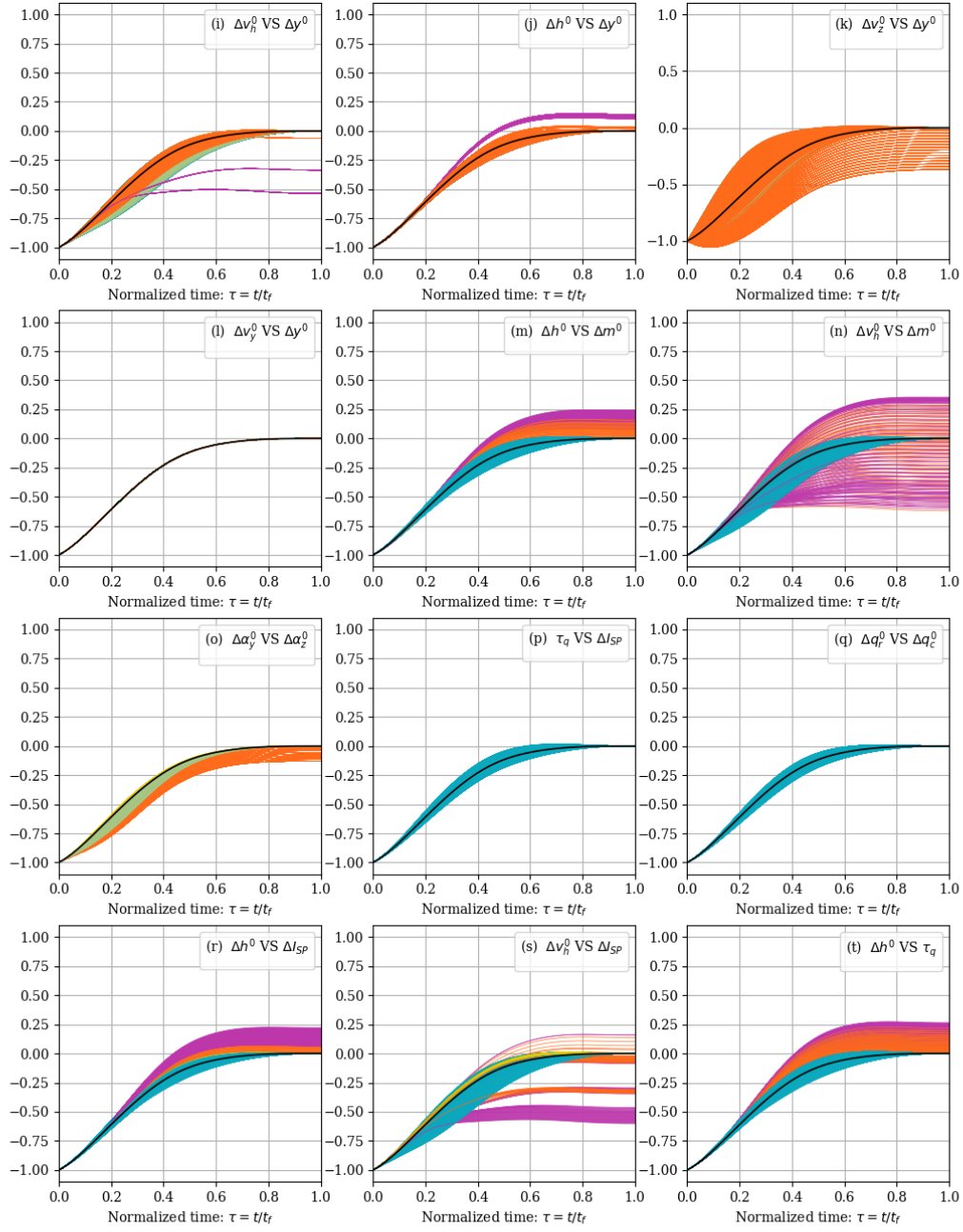


Figure B.12: Horizontal position z (in-plane) w.r.t normalized time. Batch B.

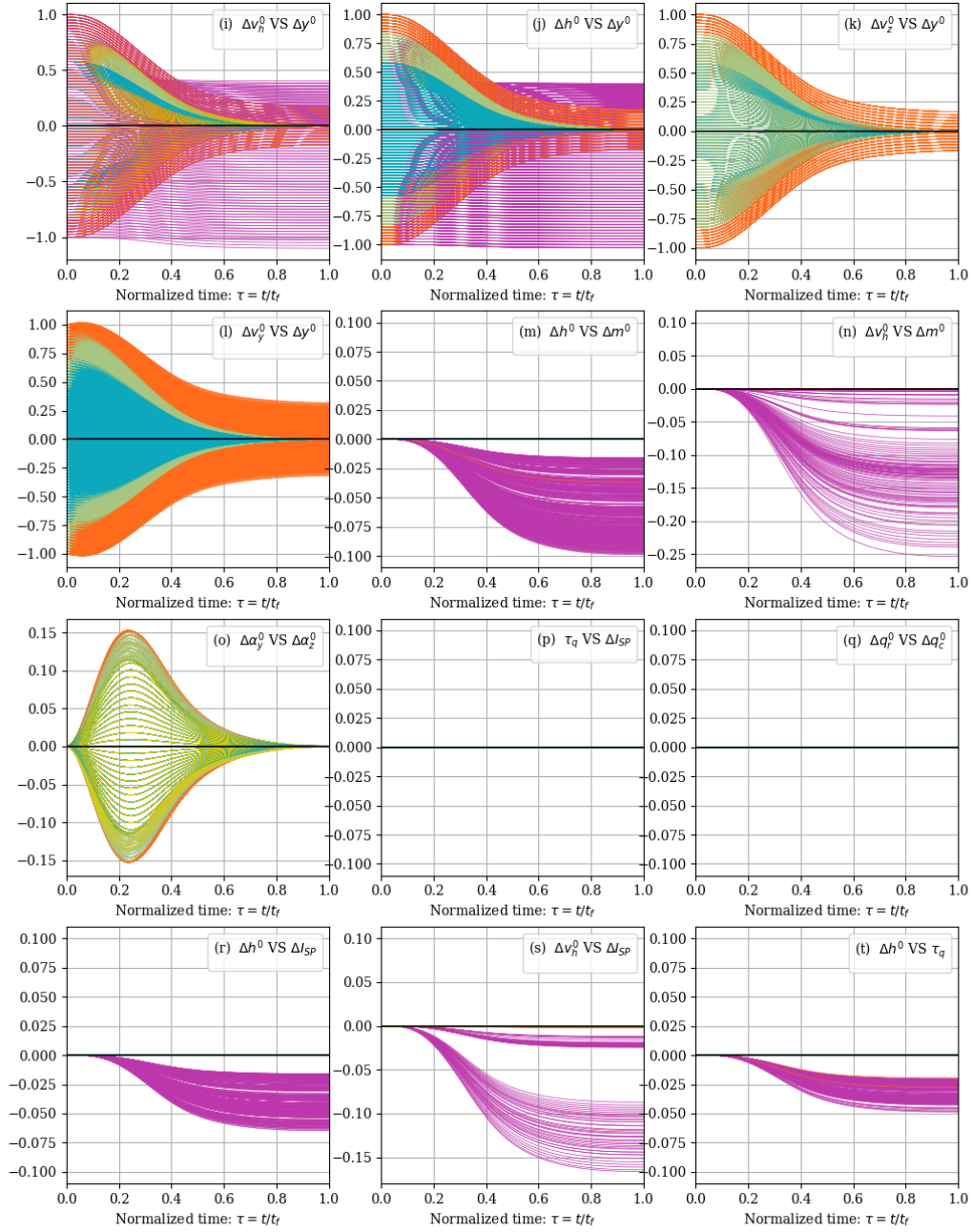
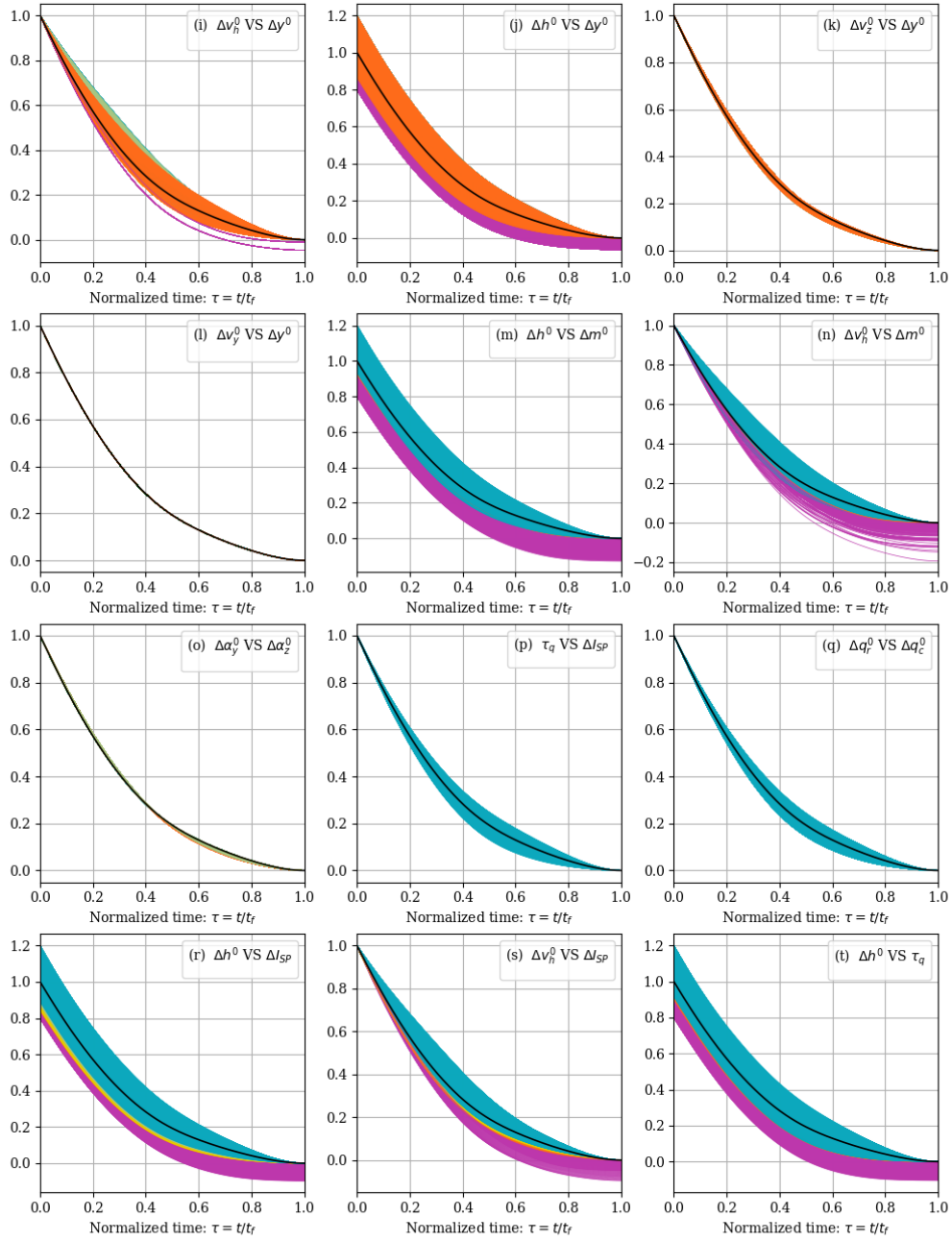


