MULTISTAGE LAUNCH VEHICLE DESIGN WITH THRUST PROFILE AND TRAJECTORY OPTIMIZATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

EZGİ CİVEK COŞKUN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
MECHANICAL ENGINEERING

SEPTEMBER 2014

Approval of the thesis:

MULTISTAGE LAUNCH VEHICLE DESIGN WITH THRUST PROFILE AND TRAJECTORY OPTIMIZATION

submitted by **EZGİ CİVEK COŞKUN** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Mechanical Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____________________
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Süha Oral _____________________
Head of Department, **Mechanical Engineering**

Prof. Dr. M. Kemal Özgören _____________________
Supervisor, **Mechanical Engineering Department, METU**

Examining Committee Members:

Prof. Dr. Reşit Soylu _____________________
Mechanical Engineering Department, METU

Prof. Dr. M. Kemal Özgören _____________________
Mechanical Engineering Department, METU

Prof. Dr. Tuna Balkan _____________________
Mechanical Engineering Department, METU

Prof. Dr. Ozan Tekinalp _____________________
Aerospace Engineering Department, METU

Prof. Dr. Yücel Ercan _____________________
Mechanical Engineering Department, TOBB-ETU

Date: 05.09.2014

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Ezgi CİVEK COŞKUN
Signature :

ABSTRACT

MULTISTAGE LAUNCH VEHICLE DESIGN WITH THRUST PROFILE AND TRAJECTORY OPTIMIZATION

Civek Coşkun, Ezgi
Ph.D., Department of Mechanical Engineering
Supervisor: Prof. Dr. M. Kemal Özgören

September 2014, 198 pages

In the frame of this thesis, a Matlab® based design tool utilizing a general purpose optimal control solver GPOPS-II® was developed for the optimization of ascent trajectories of multistage launch vehicles. This tool can be utilized both for preliminary mission design of an existing launch vehicle and conceptual design of a new launch vehicle.

In the preliminary mission design of an existing launch vehicle, maximizing payload mass is a very common performance measure. Therefore, a trajectory optimization code was developed to determine the optimal trajectory in terms of position and velocity, and the control functions corresponding to that trajectory while maximizing the payload mass.

Whereas in the conceptual design of a new launch vehicle, minimization of gross lift-off mass for a specific mission is the primary objective, and staging is an effective way to reduce the vehicle gross lift-off mass. For this purpose, a staging optimization code was developed to determine the staging parameters (mass distribution between stages, propellant and structural mass of each individual stage) which minimize the gross lift-off mass.

In this thesis, staging and trajectory optimization codes were integrated by two different manners, namely the decoupled and the coupled approaches, so as to determine the sizing, burn time and thrust time history of each stage together with the optimal trajectory of the launch vehicle. Thus, a quick and effective method to find optimal vehicle configurations in the early design phases was achieved.

Keywords: Launch vehicle, staging optimization, trajectory optimization, thrust profile optimization, GPOPS-II

ÖZ

İTKİ PROFİLİ VE YÖRÜNGE ENİYİLEMESİ İLE ÇOK KADEMELİ FIRLATMA ARACI TASARIMI

Civek Coşkun, Ezgi
Doktora, Makina Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. M. Kemal Özgören

Eylül 2014, 198 sayfa

Bu tez kapsamında, çok kademeli fırlatma araçlarının yükseliş yörüngesinin eniyilenmesine yönelik olarak GPOPS-II® genel amaçlı optimal kontrol çözücüsünü kullanan Matlab® tabanlı bir tasarım aracı geliştirilmiştir. Bu araç, hem varolan bir fırlatma aracının yörünge tasarımında hem de yeni bir fırlatma aracının kavramsal tasarımında kullanılabilecektir.

Varolan bir fırlatma aracının yörünge tasarımında, faydalı yük kütlesinin ençoklanması çok yaygın bir başarım ölçüsüdür. Bu sebeple, faydalı yük kütlesini ençoklayan konum ve hız cinsinden eniyi yörüngenin ve bu yörüngeye ilişkin kontrol fonksiyonlarının belirlenmesi için bir yörünge eniyileme kodu geliştirilmiştir.

Yeni bir fırlatma aracının kavramsal tasarımında ise, belirli bir görev için toplam kalkış ağırlığının enazlanması birincil amaç olup, bunun en etkili yolu kademelendirmedir. Bu amaçla, kademelendirme parametrelerinin (kütlenin kademeler arasındaki dağılımı, her bir kademenin yakıt ve yapısal ağırlığı) belirlenmesi için bir kademelendirme eniyileme kodu geliştirilmiştir.

Bu tezde, kademelendirme ve yörünge eniyilemesi kodları ayrıştırılmış ve bağlaşık olmak üzere iki farklı yaklaşımla fırlatma aracının eniyi yörüngesi ile birlikte her bir kademenin ağırlığı, yanma süresi ve itki zaman eğrisi belirlenecek şekilde birleştirilmiştir. Böylece, kavramsal tasarım aşamasında eniyi fırlatma aracı konfigürasyonunun bulunması için hızlı ve etkili bir yöntem elde edilmiştir.

Anahtar Kelimeler: Fırlatma aracı, kademelendirme eniyilemesi, yörünge eniyilemesi, itki profili eniyilemesi, GPOPS-II

To My Parents and My Grandmother

ACKNOWLEDGMENTS

The first and the foremost, I would like to thank my thesis supervisor Prof. Dr. M. Kemal Özgören, who always makes time to listen to my studies in his busy schedule, for providing useful advices and supporting my efforts.

I am extremely grateful to Assoc. Prof. Dr. Anil Rao, who is the principal developer of the optimal control software GPOPS-II, for providing us with this great optimization tool and for his quick and helpful suggestions.

I am also thankful to my first promoter Başar Seçkin for introducing me to the field of space rockets. His strong will and determination was the source of my inspiration. I would also like to express my sincere appreciation to Dr. Mehmet Ali Ak for his understanding and encouragement throughout the completion of this thesis work. And, I would like to give special thanks to Dr. Atılgan Toker, the single audience in my thesis defense. I will always remember his concern and sincerity.

Love and thanks to my dear mom, dad and grandma for raising me as who I am today. The work presented in this thesis could not have been accomplished without their never-ending support and encouragement. I also would like to thank my brother Barış for his unique sense of humor.

The last but not the least, to love of my life, my husband, my best friend, my sweetheart Korhan Coşkun: I am indebted to you for your help in both debugging the computer code and editing the draft, and for your numerous ideas and suggestions that contribute to enhancing this thesis work. Most of all, thank you for being there for me all these years, encouraging me to face the challenges.

Ankara, 5 September 2014

TABLE OF CONTENTS

ABSTRACT v

ÖZ vi

ACKNOWLEDGMENTS viii

TABLE OF CONTENTS ix

LIST OF TABLES xiii

LIST OF FIGURES xv

LIST OF SYMBOLS xx

LIST OF ABBREVIATIONS xxv

CHAPTERS xxvi

1 INTRODUCTION 1

- 1.1 Motivation 2
- 1.2 Scope and Objectives 4
- 1.3 Thesis Outline 6

2 BACKGROUND 7

- 2.1 Trajectory Optimization Methods 7
  - 2.1.1 Numerical Methods for Optimal Control 9
  - 2.1.2 Nonlinear Programming Methods 11
- 2.2 Trajectory Optimization Software 12
  - 2.2.1 Program to Optimize Simulated Trajectories (POST) 12
  - 2.2.2 Optimal Trajectories by Implicit Simulation (OTIS) 12
  - 2.2.3 Sparse Optimal Control Software (SOCS) 13
  - 2.2.4 AeroSpace Trajectory Optimization Software (ASTOS) 13
- 2.3 Previous Research 13
  - 2.3.1 Staging Optimization 14
  - 2.3.2 Trajectory Optimization 16

3 STAGING OPTIMIZATION 23

- 3.1 Mathematical Model 23
  - 3.1.1 Tsiolkovsky’s Rocket Equation 23
  - 3.1.2 Multistage Rocket Parameters 24

3.1.2.1 Serial Staging 26
3.1.2.2 Parallel Staging 27
3.1.3 Orbital Velocity Equation 28
3.1.4 Delta-V Calculations 28
3.2 Problem Formulation 31
3.3 Method of Solution 32
3.4 Results and Discussion 35
4 TRAJECTORY OPTIMIZATION 41
4.1 Problem Formulation 41
4.1.1 System Dynamics 41
4.1.1.1 Trajectory Model 41
4.1.1.2 Propulsion Model 45
4.1.1.3 Aerodynamics Model 48
4.1.1.4 Earth and Gravity Models 49
4.1.1.5 Atmosphere Models 51
4.1.2 Objective Functions 52
4.1.3 Optimization Variables 53
4.1.4 Constraints 54
4.1.4.1 Dynamic Constraints 54
4.1.4.2 Boundary Constraints 54
4.1.4.2.1. Initial Constraints 54
4.1.4.2.2. Linkage Constraints 55
4.1.4.2.3. Terminal Constraints 55
4.1.4.3 Path Constraints 56
4.1.4.3.1. Dynamic Pressure 57
4.1.4.3.2. Angle of Attack 57
4.1.4.3.3. Bending Moment 57
4.1.4.3.4. Axial Acceleration 57
4.1.4.3.5. Aerothermal Heat Flux 58
4.1.4.3.6. Total Aerodynamic Heating Rate 58
4.1.5 GPOPS-II Implementation 58
4.2 Example 1: Delta III – GTO Mission 60
4.2.1 Vehicle Properties 60
4.2.2 Mission Characteristics 61
4.2.3 Problem Formulation 62
4.2.4 Bounds and Guesses 65
4.2.5 Results and Discussion 66

4.3 Vega Missions 70
    4.3.1 Vehicle Properties 70
    4.3.2 Mission Characteristics 73
    4.3.3 Bounds and Guesses 77
    4.3.4 Example 2: Vega – LETO Mission 78
    4.3.5 Example 3: Vega – PEO Mission 89
    4.3.6 Example 4: Vega – SSO Mission 99
4.4 Example 5: Taurus – SSO Mission 104
    4.4.1 Vehicle Properties 104
    4.4.2 Mission Characteristics 106
    4.4.3 Results and Discussion 107
5 LAUNCH VEHICLE DESIGN 113
5.1 Decoupled Approach 114
    5.1.1 Problem Formulation 114
    5.1.2 Example 1: Delta III 116
    5.1.3 Example 2: Vega 118
    5.1.4 Discussion 120
5.2 Coupled Approach 122
    5.2.1 Problem Formulation 123
    5.2.2 Example 1: Vega 124
        5.2.2.1 Constant Thrust 124
        5.2.2.2 Variable Thrust 126
        5.2.2.3 Discussion 132
    5.2.3 Example 2: Taurus 133
6 CONCLUSIONS 137
6.1 Thesis Summary and Conclusions 137
6.2 Future Developments 139
REFERENCES 141
APPENDICES 149
A SATELLITE ORBITS 151
    A.1 Orbital Elements 152
    A.2 Orbital Position, Velocity and Period 154
    A.3 Orbit Types 155
    A.4 Orbit Insertion 156
B COORDINATE SYSTEMS & TRANSFORMATIONS 159

B.1 Coordinate Systems 159
B.1.1 Rectangular ECI Coordinate System 159
B.1.2 Spherical ECI Coordinate System 160
B.1.3 Spherical ECEF Coordinate System 161
B.1.4 Geographic Coordinate System 162
B.1.5 Orbital Coordinate System 163
B.1.6 Summary 164
B.2 Coordinate Transformations 165
B.2.1 Rectangular ECI to Spherical ECI and vice versa 165
B.2.2 Spherical ECI to Spherical ECEF and vice versa 166
B.2.3 Geocentric to Geodetic and vice versa 169
B.2.4 Orbital to Rectangular ECI and vice versa 170
C MATLAB IMPLEMENTATION 173
C.1 Coordinate Transformations 173
C.1.1 Rectangular ECI to Spherical ECI and vice versa 173
C.1.2 Spherical ECI to Spherical ECEF and vice versa 175
C.1.3 Spherical ECEF to Geographic and vice versa 177
C.1.4 Orbital to Rectangular ECI and vice versa 177
C.2 Staging Optimization 179
C.3 Trajectory Optimization 181
C.3.1 Dynamic Equations 189
C.3.2 Terminal Constraint and Objective Function 191
C.3.3 Descent Trajectories of Separated Stages 192
D SOLID PROPELLANT ROCKET MOTORS 193
CURRICULUM VITAE 195

LIST OF TABLES

TABLES

Table 1.1 Use of trajectory optimization in two different tasks 4
Table 2.1 Comparison of indirect and direct methods.. 9
Table 3.1 Practical I_{sp} values for common rocket propellants 24
Table 3.2 Numerical example for staging optimization for given $\Delta V_{mission}$ 36
Table 3.3 GSLV staging data 37
Table 4.1 Parameters of WGS 84 Earth model 51
Table 4.2 Mass and propulsion characteristics of Delta III 61
Table 4.3 Delta III GTO flight sequence 61
Table 4.4 Delta III target orbit parameters 67
Table 4.5 Physical and performance characteristics of Vega 71
Table 4.6 Typical flight sequence of Vega 74
Table 4.7 Vega – LETO mission flight phases summary 80
Table 4.8 Vega – LETO mission target orbit parameters 80
Table 4.9 Vega – LETO mission AVUM propellant budget 81
Table 4.10 Vega – LETO mission separable parts impact points 85
Table 4.11 Vega – PEO mission flight phases summary 91
Table 4.12 Vega – PEO mission target orbit parameters 92
Table 4.13 Vega – PEO mission AVUM propellant budget 92
Table 4.14 Vega – PEO mission separable parts impact points 97
Table 4.15 Vega – SSO mission flight phases summary 100
Table 4.16 Vega – SSO mission target orbit parameters 100
Table 4.17 Vega – SSO mission AVUM propellant budget 100
Table 4.18 Vega – SSO mission separable parts impact points 103
Table 4.19 Physical and performance characteristics of Taurus 105
Table 4.20 Taurus flight sequence 107
Table 4.21 Taurus – SSO mission flight phases summary 108
Table 4.22 Taurus – SSO mission target orbit parameters 108
Table 5.1 Comparison of Delta III and 4SLV 116

Table 5.2 Comparison of 2SLV and Delta III 117
Table 5.3 Optimal vehicle configuration for PEO mission (decoupled) 119
Table 5.4 Decoupled vs. coupled approach for PEO mission (T_{const} & t_b: fixed) 121
Table 5.5 Optimal vehicle configuration for PEO mission (coupled – T_{const}) 124
Table 5.6 Optimal solution for PEO mission (constrained coupled – $T(t)$) 127
Table 5.7 Optimal solution for PEO mission (unconstrained coupled – $T(t)$) 127
Table 5.8 Mass comparison between optimal vehicle and actual Vega 129
Table 5.9 Delta-V and speed comparison between optimal vehicle and Vega 129
Table 5.10 Delta-V comparison between Vega, optimal vehicle and ideal vehicle 131
Table 5.11 Mass comparison between Vega, optimal vehicle and ideal vehicle 132
Table 5.12 Comparison of optimal vehicle configurations and Vega for PEO mission 132
Table 5.13 Optimal vehicle configuration for SSO mission of Taurus 134
Table 5.14 Mass comparison between optimal vehicle and actual Taurus 134
Table A.1 Special cases of orbits 154
Table A.2 Types of satellite orbits 156
Table B.1 Summary of reference frames 164
Table B.2 Summary of trajectory variables 164
Table B.3 Range of trajectory variables 164

LIST OF FIGURES

FIGURES

Figure 1.1 United States current launch vehicles 1
Figure 1.2 Launch vehicle's trajectory on its way to orbit 2
Figure 2.1 Classification of direct methods 9
Figure 2.2 Shooting (a) and collocation (b) methods 10
Figure 2.3 Classification of NLP methods 11
Figure 2.4 Types of staging (a) serial, (b) parallel 14
Figure 3.1 Mass definitions for serial staging 25
Figure 3.2 Mass definitions for parallel staging 27
Figure 3.3 ΔV losses vs. T/W 30
Figure 3.4 Velocity gain due to Earth's rotation 31
Figure 3.5 Staging optimization code description 35
Figure 3.6 Variation of GLOM with number of stages 38
Figure 3.7 Variation of GLOM with $\Delta V_{mission}$ 38
Figure 3.8 Variation of GLOM with payload mass 39
Figure 4.1 Euler angles 43
Figure 4.2 Pitch (θ), angle of attack (α) and flight path angle (γ) 44
Figure 4.3 Delta III launch vehicle 60
Figure 4.4 Delta III flight sequence 61
Figure 4.5 Delta III position profile 66
Figure 4.6 Delta III inertial velocity profile 66
Figure 4.7 Delta III control time history 67
Figure 4.8 Delta III mass profile 67
Figure 4.9 Delta III altitude profile 68
Figure 4.10 Delta III speed profile 68
Figure 4.11 Delta III latitude vs. longitude 68
Figure 4.12 Delta III altitude versus range 68
Figure 4.13 Delta III flight path angle 69
Figure 4.14 Delta III heading angle profile 69

Figure 4.15 Delta III acceleration profile 69
Figure 4.16 Delta III dynamic pressure profile 69
Figure 4.17 Vega launch vehicle 70
Figure 4.18 Thrust and propellant mass profiles of solid rocket motors of Vega 71
Figure 4.19 Aerodynamic configurations of Vega 72
Figure 4.20 Drag coefficients of Vega 73
Figure 4.21 Vega flight sequence 75
Figure 4.22 LETO position profile 79
Figure 4.23 LETO inertial velocity profile 79
Figure 4.24 LETO mass time history 79
Figure 4.25 LETO control time history 79
Figure 4.26 LETO perigee & apogee heights 81
Figure 4.27 LETO orbit inclination profile 81
Figure 4.28 LETO speed profile 83
Figure 4.29 LETO relative speed comparison 83
Figure 4.30 LETO altitude comparison 84
Figure 4.31 LETO launch vehicle’s ground track comparison 84
Figure 4.32 LETO separated stages impact points 85
Figure 4.33 LETO altitude vs. downrange (incl. separated stages) 85
Figure 4.34 LETO thrust profile 86
Figure 4.35 LETO acceleration profile 86
Figure 4.36 LETO density profile 86
Figure 4.37 LETO Mach number profile 86
Figure 4.38 LETO dynamic pressure profile 87
Figure 4.39 LETO drag force profile 87
Figure 4.40 LETO FPA & pitch profile 87
Figure 4.41 LETO heading & yaw profile 87
Figure 4.42 LETO angle of attack profile – constrained vs. unconstrained 88
Figure 4.43 LETO bending moment profile 88
Figure 4.44 LETO heat flux profile 88
Figure 4.45 PEO TO altitude profile 90
Figure 4.46 PEO TO speed profile 90
Figure 4.47 PEO mass time history 93
Figure 4.48 PEO relative speed comparison 93

Figure 4.49 PEO altitude comparison 93
Figure 4.50 PEO flight path angle profile comparison 94
Figure 4.51 PEO pitch angle profile comparison 95
Figure 4.52 PEO angle of attack profile comparison 95
Figure 4.53 PEO a, r_a, r_p comparison 96
Figure 4.54 PEO eccentricity comparison 96
Figure 4.55 PEO inclination comparison 96
Figure 4.56 PEO true anomaly comparison 96
Figure 4.57 PEO launch vehicle’s ground track comparison 97
Figure 4.58 PEO separated stages impact points 97
Figure 4.59 PEO altitude vs. downrange (incl. separated stages) 98
Figure 4.60 PEO bending moment profile 98
Figure 4.61 PEO heat flux profile 98
Figure 4.62 PEO acceleration profile 99
Figure 4.63 PEO dynamic pressure profile 99
Figure 4.64 SSO mass time history 101
Figure 4.65 SSO relative speed comparison 101
Figure 4.66 SSO altitude comparison 101
Figure 4.67 SSO FPA profile 101
Figure 4.68 SSO pitch angle profile 101
Figure 4.69 SSO angle of attack profile 102
Figure 4.70 SSO bending moment profile 102
Figure 4.71 SSO heat flux profile 102
Figure 4.72 SSO acceleration comparison 102
Figure 4.73 SSO dynamic pressure profile 102
Figure 4.74 SSO launch vehicle’s ground track comparison 103
Figure 4.75 SSO separated stages impact points 103
Figure 4.76 Taurus launch vehicle 104
Figure 4.77 Thrust and propellant mass profiles of solid rocket motors of Taurus 105
Figure 4.78 Taurus flight sequence 106
Figure 4.79 Taurus altitude comparison 109
Figure 4.80 Taurus speed comparison 109
Figure 4.81 Taurus altitude-range comparison 109
Figure 4.82 Taurus mass profile 109

Figure 4.83 Taurus ground track comparison ... 110

Figure 4.84 Taurus 1st & 2nd impact points ... 110

Figure 4.85 Taurus separated 3rd stage ground track and impact point ... 110

Figure 4.86 Taurus thrust profile ... 110

Figure 4.87 Taurus acceleration profile ... 110

Figure 4.88 Taurus dynamic pressure profile ... 111

Figure 4.89 Taurus drag force profile ... 111

Figure 4.90 Taurus bending moment profile ... 111

Figure 4.91 Taurus heat flux profile ... 111

Figure 4.92 Taurus FPA & pitch profile ... 112

Figure 4.93 Taurus angle of attack profile ... 112

Figure 4.94 Taurus perigee-apogee heights ... 112

Figure 4.95 Taurus inclination profile ... 112

Figure 5.1 Schematic for the decoupled approach ... 114

Figure 5.2 Delta-4SLV altitude comparison ... 117

Figure 5.3 Delta-4SLV speed comparison ... 117

Figure 5.4 2SLV acceleration profile ... 118

Figure 5.5 2SLV dynamic pressure profile ... 118

Figure 5.6 Altitude profile (Vega decoupled) ... 119

Figure 5.7 Speed profile (Vega decoupled) ... 119

Figure 5.8 Thrust profile (Vega decoupled) ... 120

Figure 5.9 Acceleration (Vega decoupled) ... 120

Figure 5.10 Dyn. pressure (Vega decoupled) ... 120

Figure 5.11 Bend. moment (Vega decoupled) ... 120

Figure 5.12 Schematic for the coupled approach ... 122

Figure 5.13 Thrust time curve modeling ... 123

Figure 5.14 Altitude profile (Vega coupled) ... 125

Figure 5.15 Speed profile (Vega coupled) ... 125

Figure 5.16 Thrust profile (Vega coupled) ... 125

Figure 5.17 Acceleration (Vega coupled) ... 125

Figure 5.18 Dyn. pressure (Vega coupled) ... 126

Figure 5.19 Bending moment (Vega coupled) ... 126

Figure 5.20 Altitude (cons. vs. unc.) ... 128

Figure 5.21 AOA (cons. vs. unc.) ... 128

Figure 5.22 Thrust (cons. vs. unc.) 128
Figure 5.23 Acceleration (cons. vs. unc.) 128
Figure 5.24 Dyn. pressure (cons. vs. unc.) 128
Figure 5.25 Bending moment (cons. vs. unc.) 128
Figure 5.26 Thrust profile comparison of 1st stage SRM 130
Figure 5.27 Thrust profile comparison of 2nd stage SRM 130
Figure 5.28 Thrust profile comparison of 3rd stage SRM 131
Figure 5.29 Thrust comparison of 1st stage 135
Figure 5.30 Thrust comparison of 2nd stage 135
Figure 5.31 Thrust comparison of 3rd stage 135
Figure 5.32 Thrust comparison of 4th stage 135
Figure 5.33 Altitude (Taurus coupled) 135
Figure 5.34 Acceleration (Taurus coupled) 135
Figure 5.35 Dyn. pressure (Taurus coupled) 136
Figure 5.36 Bend. moment (Taurus coupled) 136
Figure A.1 Elliptical orbit 151
Figure A.2 Orbital elements (i, Ω, ω, θ^*) 152
Figure A.3 Orbital injection 157
Figure B.1 Rectangular ECI coordinate system 160
Figure B.2 Spherical ECI coordinate system 161
Figure B.3 Spherical ECEF coordinate system 162
Figure B.4 Geographic coordinate system 163
Figure B.5 Orbital frame 163
Figure B.6 Difference in longitude definitions in ECI and ECEF frames 167
Figure D.1 SRM grain cross sections and influence on thrust profile 193
Figure D.1 Burn time of SRMs used in launch vehicles 194

LIST OF SYMBOLS

Latin Letters

A	Velocity azimuth angle
A_e	Nozzle exit area
a	Semimajor axis or
	Speed of sound or
	Axial acceleration (depending on the context)
C	Effective exhaust velocity
C_D	Drag coefficient
c	Half length between the two foci of the ellipse
D	Drag force
d	Geodetic ground distance, i.e. range
e	Orbit eccentricity
e_e	First eccentricity of the reference ellipsoid for the Earth
g_0	Gravitational acceleration at sea level
$\mathbf{h}$	Angular momentum vector
h	Geocentric altitude
h_a	Apogee altitude
h_p	Perigee altitude
$\mathbf{I}$, $\mathbf{J}$, $\mathbf{K}$	Unit vectors in inertial frame
I_{sp}	Vacuum specific impulse
$\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$	Unit vectors in relative frame
i	Orbit inclination
k	Specific heat ratio for air
M	Mach number
m	Mass

$\dot{m}$	Mass flow rate
m_0	Initial mass
m_f	Final mass
m_p	Propellant mass
m_{pl}	Payload mass
m_{plf}	Payload fairing mass
m_s	Structural mass
N	Number of stages
$\mathbf{n}$	Ascending node vector
P	Number of phases or Pressure (depending on the context)
P_a	Ambient static pressure, i.e. the atmospheric pressure
P_e	Static pressure at the nozzle exit
p	Lagrange multiplier
q	Dynamic pressure
$q\alpha$	Bending moment indicator
$\mathbf{R_1}$, $\mathbf{R_2}$, $\mathbf{R_3}$	Elementary rotation matrices around three orthogonal axes
R	Earth radius at a given geocentric latitude considering the oblateness or Specific gas constant for air (depending on the context)
R_e	Equatorial radius of the Earth
R_m	Earth’s mean radius
$\mathbf{r}$	Position vector
r	Geocentric radius, i.e. radial distance from the Earth’s center to the point of interest
$\tilde{r}$	Radius ratio
r_a	Apogee radius
$\tilde{r}_a$	Apogee radius ratio
r_p	Perigee radius
$\tilde{r}_p$	Perigee radius ratio
S	Aerodynamic reference area

T	Thrust or Temperature (depending on the context)
T_{ave}	Average thrust
T_{vac}	Thrust in vacuum
TR	Maximum over minimum thrust ratio
T/W	Thrust-to-weight ratio
t	Time
$\mathbf{u}$	Control vector, i.e. thrust direction
V_{circ}	Circular orbit velocity at a given altitude
V_e	Exhaust velocity at the nozzle exit
V_{orbit}	Orbital velocity
$\mathbf{v}$	Velocity vector
v_i	Inertial speed
v_{rel}	Relative speed with respect to the Earth
$\mathbf{x}$	States

Greek Letters

α	Angle of attack
β	Sideslip angle
γ	Flight path angle
γ_i	Inertial flight path angle
δ	Declination, i.e. geocentric latitude or Thrust deflection angle (depending on the context)
ΔV	Velocity change
ΔV_d	Velocity loss due to aerodynamic drag
ΔV_g	Velocity loss due to gravity
ΔV_{gain}	Velocity gain due to Earth’s rotation or initial altitude or initial velocity
ΔV_i	Initial velocity gain due to Earth’s rotation or initial altitude
$\Delta V_{mission}$	Total velocity increment required to reach orbit

ΔV_p	Velocity loss due to propulsive steering and nozzle efficiency
$\Delta V_{vehicle}$	Velocity change provided by the vehicle
ε	Structural ratio
ζ	Propellant ratio
θ	Pitch angle
θ^*	True anomaly
Λ	Burn-out mass ratio
λ	Payload ratio or Geodetic longitude (depending on the context)
λ_i	Right ascension, i.e. inertial longitude
μ	Earth's gravitational parameter
ρ	Atmospheric density
τ	Orbital period
ϕ	Geodetic latitude
χ	Heading angle
ψ	Yaw angle
Ω	Right ascension of the ascending node
ω	Argument of perigee
ω_e	Earth's angular rotation rate

Subscripts

0	Initial
b	Booster
c	Core stage
f	Final
k	Index for stage number
max	Maximum
x, *y*, *z*	First, second and third coordinates in rectangular ECI frame

Superscripts

T	Transpose

Note: Vectors and matrices are represented by bold case characters.

LIST OF ABBREVIATIONS

2PBVP	Two point boundary value problem
3DOF	Three degrees of freedom
6DOF	Six degrees of freedom
AOA	Angle of attack
ASTOS	Aerospace trajectory optimization software
AVUM	Attitude and vernier upper module (Vega launch vehicle liquid upper stage)
ECEF	Earth-centered Earth-fixed
ECI	Earth-centered-inertial
ENU	East-north-up
FPA	Flight path angle
GEO	Geostationary orbit
GLOM	Gross lift-off mass
GPOPS-II	General purpose Matlab toolbox for solving optimal control problems
GTO	Geostationary transfer orbit
IP	Impact point of a separated part
LEO	Low Earth orbit
LETO	Low Earth elliptical transfer orbit
LV	Launch vehicle
MDO	Multidisciplinary design optimization
ME	Main engine
MEO	Medium Earth orbit
NLP	Nonlinear programming problem
OCP	Optimal control problem
OTIS	Optimal trajectories by implicit simulation
PEO	Polar Earth orbit

PLF	Payload fairing
PLSF	Payload scaling factor
POST	Program to optimize simulated trajectories
RAAN	Right ascension of the ascending node
RPM	Radau pseudospectral method
SNOPT	Sparse nonlinear optimizer
SOCS	Sparse optimal control software
SQP	Sequential quadratic programming
SRM	Solid propellant rocket motor
SSO	Sunsynchronous orbit
SSTO	Single-stage-to-orbit
TO	Transfer orbit

CHAPTER 1

INTRODUCTION

Since the launch of first artificial satellite *Soviet Sputnik 1* in 1957, thousands of payloads have been placed into Earth orbits using space launch vehicles. The payload can be anything from satellites to astronauts or from scientific instruments to cargo depending on the mission.

In the mid-1950s, the Soviet Union and the United States were the only countries having space launch capability, but today several other nations (Europe, China, Ukraine, Japan, India, Israel, Brazil, North Korea and Iran) have produced their own launch vehicles. Some of the launch vehicles developed by the United States for space access and exploration are illustrated by size and shape in Figure 1.1.

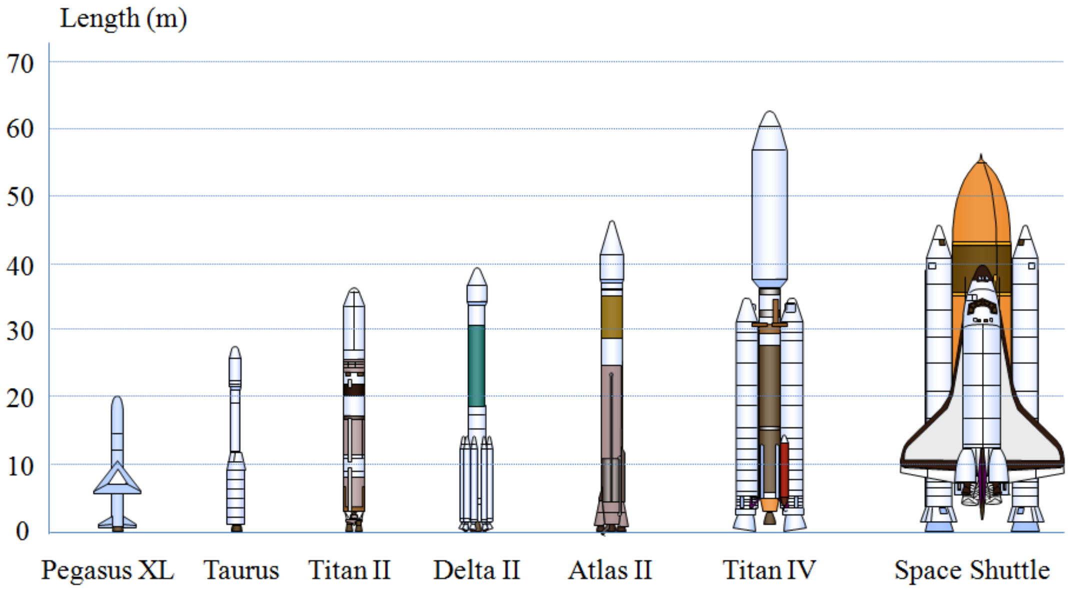


Figure 1.1 United States current launch vehicles
(adapted from http://www.docstoc.com/docs/53998514/Launch-Vehicles)

Launch vehicles used in space applications are mostly multistage rockets composed of two or more stages, each of which contains its own propellant and structure. Stages are stacked on top of (serial or tandem staging) or attached next to (parallel staging) each other and they are numbered as the order of firing.

The idea behind staging is to improve performance by reducing the vehicle's mass on the way to orbit. Once the propellant of a stage is consumed, the empty stage which is no longer useful and only adds weight to the vehicle is discarded and the next stage is ignited. This stage then accelerates the rest of the vehicle much faster. As a result, less propellant is required to reach the desired orbit.

Space launch vehicles deliver payloads into orbit by moving so fast that they can overcome Earth's gravity and after they reach the desired altitudes they release the payloads at very high speeds. Almost all rocket powered launch vehicles are launched vertically from the ground and they usually try to get out of the dense atmosphere as soon as possible, since the atmosphere increases drag losses. However, when the ascent is too steep, energy is spent to lift the propellant to high altitudes, which causes gravity losses. Therefore, launch vehicles turn down gradually to have almost horizontal acceleration after leaving the atmosphere.

The path followed by the launch vehicle from the ground to orbit is called the ascent trajectory (Figure 1.2). Ascent trajectory depends on the location of the launch site, propulsive, gravitational and aerodynamic forces acting on the vehicle during flight, safety and operational constraints on the trajectory and the parameters of the injection orbit that the vehicle tries to achieve. Small changes in the ascent trajectory can have significant effects on the payload mass that can be delivered into specified orbit, as well as on the design of the vehicle and consequently the production and operating costs. Therefore, trajectory optimization is required to achieve the best performance. Brief overview of satellite orbits are given for reference in APPENDIX A.

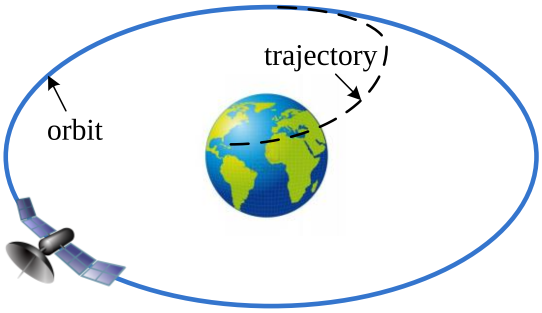


Figure 1.2 Launch vehicle's trajectory on its way to orbit

1.1 Motivation

Design of a launch vehicle is a very challenging activity because of highly complex and interconnected nature of the launch vehicles. In the early phases of design, various vehicle concepts are first proposed for a specific mission, next an optimum configuration for each concept is found and finally the best configuration to proceed with the preliminary design is selected by evaluating the performance of the configuration alternatives.

Launch vehicle performance is greatly influenced by both the vehicle characteristics and its flight trajectory. Therefore, system level engineering tools optimizing the vehicle design and the trajectory simultaneously are needed to predict the performance capabilities of the launch vehicle configurations. These tools rely on extensive numerical modeling, simulation and optimization.

In principle, vehicle design and trajectory optimization problem can be solved by collecting all elements of the trajectory control vector and system design variables in one vector of optimization parameters to be manipulated by an appropriate optimization algorithm. However, system design is a multidisciplinary process in which aerodynamics, propulsion, weights and sizing, trajectory and even reliability and cost must be addressed (Stanley, et al., 1994). For multistage launch vehicles, resulting search space is extremely large with continuous and discrete variables which may cause convergence problems even major system design parameters such as mass split of stages or engine sizing are included.

Furthermore, during conceptual design phase, requirements are not so strict and there are many unknowns. Therefore, low-fidelity models are implemented and relationships among design objectives and design parameters are not well-defined, which may end up with inefficient designs. Rowell and Korte (2003) suggested three approaches for improving results. One is the improvement in computational methods, the other is the continued improvement of disciplinary analysis models and tools to capture sufficient fidelity, and the last one is the development of methods for integrating the disciplines and optimizing the system as a whole. They also pointed out that multidisciplinary design optimization plays an important role in identifying a near-optimum design for each concept so that subsystem level decision makers use this system level information to make choices in smaller design spaces.

A few books or book chapters have been written about the launch vehicle design (Griffin & French, 2004; Moss, 1995; Loftus & Teixeira, 1999; Peters, 2004; Ndefo, Encalada, Hallman, Wang, & Abbott, 2004; Pisacane, 2005). These books give many details about how to construct analysis models for each of the different disciplines. However, none of these describe how to create an integrated environment that incorporates all of the disciplines required for system level conceptual design of a launch vehicle. Only Hammond (1999 & 2001) describes the data flow between the disciplines and discusses the system level collaborative optimization in his books. Nevertheless, it is left to the system designer to develop his/her own approach for integrating required disciplines and to determine the details of implementing these approaches.

In the market on the other hand, there are a number of powerful trajectory optimization software packages, which offer simultaneous optimization of ascent trajectory and launch vehicle design. However, these software packages have been developed by governmental agencies or universities for national space projects and their sale to other countries is restricted due to export control regulations. Therefore, there is still a growing demand for an integrated and efficient system design and trajectory optimization tool for vehicle performance calculations and simulations.

1.2 Scope and Objectives

Main objective of this thesis is to develop a trajectory optimization tool well suited to be used in the following tasks.

- Preliminary mission design of an existing launch vehicle (trajectory optimization of a fixed system)
- Conceptual design of a new launch vehicle (simultaneous optimization of system and trajectory)

Use of trajectory optimization in above mentioned tasks in terms of inputs and outputs are tabulated in Table 1.1.

Table 1.1 Use of trajectory optimization in two different tasks

Trajectory optimization of a fixed system	
Inputs: All physical and performance parameters of the launch vehicle are known. • number and type of stages • propellant and structural masses • thrust vs. time • mass vs. time	**Outputs:** Only trajectory control variables are optimized. • thrust direction vs. time • free flight time durations • thrust profile for the stages having throttling capability
Simultaneous optimization of system and trajectory	
Inputs: Physical and performance parameters are either specified by design team considering the vehicle concept, operational scenarios and technology options or calculated from a statistical database of existing launch vehicles. • specific impulses of stages • structural ratios of stages • throttling and restart capabilities of stages	**Outputs:** Both system and trajectory variables are optimized. • number and type of stages • propellant and structural masses of stages • thrust profiles for all stages • thrust direction vs. time • free flight time durations

In the conceptual design of a launch vehicle, minimization of gross lift-off mass for a specific mission is the primary objective, and staging is an effective way to reduce the vehicle gross lift-off mass. Therefore, problem arises as to what is the optimum staging to achieve the given mission.

In the scope of this thesis, a Matlab® based computer code was written to determine the staging parameters (number of stages, mass distribution between stages, and the propellant and structural masses for each individual stage) which minimize the gross lift-off mass of the launch vehicle for a specific mission.

The staging optimization problem was formulated based on the delta-V (velocity change) equations and solved by the method of Lagrange Multipliers. The problem was stated in a general form to handle launch vehicles having arbitrary number of stages and with various configurations involving serial, parallel and clustered stages; and with different structural ratios and propellant exhaust velocities in each stage. Velocity increment required to get into orbit can also be calculated by considering the gravitational, aerodynamic and propulsive loss factors with proper margins under the simplest possible conditions without determining the flight trajectory.

On the other hand, a Matlab® based trajectory optimization code well suited to be used in the preliminary mission design phase of a generic multistage Earth-to-orbit launch vehicle was developed. It was aimed to determine the optimal ascent trajectory so as to maximize the payload mass delivered to orbit while satisfying the mission constraints such as limits on acceleration, dynamic pressure and aerodynamic heating rate to prevent the launch vehicle or the payload from damage.

Launch vehicle dynamics were modeled using three-degrees-of-freedom equations of motion and standard models of gravity and atmosphere were implemented in the trajectory optimization problem. A general purpose Matlab software program called GPOPS-II developed by Patterson and Rao (2013a) was employed to solve the trajectory optimization problem.

GPOPS-II utilizes one of the most recent methods known as Radau pseudospectral method, which is based on the discretization of states and controls using global polynomials collocated at a set of Legendre-Gauss-Radau points. After discretization, resulting nonlinear programming problem was solved by SNOPT (Sparse Nonlinear Optimizer) developed by Gill, Murray and Saunders (2005) based on a sequential quadratic programming algorithm for large scale constrained optimization problems.

After having developed these two independent computer codes for staging and trajectory optimization, they are integrated by an overhead program to be used for the simultaneous optimization of the vehicle and the trajectory parameters. Thus, a quick and effective method to find optimal vehicle configurations in the early design phases was achieved in the frame of this thesis.

Space transportation covers a wide range of vehicles from interplanetary or deep space transfer vehicles to reentry vehicles for human spaceflight and expendable launch vehicles for non-crewed payloads. Since the main characteristics of these vehicles differ from each other, disciplinary modeling needs might also be different. In this thesis, only the multistage launch vehicles for satellite delivery into Earth's orbit were covered. Developed tools are appropriate for several types of orbits like sunsynchronous and polar orbits, geostationary transfer orbits, as well as low and high circular or elliptical orbits.

Launch vehicle design is a complex activity involving all aerospace disciplines, and such a design activity requires many years of team work with successive iterations achieved with numerical and experimental methods in the different disciplinary areas. In this thesis, it was not attempted to cover the launch vehicle detailed design process completely, but a representative conceptual design model was identified with few design variables and easy to compute sizing equations.

1.3 Thesis Outline

This thesis is organized as follows:

Chapter 1 is the introduction to the thesis. It provides an overall view of the work and discusses the motivation of this research and presents the scope, objectives and outline of the thesis.

Chapter 2 gives the background on historical development of trajectory optimization techniques, and discusses the currently available trajectory optimization software. The existing research concerned with the design and trajectory optimization of space launch vehicles in the literature is also presented.

Chapter 3 describes the staging optimization code developed in the frame of this thesis in order to determine the staging parameters which minimize the gross lift-off mass of the launch vehicle for a specific mission.

Chapter 4 focuses on the trajectory optimization code in order to determine the optimal ascent trajectory so as to maximize the payload mass while satisfying the mission constraints.

Chapter 5 presents the mathematical framework and illustrative examples to find optimal vehicle configuration in the early design phases by simultaneous optimization of thrust profile and trajectory.

Chapter 6 draws the conclusion of the whole study. It also includes a brief summary of the contributions of this thesis and recommendations for future research.

CHAPTER 2

BACKGROUND

The purpose of this chapter is to present the practical foundations of the staging and trajectory optimization of launch vehicles. Since the majority of the work depends on the application of the optimization theory, a brief overview of several applicable optimization methods and their certain advantages and limitations are first highlighted to provide general background knowledge.

This chapter is organized in three sections. Section 2.1 presents available literature on the trajectory optimization methods, Section 2.2 gives an introductory summary of available software packages for trajectory and design optimization of launch vehicles. Finally, the existing research on staging and trajectory optimization of launch vehicles are given in Section 2.3 including their relevance to the subject of this thesis as well.

2.1 Trajectory Optimization Methods

Trajectory optimization of launch vehicles has been a topic of considerable research and development for almost 50 years. Prior to the development of digital computers, much of the work was focused on obtaining analytical solutions which required making numerous simplifying assumptions as Lawden (1963) declared. With the advent of digital computers, numerical methods were successfully applied to the launch vehicle trajectory optimization problems.

Since the multistage launch vehicles are nonlinear dynamical systems including discontinuities in state variables, and their trajectories are characterized by the time varying differential equations, the launch vehicle trajectory optimization problem can be regarded as a nonlinear, multiphase *optimal control problem* (OCP). A multiphase OCP is formulated as follows.