Figure B.13: Horizontal position y (out-of-plane) w.r.t normalized time. Batch B.

Figure B.14: Altitude h w.r.t normalized time. Batch B.

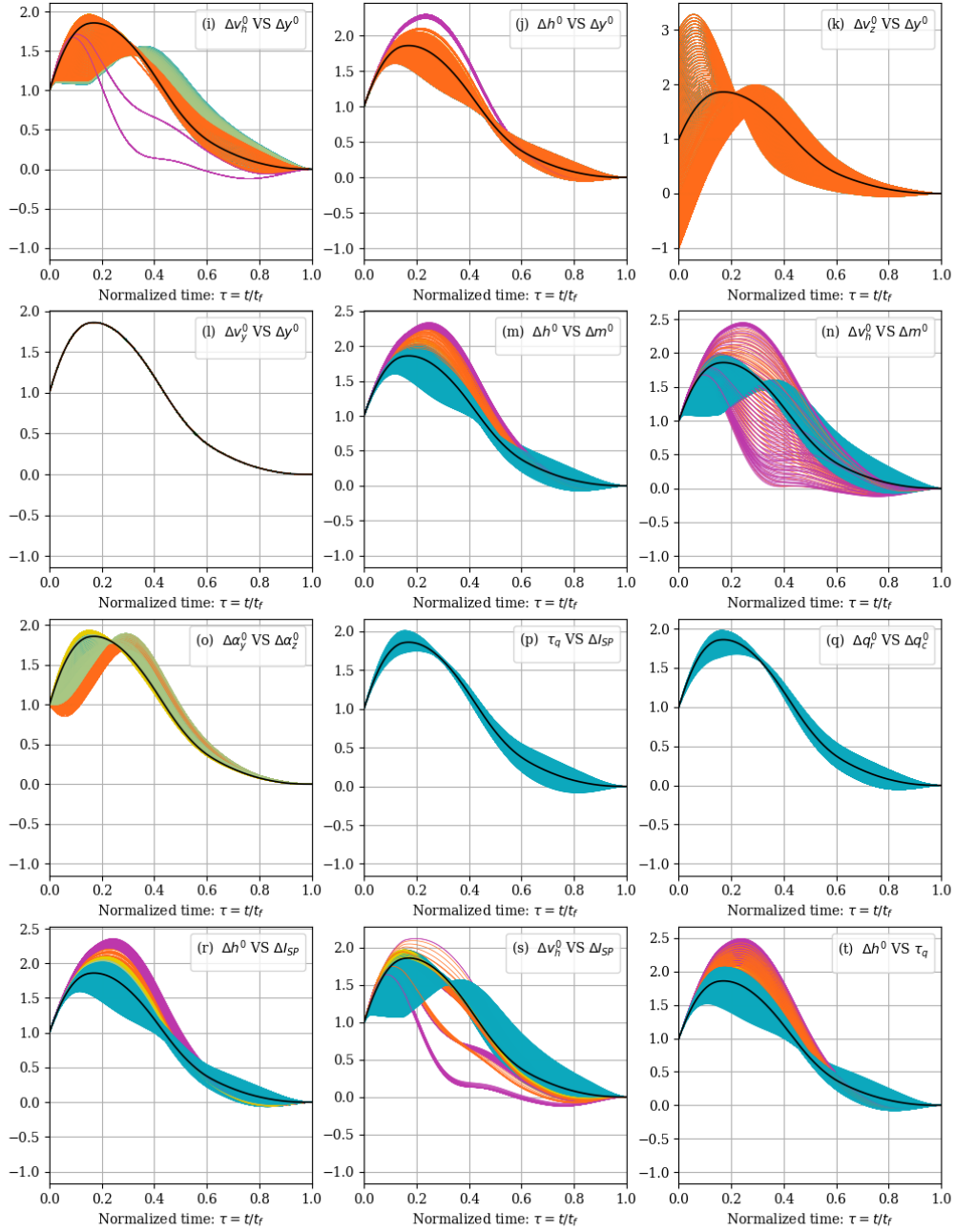


Figure B.15: Horizontal speed v_z (in-plane) w.r.t normalized time. Batch B.