Given a set of P *phases* where $p \in [1, \ldots, P]$, determine the state $\mathbf{x}^{(p)}(t)$, control $\mathbf{u}^{(p)}(t)$, initial time $t_0^{(p)}$, final time $t_f^{(p)}$, integrals $\mathbf{q}^{(p)}$ in each phase and the static parameters $\mathbf{s}$ that minimize the *cost functional* (also called *objective functional*)

$$J = \Phi\left(\begin{array}{l} \left[\mathbf{x}^{(1)}\left(t_0^{(1)}\right), t_0^{(1)}, \mathbf{x}^{(1)}\left(t_f^{(1)}\right), t_f^{(1)}, \mathbf{q}^{(1)}\right], \\ \ldots, \left[\mathbf{x}^{(P)}\left(t_0^{(P)}\right), t_0^{(P)}, \mathbf{x}^{(P)}\left(t_f^{(P)}\right), t_f^{(P)}, \mathbf{q}^{(P)}\right], \mathbf{s} \end{array} \right) \tag{2.1}$$

subject to the *dynamic constraints*

$$\dot{\mathbf{x}}^{(p)} = \mathbf{a}^{(p)}\left(\mathbf{x}^{(p)}, \mathbf{u}^{(p)}, t^{(P)}, \mathbf{s}\right) \tag{2.2}$$

the *event (boundary) constraints*

$$\mathbf{b}_{\mathbf{min}} \leq \mathbf{b}\left(\begin{array}{l}\left[\mathbf{x}^{(1)}\left(t_0^{(1)}\right), t_0^{(1)}, \mathbf{x}^{(1)}\left(t_f^{(1)}\right), t_f^{(1)}, \mathbf{q}^{(1)}\right], \\ \ldots, \left[\mathbf{x}^{(P)}\left(t_0^{(P)}\right), t_0^{(P)}, \mathbf{x}^{(P)}\left(t_f^{(P)}\right), t_f^{(P)}, \mathbf{q}^{(P)}\right], \mathbf{s}\end{array}\right) \leq \mathbf{b}_{\max} \tag{2.3}$$

the *inequality path constraints*

$$\mathbf{c}_{\min}^{(p)} \leq \mathbf{c}^{(p)}\left(\mathbf{x}^{(p)}, \mathbf{u}^{(p)}, t^{(P)}, \mathbf{s}\right) \leq \mathbf{c}_{\max}^{(p)} \tag{2.4}$$

the *static parameter constraints*

$$\mathbf{s}_{\min} \leq \mathbf{s} \leq \mathbf{s}_{\max} \tag{2.5}$$

the *integral constraints*

$$\mathbf{q}_{\min}^{(p)} \leq \mathbf{q}^{(p)} \leq \mathbf{q}_{\max}^{(p)} \tag{2.6}$$

where the *integrals* are defined as

$$q_i^{(p)} = \int_{t_0^{(p)}}^{t_f^{(p)}} \mathcal{L}^{(p)}\left(\mathbf{x}^{(p)}, \mathbf{u}^{(p)}, t^{(P)}, \mathbf{s}\right) dt \tag{2.7}$$

In an OCP, the control functions to be applied on the system in order to minimize or maximize a specified performance measure are determined while satisfying any constraints on the motion of the system. Primary objective of the launch vehicle trajectory optimization problem is to find an optimal trajectory between the launch site and the target injection orbit. Such kind of OCP with endpoint constraints is referred to as *two point boundary value problem* (2PBVP).

Numerical methods for solving OCPs are divided into two major classes: indirect methods and direct methods. In *indirect methods*, optimality conditions are derived by calculus of variations, and resulting 2PBVP is then solved numerically where some sort of discretization is introduced. In *direct methods*, original problem is discretized in time and it is converted into a *nonlinear programming problem* (NLP) and an approximate solution is then obtained by using NLP techniques.

There are three comprehensive survey papers reviewing and discussing various trajectory optimization methods in the literature. Betts (1998) described the direct and indirect optimization methods in detail, presented the pros and cons and cited relevant work in the literature. Later, Rao (2009) investigated the numerical methods for optimal control problems and introduced well known software programs. Finally, Conway (2012) reported the recent advancements in the numerical solution methods and explained the advantages and disadvantages of the recently developed evolutionary methods.

2.1.1 Numerical Methods for Optimal Control

Advantages and disadvantages of direct and indirect methods were evaluated in the book of Betts (2010) and are herein tabulated in Table 2.1.

Table 2.1 Comparison of indirect and direct methods

	Advantages	**Disadvantages**
Indirect methods	High accuracy	Analytical expressions for necessary conditions must be derived
		Small convergence area, requires good initial guess
		Requires also guesses for costates which do not have physical meaning
Direct methods	Better convergence	Low accuracy

In order to eliminate the disadvantages of indirect methods and to obtain better convergence properties, direct methods were examined in more detail in the scope of this thesis. In a direct method, the state and/or control of the original OCP are approximated/discretized/parameterized in some appropriate manner and direct methods are classified according to how this discretization is handled (Figure 2.1).

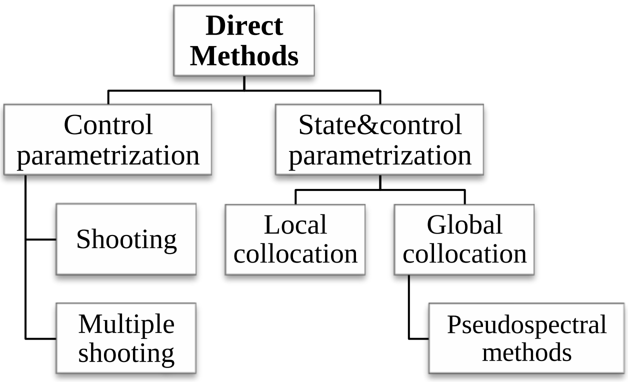
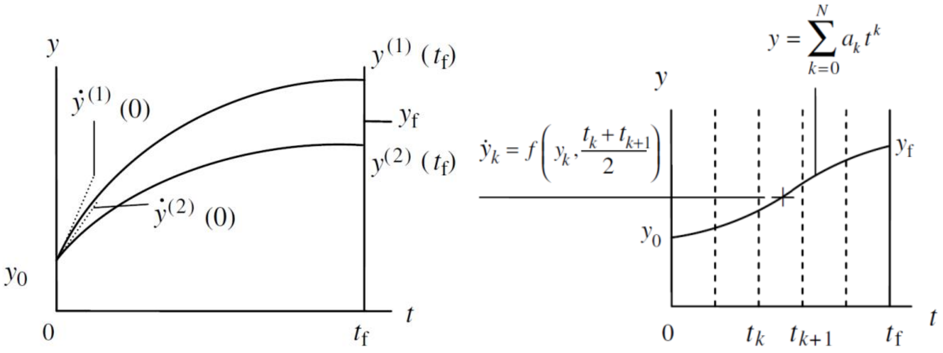


Figure 2.1 Classification of direct methods

In the case where only the control is approximated, the method is called a *shooting method*. When both the state and control are approximated, the method is called a *collocation method*, also called transcription. Schematics of these two methods are given in Figure 2.2 (Tewari, 2011).



(a) (b)

Figure 2.2 Shooting (a) and collocation (b) methods

Shooting methods treat the boundary value problem as if it is an initial value problem and initial slope $\dot{y}(0)$ is adjusted iteratively so that terminal conditions are satisfied. The simplest method to implement is the *ordinary shooting* method and a variant of it is the *multiple shooting* wherein intermediate shooting points are introduced allowing state/costate guesses at these points. Shooting methods are not suitable when the number of design variables is large and they are sensitive to the changes in initial conditions.

The major advantage of collocation methods compared to multiple shooting methods is their better run time performance. Moreover, as Betts (2010) expressed collocation methods offer a convenient way to place a path constraint evaluation point at the same locations where the dynamics are evaluated.

Collocation methods are further subdivided into local and global collocation methods. In *local collocation*, time interval considered is divided into a series of subintervals within which the integration rule must be satisfied. Trapezoidal, Euler, Runge-Kutta and Hermite-Simpson methods are local collocation methods having wide application. In *global collocation*, the state and control variables are expanded into piecewise-continuous polynomials, the derivative of the state variables are approximated by combinations of these interpolating polynomials and their derivatives.

In recent years, global form of orthogonal collocation, i.e. *pseudospectral methods* become very popular. In these methods, states are approximated using global polynomials, and collocation is performed at chosen points. Costates can also be estimated directly from Lagrange Multipliers of NLP, thus advantage of both direct and indirect methods, such as fast convergence and better accuracy are provided. As the name implies, solution converges spectrally (at an exponential rate) as a function of the number of collocation points.

Following types of piecewise polynomials can be used to approximate the differential equations at collocation points.

- Legendre-Gauss polynomials
- Legendre-Gauss-Radau polynomials
- Legendre-Gauss-Lobatto polynomials
- Chebyshev-Gauss polynomials
- Chebyshev-Gauss-Radau polynomials
- Chebyshev-Gauss-Lobatto polynomials

2.1.2 Nonlinear Programming Methods

After a continuous infinite dimensional optimal control problem is discretized using one of the above mentioned collocation methods and converted to an NLP, then an NLP solver is required to obtain optimal trajectory and the controls. Numerical methods for solving NLP are categorized in Figure 2.3.

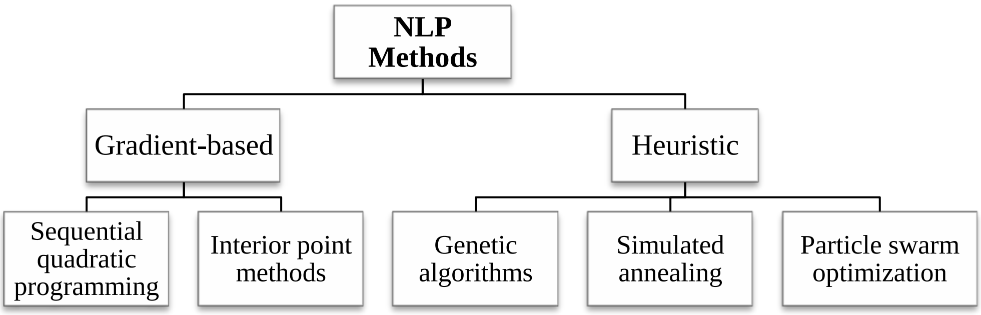


Figure 2.3 Classification of NLP methods

In *gradient-based methods*, an initial guess is made, and a search direction and a step length are determined at every iteration. Most commonly used gradient-based NLP solution methods are *sequential quadratic programming* (SQP) and *interior point methods*. Large numbers of variables and constraints can be handled, and fast and accurate solutions are provided through gradient-based methods. But, they suffer from the possibility of convergence to the local optima closest to the given starting point when the step length is too small to find an improving move.

In *heuristic methods*, search is performed in a stochastic manner and discrete variables can easily be accommodated. They are fairly good at locating the global optimum among local optima, but often provide near optimum solutions, and they are poor in handling path constraints. Main problems that limit the application of stochastic methods to trajectory optimization problems are slow convergence rates and unaffordable computational times due to the requirement of large number of function evaluations before convergence.

2.2 Trajectory Optimization Software

A number of software packages exist for performing trajectory optimization of launch vehicles. POST, OTIS, SOCS and ASTOS are the most popular ones developed by different organizations.

2.2.1 Program to Optimize Simulated Trajectories (POST)

The Program to Optimize Simulated Trajectories (POST) was developed by Lockheed-Martin Astronautics and NASA (United States National Aeronautics and Space Administration) Langley Research Center in 1970 as a space shuttle trajectory optimization program and since then it has been significantly improved with additional capabilities added in the area of vehicle modeling, trajectory simulation and optimization (Brauer, Cornick, & Stevenson, 1977).

POST provides the capability to simulate and optimize 3DOF and 6DOF trajectories for powered or unpowered vehicles and it has been widely used successfully to solve a wide variety of atmospheric ascent and reentry problems, as well as exoatmospheric orbital transfer problems.

POST is a generalized event-oriented code that numerically integrates the equations of motion of a flight vehicle given definitions of aerodynamic coefficients, propulsion system characteristics, atmosphere tables, and gravitational models. Guidance algorithms used in each phase are user-defined. Numerical optimization, specifically nonlinear programming and direct shooting, is used to satisfy trajectory constraints and minimize a user-defined objective function by changing independent steering and propulsion variables along the flight path. POST runs in a batch execution mode and depends on an input file to define the trajectory event structure, vehicle parameters, independent variables, constraints, and objective function.

2.2.2 Optimal Trajectories by Implicit Simulation (OTIS)

Optimal Trajectories by Implicit Simulation (OTIS) was originally developed by the Boeing Company in 1980s under contract to the Air Force and later versions were released by NASA Glenn Research Center (OTIS, 2011).

OTIS can accommodate varying models of propulsion, weights, atmospheres, and aerodynamics. As a result of the trajectory variables being parameterized over specific time intervals, constraint boundaries, such as a dynamic pressure boundary, are easily simulated in OTIS. Vehicle models can be very sophisticated and can be simulated through six degrees of freedom.

Earlier versions of OTIS have primarily been launch vehicle trajectory and analysis programs, but have since been updated for robust and accurate interplanetary mission analyses, including low-thrust trajectories.

OTIS primarily uses nonlinear programming and collocation to solve the nonlinear programming problem associated with the solution of the implicit integration method, although shooting is an option.

2.2.3 Sparse Optimal Control Software (SOCS)

Sparse Optimal Control Software (SOCS) was developed by the Boeing Company in 1990s for solving optimal control problems such as trajectory optimization, chemical process control and machine tool path definition.

SOCS is based on the sparse optimization. Sparsity allows the solver to be fast in comparison with other methods. Betts and Huffman (1990) incorporated sparse matrix methods into a nonlinear programming algorithm and brought a major improvement to collocation methods. Their sparse matrix methods include an automatic mesh refinement algorithm and a Sequential Quadratic Programming (SQP) method that is specialized on solving very large, sparse problems including several ten-thousand variables and constraints.

2.2.4 AeroSpace Trajectory Optimization Software (ASTOS)

AeroSpace Trajectory Optimization Software (ASTOS) has been developed by University of Stuttgart as a part of an ESA (European Space Agency) project since 1989 (ASTOS, 2011).

ASTOS is a trajectory simulation and optimization tool covering 3DOF point mass motion and trimmed 6DOF trajectories for launch vehicle ascent, suited to analyze a wide range of aerospace missions from atmospheric over orbital to interplanetary scenarios. It can be used for fast preliminary design studies as well as for operational tasks with high requirements for accuracy.

ASTOS use two kinds of third party NLP solvers (SNOPT and SOCS) and offers simultaneous optimization of ascent trajectory and launch vehicle design using all-at-once approach with trajectory, aerodynamics, weight, structure, propulsion and cost (as post analysis) disciplines.

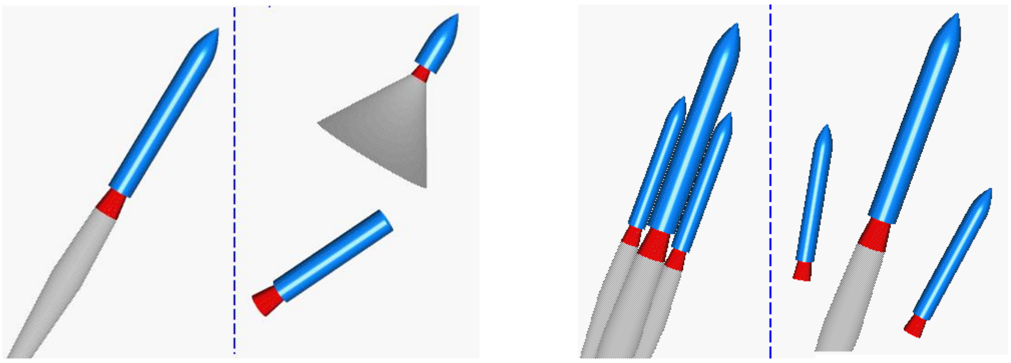
2.3 Previous Research

Existing research in the field of staging and trajectory optimization of launch vehicles are reviewed and discussed in this chapter.

2.3.1 Staging Optimization

Soviet rocket scientist Konstantin Tsiolkovsky first realized the advantages of staging a rocket and leaded to the development of the concept of multistage rockets. He proposed his idea of multistage rockets in the year 1924 in his book called *Cosmic Rocket Trains* (as cited in Todd & Angelo, 2005).

Mainly, there are two types of staging so called serial staging and parallel staging as illustrated in Figure 2.4. In serial staging, several stages are stacked on top of each other and they burn successively. While in parallel staging, stages are mounted in parallel and they burn simultaneously.



(a) (b)

Figure 2.4 Types of staging (a) serial, (b) parallel

(adapted from http://exploration.grc.nasa.gov/education/rocket/rktstage.html)

In serial staging, the first and the largest stage is usually at the bottom and the subsequent upper stages are above it, usually decreasing in size. In parallel staging, strap-on boosters are used to assist the lift capacity of the launch vehicle. Boosters are often referred to as zeroth stage, and they are generally ignited simultaneously with the first stage at lift-off to accelerate the entire vehicle upwards. In general, burn time of the boosters is shorter than the first stage. So, when the boosters run out of propellant they separate from the rest of the vehicle, while the first stage continues to burn. A special case of parallel staging is the cluster staging in which the burn time of the boosters and the core stage are the same.

Different types of vehicle configurations exist in the world market. As shown in Figure 1.1 there are a number of vehicles having stages in series such as Pegasus, Taurus, Titan II and Atlas II. Space Shuttle is the most obvious example of a parallel staged vehicle which has two solid rocket boosters burning simultaneously with the main cryogenic engine. Some other vehicles use both serial and parallel staging such as Delta II and Titan IV. These vehicles have boosters around the 1st stage and a 2nd stage on top of the 1st stage.

On the other hand, single-stage-to-orbit (SSTO) vehicles are so attractive because they eliminate cost, complexities and reliability problems of staging. Fully reusable SSTO vehicles that can fly into orbit and return to landing site like an airplane are the ultimate goal of current research and development activities. However, it is extremely difficult to build such a vehicle with today's technology. The key engineering challenges are the need for very efficient propulsion systems having higher specific impulses and mass ratios.

Another advantage of staging is that launch vehicle configuration can be optimized for the requirements of a particular mission by adjusting the amount of propellant and engine thrust, and using different types of engines, propellants and structural materials for various stages. Stages can also be designed for best performance considering their operating conditions. Lower stages operate in high atmospheric pressure which leads to high aerodynamic drag, while upper stages operate in the vacuum and there is no drag and lift. On the other hand, rocket efficiency is strongly affected by the ambient pressure at the nozzle exit. For optimum performance, nozzle should be short in the lower atmosphere and should be long in the vacuum of space.

Generally, it is meaningful to use solid propellant motors for the lower stages, and liquid propellant engines for the upper stages. This is because solid motors are well suited to produce the high thrust at lift-off, while high energy and accordingly more efficient liquid engines are lighter and thus reduce the work of the lower stages. Therefore, light weight construction quality and the exhaust velocity have to increase with the stage number. Consequently, investment in high-energy propellant upper stages also results in a reduction of the mass of the lower stages. Liquid engines are preferred in the upper stages especially for orbit injection purposes as well because of their stop/restart and throttling capabilities.

The optimum stage mass distribution between stages for a multistage launch vehicle can be determined by staging optimization for a given set of technology options. Previous staging optimization studies are based on the maximization of the payload velocity for a given gross lift-off mass constraint or the minimization of gross lift-off mass for a given payload velocity constraint. While designing a new launch vehicle, minimization of gross lift-off mass for a specific mission is the primary objective.

Till today, many efforts have been made to optimize launch vehicle staging for minimum gross lift-off mass. In the early papers, problem was solved with many simplifying assumptions. Malina and Summerfield (1947) were the first to optimize staging; however, the solution was limited to stages having equal propellant exhaust velocities and structural ratios. Vertregt (1956) extended the solution to the case when all stages have different exhaust velocities. Goldsmith (1957) offered a solution for two stage rockets when the structural masses are proportional to the propellant masses. Weisbord (1958), Subotowicz (1958), Hall and Zambelli (1958) have all presented general solutions for minimum gross lift-off mass with non-homogenous stages and the solution holds for arbitrary number of stages.

All of these authors used the method of Lagrange Multipliers, which was proven many times, and they ignored the effects of gravity, drag and steering in order to obtain the derivatives of Lagrangian analytically. Cobb (1961) minimizes the gross lift-off mass with the constraint that the total energy imparted to the payload is fixed. He included the effect of steering by assuming an average attitude angle for each stage. Later on, Srivastava (1966), Tawakley (1967) and Adkins (1970) have examined the isolated effects of gravity, steering and drag, respectively.

In the frame of this thesis, staging parameters were optimized so as to obtain the initial sizing of the vehicle by considering the gravitational, aerodynamic and propulsive loss factors with proper margins under the simplest possible conditions without determining the flight trajectory.

2.3.2 Trajectory Optimization

The ascent trajectory of a launch vehicle begins with lift-off from the launch site and ends up with the placement of the payload into a specified orbit. Most launch vehicles are launched vertically from the ground, and the vehicle first flies straight up in the dense atmosphere gaining both vertical speed and altitude, and then turns slightly to achieve the horizontal velocity to get into the orbit.

During flight, a number of energy loss mechanisms affect the performance of the launch vehicle. These can be classified as drag losses, gravity losses, steering losses and thrust losses. Determination of the ascent trajectory is governed by the desire to minimize these losses subject to the operational constraints such as limits on acceleration, dynamic pressure, structural loads and aerodynamic heating rate to prevent the launch vehicle or the payload from damage.

Another factor that must be taken into consideration during trajectory optimization is the safety precautions. Launch site and the launch azimuth angle should be selected such that the spent stages, or the launch vehicle itself in case of any emergency, can be safely aborted into open water or unsettled areas.

Such kind of severe constraints on the flight path and the high costs of operating launch vehicles demand optimal flight trajectories, and the trajectory optimization becomes an unavoidable activity for space transportation. Certain applications are:

- Determining the maximum payload mass that a launch vehicle can deliver to orbit
- Sizing the stages of a launch vehicle for best performance
- Analyzing how changes in design affect the overall performance
- Computing the flight corridor and the impact points of spent stages
- Determining the launch window

Typical design variables used in the trajectory optimization of a fixed system for a particular mission are time histories of thrust direction, pitch and yaw angles, fairing jettison time and free flight (coasting) time durations. If it is intended to optimize the vehicle simultaneously with the trajectory, then propellant weights, staging times and thrust profiles will also be considered as optimization variables.

After optimal trajectory has been calculated it is the task of the guidance system to generate steering commands which will direct the launch vehicle over the flight path from the point of departure to the destination. Traditionally, the complete ascent trajectory of a launch vehicle is partitioned into two portions: endoatmospheric and exoatmospheric or atmospheric and vacuum.

In the atmospheric portion, ascent guidance is performed in open loop mode. After the high dynamic pressure portion of the trajectory is over, closed loop guidance mode is switched on. Open loop guidance implies that steering commands and engine throttle settings along the trajectory are predetermined and installed on the guidance computer before flight. Whereas, closed loop guidance means trajectory optimization problem is solved online during flight and guidance commands are generated by the onboard computer in every guidance update cycle using the current condition as the initial condition.

The need for such a partitioning stems from the presence of aerodynamic forces which significantly complicate the vehicle dynamics model and consequently the solution process. However, after the launch vehicle clears the atmosphere, aerodynamic forces and wind effects can be neglected and the optimal ascent trajectory can be determined more quickly and reliably.

Trajectory optimization is a complex and computationally intensive task and requires sophisticated computer simulations. Trajectory optimization is a part of the earliest conceptual design activities with simplified propulsion, aerodynamics and mass properties models, and then it iteratively matures its output through the design process and evolves to a mission specific trajectory determination program in the operational phase.

Work on launch vehicle trajectory optimization has a long history. Over the last 40 years, several authors carried out researches on different methods to solve various problems of this type. The methods differ principally by the choice of unknowns, the types of methods used to integrate the differential equations and the order of integration.

The early methods employed were based on the calculus of variations so called indirect methods. However, because of the disadvantages stated in Table 2.1, finding an optimal solution using indirect methods was a lengthy and labor intensive process. In later years, emergence of faster computers and improvement of numerical techniques leaded to the development of direct methods. Direct methods were then widely used for solving trajectory optimization problems.

One of the simplest methods to implement among direct methods is the direct shooting method. It was first proposed by Brusch and Peltier (1974) and applied to trajectory optimization problems of Space Shuttle (Brusch, 1974) and Atlas-Centaur (Brusch, 1977). Later, Jänsch, Schnepper and Well (1990) proposed multiple shooting method for integrating the dynamic equations numerically and they successfully applied this method to the ascent trajectory optimization of Ariane 5.

During the same years, direct collocation method was developed by Hargraves and Paris (1987) and they implemented this method in OTIS software, which was described in Section 2.2.2. Subsequent implementations of direct collocation algorithms can be found in the works of Jänsch, Schnepper and Well (1990), Enright and Conway (1992) and Styrk (1993). A major improvement to collocation methods was made by Betts and Huffman (1990) with the introduction of SOCS software presented above in Section 2.2.3.

Although the direct methods show better convergence properties, NLP can still be sensitive to initial guesses when the gradient-based search algorithms are used in the solution. That is, a poor initial condition may terminate in an infeasible solution or converge to a local optimum. In order to find an appropriate initial solution for direct methods, hybrid methods, which can be a combination of either direct and indirect methods or direct and heuristic methods, have been proposed by several authors. Hybrid methods provide quick convergence toward the optimal solution.

Gath and Well (2002) implemented an indirect optimization algorithm for initial guess calculation. They first generated the nearly analytic vacuum solution and then introduced the atmospheric effects. Afterwards, they implemented a direct optimization method to solve the optimal trajectory considering path constraints such as dynamic pressure, heat flux and empty stage splash down.

Yokoyama and Suzuki (2005) proposed a real-coded genetic algorithm to provide an appropriate initial solution to gradient-based direct trajectory optimization of a space plane reentry problem. A new hybrid algorithm called genetic algorithm guided gradient search was proposed by Geethaikrishnan, Mujumdar, Sudhakar and Adimurthy (2008) to produce good initial guesses. A case study with Space Shuttle ascent trajectory optimization problem was also carried out in the scope of this work.

Zotes and Peñas (2010) studied the use of a genetic algorithm to optimize the ascent trajectory of a conventional two-stage launcher. Karslı and Tekinalp (2005) successfully applied simulated annealing based single and multiobjective optimization algorithms to the ascent trajectory optimization of a two stage rocket powered launch vehicle. Arslantaş (2012) performed design optimization of a nano-satellite launcher using a multiple cooling multi objective simulated annealing algorithm. In this study, the angle of attack profile and the propulsion characteristics of the motors were formulated as optimization variables.

Chenglong, Xin and Leni (2008) used a particle swarm optimization algorithm in order to solve the trajectory optimization problem of a reusable launch vehicle. They optimized the time histories of angle of attack and bank angle considering the physical constraints such as dynamic pressure, heat flux, total aerodynamic heating and aerodynamic forces.

Main problem that limits the application of heuristic methods to trajectory optimization problem is the requirement of large number of function evaluations before convergence, which results in unaffordable computational times for complex problems.

There are numerous studies covering the optimization of system variables along with the trajectory. In these studies, launch vehicle design problem is generally decomposed into different physical disciplines such as aerodynamics, propulsion, structure, weights and sizing and trajectory optimization and it is called *multidisciplinary design optimization* (MDO) problem. These disciplines can be optimized separately and sequentially or simultaneously within a same system. Hybrid algorithms that combine heuristic methods and gradient-based methods have usually been preferred in MDO studies as also pointed out in the review paper of Balesdent, Bérend, Dépincé and Chriette (2012). Several researches on the applications of MDO methods to launch vehicle design are presented in the following.

Braun, Moore and Kroo (1997) applied the collaborative optimization to the multidisciplinary design of an SSTO vehicle. Vehicle design, trajectory and cost issues were directly modeled in the problem, which is characterized by 95 design variables and 16 constraints.

Akhtar and Linshu (2005) applied hybrid optimization algorithm using genetic algorithm as a global optimizer and sequential quadratic programming as a local optimizer to optimal design of a launch vehicle based on liquid rocket engines. Bayley, Hartfield, Burkhalter and Jenkins (2008) used genetic algorithm to optimize multistage launch vehicles with the goal of minimizing vehicle weight and ultimately the cost.

Rafique, He, Zeeshan, Kamran and Nisar (2011) proposed a hybrid heuristic search algorithm to overcome inherent disadvantages of genetic algorithm, which requires huge number of function evaluations to obtain near optimal solutions, and simulated annealing, which requires feasible direction at initial stage. They applied their method for the design and trajectory optimization of an air launched satellite launch vehicle and proved that the proposed method is more efficient.

Ebrahimi, Farmani and Roshanian (2011) optimized the design of a small solid propellant launch vehicle so as to minimize the gross lift off mass using particle swarm optimization method. Pontani (2014) applied swarming algorithm in order to determine the optimal ascent trajectory after expressing the control variables as functions of the adjoint variables by employing necessary conditions of optimality.

Castellini, Lavagna and Erb (2008) and Castellini and Lavagna (2012) carried out a comparative study to analyze different heuristic algorithms for concurrent optimization of ascent trajectory and system level design of expendable launch vehicles.

Over the last few years, a new approach based on pseudospectral collocation has been developed for rapid trajectory optimization of complex problems. Pseudospectral methods are used to transcribe a given optimal control problem into a nonlinear programming problem and have advantages of big convergence radius, low requirement for initial guesses, small number of function evaluations and good robustness.

Fahroo and Ross (2002) proposed a chebyshev pseudospectral method for solving a generic optimal control problem and pointed out that further tests and analyses are needed to investigate the stability and accuracy of the method.

Ross, D'Souza, Fahroo and Ross (2003) presented the application of a pseudospectral knotting method implemented in the Matlab based software package called DIDO to compute optimal trajectory of a three stage solid launch vehicle. They determined the time history of thrust direction during the whole trajectory including powered and coast phases of flight while maximizing the payload mass.

Rea (2003) applied a legendre pseudospectral method successfully to several launch vehicle trajectory optimization problems. In this study, dynamic pressure and sensed acceleration constraints were imposed and Euler angle quaternions were optimized. Proposed method in this study offered as a potential real time predictive guidance algorithm. Xuan, Zhang and Zhang (2009) also used legendre pseudospectral method to determine the optimal trajectory of a small solid launch vehicle. They analyzed the fairing jettison time as well.

Benson (2004) first implemented the integral and differential gauss pseudospectral method in launch vehicle trajectory optimization problem, and then Huntington (2007) extended the method by a revised pseudospectral transcription for the computation of the control at the boundaries, and examined the local versus global implementation of the method. Later, Jorris and Cobb (2009) addressed the ability of gauss pseudospectral method to generate an optimal reentry trajectory satisfying the waypoints for multiple payload deployments and no-fly zone constraints for geopolitical restrictions or threat avoidance.

Different ascent guidance methods have been developed and implemented in trajectory optimization studies carried out until now. During early times, only the exoatmospheric phase of the flight was attempted to solve. Brown and Johnson (1967) solved the trajectory optimization problem as a boundary value problem in ordinary differential equations to find the optimal steering laws for orbital injection and rendezvous missions.

Later, methods that also treat the endoatmospheric phase have been developed. Bradt, Jessick and Hardtla (1987) used Hermite interpolation and collocation to implicitly integrate the equations of motion and with penalty function to reduce bending moment loads.

Hanson, Shrader and Cruzen (1995) examined and tested several atmospheric open loop and closed loop ascent guidance options for a variety of launch vehicle models. Calise, Melamed and Lee (1998) proposed a hybrid collocation approach and achieved reliable convergence for a single stage vehicle model. They used an iterative method, which starts from a vacuum solution and gradually introduces atmospheric effects until a converged solution is obtained.

Dukeman (2002) developed a closed loop ascent guidance algorithm which cyclically solves the 2PBVP by calculus of variations starting at vertical rise completion through main engine cut off, taking into account atmospheric effects.

Lu, Zhang and Sun (2005) integrated the vacuum multiburn ascent portion with the atmospheric ascent portion via a fixed point iteration algorithm. They specifically addressed convergence issues and were trying to obtain physically correct solution for endoatmospheric portion. Several refinements to this algorithm were presented by Zhang and Lu (2008) and they verified and validated the algorithm by studying several test cases.

Another approach followed in many trajectory optimization problems is dividing the flight trajectory into phases depending on the specific guidance laws. This approach rose again from the need for initial estimates of controls for the numerical solution of the optimal trajectory. Even if the algorithms do not strictly require an initial guess, good guess often help speed up the optimization process. In the presence of local minima, supplying an initial guess, which is as close as possible to the optimal solution also influence the correctness of the results.

Edge and Powers (1976), Markl (2001), Well (2003) and Castellini (2008) adopted this approach and they imposed initial guesses estimated by applying a sequence of guidance laws. A typical reference guidance program consists of four sections: vertical ascent, pitch over, gravity turn and tangent steering laws. However, imposing initial guesses in advance for each trajectory segment, may lead to near optimal solutions rather than optimal solutions. Moreover, these guidance schemes either need some precomputations or involve a specific kind of optimization on their own.

CHAPTER 3

STAGING OPTIMIZATION

Staging optimization starts with the parameters of the desired orbit and payload mass, and serves to determine the staging parameters (number of stages, mass split among stages, propellant masses, etc.) so as to minimize the launch vehicle's gross lift-off mass which can be considered as a key driver of both performance and cost. Staging optimization gives a quick insight about the vehicle performance capability prior to trajectory design with minimum basic vehicle data such as the state of the art values of specific impulse and structural ratios.

3.1 Mathematical Model

3.1.1 Tsiolkovsky's Rocket Equation

The famous fundamental rocket equation derived from Newton's second law of motion governs the relationship between the rocket velocity and the mass change while the propellant is consumed. It allows the estimation of propellant weight necessary to accelerate the vehicle to a given velocity and expressed as

$$\Delta V_{vehicle} = C \cdot \ln \Lambda \tag{3.1}$$

In Eq. (3.1),

$\Delta V_{vehicle}$ is the maximum change of speed that can be attained by the vehicle,

C is the effective exhaust velocity,

Λ is the burn-out mass ratio.

Rocket equation shows that mainly two parameters affect the overall performance of the rocket: the exhaust velocity (C) and the burn-out mass ratio (Λ). C is a common performance figure for the propulsion system and depends mainly on the chemical composition of the propellant, while Λ is a measure of structural efficiency of the rocket.

The *effective exhaust velocity* (C) is defined by

$$C = I_{sp} \cdot g_0 \tag{3.2}$$

In Eq. (3.2),

I_{sp} is the vacuum specific impulse,

g_0 is the gravitational acceleration at sea level, i.e. $g_0 = 9.80665$ m/s^2.

In today's technology, I_{sp} values of different type of rocket motors vary between 180 sec up to 475 sec depending on the propellant used as tabulated in Table 3.1 (Tewari, 2007).

Table 3.1 Practical I_{sp} values for common rocket propellants

Propellant type	**I_{sp} [sec]**
Solid	180 - 270
N2O4/MMH	260 - 310
N2O4/UDMH	
Kerosene/LO2	300 - 350
LH2/LO2	455
LH2/LF	475

The *burn-out mass ratio* (Λ) is just the ratio of the initial mass to the final mass.

$$\Lambda = \frac{m_0}{m_f} = \frac{m_0}{m_0 - m_p} = \frac{m_s + m_p + m_{pl}}{m_s + m_{pl}} \tag{3.3}$$

In Eq. (3.3),
m_0 is the initial mass,
m_f is the final mass,
m_s is the structural mass,
m_p is the propellant mass,
m_{pl} is the payload mass.

For a typical multistage rocket, Λ is in the range from 4 to 14 depending on today's material technology (Turner, 2009). Since I_{sp} and Λ take values within the limits of technology, maximum attainable velocity by the vehicle ($\Delta V_{vehicle}$) is also limited.

Turner (2009) noted that the total velocity requirement to achieve a 500 km circular orbit is about 8.7 km/s; when the gravity, drag and steering losses are included. The mass ratio for such a velocity requirement is too high to achieve, considering the fact that current rocket motors have a maximum specific impulse of about 475 sec. Therefore, staging is essential to inject satellites into orbit.

3.1.2 Multistage Rocket Parameters

The ideal velocity increment for an *N*-stage rocket is the sum of the velocity increments of the individual stages. That is

$$\Delta V_{vehicle} = \sum_{k=1}^{N} C_k \cdot \ln \Lambda_k \tag{3.4}$$

In the analysis of an N-stage rocket, the payload of any particular stage (k) can be considered as the mass of the subsequent stages (k+1, … , N) as illustrated in Figure 3.1.

$$m_{pl,k} = m_{0,k+1} \tag{3.5}$$

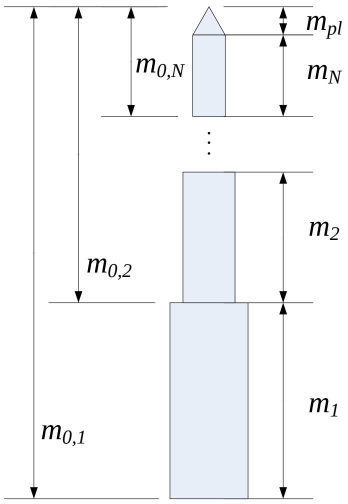


Figure 3.1 Mass definitions for serial staging

Accordingly, the payload of the last (Nth) stage is the actual payload of the launch vehicle. That is

$$m_{pl,N} = m_{pl} \tag{3.6}$$

The total payload ratio is a measure of how much of the initial mass of the vehicle is payload. For launches to low Earth orbit, the payload is about 2-4% of the vehicle mass, and for launches to geostationary transfer orbit, it is only 1% of the vehicle mass (Fischer, 2005). The *total payload ratio* (λ_t) is defined as

$$\lambda_t = \frac{m_{pl}}{m_0} = \frac{m_{pl}}{m_{0,1}} \tag{3.7}$$

In Eq. (3.7), $m_{0,1}$ is the *gross lift-off mass* (GLOM) of the launch vehicle.

The mass of the kth stage, and the initial and final mass before and after the operation of the kth stage are given by the following formulas:

$$m_k = m_{s,k} + m_{p,k} \tag{3.8}$$

$$m_{0,k} = m_k + m_{0,k+1} \tag{3.9}$$

$$m_{f,k} = m_{s,k} + m_{0,k+1} \tag{3.10}$$

3.1.2.1 Serial Staging

The relevant dimensionless ratios for stages in series configuration such as illustrated in Figure 3.1 are written as follows.

The *mass ratio* of the *k*th stage (Λ_k) is

$$\Lambda_k = \frac{m_{0,k}}{m_{f,k}} = \frac{m_{0,k}}{m_{s,k} + m_{0,k+1}} \tag{3.11}$$

The structural ratio is a measure of how much of the stage is structure. The quantity of structural mass comprises of the mass of the structures, mechanisms, engines, fuel tanks, control and measurement systems, etc. excluding the propellant and the payload. The *structural ratio* of the *k*th stage (ε_k) is defined by

$$\varepsilon_k = \frac{m_{s,k}}{m_{s,k} + m_{p,k}} = \frac{m_{s,k}}{m_k} \tag{3.12}$$

It is also convenient to define the *propellant ratio* of the *k*th stage (ζ_k) as

$$\zeta_k = \frac{m_{p,k}}{m_{s,k} + m_{p,k}} = \frac{m_{p,k}}{m_k} \tag{3.13}$$

Following relation exists between the structural ratio and the propellant ratio according to the definitions stated in Eqs. (3.12) and (3.13).

$$\varepsilon_k + \zeta_k = 1 \tag{3.14}$$

The structural mass varies typically between 5% and 15% of the stage mass depending on the size of the stage, and the types of propellant and structural materials used (Ley, Wittmann, & Hallmann, 2009). Accordingly, typical values of ε_k and ζ_k are in the range of $0.05 < \varepsilon_k < 0.15$ and $0.85 < \zeta_k < 0.95$, respectively.

For instance, if the motor casing and/or the propellant tanks are manufactured from composite material, ε_k is smaller compared to a stage made up of metallic structures. The use of high density propellant assists to decrease the structural ratio as well by leading to a smaller size of stage. On the other hand, expendable launch vehicles designed to be used only once have relatively low structural ratios, while reusable ones have higher structural ratios.

The *payload ratio* of the *k*th stage (λ_k) is defined as

$$\lambda_k = \frac{m_{pl,k}}{m_{s,k} + m_{p,k}} = \frac{m_{0,k+1}}{m_k} \tag{3.15}$$

The parameters λ, ε and Λ are not independent. If m_s and m_{pl} are expressed in terms of m_p using Eqs. (3.12) and (3.15), respectively and inserted in Eq. (3.11), then the mass ratio for the *k*th stage (Λ_k) is obtained as follows.

$$\Lambda_k = \frac{1+\lambda_k}{\varepsilon_k + \lambda_k} \tag{3.16}$$

3.1.2.2 Parallel Staging

The analysis of a parallel staged rocket as illustrated in Figure 3.2 is quite similar to that presented above for a serially staged rocket. The main difference is the stage numbering and the need for calculation of average exhaust velocity.

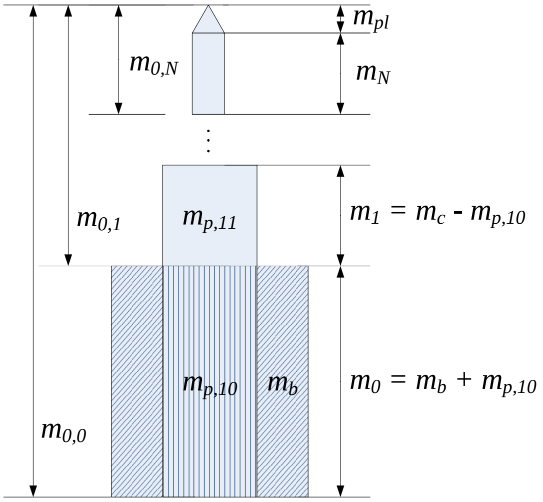


Figure 3.2 Mass definitions for parallel staging

When the parallel boosters and the core of the first stage are burning simultaneously, they are taken together and called the zeroth stage, while the propellant remaining in the core's first stage after discarding the parallel boosters is called the first stage of the rocket.

According to Figure 3.2, the mass, structural and payload ratios of the zeroth stage are equivalent to a serial rocket and they are given by the following formulas:

$$\Lambda_0 = \frac{m_{0,0}}{m_{s,b} + m_{0,1}} \tag{3.17}$$

$$\varepsilon_0 = \frac{m_{s,b}}{m_b + m_{p,10}} \tag{3.18}$$

$$\lambda_0 = \frac{m_{0,1}}{m_b + m_{p,10}} \tag{3.19}$$

In Eqs. (3.17) - (3.19),
b is the subscript for the boosters,
$m_{p,10}$ is the propellant mass burned in parallel with the boosters of the zeroth stage.

The average exhaust velocity of the zeroth stage is

$$C_0 = g_0 \frac{I_{sp,b} \cdot m_{p,b} + I_{sp,c} \cdot m_{p,10}}{m_{p,b} + m_{p,10}} \tag{3.20}$$

Similarly, the equivalent ratios for the first stage are as expressed below:

$$\Lambda_1 = \frac{m_{0,1}}{m_{s,c} + m_{0,2}} \tag{3.21}$$

$$\varepsilon_1 = \frac{m_{s,c}}{m_c - m_{p,10}} \tag{3.22}$$

$$\lambda_1 = \frac{m_{0,2}}{m_c - m_{p,10}} \tag{3.23}$$

In Eqs. (3.20) - (3.23), c is the subscript for the core of the first stage.

3.1.3 Orbital Velocity Equation

A launch vehicle must provide the required energy to insert a spacecraft into a desired orbit represented by its altitude and velocity. It is easy to calculate the velocity required to keep an object in a specified orbit (V_{orbit}) using Eq. (A.10) given in Appendix A.2. For elliptical orbits, injection point can be considered to remain on the safe side as perigee, where it requires higher velocity as explained in Appendix A.4.

3.1.4 Delta-V Calculations

The velocity change of an N-stage launch vehicle ($\Delta V_{vehicle}$) can be calculated from the rocket equation (3.4) knowing the fact that it is the ideal velocity change, for which all the effects due to gravity, aerodynamics and flight maneuvers are neglected. The launch vehicles must also overcome these effects, in addition to accelerating the payload to orbital velocity (V_{orbit}).

Therefore, losses/gains due to various effects should be considered in the calculation of the total velocity increment needed to get into orbit ($\Delta V_{mission}$). That is,

$$\Delta V_{mission} = V_{orbit} + \Delta V_g + \Delta V_d + \Delta V_p - \Delta V_{gain} + \Delta V_m \qquad (3.24)$$

In Eq. (3.24),

V_{orbit} is the orbital velocity,
ΔV_g is the gravitational loss,
ΔV_d is the aerodynamic drag loss,
ΔV_p is the propulsive loss due to steering and ambient pressure change,
ΔV_{gain} is the velocity gain due to Earth's rotation or initial altitude or initial velocity,
ΔV_m is the performance margin for unexpected disturbances and inaccuracies.