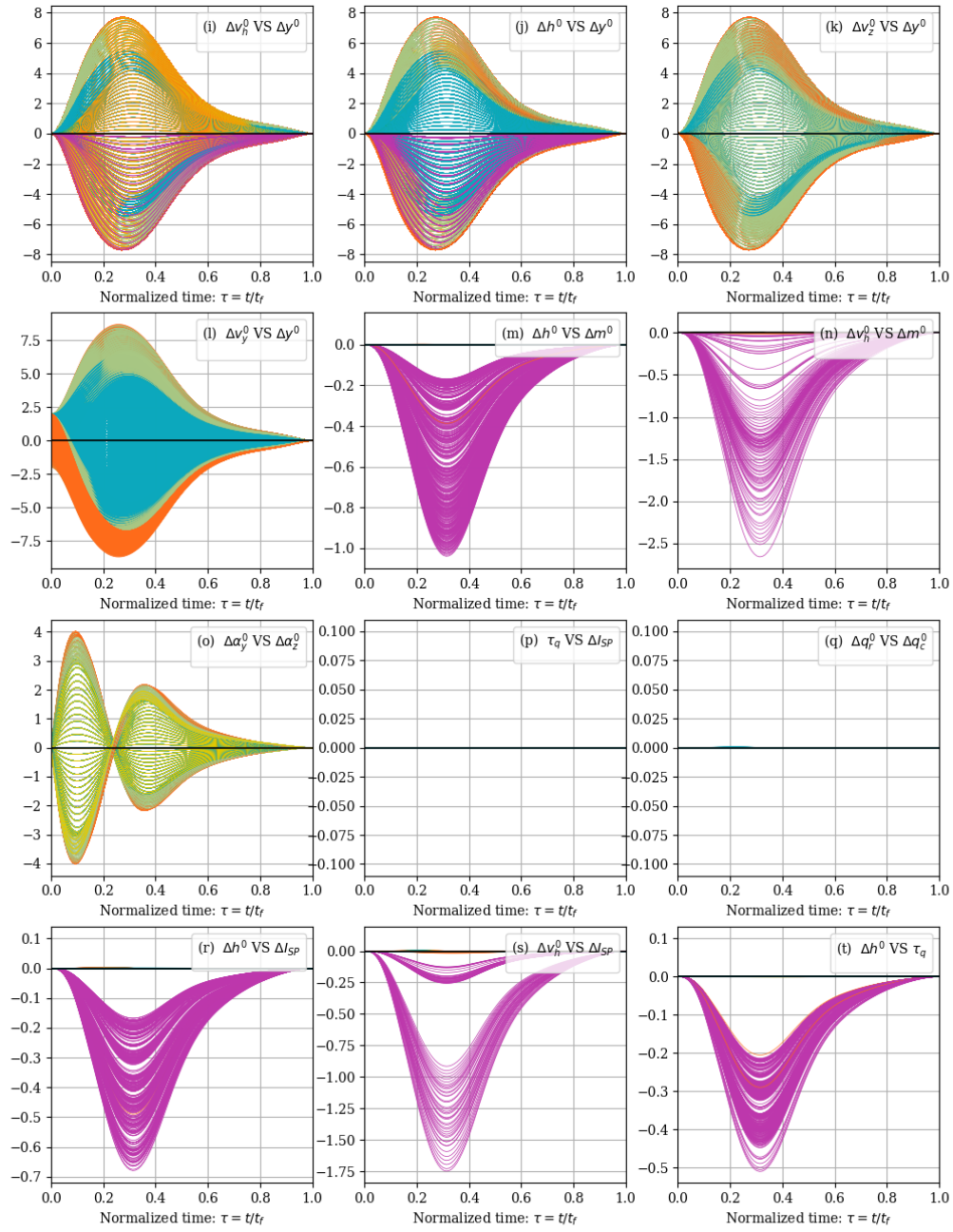


Figure B.16: Horizontal speed v_y (out-of-plane) w.r.t normalized time. Batch B.

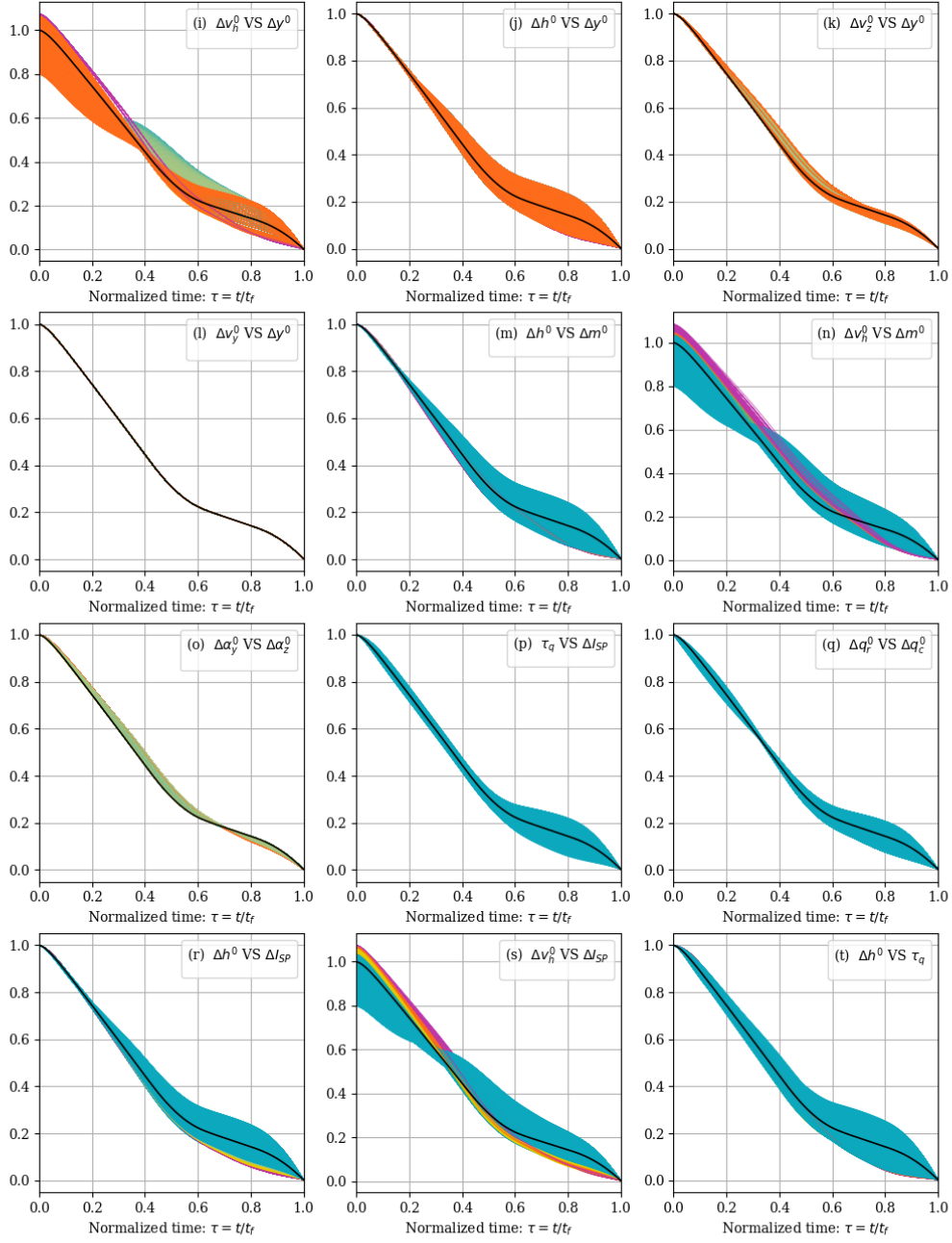
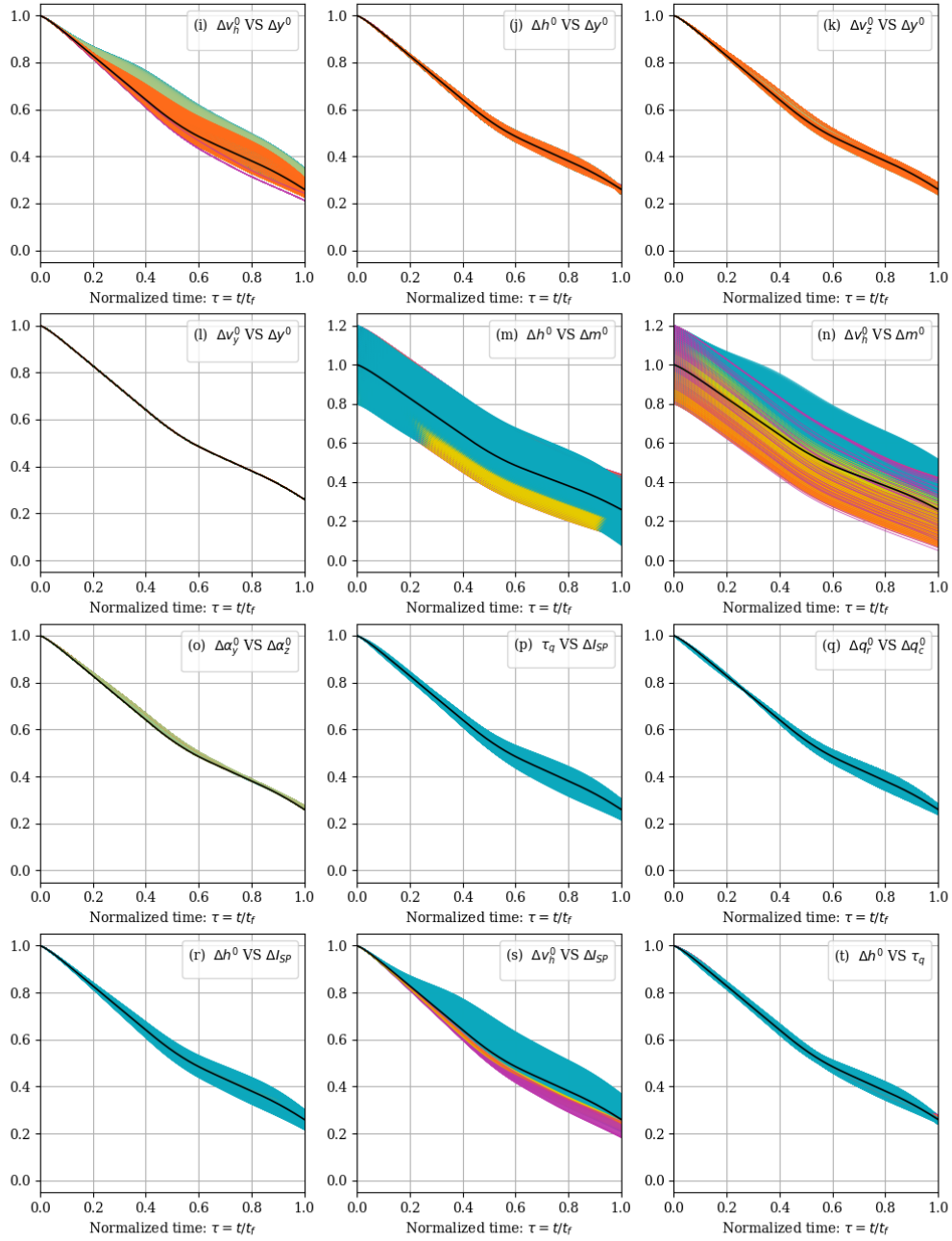


Figure B.17: Vertical speed v_h w.r.t normalized time. Batch B.