Tewari (2007) proposed to add a total of 1.5 km/s margin for the possible velocity losses/gains as applied for a rocket launched to a low Earth orbit and 2 km/s as applied for a rocket launched to a geosynchronous orbit.

If ΔV loss terms are examined separately, it is seen that the gravity losses (ΔV_g) and the drag losses (ΔV_d) are the most significant ones, and they are primarily dependent upon the lift-off thrust-to-weight ratio (T/W). T/W needs to be greater than unity for the vehicle to leave the launch pad, and typical lift-off T/W values are in the range 1.3 to 2 (Curtis, 2005). Propellant type and stage configuration (serial or parallel staging) affects the magnitude of T/W as well.

The higher the T/W, the faster the rocket flies in the dense atmosphere and drag losses increase, while gravity losses decrease. However, the lower the T/W, the longer it takes the rocket to turn over to align itself tangentially to the Earth's surface. Therefore, gravity losses increase.

Approximated values obtained from real data samples can be used for rough estimations of gravity and drag losses. In this study, the variations of ΔV_g and ΔV_d versus T/W are used as illustrated in Figure 3.3. These variations are taken from the study of Loftus and Teixeira (1999) and they are valid for vertical take-off vehicles. For horizontal take-off vehicles, the thrust losses will be higher, but the gravity loss will be much lower. For the horizontal take-off vehicles, the highest value of T/W can be used to approximate this characteristic.

The launch vehicles also experience propulsive losses due to the maneuvering and static pressure difference at the nozzle exit during their flight. These losses are smaller compared to ΔV_g and ΔV_d especially for the vertical take-off vehicles and it is difficult to estimate the magnitude without having the flight trajectory. According to Ley et al. (2009), steering losses is around 20-50 m/s for reaching a low Earth orbit at 200 km altitude. They also pointed out that a margin of 1-2% must be included in the ΔV budget for unexpected orbit/trajectory disturbances and inaccuracies.

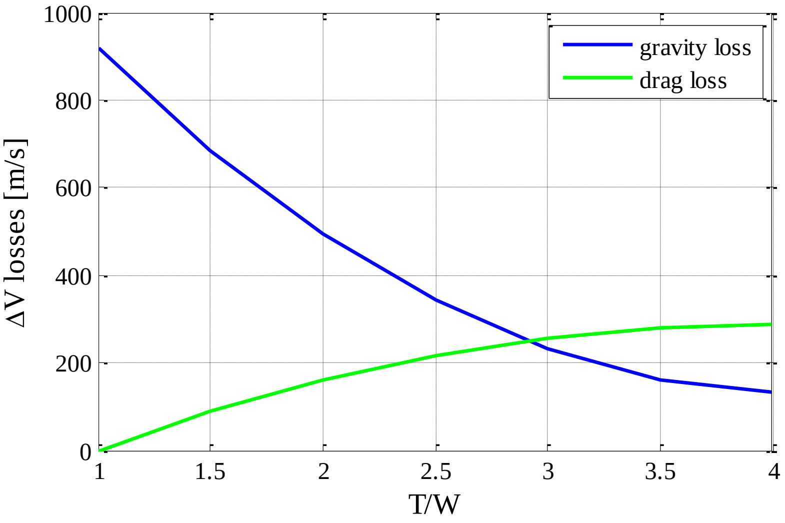


Figure 3.3 ΔV losses vs. T/W

On the other hand, the rotation of the Earth fortunately assists in launching. A launch vehicle has a latitude-dependent initial velocity at the time of launch owing to the eastward rotational velocity of the Earth. Launching from an altitude (h_0) also helps decreasing the total required ΔV. The component of velocity due to Earth's rotation can be calculated as

$$V_{r_0,\phi_0} = \omega_e \cdot r_0 \cdot \cos\delta_0 \tag{3.25}$$

In Eq. (3.25),

ω_e is the Earth's angular velocity, i.e. $\omega_e = 7.292 \times 10^{-5}$ rad/s,

r_0 is the radial distance from the center to the surface of the Earth, i.e. $r_0 = R_e + h_0$,

δ_0 is the geocentric latitude of the launch site,

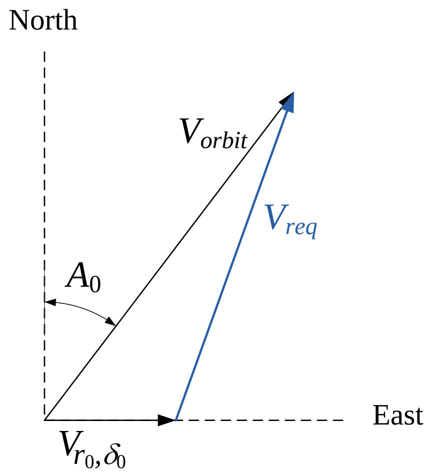
V_{r_0,δ_0} is the Earth's speed at radius r_0 and latitude δ_0 (eastward).

Considering the launch azimuth (A_0) and the Earth's rotation, one can find the velocity required to obtain the target orbit velocity as illustrated in Figure 3.4. According to Figure 3.4, the speed gained due to the Earth's rotation can be calculated as

$$\Delta V_{rot} = V_{orbit} - \sqrt{\left(V_{orbit}\sin A_0 - V_{r_0,\delta_0}\right)^2 + \left(V_{orbit}\cos A_0\right)^2} \tag{3.26}$$

Launching from a carrier aircraft with an initial velocity (V_i) also helps decreasing the total required ΔV. Therefore, ΔV_{gain} in Eq. (3.24) happens to be

$$\Delta V_{gain} = \Delta V_{rot} + V_i \tag{3.27}$$



$$A_0 = \sin^{-1}\left(\frac{\cos i}{\cos \delta_0}\right)$$

i: orbit inclination

Figure 3.4 Velocity gain due to Earth's rotation

3.2 Problem Formulation

The objective of staging optimization problem is to find the optimal mass ratios of stages (Λ_k) which will minimize the gross lift-off mass of the launch vehicle (GLOM, m_0) for specified payload mass (m_{pl}).

$$m_0 = m_{0,1} = \sum_{k=1}^{N} m_k + m_{pl} \tag{3.28}$$

Dividing Eq. (3.28) by m_{pl}, one can write the objective function (minimization of m_0) with respect to the mass ratios of stages (Λ_k). Curtis (2005) showed that the following relation could be obtained by using the definitions of Λ_k and ε_k given in Eqs. (3.11) and (3.12), respectively.

$$\frac{m_0}{m_{pl}} = \prod_{k=1}^{N} \frac{(1-\varepsilon_k)\Lambda_k}{1-\varepsilon_k \Lambda_k} \tag{3.29}$$

It is more convenient to take the natural logarithm of both sides to obtain the following easily differentiable function.

$$\ln\left(\frac{m_0}{m_{pl}}\right) = \sum_{k=1}^{N} \ln \frac{(1-\varepsilon_k)\Lambda_k}{1-\varepsilon_k \Lambda_k} \tag{3.30}$$

The launch vehicle must provide the required energy to insert the satellite into the desired orbit. Thus, $\Delta V_{vehicle}$ must be equal to $\Delta V_{mission}$.

$$\Delta V_{vehicle} = \sum_{k=1}^{N} C_k \cdot \ln \Lambda_k = \Delta V_{mission} \tag{3.31}$$

Finally, the optimization problem can be formulated as described below:

Minimize

$$f = \sum_{k=1}^{N} \ln \frac{(1-\varepsilon_k)\Lambda_k}{1-\varepsilon_k \Lambda_k} \qquad \text{(objective function)} \tag{3.32}$$

Subject to

$$g = \sum_{k=1}^{N} C_k \cdot \ln \Lambda_k - \Delta V_{mission} = 0 \quad \text{(constraint equation)} \tag{3.33}$$

3.3 Method of Solution

If the Method of Lagrange Multipliers is applied, the optimal mass ratios (Λ_k) can be found by specifying the values for $\Delta V_{mission}$, ε_k, C_k and N. Introducing the Lagrange multiplier as p, and combining Eqs. (3.32) and (3.33), one can have the following augmented objective function $f^* = f + pg$:

$$f^* = \sum_{k=1}^{N} \ln \frac{(1-\varepsilon_k)\Lambda_k}{1-\varepsilon_k \Lambda_k} + p\left(\sum_{k=1}^{N} C_k \cdot \ln \Lambda_k - \Delta V_{mission} \right) \tag{3.34}$$

Expanding the logarithm, Eq. (3.34) can be rewritten as

$$\begin{aligned} f^* &= \sum_{k=1}^{N} \left[\ln(1-\varepsilon_k) + \ln \Lambda_k - \ln(1-\varepsilon_k \Lambda_k) \right] \\ &\quad + p\left(\sum_{k=1}^{N} C_k \cdot \ln \Lambda_k - \Delta V_{mission} \right) \end{aligned} \tag{3.35}$$

Differentiating f^* with respect to Λ_k and equating the result to zero, the optimality condition is obtained as

$$\frac{\partial f^*}{\partial \Lambda_k} = \frac{1}{\Lambda_k} + \frac{\varepsilon_k}{1-\varepsilon_k \Lambda_k} + p \cdot C_k \frac{1}{\Lambda_k} = 0 \tag{3.36}$$

Hence, one can find Λ_k as

$$\Lambda_k = \frac{1 + p \cdot C_k}{p \cdot C_k \cdot \varepsilon_k} \tag{3.37}$$

For f^* to be minimum at the mass ratios Λ_k given by Eq. (3.37), the second derivatives of f^* must all be positive for all values of Λ_k. This condition is expressed as follows:

$$\frac{\partial^2 f^*}{\partial \Lambda_k^{\,2}} = -\frac{1+p\cdot C_k}{\Lambda_k^{\,2}} + \left(\frac{\varepsilon_k}{1-\varepsilon_k \Lambda_k}\right)^2 > 0 \tag{3.38}$$

On the other hand, the substitution of Eq. (3.37) into Eq. (3.33) leads to the following equation:

$$\sum_{k=1}^{N} C_k \cdot \ln \frac{1+p\cdot C_k}{p\cdot C_k \cdot \varepsilon_k} = \Delta V_{mission} \tag{3.39}$$

It is clear from Eq. (3.4) that the vehicle performance increases with an increasing number of stages. However, the increase becomes very small after a certain point and staging only brings complexity and cost to the system. Burghes (1974) showed that the optimum number of stages is between 2 and 4 for most satellite launching operations. Based on this fact, the minimum number of stages that is practical should be chosen and solved first, and then significant differences should be compared after evaluating different values of N.

As noted in Sections 3.1.1 and 3.1.2.1, the exhaust velocities (C_k) and the structural ratios (ε_k) of the stages strongly depend on the operational aspects, and the state of the art of propellant and materials technology. Therefore, the values for ε_k and C_k are initially specified by the designer considering available technology options, and $\Delta V_{mission}$ is calculated using the method given in Section 3.1.4.

Eq. (3.39) is a transcendental equation with single unknown (p), and can be solved by iterative methods. Having evaluated p for a given set of ε_k and C_k, the optimal mass ratios of stages (Λ_k) can be found by substituting p into Eq. (3.37).

In this study, Newton-Raphson method, which is a widely used method for solving transcendental equations, was used to solve Eq. (3.39). The Newton-Raphson method in one variable (p) was implemented starting with an initial guess p_0.

The initial guess plays an important role in the solution to overcome the shortcomings such as infinite iteration cycles and non-convergence. Determining the bounds for the solution will help define the initial guess. The bounds can be determined as explained below.

It is clear that the natural logarithm (ln) function in Eq. (3.32) is defined only for positive values, i.e.

$$\frac{(1-\varepsilon_k)\Lambda_k}{1-\varepsilon_k \Lambda_k} > 0 \tag{3.40}$$

The following facts are also known:

- $\Lambda_k > 1$
- $0 < \varepsilon_k < 1$
- $C_k > 0$

$1 - \varepsilon_k \Lambda_k$ must be greater than zero in order to satisfy Ineq. (3.40). Using Eq. (3.37)

$$\varepsilon_k \Lambda_k = \frac{1 + p \cdot C_k}{p \cdot C_k} = \frac{1}{p \cdot C_k} + 1 < 1 \quad \rightarrow \quad p < 0 \tag{3.41}$$

$$\Lambda_k = \frac{1 + p \cdot C_k}{p \cdot C_k \cdot \varepsilon_k} > 1 \tag{3.42}$$

Rearranging the preceding inequalities, one gets

$$1 + p \cdot C_k < p \cdot C_k \cdot \varepsilon_k \quad \rightarrow \quad p < \frac{-1}{C_k \left(1 - \varepsilon_k\right)} \tag{3.43}$$

Since Eq. (3.43) must hold for every k, the upper bound on p is obtained as

$$p < \frac{-1}{\min\left[C_k \left(1 - \varepsilon_k\right)\right]} \tag{3.44}$$

After the calculation of Λ_k, λ_k can be obtained from Eq. (3.16) as

$$\lambda_k = \frac{\Lambda_k \varepsilon_k - 1}{1 - \Lambda_k} \tag{3.45}$$

With the known values of λ_k, the mass of each stage can be calculated from Eq. (3.15) by the following recursive equation beginning from the Nth stage down to the first stage.

$$\begin{aligned} m_N &= \frac{m_{pl}}{\lambda_N} \\ m_{N-1} &= \frac{m_{pl} + m_N}{\lambda_{N-1}} \\ &\vdots \\ m_1 &= \frac{m_{pl} + m_N + m_{N-1} + \cdots + m_2}{\lambda_1} \end{aligned} \tag{3.46}$$

Finally, GLOM (minimized value) is obtained as

$$m_0 = \sum_{k=1}^{N} m_k + m_{pl} \tag{3.47}$$

The structural mass of each stage is found from Eq. (3.12)

$$m_{s,k} = \varepsilon_k \cdot m_k \tag{3.48}$$

As for the propellant mass of each stage, it is found as

$$m_{p,k} = m_k - m_{s,k} \tag{3.49}$$

So far, the staging has been optimized based on the equations of serial staging. For the launch vehicles with parallel staging, an equivalent launch vehicle with serial staging can be defined as described in Section 3.1.2.2. After the staging has been optimized based on the serial staging, the optimal staging data can be converted to the actual parallel configuration again as described in Section 3.1.2.2.

3.4 Results and Discussion

A Matlab script given in Appendix C.2 was written to solve the staging optimization problem. Thus, a quick and effective tool to find optimal vehicle configurations in the conceptual design phase of a generic multistage launch vehicle was achieved. A brief description of the code as a simple flow diagram is given in Figure 3.5.

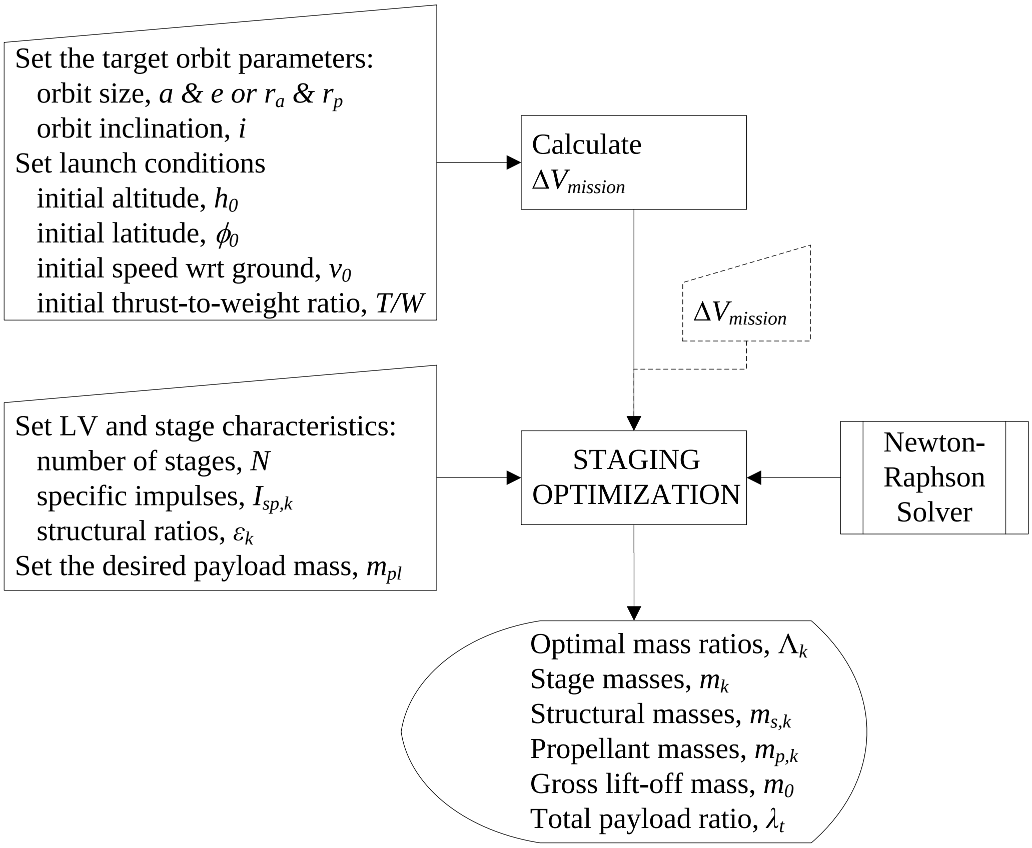

Set the target orbit parameters:
orbit size, a & e or r_a & r_p
orbit inclination, i
Set launch conditions
initial altitude, h_0
initial latitude, ϕ_0
initial speed wrt ground, v_0
initial thrust-to-weight ratio, T/W
Calculate $\Delta V_{mission}$
$\Delta V_{mission}$
Set LV and stage characteristics:
number of stages, N
specific impulses, $I_{sp,k}$
structural ratios, ε_k
Set the desired payload mass, m_{pl}
STAGING OPTIMIZATION
Newton-Raphson Solver
Optimal mass ratios, Λ_k
Stage masses, m_k
Structural masses, $m_{s,k}$
Propellant masses, $m_{p,k}$
Gross lift-off mass, m_0
Total payload ratio, λ_t

Figure 3.5 Staging optimization code description

As seen from this diagram, $\Delta V_{mission}$ can be given by the user directly as an input to the program or it is calculated by the program based on the mission and considering the possible velocity losses explained in Section 3.1.4.

The code was verified in two steps. First, the staging optimization for a given value of $\Delta V_{mission}$ was verified by the results of previous studies based on different methods (Weisbord, 1958; Builder, 1959; Gray & Alexander, 1965; Lubowe, 1965; Hill, 1967; Curtis, 2005; Tewari, 2007). In the next step, code was verified as a whole including the estimation of $\Delta V_{mission}$ for different missions using the available data of present launch vehicles in the market.

As a numerical example, results of staging optimization for two different missions are presented in Table 3.2 for gaining insight about vehicle configurations. $I_{sp,k}$ and ε_k values were specified among the available technology options. First, two stage vehicle was solved and then three stage vehicle was tried to see the effect of number of stages. Hence, it is shown that staging provides considerable performance gain in terms of the decrease in GLOM and increase in payload ratio.

Table 3.2 Numerical example for staging optimization for given $\Delta V_{mission}$

Mission definition				
Payload mass, m_{pl}	150 kg		2000 kg	
Target orbit	200 km altitude circular		350 km x 1000 km altitude elliptic	
$\Delta V_{mission}$	≈ 8 km/s		≈ 10.6 km/s	
Specified				
# of stages	2	3	2	3
Specific impulses, $I_{sp,k}$ (sec)	270; 305	270; 305; 305	360; 450	360; 450; 400
Structural ratios, ε_k	0.15; 0.14	0.15; 0.14; 0.12	0.15; 0.12	0.15; 0.12; 0.11
Results				
Optimal mass ratios, Λ_k (kg)	3.82; 4.44	1.92; 2.64; 3.08	2.98; 4.65	1.53; 3.19; 2.78
Stage masses, m_k (kg)	9993; 1365	3025; 1686; 496	66190; 16486	22242; 25395; 5140
Gross lift-off mass, m_0 (kg)	11508.4	5356.81	84676	54777
Total payload ratio, λ_t	0.013	0.028	0.024	0.037

As an example for existing launch vehicles, India's geosynchronous satellite launch vehicle (GSLV) whose staging data is available on the web site of Encyclopedia Astronautica (2012) was selected. For a mission inserting a 2500 kg payload into geostationary transfer orbit, staging was optimized for minimum GLOM (Table 3.3).

Table 3.3 GSLV staging data

Original data							
Stage #	I_{sp} (sec)	ε (-)	m_p (kg)	m_s (kg)	m (kg)	Λ (-)	ΔV (m/s)
0	281	0.123	160,000	22,400	182,400	1.667501	1,410
1	266	0.18	129,000	28,300	157,300	2.460929	2,350
2	295	0.126	37,500	5,400	42,900	2.666667	2,838
3	460	0.151	12,400	2,200	14,600	3.638298	5,828
Σ			338,900	58,300	397,200		12,426
Serial staging – optimized for minimum GLOM							
Stage #	I_{sp} (sec)	ε (-)	m_p (kg)	m_s (kg)	m (kg)	Λ (-)	ΔV (m/s)
0	281	0.123	209,340	29,360	238,700	2.48287	2,507
1	266	0.18	36,213	7,949	44,163	1.47902	1,021
2	295	0.126	42,458	6,121	48,578	2.68538	2,859
3	460	0.151	14,069	2,502	16,571	3.81249	6,039
Σ			302,080	45,932	348,012		12,426
Parallel staging – converted from serial staging data							
Stage #	I_{sp} (sec)	ε (-)	m_p (kg)	m_s (kg)	m (kg)	Λ (-)	ΔV (m/s)
0	277.7	0.099	$m_{p,b}$: 180,000 $m_{p,10}$: 50,000	25,245	255,245	2.416173	2,404
1	266	0.309	48,000	21,512	69,512	1.538331	1,124
2	295	0.126	42,437	6,118	48,555	2.682981	2,856
3	460	0.151	14,091	2,506	16,597	3.814745	6,042
Σ			334,528	55,381	389,910		12,425

When results are compared with the original vehicle data, a considerable difference has been observed in the first two stages. It is important to note that this difference is due to the parallel staging of GSLV.

One can easily find the equivalent parallel staged vehicle after staging has been optimized based on serial staging using the definitions given in Section 3.1.2.2. However, resulting parallel staged vehicle will be heavier than its serial equivalent to provide same ΔV. This shows that serial staging is more efficient from ΔV point of view and parallel staging is more efficient from thrust point of view. Thus, boosters are only used in initial stages.