Figure B.18: Mass m w.r.t normalized time. Batch B.

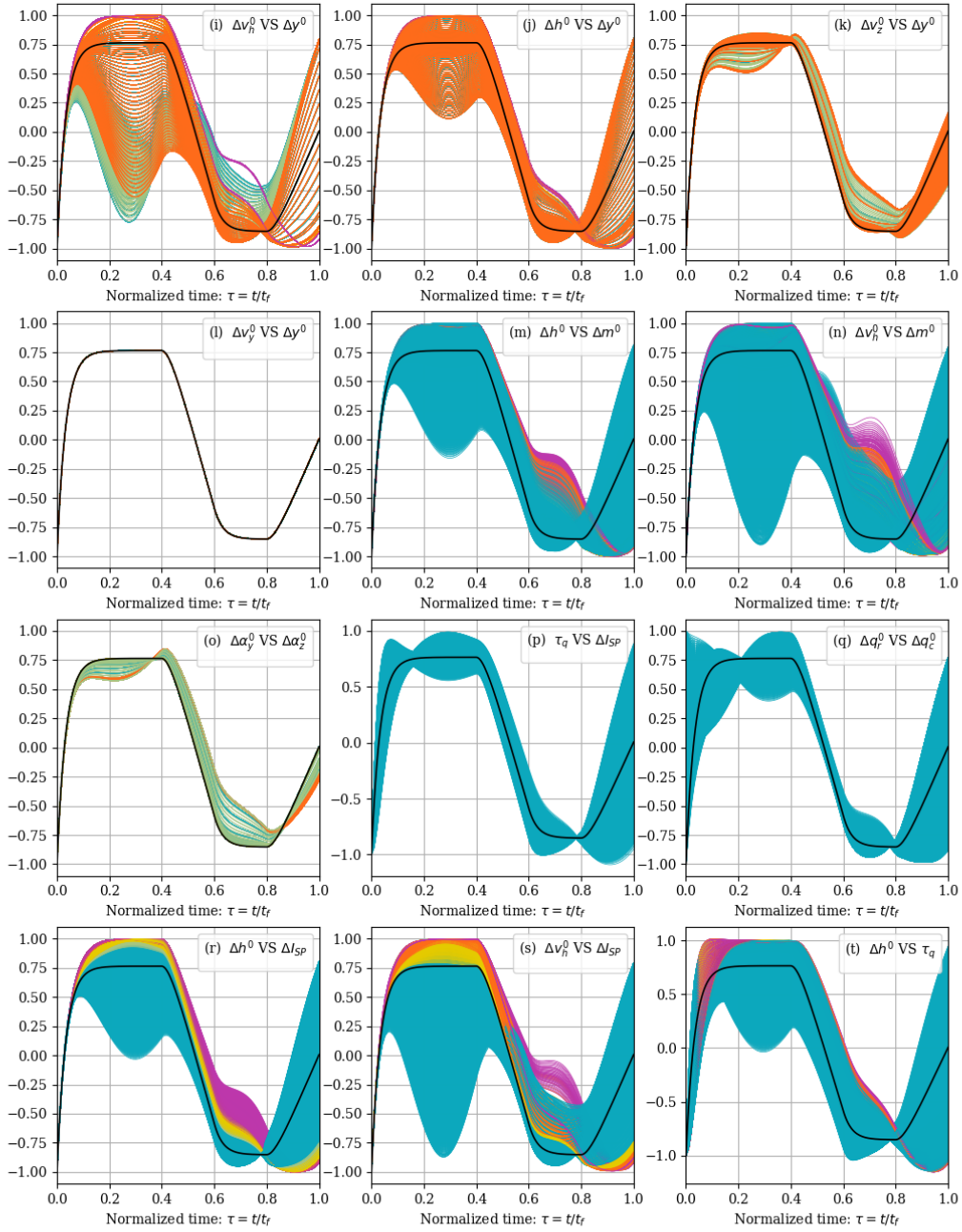
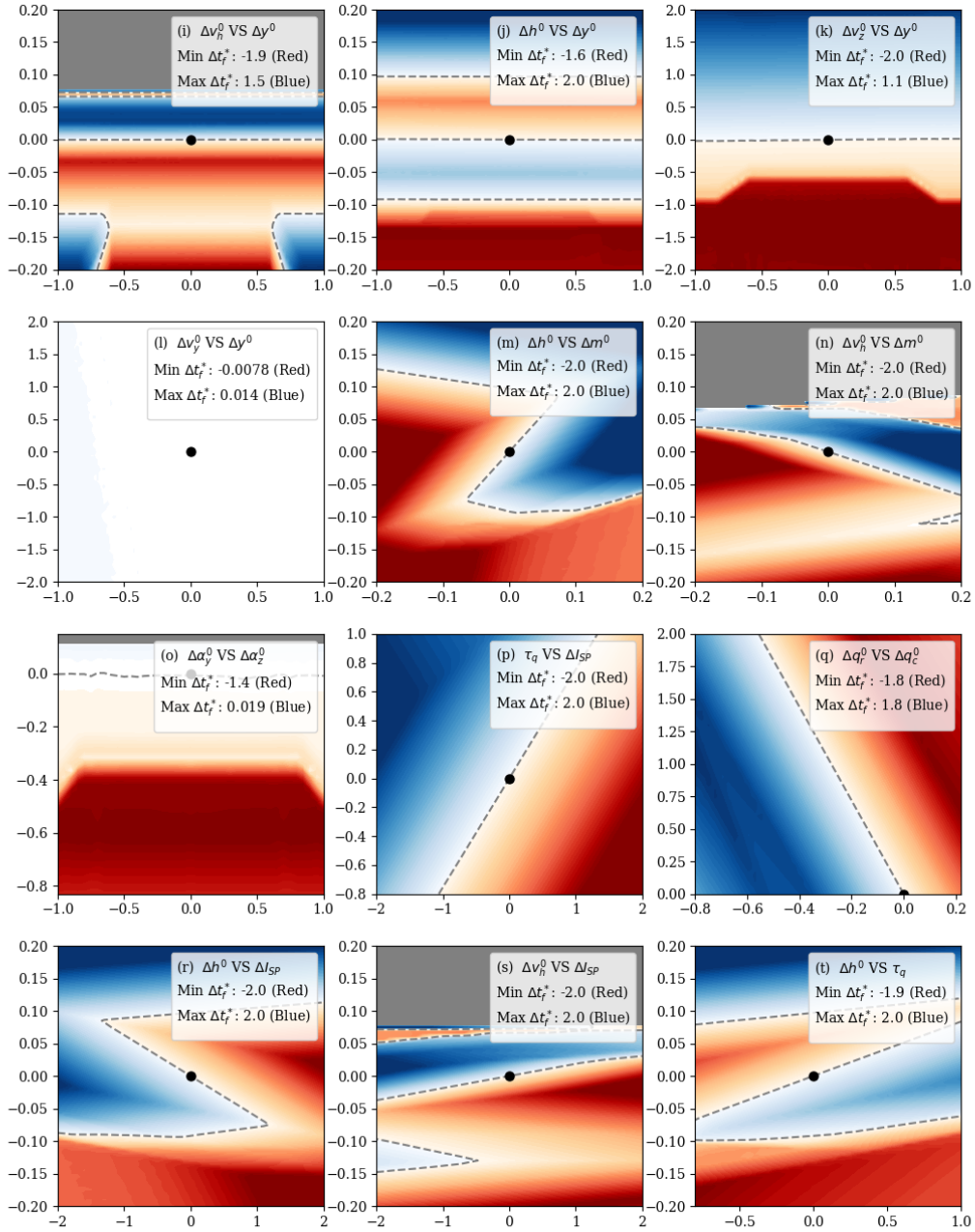


Figure B.19: Real engine flow q_r w.r.t normalized time. Batch B.

Figure B.20: Optimal time-of-flight change Δt_f^* . Batch B.

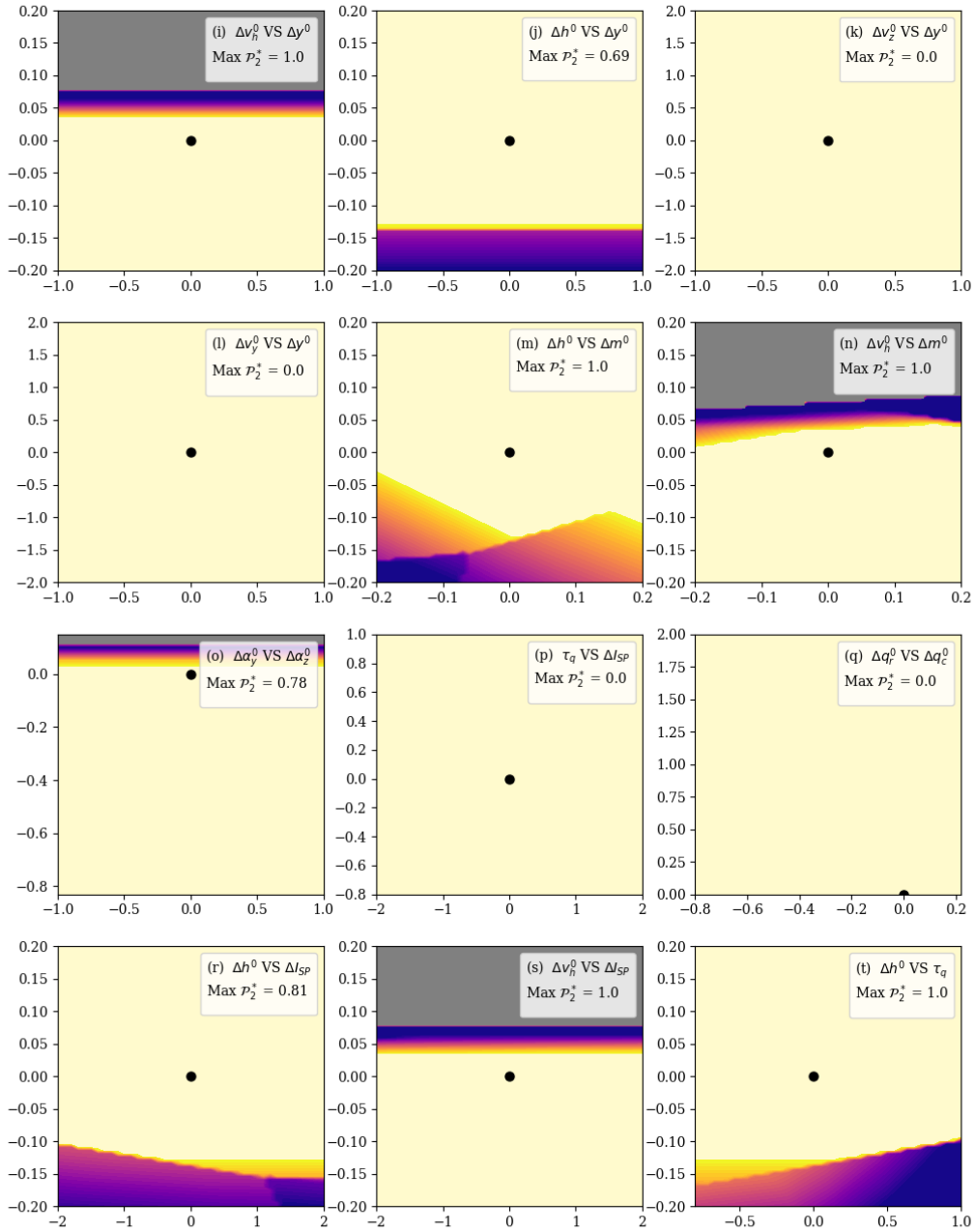


Figure B.21: Heatmap of second negotiation $\mathcal{P}_2^* = |\Delta a_{\text{nor}}^{\max}|$. Batch B.

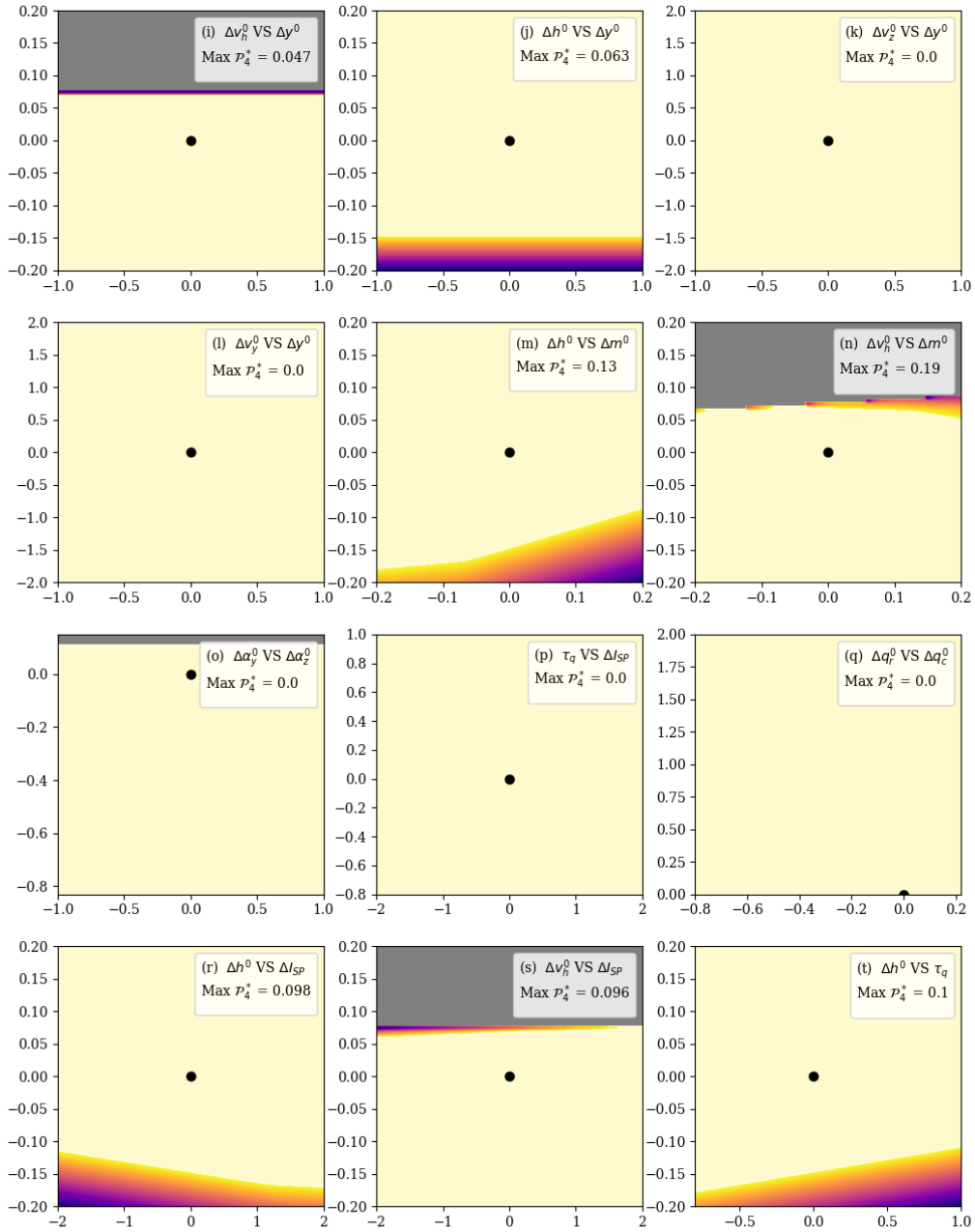


Figure B.22: Heatmap of fourth negotiation $\mathcal{P}_4^* = |\Delta h^f|$. Batch B.