Finally, in order to illustrate the application of the method proposed in this thesis, the variation of GLOM with N, $\Delta V_{mission}$ and m_{pl} were shown graphically in Figure 3.6 to Figure 3.8.

Figure 3.6 illustrates the variation of GLOM with the number of stages. The decrease in GLOM between N = 2 and N = 3 is quite appreciable, but for higher values of N, the decrease becomes very small.

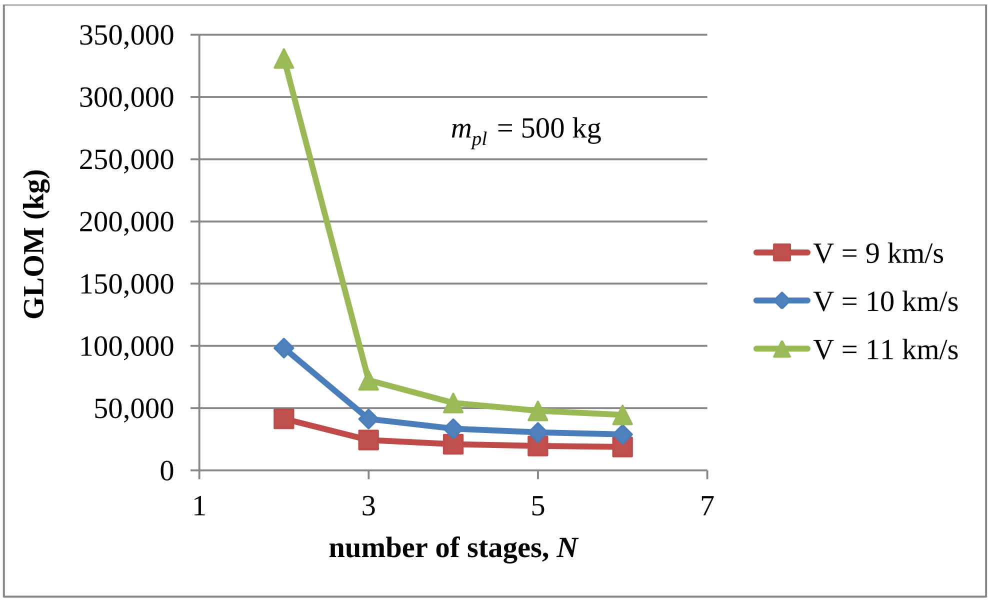


Figure 3.6 Variation of GLOM with number of stages

Figure 3.7 and Figure 3.8 illustrate the variation of GLOM with $\Delta V_{mission}$ and the payload mass (m_{pl}), respectively. These figures show that GLOM increases exponentially with increasing $\Delta V_{mission}$ and increases linearly with the increasing m_{pl}.

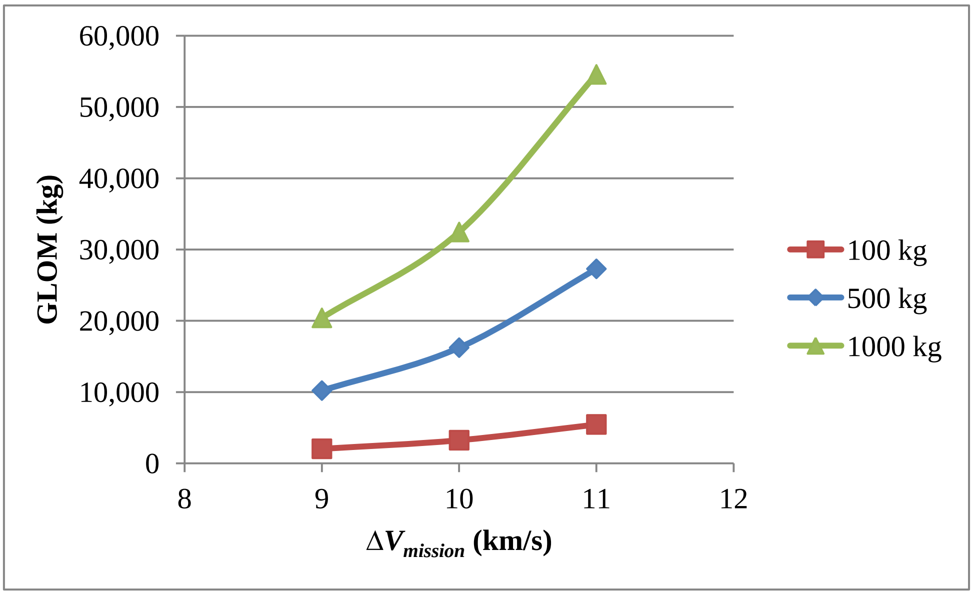


Figure 3.7 Variation of GLOM with $\Delta V_{mission}$

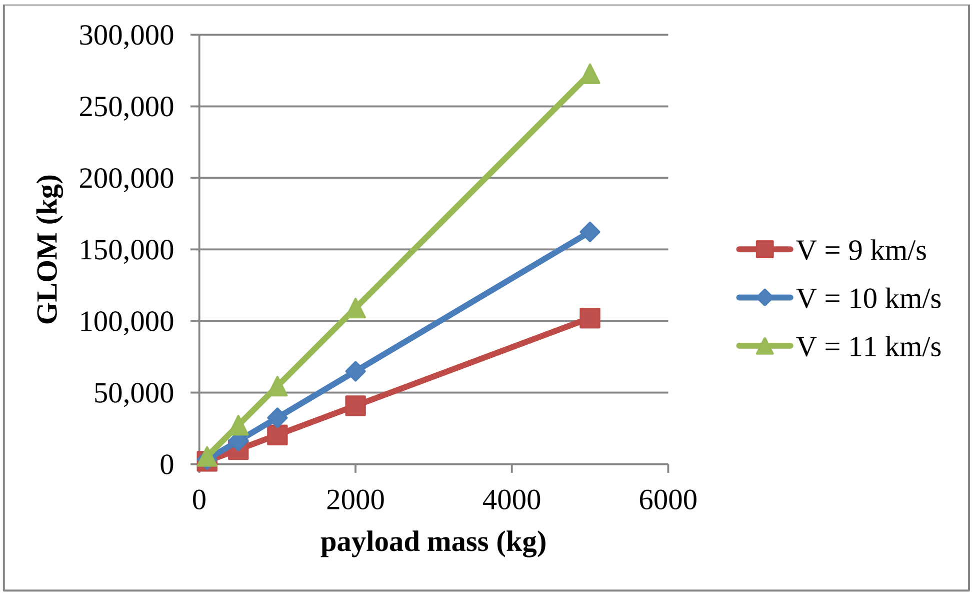


Figure 3.8 Variation of GLOM with payload mass

It is necessary to point out that the method proposed in this study with approximate estimations of the gravitational, aerodynamic and propulsive losses during the flight is not very accurate, but nonetheless it is still useful for a preliminary evaluation during the conceptual design phase. It is always necessary to carry out trajectory simulations by solving the equations of motion for a more detailed and accurate performance analysis.

CHAPTER 4

TRAJECTORY OPTIMIZATION

Launch vehicle trajectory optimization deals with to ensure desired terminal conditions with minimum energy effort as well as satisfying mission constraints. The aim of this thesis is to develop a trajectory optimization tool to be used for both the preliminary mission design of an existing launch vehicle and the conceptual design of a new launch vehicle. For this purpose, a simplified formulation of an ascent trajectory optimization for a typical multistage Earth-to-orbit launch vehicle was first established, and then new features and capabilities were included to improve the modeling and simulation fidelity and to meet the launch vehicle design needs.

4.1 Problem Formulation

The multistage launch vehicle trajectory optimization problem is an example of an optimal control problem (OCP) with discontinuities in the states, where they occur at time instants of empty stages jettisoning. Such kind of problems is called multiphase OCPs and a brief description of the formulation was given in Section 2.1.

The details of the trajectory optimization problem addressed in this thesis are stated in the following sections in terms of system dynamics, objective functions, optimization variables and constraints. Furthermore, general purpose optimization software used to solve the problem and its implementation are described in Section 4.1.5.

4.1.1 System Dynamics

The differential equations representing the system dynamics of the launch vehicle are the equations of motion and the mass flow rate equation since it is a moving body with variable mass. Mathematical models used in the problem formulation are related to the disciplines of trajectory, propulsion and aerodynamics, and also geophysical models such as earth, gravity and atmosphere are utilized in order to simulate the environmental conditions during flight. Overview of these disciplinary models and the interactions among them are presented in the following sections before applying them to solve a given problem.

4.1.1.1 Trajectory Model

The complete description of a launch vehicle's motion comprises three translational and three rotational degrees of freedom. However, in the early phases of design, the trajectory of the center of mass of the vehicle is of greater interest than its attitude motion. Therefore, in this study, rotational dynamics is neglected and the vehicle is considered as a point mass which leads to three degrees of freedom (3DOF) equations of motion governed by Newton’s second law.

$$\dot{\mathbf{r}}(t) = \mathbf{v}(t) \tag{4.1}$$

$$\dot{\mathbf{v}}(t) = -\frac{\mu}{\left\|\mathbf{r}(t)\right\|^3}\mathbf{r}(t) + \frac{T(t)}{m(t)}\mathbf{u}(t) + \frac{\mathbf{D}(t)}{m(t)} \tag{4.2}$$

Eq. (4.1) is the kinematic equation defining the time rate of change of the vehicle’s position and Eq. (4.2) is the dynamic equation describing the motion of the vehicle under the external forces. The forces acting on the vehicle during powered atmospheric flight are gravitational, propulsive and aerodynamic forces.

In Eqs. (4.1) and (4.2),
$\mathbf{r}(t)$ is the inertial position vector at any time instant and represented by $[r_x\ r_y\ r_z]^T$,
$\mathbf{v}(t)$ is the inertial velocity vector at any time instant and represented by $[v_x\ v_y\ v_z]^T$,
$\mathbf{u}(t)$ is the inertial thrust direction vector at any time instant and represented by $[u_x\ u_y\ u_z]^T$,
$\mathbf{D}(t)$ is the drag force vector at any time instant and represented by $[D_x\ D_y\ D_z]^T$,
μ is the Earth’s gravitational parameter, i.e. μ = 398600.4 km^3/s^2,
$T(t)$ is the magnitude of thrust at any time instant,
$m(t)$ is the mass at any time instant.

Knowing the fact that Newton’s laws are valid only with respect to the inertial frame, it is the easiest and the fastest way to define and integrate the equations of motion in Earth-centered-inertial (ECI) frame. And also, rectangular coordinates are preferred since they offer a simpler formulation and eliminate singularity problems at lift-off (zero initial velocity and 90° flight path angle).

While on the other hand, the relative spherical coordinates give a better insight and understanding about the vehicle’s motion, so they are used as the output coordinates to present the results to the user. Furthermore, since the final state of the launch vehicle’s ascent trajectory is defined with respect to the orbital frame, it is also required to compute the orbital elements all along the trajectory in order to assess whether the terminal boundary constraints are satisfied or not. Different coordinate systems and the related transformations between them are presented in Appendix B.1 and Appendix B.2, respectively.

Although the rotational dynamics are not modeled in the equations of motion, the attitude of the launch vehicle can be defined based on certain simplifying assumptions, and thus trajectory control variables can be physically interpreted easily. The attitude, in general, describes the orientation of a body-fixed reference frame with respect to an external reference frame. Euler angles or aerodynamic angles are often used to specify the attitude of the launch vehicle during flight.

Body-fixed reference frame denoted by oxyz, is a reference frame moves with the rigid body and has its origin at the instantaneous mass center of the vehicle. As illustrated in Figure 4.1, the x axis coincides with the longitudinal axis of the vehicle and points forward (flight direction), while the y axis is the lateral or transverse axis and points right. The axis completing the right hand triad is called the z axis or normal axis and points downward.

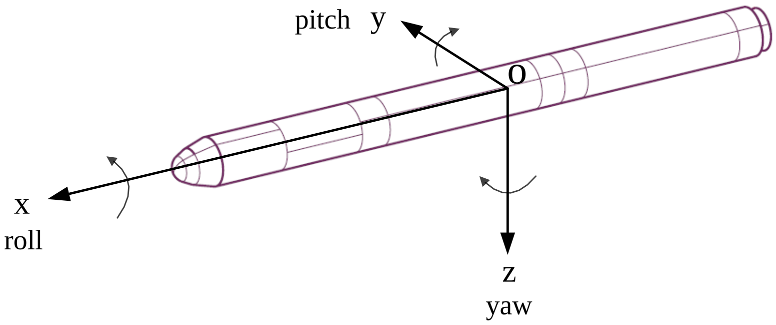


Figure 4.1 Euler angles

The Euler angles, namely roll, pitch and yaw angles define the attitude of the vehicle with respect to the vehicle carried local frame defined in Section B.1.2. According to Figure 4.1, rotation about the x axis (positive, if clockwise) is called the *roll angle*, rotation about the y axis is called the *pitch angle* (θ) and finally rotation about the z axis is called the *yaw angle* (ψ).

For the launch vehicles having a rotational symmetry around the x axis, specifying a roll angle would not make any sense. However, roll angle becomes significant for the vehicles which are nonaxisymmetric those having strap-on boosters. In such cases, the roll angle can be assumed as fixed so that both the longitudinal axis and the velocity vector lie in the vehicle's symmetry plane. Therefore, the variation of roll angle is not considered in this thesis.

The pitch and yaw control of launch vehicles are achieved mainly by thrust vectoring i.e., by changing the direction of the thrust vector relative to the longitudinal axis of the vehicle. In Eq. (4.2), the thrust direction vector is represented by a unit vector in rectangular coordinates in ECI frame, and its physical interpretation is not quite clear. In general, the thrust direction is expressed by two angles in pitch and yaw planes and called as thrust deflection angles. Thrust deflection angle in pitch plane is illustrated in Figure 4.2a.

Considering that the commanded thrust deflection angles can be achieved instantaneously by the launch vehicle's control system, the thrust vector can be assumed to be coincident with the longitudinal (roll) axis of the vehicle in 3DOF trajectory models (Figure 4.2b).

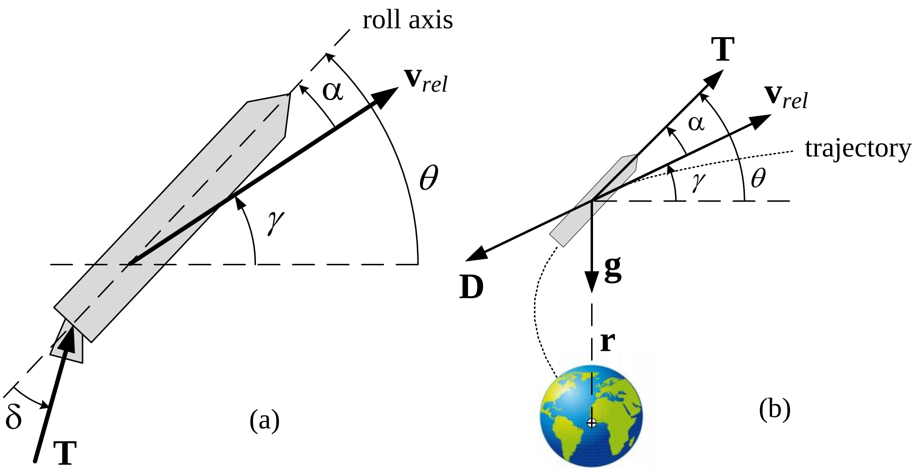


Figure 4.2 Pitch (θ), angle of attack (α) and flight path angle (γ)

Aerodynamic angles, namely the angle of attack, side slip and bank angle define the attitude of the vehicle with respect to the relative velocity vector in the absence of wind. Under the assumption that the thrust is aligned with the body axis represented by Figure 4.2b, the aerodynamic angles are the two angles exist between the projections of the thrust vector and the relative velocity vector (tangent to the trajectory) into the vehicle carried local frame defined in Section B.1.2.

As illustrated in Figure 4.2, the *angle of attack* (α) is the angle between the thrust vector and the velocity vector in the vertical plane and is defined positive for a nose-up attitude.

$$\alpha = \theta - \gamma \tag{4.3}$$

In Eq. (4.3),

θ is the pitch angle,

γ is the flight path angle which is defined in Section B.1.3.

Similarly, the *side slip* (β) is the angle in the horizontal plane and is defined as positive for a nose-left attitude.

$$\beta = \psi - \chi \tag{4.4}$$

In Eq.(4.4),

ψ is the yaw angle,

χ is the heading angle which is defined in Section B.1.3.

Rotation around the relative velocity vector is called the *bank angle* and it can be assumed as zero for vertically launched nonwinged vehicles that are in the subject of this thesis. Additionally, it is important to point out that for the launch vehicles having the thrust vector aligned with the body axis, the thrust deflection angles can be considered to be equal to the aerodynamic angles.

4.1.1.2 Propulsion Model

The propulsion system generates the required thrust to overcome the gravity and drag forces, and lifts the vehicle from the ground and carries it into the orbit. The *thrust equation* is derived from the conservation of momentum and expressed as

$$T = \dot{m}V_e + \left(P_e - P_a\right)A_e \tag{4.5}$$

In Eq. (4.9),

T is the magnitude of the thrust force,

$\dot{m}$ is the mass flow rate,

V_e is the exhaust velocity at the nozzle exit,

P_e is the static pressure at the nozzle exit,

P_a is the ambient static pressure, i.e. the atmospheric pressure,

A_e is the nozzle exit area.

The first term in Eq. (4.5) is called the momentum thrust, and the second term is the pressure thrust. Pressure thrust increases as the atmospheric pressure decreases with increasing altitude. In the vacuum of space, the ambient pressure drops almost to zero and the thrust becomes

$$T_{vac} = \dot{m}V_e + P_e A_e \tag{4.6}$$

Sutton and Biblarz (2001) emphasized that the change in pressure thrust due to altitude changes can amount between 10% and 30% of the overall thrust. Therefore, a correction has to be made to consider this effect in order to obtain accurate results. Using Eqs. (4.5) and (4.6), the actual thrust at any altitude can be calculated from the vacuum thrust through

$$T = T_{vac} - P_a A_e \tag{4.7}$$

Another important measure of the rocket performance is the *specific impulse* (I_{sp}) and it represents the thrust per unit weight flow rate of propellant at sea level on Earth by convention.

$$I_{sp} = \frac{T}{\dot{m} \cdot g_0} \tag{4.8}$$

In Eq. (4.8),

I_{sp} is the specific impulse of the engine,

g_0 is the gravitational acceleration at sea level, i.e. $g_0 = 9.80665$ m/s^2.

To complete the propulsion model, Eq. (4.8) is rewritten to obtain the *mass flow rate equation* knowing that the mass is continuously decreases during flight.

$$\dot{m}(t) = -\frac{T(t)}{I_{sp} \cdot g_0} \tag{4.9}$$

When the upper stages of the launch vehicles are equipped with restartable liquid engines, efficient use of the available propellant becomes an important consideration. The total amount of propellant in the upper stage is consumed mainly in four different tasks.

- Propellant needed to get into the orbit
- Propellant reserve allocated for trajectory corrections
- Propellant needed for deorbiting maneuver
- Unusable propellant

Orbit acquisition can be achieved by a single firing or with two burns of the upper stage which are separated by a coast phase. Propellant consumed during these burns is calculated by solving differential equations of motion and the mass flow rate equation. However, the propellant mass available for orbit acquisition is limited because of the other items explained below.

Propellant reserve should be allocated to ensure mission success against any uncertainties and dispersions. Unforeseen variations in the propulsive characteristics of the lower stages, errors in the mass properties determination and the limited predictability of the aerodynamic coefficients and the atmospheric conditions are the sources of trajectory dispersions. Therefore, a certain amount of propellant is used to compensate these dispersions occur in previous flight phases. In this thesis, propellant amount corresponding to 0.5% of the total delta-V generated by the lower stages is considered for trajectory corrections.

$$\Delta V_{1 \to N-1} = \sum_{k=1}^{N-1} C_k \cdot \ln \Lambda_k \tag{4.10}$$

$$m_{p,corrections} = m_0 \cdot \left(1 - e^{\left(-\frac{0.005 \Delta V_{1 \to N-1}}{I_{sp,N} \cdot g_0} \right)} \right) \tag{4.11}$$

The upper stages of launch vehicles leave in the orbit for years after spacecraft separation unless they are removed from the orbit. Since the debris and consequently the risk of collision gradually increases in the densely populated orbit regions, deorbiting of the launch vehicle upper stages has become a common practice in recent years due to the international agreements. Deorbiting means intentional departing of an object from the orbit for reentry into the Earth’s atmosphere by applying a tangential and decelerative impulse at the apogee point.

Delta-V required for deorbiting from a circular or an elliptical orbit can be estimated using the following equations derived by Milstead (*Deboost from circular orbits*, 1966) as cited in the website of Eagle. The user-defined mission constraints are the entry altitude and the entry flight path angle relative to the spherical Earth.

$$\Delta V_{deorbit,circular} = \sqrt{\frac{\mu}{r_i}} \left\{ 1 - \sqrt{\frac{2(\tilde{r}-1)}{\left(\dfrac{\tilde{r}}{\cos\gamma_e}\right)^2 - 1}} \right\} \tag{4.12}$$

In Eq. (4.12),

r_i is the radius of the initial circular orbit,

r_e is the radius at the entry interface,

γ_e is the flight path angle at the entry interface,

$\tilde{r}$ is the radius ratio, i.e. $\tilde{r} = \dfrac{r_i}{r_e}$.

$$\Delta V_{deorbit,elliptical} = \sqrt{\frac{\mu}{r_e}} \left\{ \sqrt{\frac{2\tilde{r}_p}{\tilde{r}_a\left(\tilde{r}_a + \tilde{r}_p\right)}} - \sqrt{\frac{2\left(\tilde{r}_a - 1\right)}{\tilde{r}_a\left(\tilde{r}_a^{\,2} - \cos^2\gamma_e\right)}} \cos\gamma_e \right\} \tag{4.13}$$

In Eq. (4.13),

r_a is the apogee radius of the initial elliptical orbit,

r_p is the perigee radius of the initial elliptical orbit,

$\tilde{r}_a$ is the apogee radius ratio, i.e. $\tilde{r}_a = \dfrac{r_a}{r_e}$,

$\tilde{r}_p$ is the perigee radius ratio, i.e. $\tilde{r}_p = \dfrac{r_p}{r_e}$.