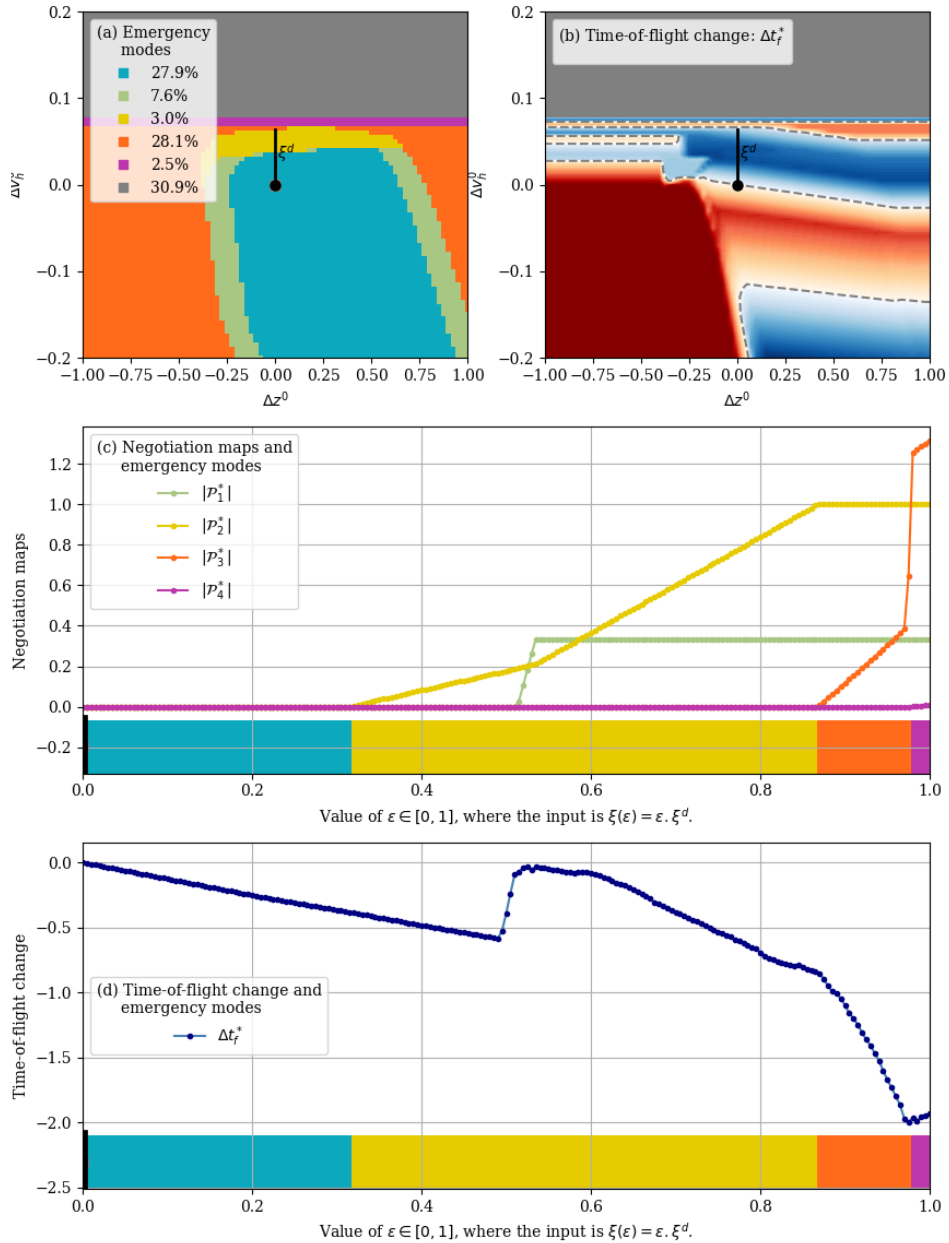


Figure B.23: *Cross-section view through the pair $(\Delta z^0, \Delta v_h^0)$.* The latter pair has a non-trivial heatmap regarding the optimal time-of-flight. To better understand how Δt_f^* changes depending on the emergency modes, the figures above show a cut in the direction ξ^d , which has only a single non-zero component along Δv_h^0 . The coordinate ϵ used in the sub-Figures (c) and (d) denotes the position along the thick black line from the sub-Figures (a) and (b).

Bibliography

- [1] B. Açikmeşe, M. Aung, J. Casoliva, S. Mohan, A. Johnson, D. Scharf, D. Masten, J. Scotkin, A. Wolf, and M. W. Regehr. Flight Testing Of Trajectories Computed By G-FOLD: Fuel Optimal Large Divert Guidance Algorithm For Planetary Landing. page 15, 2012.
- [2] B. Açikmeşe and L. Blackmore. Lossless convexification of a class of optimal control problems with non-convex control constraints. *Automatica*, pages 341–347, 2011.
- [3] B. Açikmeşe, J. Casoliva, J. M. Carson, and L. Blackmore. G-FOLD: A Real-Time Implementable Fuel Optimal Large Divert Guidance Algorithm for Planetary Pinpoint Landing. 2012.
- [4] B. Açikmeşe and S. R. Ploen. Convex Programming Approach to Powered Descent Guidance for Mars Landing. *Journal of Guidance, Control, and Dynamics*, 30(5):1353–1366, Sept. 2007.
- [5] I. Adler and R. D. C. Monteiro. A geometric view of parametric linear programming. *Algorithmica*, 8(1-6):161–176, Dec. 1992.
- [6] A. A. Agrachev and A. V. Sarychev. On Abnormal Extremals for Lagrange Variational Problems. *Journal of Mathematical Systems Estimation and Control*, 8(1):87–118, 1995.
- [7] E. D. Andersen and K. D. Andersen. The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm. In *High performance optimization*, pages 197–232. Springer, 2000.
- [8] M. Andersen, J. Dahl, and L. Vandenberghe. CVXOPT: Convex Optimization, 2020.
- [9] J.-P. Aubin. Lipschitz Behavior of Solutions to Convex Minimization Problems. *Mathematics of Operations Research*, 9(1):87–111, Feb. 1984.
- [10] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust Optimization*. Applied Mathematics. Princeton University Press, 2009.
- [11] A. Bensoussan. *Perturbation methods in optimal control*, volume 5. Wiley, 1988.

- [12] A. B. Berkelaar, B. Jansen, C. Roos, and T. Terlaky. Sensitivity analysis in (degenerate) quadratic programming. Technical Report No. EI 9611-/A, Delft University of Technology, Delft, Netherlands, 1996.
- [13] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, second edition edition, 1995.
- [14] J. T. Betts. *Practical methods for optimal control and estimation using nonlinear programming*. Advances in Design and Control. SIAM, 2001.
- [15] L. Blackmore. Autonomous Precision Landing of Space Rockets. *The Bridge*, 2016.
- [16] L. Blackmore, B. Açıkmeşe, and J. M. Carson. Lossless convexification of control constraints for a class of nonlinear optimal control problems. *Systems & Control Letters*, 61(8):863–870, Aug. 2012.
- [17] L. Blackmore, B. Açıkmeşe, and D. P. Scharf. Minimum-Landing-Error Powered-Descent Guidance for Mars Landing Using Convex Optimization. *Journal of Guidance, Control, and Dynamics*, pages 1161–1171, 2010.
- [18] J. F. Bonnans. *Course on Optimal Control, Part I: the Pontryagin approach*. SOD311 Ensta Paris Tech. Aug. 2019.
- [19] J. F. Bonnans and A. Shapiro. *Perturbation Analysis of Optimization Problems*. Springer New York, New York, NY, 2000.
- [20] B. Bonnard, L. Faubourg, and E. Trelat. Optimal control of the atmospheric arc of a space shuttle and numerical simulations with multiple-shooting method. *Mathematical Models and Methods in Applied Sciences*, 15(01):109–140, Jan. 2005.
- [21] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge. Cambridge University Press, 2004.
- [22] E. Brendel, B. Hérissé, and E. Bourgeois. Optimal guidance for toss back concepts of Reusable Launch Vehicles. In *EUCASS 2019*, July 2019.
- [23] J. W. Bruce and Giblin. *Curves and Singularities*. Cambridge university press edition, 1984.
- [24] A. E. Bryson and Y.-C. Ho. *Applied optimal control: optimization, estimation and control*. CRC Press, 1975.
- [25] C. Büskens and H. Maurer. Sensitivity Analysis and Real-Time Optimization of Parametric Nonlinear Programming Problems. In *Online Optimization of Large Scale Systems*, pages 3–16. 2001.
- [26] A. J. Calise and N. Brandt. Generation of Launch Vehicle Abort Trajectories Using a Hybrid Optimization Method. *Journal of Guidance, Control, and Dynamics*, 27(6):929–929, Nov. 2004.

- [27] J. M. Carson, B. Açikmeşe, and L. Blackmore. Lossless convexification of Powered-Descent Guidance with non-convex thrust bound and pointing constraints. In *Proceedings of the 2011 American Control Conference*, pages 2651–2656, June 2011. ISSN: 0743-1619.
- [28] J. W. Chinneck. *Feasibility and Infeasibility in Optimization: Algorithms and Computational Methods*. Springer Science & Business Media, 2007.
- [29] M. Clark, X. Koutsoukos, J. Porter, R. Kumar, G. Pappas, O. Sokolsky, I. Lee, and L. Pike. A Study on Run Time Assurance for Complex Cyber Physical Systems:. Technical report, Defense Technical Information Center, Fort Belvoir, VA, Apr. 2013.
- [30] M. Cococcioni, M. Pappalardo, and Y. D. Sergeyev. Lexicographic multi-objective linear programming using grossone methodology: Theory and algorithm. *Applied Mathematics and Computation*, 318:298–311, 2018.
- [31] J.-P. Demailly. *Analyse numérique et équations différentielles*. Grenoble Sciences. Presses Universitaires de Grenoble, 1996.
- [32] S. A. Deshpande, D. Bonvin, and B. Chachuat. Directional Input Adaptation in Parametric Optimal Control Problems. *SIAM Journal on Control and Optimization*, 50(4):1995–2024, Jan. 2012.
- [33] C. D’Souza. An optimal guidance law for planetary landing. In *Guidance, Navigation, and Control Conference*, page 3709, 1997.
- [34] D. Dueri, B. Açikmeşe, D. P. Scharf, and M. W. Harris. Customized Real-Time Interior-Point Methods for Onboard Powered-Descent Guidance. *Journal of Guidance, Control, and Dynamics*, 40(2):197–212, 2017.
- [35] U. Eren, D. Dueri, and B. Açikmeşe. Constrained Reachability and Controllability Sets for Planetary Precision Landing via Convex Optimization. *Journal of Guidance, Control, and Dynamics*, 38(11):2067–2083, Nov. 2015.
- [36] F. Fahroo and I. M. Ross. Direct Trajectory Optimization by a Chebyshev Pseudospectral Method. page 6, Chicago, Illinois, June 2000.
- [37] A. V. Fiacco. *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming*, volume 165 of *Mathematics in Science and Engineering*. Elsevier, 1983.
- [38] FSF. glpk, GNU Linear Programming Kit, 2012. gnu.org/software/glpk/.
- [39] R. Furfaro and R. Linares. Waypoint-Based Generalized Zem/Zev Feedback Guidance For Planetary Landing Via A Reinforcement Learning Approach. *3rd IAA Conf on Dynamics and Control of Space Systems*, page 16, 2017.
- [40] J. Garcia and F. Fernandez. A Comprehensive Survey on Safe Reinforcement Learning. *Journal of Machine Learning Research*, 16:1437–1480, 2015.