The amount of propellant required for deorbiting can then easily be calculated by using the famous rocket equation given in Eq. (3.1) and can be expressed as

$$m_{p,deorbit} = m_0 \cdot \left(1 - e^{\left(-\frac{\Delta V_{deorbit}}{I_{sp} \cdot g_0}\right)} \right) \tag{4.14}$$

Apart from all these propellant consumption items, a certain amount of unusable propellant always exists, remaining in the tanks or trapped in the engine, pipelines and valves. In this thesis, unusable propellant mass is assumed as 5% of the total propellant loaded into the stage.

4.1.1.3 Aerodynamics Model

Aerodynamic forces arise due to the relative motion between the launch vehicle and the surrounding air and can actually be broken down into three components. These are called drag, lift and side force. The drag force is opposite to the direction of motion, while the lift force acts perpendicular to the direction of motion, and the side force is mutually perpendicular to the lift and the drag. If the launch vehicle's trajectory is maintained such that the velocity vector is parallel to the vehicle's body axis, i.e. angle of attack and side slip are zero, then no lift or side force is generated.

In this thesis, only the drag force is considered for simplification reasons. It is generally a valid assumption in the early design phases, since the launch vehicle trajectories are designed to keep the aerodynamic angles small during the atmospheric flight in order to avoid high aerodynamic loads. The expression for drag force is given by

$$\mathbf{D}(t) = -\frac{1}{2}\rho S C_D \left\|\mathbf{v}_{rel}(t)\right\| \mathbf{v}_{rel}(t) \tag{4.15}$$

In Eq. (4.15),

ρ is the atmospheric density at the altitude of interest,
S is the aerodynamic reference area,
C_D is the drag coefficient,
$\mathbf{v}_{rel}$ is the relative velocity vector,

The *relative velocity vector*, $\mathbf{v}_{rel}(t)$ with respect to the Earth's atmosphere is defined by

$$\mathbf{v}_{rel}(t) = \mathbf{v}(t) - \boldsymbol{\omega}_\mathbf{e} \times \mathbf{r}(t) \tag{4.16}$$

In Eq. (4.16),

$\mathbf{v}(t)$ is the inertial velocity vector,
$\boldsymbol{\omega}_e$ is the Earth's angular velocity vector relative to the inertial space,
$\mathbf{r}(t)$ is the inertial position vector.

Drag coefficients (C_D) are dependent on the size and shape of the vehicle and calculated for a reference area (S), which is usually the maximum cross sectional area of the body. They vary also with Mach number (M), altitude (h) and the angle of attack (α). Again for simplicity, variation of drag coefficients with respect to the angle of attack is not considered in this thesis.

The *Mach number* (M) is an important nondimensional aerodynamic parameter and defined as

$$M = \frac{v_{rel}}{a} \tag{4.17}$$

In Eq. (4.17),

v_{rel} is the relative speed,
a is the speed of sound at the altitude of interest.

The *speed of sound* (a) is defined as

$$a = \sqrt{kRT} \tag{4.18}$$

In Eq. (4.18),

k is the specific heat ratio for air,
R is the specific gas constant for air,
T is the temperature at the altitude of interest.

The aerodynamic prediction code Missile Datcom 97 is used to calculate the drag coefficients for the example vehicle configurations presented in this thesis. Datcom is a Fortran based computer program developed by McDonnell Douglas Corporation for the United States Air Force and is made available to the public domain as a supplement of the book of Hammond (2001).

Datcom is a semi empirical code having accuracy suitable for preliminary design and provide the users with the capability of modeling different vehicle configurations easily (Blake, 1998). However, an important limitation that should be highlighted is the fact that Datcom does not allow to model strap-on boosters. Therefore, they have to be analyzed separately and the coefficients of the boosters and the central core have to be summed up conveniently considering their reference areas.

Datcom can generate aerodynamic data for up to 20 different Mach numbers for a single configuration. Therefore, data for different vehicle configurations related to different phases of trajectory are stored in different look-up tables and the drag coefficients are interpolated by cubic spline functions from the relevant tabulated data based on Mach number.

4.1.1.4 Earth and Gravity Models

Two different Earth models are used in this study depending on the particular need. These are the spherical Earth model and the oblate Earth model. In the spherical Earth model, Earth is assumed as a perfect sphere with a constant radius, represented by Earth’s equatorial radius (R_e).

For spherical Earth, the gravitational force can be approximated from Newton's law of gravitation, which states that any two bodies attract each other with a force that is directly proportional to the product of their masses and inversely proportional to the square of the distance between them.

$$F = G\frac{Mm}{r^2} \tag{4.19}$$

In Eq. (4.19),
F is the force between the two bodies,
G is the universal gravitational constant,
M is the mass of the first body,
m is the mass of the second body,
r is the distance between the centers of the two bodies.

Or, in vector form

$$\mathbf{F} = -G\frac{Mm}{\|\mathbf{r}\|^3}\mathbf{r} \tag{4.20}$$

If M is the Earth's mass and m is the vehicle's mass, then *gravitational acceleration* (**g**), which is simply defined as the force of a unit mass due to gravity, can be solved from Eq. (4.20) by using Newton's law of motion.

$$\mathbf{g} = \frac{\mathbf{F}}{m} = -G\frac{M}{\|\mathbf{r}\|^3}\mathbf{r} = -\frac{\mu}{\|\mathbf{r}\|^3}\mathbf{r} \tag{4.21}$$

In Eq. (4.21), the product GM being a constant, is replaced by μ, which is called the Earth's gravitational parameter or geocentric gravitational constant. Eq. (4.21) is known as the Newtonian gravity model, and is valid under the assumption that the two bodies are particles whose masses are concentrated at their centers. Although the zonal harmonics perturbations used to consider the latitude variations due to nonspherical mass distribution are not modeled in the spherical Earth approximation, it is deemed adequate for relatively short term missions such as flight to the low Earth orbit.

In the numerical solution of the equations of motion, Newtonian gravity model is used as shown in Eq.(4.2), and each altitude of interest for the prediction of atmospheric pressure and density in Eqs. (4.7) and (4.15) is also calculated based on the spherical Earth assumption.

On the other hand, the oblate Earth model is used for deriving the outputs to be presented at the end of the optimization problem. These outputs are the geodetic latitude, geodetic altitude and the geodetic distance.

The oblate Earth model is a high accurate Earth model, which is based on the ellipsoidal approximation defined by the World Geodetic System 1984 (NIMA, 2000). In this model, the geoid shape of the Earth is represented as an oblate spheroid characterized by the parameters given in Table 4.1.

Table 4.1 Parameters of WGS 84 Earth model

Parameter	**Value**
Equatorial radius of Earth, R_e	6378137 m
Polar radius of Earth, R_p	6356752.3 m
Mean radius of Earth, $R_m = (2R_e + R_p)/3$	6371008.8 m
First eccentricity of the reference ellipsoid for Earth, e_e	0.08181919084
Earth's rotation rate, ω_e	7.29211585 x 10^{-5} rad/s
Geocentric gravitational constant (with Earth's atmosphere included), μ	398600.4 x 10^9 m^3/s^2

4.1.1.5 Atmosphere Models

In order to predict the aerodynamic loads on the vehicle, to consider the variation of thrust with altitude and to simulate vehicle's flight trajectory accurately, a good representation of the atmosphere is required. Atmosphere models typically provide air density, temperature and atmospheric pressure values primarily as a function of altitude. Temperature data is used to calculate the speed of sound and consequently the Mach number which is required to obtain the aerodynamic coefficients, air density data is used to compute the aerodynamic forces; whereas pressure data is used to determine the propulsive forces.

Various atmospheric models have been developed so far. The simplest atmospheric model is the exponential model, which represents the variation of the density and pressure as an exponential function of the altitude, but assumes constant temperature. Some other sophisticated models take into account the latitudinal, temporal, seasonal, geomagnetic and solar effects as well.

Since the aim of this thesis is to develop a trajectory optimization tool that allows for the evaluation of performance capabilities of different vehicles or the design of new vehicles for a specified mission, simulations are required to be performed under the same circumstances. Simplicity of the models is also important in terms of computational efficiency. Matlab offers several built-in functions for different types of atmosphere models. Two useful ones are the well known US Standard Atmosphere 1976, which was proposed by the United States Committee on Extension to the Standard Atmosphere (COESA, 1976) and NRLMSISE-00 atmosphere model developed by United States Naval Research Laboratory (Picone, Hedin, Drob, & Aikin, 2002).

NRLMSISE-00 (United States Naval Research Laboratory Mass Spectrometer and Incoherent Scatter Radar Exosphere) model calculates the neutral atmosphere empirical model from the Earth's surface to the lower exosphere up to 1,000 km of altitude. US Standard Atmosphere 1976 model serves to provide atmospheric parameters up to the geometric altitude of 86 km for fixed geodetic latitude of 45° North. Above 86 km altitude, it is common practice to extrapolate temperature values linearly and pressure values logarithmically. Density and speed of sound can then easily be calculated using the ideal gas law. Accuracy of the model above this altitude is not so critical, since the aerodynamic loads become ineffective in this region.

Matlab built-in function `'atmosnrlmsise00'` implements the mathematical representation of the NRLMSISE-00 model. Input arguments of this function are altitude, geodetic latitude, longitude and time in terms of year, day of year and seconds in day in universal time and the output arguments are temperature (T) and total mass density (ρ).

Because NRLMSISE-00 is valid up to 1000 km of altitude and it has higher fidelity than the US Standard Atmosphere 1976 in terms of latitudinal and temporal dependency, it was primarily chosen to be used during the trajectory computations in the frame of this thesis. But, since it does not provide pressure data, it was considered more practical to use both models together. The NRLMSISE-00 model was utilized to obtain temperature and density values, whereas another Matlab built-in function `'atmoscoesa'` which implements the mathematical representation of the US Standard Atmosphere 1976 was used to find pressure values for the input argument of altitude.

Transitory events such as wind were not taken into account in the atmosphere model because this would complicate the model a lot without bringing substantial advantage on the final result in terms of accuracy.

4.1.2 Objective Functions

As mentioned earlier, the trajectory optimization tool developed in the frame of this thesis can be utilized to solve two different problems. First problem aims to determine the optimal ascent trajectory and the payload capacity of an existing launch vehicle for different target orbits specified by the user, and is simply called trajectory optimization problem. While the second problem is an extension of the trajectory optimization problem and can be considered as the system level design of a new launch vehicle for a given mission characterized by the target orbit and the payload mass.

In the trajectory optimization problem, the objective function is to maximize the payload mass and can be formulated as

$$J = -m\left(t_f^{(P)}\right) \tag{4.22}$$

In Eq. (4.22),
$m(t_f^{(P)})$ is the mass at the end of mission where $t_f^{(P)}$ is the final time of the last phase P.

Payload mass maximization criterion is used to generate the performance maps of launch vehicles for a range of missions. These maps are then utilized by the payload planners to size the satellites during design phase and/or to determine the amount of propellant to be filled into the satellite for orbit maintenance maneuvers. The excess payload capacity can also be utilized as a secondary payload or additional sensors for housekeeping purposes.

In the launch vehicle design problem, the objective function is to minimize the gross lift-off mass and can be formulated as

$$J = m\left(t_0^{(1)}\right) \tag{4.23}$$

In Eq. (4.23),
$m(t_0^{(1)})$ is the mass at the beginning of mission where $t_0^{(1)}$ is the initial time of the first phase.

4.1.3 Optimization Variables

Optimization variables used in the problem formulation are listed in the following.

- Monotonically increasing independent variable: time - t
- Time-dependent variables: states - $\mathbf{x}(t)$ and controls - $\mathbf{u}(t)$
- Time-independent variable: static parameters - $\mathbf{s}$

The dynamic behavior of the launch vehicle's motion was described by Eqs. (4.1), (4.2) and (4.9) with respect to time. These differential equations involve the position ($\mathbf{r}$), velocity ($\mathbf{v}$) and mass (m) as state variables and the thrust direction ($\mathbf{u}$) as a control variable. As being vectors, position, velocity and thrust direction are defined by three rectangular coordinates, which yield a total of seven states and three controls for each time point.

As mentioned earlier in Section 4.1.1.1, although the equations of motion are formulated using rectangular ECI coordinates, the relative spherical coordinates and the orbital elements representing the position and velocity are also computed all along the trajectory. Moreover, attitude variables including the reduced Euler angles and the reduced aerodynamic angles are also derived utilizing the thrust vector and the relative velocity vector. Thus, the state and control variables used in this thesis can be summarized as follows. (Please refer to APPENDIX B for the definitions of the position and velocity variables and Section 4.1.1.1 for the definitions of attitude variables.)

- 3 sets of position and velocity variables (states)
    - Rectangular ECI coordinates: $\{r_x, r_y, r_z, v_x, v_y, v_z\}$
    - Spherical ECEF and geographic coordinates: $\{h, \lambda, \phi, v_{rel}, \chi, \gamma\}$ and d
    - Orbital elements: $\{a, e, i, \Omega, \omega, \theta^*\}$

- 3 sets of attitude variables (controls)
    - Rectangular ECI coordinates: unit vector of thrust direction - $\{u_x, u_y, u_z\}$
    - Euler angles: $\{\theta, \psi\}$
    - Aerodynamic angles: $\{\alpha, \beta\}$

In the trajectory optimization problem, maximized payload mass is formulated as the product of a factor and the estimated value of the payload mass. This factor is called the *payload scaling factor* (PLSF) and formulated as a static parameter. Whereas, in the launch vehicle design problem, thrust profile of each stage are optimized using static parameters. Details of the thrust profile model will be described in Section 5.2.1.

In the optimization process, state variables are determined by integrating the differential equations for each individual time step, whereas control variables and static parameters evolve during the iterations within the user defined bounds in order to find the optimal solution satisfying the constraints. Therefore, control variables and static parameters are also known as design variables.

4.1.4 Constraints

A general OCP is formulated as to determine the states, controls and static parameters that minimize the objective function subject to the following type of constraints.

- Dynamic constraints
- Boundary constraints
- Path constraints

4.1.4.1 Dynamic Constraints

The dynamic constraints are a set of ordinary differential equations in state-space form, which represents the simplest mathematical description of the system. In the trajectory optimization problem having a total number of P flight phases where $p \in [1, \ldots, P]$, the dynamic constraints are

$$\dot{\mathbf{r}}^{(p)} = \mathbf{v}^{(p)} \tag{4.24}$$

$$\dot{\mathbf{v}}^{(p)} = -\frac{\mu}{\left\|\mathbf{r}^{(p)}\right\|^3}\mathbf{r}^{(p)} + \frac{T^{(p)}}{m^{(p)}}\mathbf{u}^{(p)} + \frac{\mathbf{D}^{(p)}}{m^{(p)}} \tag{4.25}$$

$$\dot{m}^{(p)} = -\frac{T^{(p)}}{I_{sp} g_0} \tag{4.26}$$

4.1.4.2 Boundary Constraints

The boundary constraints or so called event constraints are equality or inequality type point constraints defined only for state variables. Initial and terminal constraints defined at the start and/or end time of each phase and the continuity constraints defined at the interior interfaces between phases are classified as boundary constraints. The following boundary constraints are imposed in this thesis.

4.1.4.2.1. Initial Constraints

The initial constraints are position and velocity coordinates at the beginning of the mission.

$$\mathbf{r}(t_0) = \mathbf{r}_0 = R_e[\cos\delta_0 \cos\lambda_0 \quad \cos\delta_0 \sin\lambda_0 \quad \sin\delta_0]^T \tag{4.27}$$

$$\mathbf{v}(t_0) = \mathbf{v}_0 = \boldsymbol{\omega}_\mathbf{e} \times \mathbf{r}_0 \tag{4.28}$$

In Eqs. (4.27) and (4.28),
R_e is the equatorial radius of the Earth,
δ_0 is the launch site’s geocentric latitude,
λ_0 is the launch site’s geodetic longitude,
$\mathbf{v_0}$ is the inertial velocity vector due to Earth’s rotation.

Eqs. (4.27) and (4.28) are formulated for ground launched vehicles in line with the examples presented in this thesis. In case of a launching from a carrier aircraft, these equations should be updated so as to include the initial altitude and velocity at the instant of launch.

4.1.4.2.2. Linkage Constraints

The interior point constraints also called phase continuity or linkage constraints are written for time and states.

$$t_f^{(p)} - t_0^{(p+1)} = 0 \tag{4.29}$$

$$\mathbf{r}^{(p)}(t_f^{(p)}) - \mathbf{r}^{(p+1)}(t_0^{(p+1)}) = \mathbf{0} \tag{4.30}$$

$$\mathbf{v}^{(p)}(t_f^{(p)}) - \mathbf{v}^{(p+1)}(t_0^{(p+1)}) = \mathbf{0} \tag{4.31}$$

$$m^{(p)}(t_f^{(p)}) - m_s^{(p)} - m^{(p+1)}(t_0^{(p+1)}) = 0 \tag{4.32}$$

In Eqs. (4.29) - (4.32),
p is the phase number and $p \in [1, \ldots, P\text{-}1]$,

Eqs. (4.29) - (4.31) enforce the time, position and velocity to be continuous, and also mass drops at the end of the applicable phases are considered with Eq. (4.32). No mass drop occurs in multiburn missions including a coasting phase between two powered phases of the same engine. Stage separation/jettison events are assumed to take place instantaneously at the end of phases for simplicity, i.e. there is no gap between the motor burn-out and the stage separation.

4.1.4.2.3. Terminal Constraints

The terminal constraints can be applied to both for the endpoint of the last phase and/or the endpoints of the intermediate phases. Terminal constraints corresponding to the target orbit are

$$\mathbf{r}^{(P)}(t_f^{(P)}) = \text{given} \tag{4.33}$$

$$\mathbf{v}^{(P)}(t_f^{(P)}) = \text{given} \tag{4.34}$$

All six components of position and velocity or a couple of them can be enforced as terminal constraints depending on the particular mission requirements. For missions including a transfer/parking orbit, two distinct terminal constraints for different phases can be defined representing the intermediate and the target orbits, leading to a multipoint boundary value problem.

Another terminal constraint taken into account is the maximum allowable amount of propellant to be used by the upper stage engine during orbit insertion, which is valid only for restartable liquid engines. The reason and the rules for applying this constraint were explained in Section 4.1.1.2.

Descent trajectories of the separated stages are solved and the instantaneous impact points are evaluated in the follow-on analyses as other terminal constraints. Separated stages must be dropped into the sea or unpopulated areas in order not to cause any hazards to people or property. If the resulting impact point is not acceptable, then the trajectory can be reshaped, or thrust time profile of stage motors can be adjusted if the vehicle parameters are free to optimize.

For descent trajectories, the differential equations (4.24) - (4.26) given for the powered ascent phase can be modified by omitting the thrust and the mass flow rate from the equations since there is no propulsion system operating during descent. Nominal impact points are calculated for the aerodynamically stable attitude of the separated stages that is uncontrolled motion such as tumbling is not considered.

Matlab built-in function `'ode45'` based on Runge-Kutta (4,5) method is used to integrate the differential equations of descent for a certain time interval with initial conditions of position and velocity at the separation instant and the mass of the separated stage. The integration is terminated when altitude equals to zero, which means that the separated stage hits the ground. Related Matlab code is given in Appendix C.3.3 for reference.

4.1.4.3 Path Constraints

The path constraints are equality or inequality type constraints defined for restricting the range of values taken by separate or combined functions of the state and the control variables. They can be imposed over the entire trajectory or within a particular flight phase. The following path constraints due to the definition of the mathematical models are taken into account in the problems solved within this thesis.

$$|\mathbf{r}| \geq R_e \tag{4.35}$$

$$\|\mathbf{u}\|^2 = u_x{}^2 + u_y{}^2 + u_z{}^2 = 1 \tag{4.36}$$

The state constraint in Eq. (4.35) guarantees to keep the vehicle’s altitude above the surface of the Earth, and the control constraint in Eq. (4.36) ensures that the thrust direction **u** is a unit vector. Trajectory or mission constraints arising from the limitations of the vehicle and the payload are described in the following sections.

4.1.4.3.1. Dynamic Pressure

The dynamic pressure (q) is a function of the air density (ρ) and the air speed (v_{rel}), and expressed as

$$q(t) = \frac{1}{2}\rho(t)v_{rel}^2(t) \le q_{\max} \tag{4.37}$$

Dynamic pressure acts on the outer surface of the vehicle and it is an important consideration of the structural loads. Aerodynamic stress, i.e. stress within the structure subject to aerodynamic forces is proportional to q. The point of maximum aerodynamic stress is often referred to as max q and it is one of the critical load conditions for launch vehicles. Therefore, maximum allowable dynamic pressure (q_{max}) is defined by the user as a constraint for trajectory optimization.

4.1.4.3.2. Angle of Attack

In order to avoid unrealistic trajectories with regard to the controllability and structural loads, the maximum angle of attack (α) is constrained especially within the atmospheric portion of the flight.

4.1.4.3.3. Bending Moment

Dynamic pressure and angle of attack constraints together introduce a third constraint which is known as $q\alpha$, which describes the limit of the product of the two and this term is considered as another critical load condition which is an indicator of bending moment due to dynamic pressure.

4.1.4.3.4. Axial Acceleration

The acceleration (a) is simply the force per unit mass. Since the thrust direction is assumed to be aligned with the vehicle's longitudinal axis within this thesis, and if the small angle of attack values are neglected then nongravitational acceleration can be calculated through the following equation and referred as the axial acceleration.

$$a(t) = \frac{T(t) - D(t)}{m(t)} \le a_{\max} \tag{4.38}$$

If Eq. (4.38) is divided by the sea level gravitational acceleration (g_0), the resulting relation corresponds to the axial g-load, which is a common measure of acceleration.

The majority of the total mass of the launch vehicle is the propellant mass. Assuming the thrust is constant; the acceleration of the vehicle increases when propellant is burnt and expelled. At this point, thrust must be reduced in order not to destroy the payload or the vehicle. The magnitude of the thrust can be controlled in response to the needs during flight if and only if throttling capability exists. Only some liquid engines have this capability, whereas the thrust of solid propellant motors can be adjusted by various grain shapes predetermined by design.

Implementation of acceleration constraint is different than the implementation of other constraints mentioned previously. This constraint should be actively taken into consideration during design phase, because it is due to the nature of the thrust model which is a part of the vehicle characteristics. It is difficult to satisfy acceleration constraint by trajectory shaping only and it causes considerable amount of performance loss.