- [41] J. Gauvin. Formulae for the Sensitivity Analysis of Linear Programming Problems. In M. Lassonde, editor, *Approximation, Optimization and Mathematical Economics*, pages 117–120, Heidelberg, 2001. Physica-Verlag HD.
- [42] R. H. Goddard. *A Method Of Reaching Extreme Altitudes*. Smithsonian Institution, Baltimore, MD. U.S.A., 1920. Publication 2540.
- [43] K. Graichen and N. Petit. Solving the Goddard problem with thrust and dynamic pressure constraints using saturation functions. *IFAC Proceedings Volumes*, 41(2):14301–14306, 2008.
- [44] J. Hanson, D. Coughlin, G. Dukeman, J. Mulqueen, and J. McCarter. Ascent, transition, entry, and abort guidance algorithm design for the X-33 vehicle. In *Guidance, Navigation, and Control Conference and Exhibit*, Boston, MA, U.S.A., Aug. 1998. American Institute of Aeronautics and Astronautics.
- [45] C. Hargraves and S. Paris. Direct trajectory optimization using nonlinear programming and collocation. *Journal of Guidance, Control, and Dynamics*, 10(4):338–342, 1987. American Institute of Aeronautics and Astronautics.
- [46] R. F. Hartl, S. P. Sethi, and R. G. Vickson. A Survey of the Maximum Principles for Optimal Control Problems with State Constraints. *SIAM Review*, 37(2):181–218, June 1995.
- [47] D. G. Hull. *Optimal control theory for applications*. Springer, New York; London, 2011. OCLC: 1063549889.
- [48] K. Jittorntrum. Solution point differentiability without strict complementarity in nonlinear programming. *Sensitivity, Stability and Parametric Analysis*, 21:127–138, 1984.
- [49] H. K. Khalil. *Nonlinear Systems*. Prentice Hall, 2 sub edition, 1995.
- [50] A. R. Klumpp. Apollo Lunar Descent Guidance. *Automatica*, pages 133–146, 1974.
- [51] D. Kraft. On Converting Optimal Control Problems into Nonlinear Programming Problems. In *Schittkowski K. (eds) Computational Mathematical Programming*, volume 15 of *NATO ASI Series (Series F: Computer and Systems Sciences)*, 1985.
- [52] H. Lampazzi. Intact Ascent Aborts Workbook 21002. Technical report USA007151 Rev A, United Space Alliance, Oct. 2006. Contract NNJ06VA01C.
- [53] G. M. Lee, N. N. Tam, and N. D. Yen. Continuity of the Solution Map in Quadratic Programs under Linear Perturbations. *Journal of Optimization Theory and Applications*, 129(3):415–423, Dec. 2006.
- [54] U. Lee and M. Mesbahi. Constrained Autonomous Precision Landing via Dual Quaternions and Model Predictive Control. *Journal of Guidance, Control, and Dynamics*, 40(2):292–308, 2017.

- [55] C. Leparoux, B. Hérisse, and F. Jean. Structure of optimal control for planetary landing with control and state constraints. *arXiv:2204.06794*, Apr. 2022.
- [56] F. W. Leslie and C. G. Justus. *The NASA Marshall Space Flight Center Earth Global Reference Atmospheric Model, 2010 Version*. National Aeronautics and Space Administration, Marshall Space Flight Center, June 2011.
- [57] P. Lu and S. A. Sandoval. Abort Guidance during Powered Descent for Crewed Lunar Missions. In *AIAA Scitech 2021 Forum*, Virtual Event, Jan. 2021. American Institute of Aeronautics and Astronautics.
- [58] D. Malyuta, T. P. Reynolds, M. Szmuk, T. Lew, R. Bonalli, M. Pavone, and B. Açikmeşe. Convex Optimization for Trajectory Generation. *arXiv:2106.09125*, June 2021.
- [59] O. L. Mangasarian and T.-H. Shiau. Lipschitz Continuity of Solutions of Linear Inequalities, Programs and Complementarity Problems. *SIAM Journal on Control and Optimization*, 25(3):583–595, May 1987.
- [60] Y. Mao, M. Szmuk, and B. Açikmeşe. Successive convexification of non-convex optimal control problems and its convergence properties. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 3636–3641, Dec. 2016.
- [61] Y. Mao, M. Szmuk, X. Xu, and B. Açikmeşe. Successive Convexification: A Superlinearly Convergent Algorithm for Non-convex Optimal Control Problems. *arXiv:1804.06539*, Feb. 2019.
- [62] A. Marwege, J. Riehmer, J. Klevanski, A. Gülhan, T. Ecker, B. Reimann, and E. Dumont. First Wind Tunnel Data of CALLISTO Reusable VTVL Launcher First Stage Demonstrator. page 15, 2019.
- [63] J. Mattingley and S. Boyd. CVXGEN: a code generator for embedded convex optimization. *Optimization and Engineering*, 13(1):1–27, Mar. 2012.
- [64] J. Meditch. On the problem of optimal thrust programming for a lunar soft landing. *IEEE Transactions on Automatic Control*, 9(4):477–484, Oct. 1964.
- [65] H. Ménou, E. Bourgeois, and N. Petit. Fuel-optimal program for atmospheric vertical powered landing. In *60th Conference on Decision and Control*, 2021.
- [66] H. Ménou, E. Bourgeois, and N. Petit. Nominal And Emergency Rocket Landing Guidance Using Quadratic Programming. In *AAS/AIAA Astrodynamics Specialist Conference*, Charlotte, NC, 2022. Univelt Inc. Awaiting proceedings.
- [67] H. Ménou, E. Bourgeois, and N. Petit. Sensitivity Analysis for Powered Descent Guidance: Overcoming degeneracy. In *2022 European Control Conference (ECC)*, pages 1218–1223, 2022.
- [68] R. Montgomery. Abnormal minimizers. *Siam Journal on control and optimization*, 32(6):1605–1620, 1994.

- [69] K. G. Murty. *Linear and Combinatorial Programming*. John Wiley, New York, 1976.
- [70] K. G. Murty. *Linear Programming*. John Wiley & Sons edition, 1983.
- [71] S. Nonaka, H. Nishida, H. Kato, H. Ogawa, and Y. Inatani. Vertical Landing Aerodynamics of Reusable Rocket Vehicle. *Transactions Of The Japan Society For Aeronautical And Space Sciences, ATJ*, 10(0):1–4, 2012.
- [72] A.-I. Onel, O.-I. Popescu, A.-M. Neculaescu, T.-P. Afilipoe, and T.-V. Chelaru. Liquid rocket engine performance assessment in the context of small launcher optimisation. *INCAS BULLETIN*, 11(3):135–145, Sept. 2019.
- [73] H. Oniki. Comparative Dynamics (Sensitivity Analysis) in Optimal Control Theory. *Journal of Economic Theory*, 6:265–283, 1973.
- [74] A. G. Pandala, Y. Ding, and H.-W. Park. qpSWIFT: A real-time sparse quadratic program solver for robotic applications. *IEEE Robotics and Automation Letters*, pages 3355–3362, 2019.
- [75] R. Pytlak. *Numerical methods for optimal control problems with state constraints*. Number 1707 in Lecture Notes in Mathematics. Springer Verlag, 1999.
- [76] A. V. Rao. Trajectory Optimization: A Survey. In H. Waschl, I. Kolmanovsky, M. Steinbuch, and L. del Re, editors, *Optimization and Optimal Control in Automotive Systems*, Lecture Notes in Control and Information Sciences, pages 3–21. Springer International Publishing, Cham, 2014.
- [77] T. P. Reynolds and M. Mesbahi. Optimal Planar Powered Descent with Independent Thrust and Torque. *Journal of Guidance, Control, and Dynamics*, 43(7):1225–1231, July 2020.
- [78] B. Rmili, D. Monchaux, O. Boisneau, J. Hassin, S. Querry, S. Besson, G. Poirey, R. Bore, I. Hamada, H. Amrouchi, J. Franc, M. Barreau, N. Mercadie, T. Labois, and D. Grinco. FROG, a Rocket for GNC demonstrations: Firsts flights attempts of the FROG turbojet version and preparation of the future mono-propellant rocket engine. In *8th European Conference for Aeronautics and Space Sciences*, 2019.
- [79] I. M. Ross and M. Karpenko. A review of pseudospectral optimal control: From theory to flight. *Annual Reviews in Control*, 36(2):182–197, Dec. 2012.
- [80] N. Rouche, P. Habets, and M. Laloy. *Stability Theory by Liapunov’s Direct Method*, volume 4 of *Applied Mathematical Sciences*. Springer, 1977.
- [81] M. Sagliano, A. Heidecker, J. M. Hernández, S. Farì, M. Schlotterer, S. Woicke, D. Seelbinder, and E. Dumont. Onboard Guidance for Reusable Rockets: Aerodynamic Descent and Powered Landing. page 35, Jan. 2021.

- [82] M. Sagliano, T. Tsukamoto, J. A. Macés-Hernández, D. Seelbinder, S. Ishimoto, and E. Dumont. Guidance and Control Strategy for the CALLISTO Flight Experiment. page 13, 2019.
- [83] H. Schättler. The Local Structure of Time-Optimal Trajectories in Dimension Three under Generic Conditions. *SIAM Journal on Control and Optimization*, 26(4):899–918, July 1988.
- [84] H. Schättler and U. Ledzewicz. *Geometric Optimal Control: Theory, Methods and Examples*, volume 38. Springer Science & Business Media, New York, 2012.
- [85] D. Seelbinder. *On-board Trajectory Computation for Mars Atmospheric Entry Based on Parametric Sensitivity Analysis of Optimal Control Problems*. PhD Dissertation, University Bremen, 2017.
- [86] L. Sha. Using simplicity to control complexity. *IEEE Software*, 18(4):20–28, July 2001.
- [87] E. Shapiro and D. Akin. Survivability of Emergency Escape from a Simulated Shuttle Entry Trajectory. In *43rd AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, Jan. 2005. American Institute of Aeronautics and Astronautics.
- [88] Y.-Y. Shi and M. Eckstein. An exact solution for optimum controlled soft lunar landing. *Astronautica Acta*, 16:9–18, 1971.
- [89] Z.-y. Song, C. Wang, S. Theil, D. Seelbinder, M. Sagliano, X.-f. Liu, and Z.-j. Shao. Survey of autonomous guidance methods for powered planetary landing. *Frontiers of Information Technology & Electronic Engineering*, 21(5):652–674, May 2020.
- [90] E. D. Sontag. *Mathematical Control Theory*, volume 6 of *Texts in Applied Mathematics*. Springer New York, New York, NY, 1998.
- [91] R. Sostaric and J. Rea. Powered Descent Guidance Methods For The Moon and Mars. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Francisco, California, Aug. 2005. American Institute of Aeronautics and Astronautics.
- [92] R. R. Sostaric. Powered Descent Trajectory Guidance and Some Considerations for Human Lunar Landing. page 17, Breckenridge, Colorado, 2007. AAS 07-051.
- [93] B. A. Steinfeldt, M. J. Grant, D. A. Matz, R. D. Braun, and G. H. Barton. Guidance, Navigation, and Control System Performance Trades for Mars Pinpoint Landing. *Journal of Spacecraft and Rockets*, 47(1):188–198, Jan. 2010.
- [94] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020.

- [95] H. J. Sussmann. Time-optimal control in the plane. In *Feedback Control of Linear and Nonlinear Systems*, volume 39, pages 244–260. Springer-Verlag, Berlin/Heidelberg, 1982.
- [96] H. J. Sussmann. Lie Brackets and real analyticity in control theory. volume 14, pages 515–542, Warsaw, 1985.
- [97] M. Szmuk, B. Açikmeşe, and A. W. Berning. Successive Convexification for Fuel-Optimal Powered Landing with Aerodynamic Drag and Non-Convex Constraints. In *AIAA Guidance, Navigation, and Control Conference*, San Diego, California, USA, Jan. 2016. American Institute of Aeronautics and Astronautics.
- [98] M. Szmuk, T. P. Reynolds, and B. Açikmeşe. Successive Convexification for Real-Time 6-DoF Powered Descent Guidance with State-Triggered Constraints. *arXiv:1811.10803 [math]*, Nov. 2018. arXiv: 1811.10803.
- [99] E. Trélat. Optimal control and applications to aerospace: some results and challenges. *Journal of Optimization Theory and Applications*, 154(3):713–758, 2012.
- [100] Y. Ulybyshev. Optimization of Three-Dimensional Lunar Landing Trajectories and Accessible Area Computation. In *AIAA SciTech Forum*, page 11, 2019.
- [101] P. Vana, J. Slama, J. Faigl, and P. Paces. Any-Time Trajectory Planning for Safe Emergency Landing. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5691–5696, Madrid, Oct. 2018. IEEE.
- [102] R. Vinter. *Optimal Control*. Birkhäuser Boston, Boston, 2010.
- [103] J. Vlassenbroeck. A chebyshev polynomial method for optimal control with state constraints. *Automatica*, 24(4):499–506, July 1988.
- [104] O. von Stryk. Numerical Solution of Optimal Control Problems by Direct Collocation. In R. Bulirsch, A. Miele, J. Stoer, and K. Well, editors, *Optimal Control: Calculus of Variations, Optimal Control Theory and Numerical Methods*, ISNM International Series of Numerical Mathematics, pages 129–143. Birkhäuser, Basel, 1993.
- [105] K. Wabersich and M. Zeilinger. Safe exploration of nonlinear dynamical systems: A predictive safety filter for reinforcement learning, Dec. 2018.
- [106] J. Wang and N. Cui. A Pseudospectral-Convex Optimization Algorithm for Rocket Landing Guidance. In *2018 AIAA Guidance, Navigation, and Control Conference*, Kissimmee, Florida, Jan. 2018. American Institute of Aeronautics and Astronautics.
- [107] J. Wang, N. Cui, and C. Wei. Optimal Rocket Landing Guidance Using Convex Optimization and Model Predictive Control. *Journal of Guidance, Control, and Dynamics*, 42(5):1078–1092, May 2019.

RÉSUMÉ

Cette thèse étudie le Guidage d'urgence en Descente Propulsée d'un lanceur réutilisable, sous la forme d'un problème de commande optimale en temps final libre sous contraintes. Pour ce lanceur, soumis à de forts effets aérodynamiques et disposant d'une manœuvrabilité limitée, on souhaite calculer en temps réel une trajectoire de « secours », en relâchant certains paramètres négociables, tels que la limite d'incidence, la limite d'accélération normale, ou encore le lieu d'atterrissage. A cet effet, nous introduisons une hiérarchie entre les paramètres et développons un algorithme, Optimisation Hiérarchique pour le Guidage d'Urgence (H.E.G.O.), pour la respecter strictement. L'algorithme consiste en une suite finie de Problèmes Linéaires de négociation, utilisée pour calculer les relaxations nécessaires, suivie d'un Problème Quadratique de raffinement.

Le lanceur est modélisé par huit états et trois commandes. Les paramètres de vol sont les conditions initiales de la fusée, et d'autres paramètres tels que l'Impulsion Spécifique du moteur et le profil de vent. La hiérarchie définie par l'utilisateur est exprimée via un ordre co-lexicographique. La méthode proposée est analysée d'un point de vue théorique. Entre autres, la Lipschitz-continuité de la trajectoire re-planifiée vue comme une fonction des paramètres de vol est établie. Des résultats numériques permettent de quantifier la performance et la qualité de la méthode.

MOTS CLÉS

guidage en descente propulsée, commande optimale, sensibilité de NLP, guidage d'urgence

ABSTRACT

This thesis studies emergency Powered Descent Guidance (PDG) for reusable launchers, as an Optimal Control Problem in free final-time with constraints. For such a launcher, subject to strong aerodynamic effects and having limited maneuverability, we wish to perform « emergency » trajectory planning by relaxing some negotiable parameters, such as the incidence safety bound, the normal acceleration load, or the landing site location. To this end, a hierarchy between the parameters is introduced and an algorithm, Hierarchical Emergency Guidance Optimization (H.E.G.O.), is developed to enforce it. The algorithm consists of a finite sequence of negotiation Linear Programs, followed by a refinement Quadratic Program.

The rocket is modeled by eight states, and three controls. The flight parameters are the initial conditions of the rocket states and other parameters, such as the Engine Specific Impulse and the wind profile. The user-defined hierarchy is conveyed via a co-lexicographic order. The methodology is theoretically studied. Among others, the Lipschitz-continuity of the guidance trajectory with respect to the input flight parameters is established. Extensive numerical results serve to quantify the performance and relevance of the methodology.

KEYWORDS

powered descent guidance, optimal control, NLP sensitivity, emergency guidance