4.1.4.3.5. Aerothermal Heat Flux

The aerothermal heat flux ($\dot{q}$) is a frequently used heat load constraint to determine the fairing jettisoning time. During the first minute of flight, the temperature of the payload fairing surrounding the spacecraft rises rapidly as a result of aerodynamic heating. Within two to five minutes after lift-off, the aerodynamic heating drops low enough and the fairing can be jettisoned. The aerothermal heat flux is a function of dynamic pressure (q) and air speed (v_{rel}), and it can be approximated in free molecular regime using the free stream enthalpy convective model (Cremaschi, 2013).

$$\dot{q}(t) = q(t) v_{rel}(t) = \frac{1}{2}\rho(t) v_{rel}^3(t) \leq \dot{q}_{\max} \qquad (4.39)$$

4.1.4.3.6. Total Aerodynamic Heating Rate

Another heat load constraint is the total aerodynamic heating rate (Q) which is defined as the integral of the heat flux over time.

$$Q(t) = \int_0^{t_f} \dot{q}(t) dt = \int_0^{t_f} \frac{1}{2}\rho(t) v_{rel}^3(t) dt \leq Q_{\max} \qquad (4.40)$$

Increase in total aerodynamic heating lead to increasing temperature, which causes structural degradation, and this constraint is often used to design the reentry trajectories of reusable launch vehicles. The total heating rate constraint was implemented successfully into the trajectory optimization code developed within this thesis, but was not applied to the any one of the example problems.

4.1.5 GPOPS-II Implementation

The numerical solution methods for OCPs were given in Section 2.1. Then, the existing research related to the application of these methods to the launch vehicle trajectory optimization problem was discussed in Section 2.3.2. As laid down therein, discretization is an integral part of the solution, and after discretization large number of function evaluations is required. Thus, computational efficiency of the solver is highly expected.

In this work, one of the recent methods known with their rapid convergence rates referred as pseudospectral methods were intended to use and examined in detail. The software

GPOPS-II “A General Purpose Matlab Toolbox for Solving Optimal Control Problems Using the Radau Pseudospectral Method” developed by Patterson and Rao (2013a) was finally chosen, since it can be used directly in Matlab and its high performance was demonstrated on different problems of varying complexity. This software is available to the academic researchers for a small amount of licensing fee.

GPOPS-II employs a Legendre-Gauss-Radau quadrature orthogonal collocation method where the continuous-time optimal control problem is transcribed to a large sparse NLP. The software can be interfaced with either quasi-Newton (first derivative) or Newton (second derivative) NLP solvers, and all derivatives required by the NLP solver are approximated using sparse finite-differencing of the OCP functions. The key components of the software, problem setup, input and output structures and syntax are described in details in the user’s manual of GPOPS-II prepared by Patterson and Rao (2013a).

In this thesis, various examples of the multistage launch vehicle ascent trajectory optimization problem defined in the preceding sections were solved by GPOPS-II with the NLP solver SNOPT developed by Gill, Murray and Saunders (2005) after discretized by the Radau pseudospectral method involved in GPOPS-II. Thus, a rapid design methodology including simultaneous optimization of trajectory and vehicle parameters were developed successfully.

GPOPS-II is not specifically designed to solve trajectories; it is a general purpose optimal control software. Therefore, a generic trajectory optimization code was first developed and then it was extended as a kind of launch vehicle design optimization code. Since these two codes serve to provide solutions to two different problems, the objective functions, optimization variables and the constraints used in the optimization framework differ, but the dynamic constraints are the same as formulated in the previous sections.

Each time a new problem is to be solved, numerical values for boundary and path constraints are first specified, and then initial guesses for the values of time, states, controls, integrals and static parameters are assigned for each phase by the user to initiate the iterative solution of the problem. It is also useful to limit the dynamic variables by imposing reasonable lower and upper bounds in order to improve the robustness.

According to GPOPS-II problem structure, the lower and upper bounds of time are provided separately for the start and end of the phase, whereas states require additional bounds also for the interior of the phase. Bounds on controls, integrals and path constraints are specified only for the entire phase. Bounds and guesses of static parameters are defined for the entire problem, since they are independent of phase.

The guesses can be the endpoints or any number of points along the path; therefore even an entire trajectory can be given as initial guess. If only the endpoints, namely the initial and final points are specified as a guess, then the guess is defined either a constant or a straight line over the time interval. A wide range of Earth orbit missions briefly described in Appendix A.3 were solved by the tool developed within this thesis and presented below.

4.2 Example 1: Delta III – GTO Mission

In order to evaluate the convenience of GPOPS-II for the ascent trajectory optimization of a launch vehicle and describe the methodology elaborately, a simplified example from the open literature that was studied by different authors was utilized. In his doctoral thesis, Benson (2004) solved the optimal ascent trajectory for the Delta III launch vehicle. The same problem was also included as an example in the software GPOPS-II developed by Patterson and Rao (2013b) and in the book of Betts (2010).

4.2.1 Vehicle Properties

The Delta III launch vehicle has two main stages along with nine strap-on solid rocket boosters as illustrated in Figure 4.3 (Delta III payload planners guide, 1997).

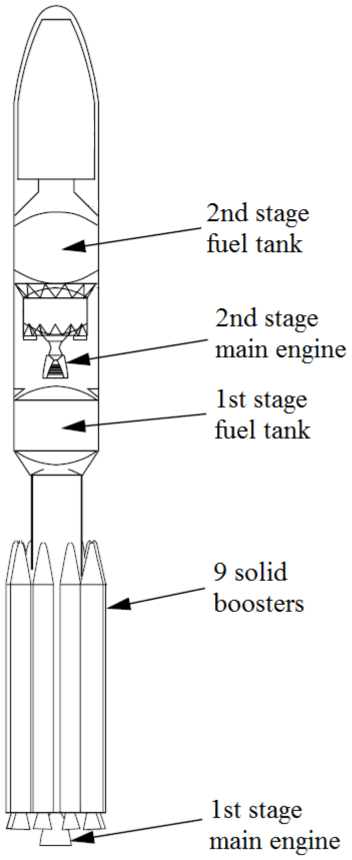


Figure 4.3 Delta III launch vehicle
(adapted from Delta III payload planners guide, 1997)

Mass and propulsion characteristics of the rocket stages used in Delta III launch vehicle are given in Table 4.2 (Rao, et al., 2010).

Table 4.2 Mass and propulsion characteristics of Delta III

	Booster	**1st stage**	**2nd stage**
Propellant type	Solid	Liquid	Liquid
Number	9	1	1
Propellant mass, m_p (kg)	17010	95550	16820
Structural mass, m_s (kg)	2280	8830	2480
Specific impulse, I_{sp} (sec)	283.3	301.7	467.2
Burn time, t_b (sec)	75.2	261	700 (max)
Payload mass m_{pl} (kg)	4164		

4.2.2 Mission Characteristics

One of the typical missions of Delta III is the direct injection into the geostationary transfer orbit (GTO). Launch is from Cape Canaveral and the flight trajectory includes four successive powered phases without any coasting periods.

The first phase begins with the rocket at rest on the ground at time t_0, when the main engine and the six of the nine solid boosters (x6) ignite. At time t_1, the boosters are depleted and their dry mass is jettisoned. The second phase begins when the remaining three solid boosters are ignited. Three solid boosters (x3) burn simultaneously with the main engine and when their fuel is exhausted at time t_2 their inert mass is ejected. Then, the main engine alone creates the thrust for the third phase. After the ejection of the inert mass of empty 1st stage at time t_3, the fourth phase begins with the second stage ignition. The thrust during fourth phase is from the second stage, which burns until the target orbit is reached at time t_4, thus completing the trajectory. Flight sequence for this mission is summarized in Figure 4.4 and Table 4.3.

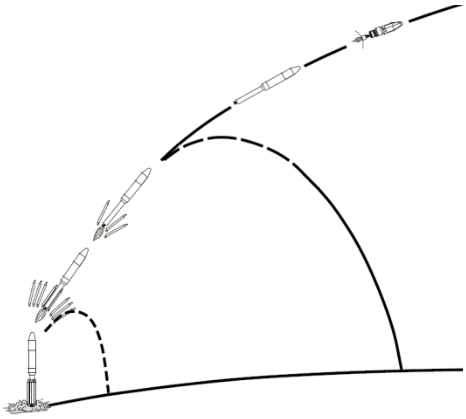


Figure 4.4 Delta III flight sequence

Table 4.3 Delta III GTO flight sequence

Event	**Time**
Ignition of 1st stage main engine Ignition of boosters (x6) Lift-off	$t_0 = 0$ sec
Boosters (x6) burn-out Boosters (x6) separation Ignition of boosters (x3)	$t_1 = t_0 + 75.2$ sec
Boosters (x3) burn-out Boosters (x3) separation	$t_2 = t_0 + 150.4$ sec
1st stage main engine cut-off 1st stage main engine separation	$t_3 = t_0 + 261$ sec
Target orbit is achieved	t_4: free ($t_{4,\max} = 961$ sec)

It is important to note that the solid boosters and the main engine operate during their entire burn time (t_1, t_2 and t_3 are fixed), while the second stage engine is shut off when the target orbit is achieved (t_4 is free), thus unused fuel can remain in the tanks.

Delta III ascent trajectory optimization problem's objective is to maneuver the launch vehicle from the ground to the target orbit while maximizing the remaining fuel in the upper stage. No trajectory constraints were applied.

4.2.3 Problem Formulation

Slightly different from the disciplinary models described in Section 4.1.1, the following assumptions were made in this first example in order to simplify the problem, reduce the computational time and focus on the trajectory optimization only.

- Dynamics is same during all flight phases, only thrust and mass change.
- Magnitude of thrust is constant within a phase, only direction changes.
- Mass flow rate is constant within a phase.
- Thrust magnitude does not depend on altitude (or atmospheric pressure).
- Drag coefficient is constant (independent of Mach number) for the entire trajectory and taken as a conservative value of $C_D = 0.5$.
- Aerodynamic reference area is constant and equal to the cross sectional area of the payload fairing having a diameter of 4 m, i.e. $S = 4\pi \text{ m}^2$.
- Exponential atmosphere model described below is implemented.

$$\rho = \rho_0 \cdot e^{\left(-h/H\right)} \tag{4.41}$$

In Eq. (4.41),

ρ_0 is the atmospheric density at sea level, i.e. $\rho_0 = 1.225 \text{ kg/m}^3$

h is the altitude, i.e., $h = r - R_e$,

H is the atmospheric scale height, i.e. $H = 7200$ m.

Since the mass flow rate ($\dot{m}$) is assumed to be constant, it can be easily calculated for each motor using the relevant data in Table 4.2.

$$\dot{m} = \frac{m_p}{t_b} \tag{4.42}$$

In Eq. (4.42),

m_p is the propellant mass,

t_b is the burn time.

Thrust can be calculated from Eq. (4.8) with the known values of I_{sp} and $\dot{m}$.

$$T = I_{sp} \cdot g_0 \cdot \dot{m} \tag{4.43}$$

The thrust (T) and the mass flow rate ($\dot{m}$) within each phase are then expressed as follows.

Phase 1:

$$0 = t_0 \le t \le t_1 = 75.2$$

$$T = 6T_b + T_1 \tag{4.44}$$

$$\dot{m}^{(1)} = 6\dot{m}_b + \dot{m}_1 \tag{4.45}$$

Phase 2:

$$75.2 = t_1 \le t \le t_2 = 150.4$$

$$T = 3T_b + T_1 \tag{4.46}$$

$$\dot{m}^{(2)} = 3\dot{m}_b + \dot{m}_1 \tag{4.47}$$

Phase 3:

$$150.4 = t_2 \le t \le t_3 = 261$$

$$T = T_1 \tag{4.48}$$

$$\dot{m}^{(3)} = \dot{m}_1 \tag{4.49}$$

Phase 4:

$$261 = t_3 \le t \le t_{4,\max} = 961$$

$$T = T_2 \tag{4.50}$$

$$\dot{m}^{(4)} = \dot{m}_2 \tag{4.51}$$

In Eqs. (4.44) - (4.51), subscripts b, 1 and 2 stands for the booster, 1st stage and 2nd stage motors, respectively. The payload mass is given and fixed in this example, only the remaining fuel in the last stage is to be maximized, thus the initial mass of the launch vehicle is known and fixed.

$$m(0) = m_{10} = 9m_b + m_1 + m_2 + m_{pl} \tag{4.52}$$

According to the vehicle definition, mass changes at time events are also known. Only the final mass is unknown being an optimization variable.

$$m_{1f} = m_{10} - \dot{m}^{(1)} \cdot t_1 \tag{4.53}$$

$$m_{20} = m_{1f} - 6m_{s,b} \tag{4.54}$$

$$m_{2f} = m_{20} - \dot{m}^{(2)} \cdot \left(t_2 - t_1\right) \tag{4.55}$$

$$m_{30} = m_{2f} - 3m_{s,b} \tag{4.56}$$

$$m_{3f} = m_{30} - \dot{m}^{(3)} \cdot \left(t_3 - t_2\right) \tag{4.57}$$

$$m_{40} = m_{3f} - m_{s,1} \tag{4.58}$$

Launch site is located at the Cape Canaveral Space Station in Florida whose geocentric latitude and longitude are δ_0 = 28.5° and λ_0 = -80.6°, respectively. It is assumed for simplicity that the ECEF frame is initially aligned with the ECI frame at time t_0 such that the X axis passes through the intersection of the equator with the meridian line at Cape Canaveral longitude. Thus, the inertial longitude can be taken as zero at time of lift-off, provided that it should then be added to the longitude values in the calculated optimal trajectory. Finally, the position and velocity vectors can be defined as

$$\mathbf{r_0} = \mathbf{r}(0) = 6378137 \cdot [\cos 28.5 \quad 0 \quad \sin 28.5]^T \tag{4.59}$$

$$\mathbf{v}_0 = \mathbf{v}(0) = 7.29211585 \text{ x } 10^{-5} \mathbf{k} \times \mathbf{r}(0) \tag{4.60}$$

The launch vehicle starts on the ground at rest (relative to the Earth) and Eq. (4.60) is the inertial velocity at lift-off, which ensures the relative velocity in Eq. (4.16) is zero. At the final time t_4, the launch vehicle must insert the payload into a GTO defined by the following orbital elements, descriptions of which are given in Appendix A.1.

$$a_f = 24361140 \text{ m} \tag{4.61}$$

$$e_f = 0.7308 \tag{4.62}$$

$$i_f = 28.5° \tag{4.63}$$

$$\Omega_f = 269.8° \tag{4.64}$$

$$\omega_f = 130.5° \tag{4.65}$$

$$\theta_f^* = \text{free} \tag{4.66}$$

Eqs. (4.61) - (4.65) define the final position and velocity of the payload to lie in the desired orbit. The true anomaly (θ^*), which represents the position of the payload along the orbit at injection time, is left as a free parameter in the optimization. Because the only concern in this example is to achieve the specified orbit, location within the orbit is not constrained.

Continuity in the position and velocity from phase to phase is enforced by imposing interior point constraints.

$$\mathbf{r}^{(1)}(t_1) = \mathbf{r}^{(2)}(t_1) \quad ; \quad \mathbf{v}^{(1)}(t_1) = \mathbf{v}^{(2)}(t_1) \tag{4.67}$$

$$\mathbf{r}^{(2)}(t_2) = \mathbf{r}^{(3)}(t_2) \quad ; \quad \mathbf{v}^{(2)}(t_2) = \mathbf{v}^{(3)}(t_2) \tag{4.68}$$

$$\mathbf{r}^{(3)}(t_3) = \mathbf{r}^{(4)}(t_3) \quad ; \quad \mathbf{v}^{(3)}(t_3) = \mathbf{v}^{(4)}(t_3) \tag{4.69}$$

It is noted that the continuity constraint on the mass at each phase interface includes an instantaneous drop of the dry mass of the particular stage. In this case, the mass drops at the end of phases 1, 2 and 3 are given respectively as $6m_{s,b}$, $3m_{s,b}$ and $m_{s,1}$.

$$m^{(1)}(t_1) - m^{(2)}(t_1) = 6m_{s,b} \tag{4.70}$$

$$m^{(2)}(t_2) - m^{(3)}(t_2) = 3m_{s,b} \tag{4.71}$$

$$m^{(3)}(t_3) - m^{(4)}(t_3) = m_{s,1} \tag{4.72}$$

The objective of the optimization problem is to determine the control vector **u***(t)*, and final time t_f to maximize the mass at the end of the fourth phase

$$J = -m\left(t_f^{(4)}\right) \tag{4.73}$$

subject to the dynamic constraints in Eqs. (4.24) and (4.25) together with Eqs. (4.44) - (4.51), the boundary constraints in Eqs. (4.59) - (4.72) and the physical path constraints in Eqs. (4.35) and (4.36).

4.2.4 Bounds and Guesses

Since the initial and final time of first three phases are fixed, lower and upper bounds and guesses for time were simply set to the certain known values. For the last phase, upper bound of final time was determined based on the maximum burn time of 2nd stage engine.

$$t_3 < t_4 < t_{4,\max} \tag{4.74}$$

The position and velocity in rectangular ECI coordinates were restricted by the following simple bounds to obtain realistic solutions.

$$-2R_e \le r_x \le 2R_e \quad ; \quad -2R_e \le r_y \le 2R_e \quad ; \quad -2R_e \le r_z \le 2R_e \tag{4.75}$$

$$-10\,\tfrac{\text{km}}{\text{s}} \le v_x \le 10\,\tfrac{\text{km}}{\text{s}}\,; -10\,\tfrac{\text{km}}{\text{s}} \le v_y \le 10\,\tfrac{\text{km}}{\text{s}}\,; -10\,\tfrac{\text{km}}{\text{s}} \le v_z \le 10\,\tfrac{\text{km}}{\text{s}} \tag{4.76}$$

The guesses for position and velocity in the first two phases were taken as constant and equal to the initial position ($\mathbf{r}_0$) and velocity ($\mathbf{v}_0$) given in Eqs. (4.59) and (4.60), while in the subsequent phases they were again taken as constant, but equal to the target orbit's position and velocity which can easily be obtained from Eqs. (4.61) - (4.66) by an appropriate transformation. The guess for free parameter θ^* was taken as simply equal to zero, since the orbit injection usually occurs near perigee point as explained in Appendix A.4.

$$\mathbf{r}_{\mathbf{f},guess} = [4397.3 \quad 4243.8 \quad 2379.5]^T \text{ km} \tag{4.77}$$

$$\mathbf{v}_{\mathbf{f},guess} = [-5826.7 \quad 7819.6 \quad -3178.4]^T \text{ m/s} \tag{4.78}$$

With the known values of initial and final mass in all phases that are given in Eqs. (4.52) - (4.58), the upper and lower bounds on mass were simply set to initial and final mass, respectively. On the other hand, the guesses for mass were given as a straight line between the initial and final mass over phase duration. The lower and upper bounds for controls were specified by definition as -1 and +1, respectively. While, the guesses for controls was arbitrarily set to a constant vector **u** = [0 1 0]^T in all phases.

4.2.5 Results and Discussion

Problem was successfully solved by GPOPS-II with the NLP solver SNOPT after assigning the bounds and initial guesses for the dynamic variables of each phase as described in Section 4.2.4.

GPOPS-II output summarizes the important characteristics of the solution obtained by GPOPS/SNOPT algorithms. According to the output summary, initial mesh consisting of ten uniformly spaced mesh intervals with four Legendre-Gauss-Radau points per mesh interval that is a total of 40 nodes per phase was used. Since the accuracy tolerance of 10^{-6} was satisfied on the initial mesh, no mesh refinement was performed to reduce the discretization error.

The resulting sparse NLP problem had 1636 nonlinear variables and 1313 nonlinear constraints. This nonlinear problem required 40 major iterations, which was completed in 5.6 seconds of CPU time on a computer with a quad-core hyperthreading capable 2.67 GHz i7 processor (i.e. 8 virtual cores) and 12 GB of memory. The total solution time including the time for discretization and scaling of the NLP variables was 8.8 seconds. This solution time was obtained by using the one eighth of the processing power, since the used optimization algorithms could not have been parallelized.

Figure 4.5 and Figure 4.6 illustrate the optimal solution for position $\mathbf{r}(t)$ and velocity $\mathbf{v}(t)$ in rectangular ECI frame, respectively. The optimal steering commands $\mathbf{u}(t)$ representing the thrust direction are also plotted in Figure 4.7. Results match quite well with the ones obtained by Benson (2004), Betts (2010), Rao et al. (2010) and Patterson and Rao (2013b) all of whom implemented different collocation and NLP algorithms. Benson (2004) and Rao et al. (2010) solved this problem by gauss pseudospectral method, whereas Betts (2010) applied the sparse optimal control (SOCS) algorithm. Finally, Patterson and Rao (2013b) utilized Radau pseudospectral method, which is implemented also in GPOPS-II.

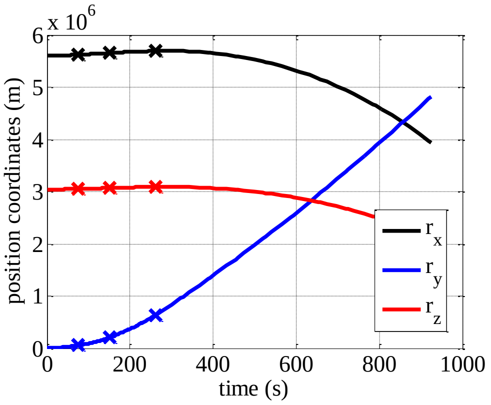


Figure 4.5 Delta III position profile

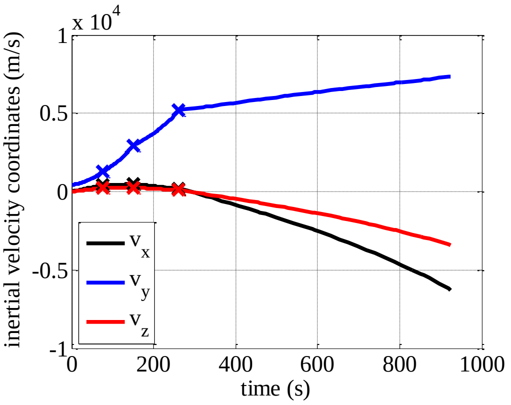


Figure 4.6 Delta III inertial velocity profile

Figure 4.8 displays the mass time profile indicating the optimal value of the objective function, which is the final mass. Final mass is obtained as 7530 kg when time t_f is equal to 924.1 sec. Subtracting the payload mass m_{pl} and the structural mass of upper stage $m_{s,2}$ from the final mass, the remaining fuel in the tank is found as **886 kg** and the burn time of the second stage, which can be at most 700 sec, is found as **663.1 sec**. As a conclusion, it is important to emphasize that the remaining fuel can be utilized as payload in the future missions if the launch vehicle follows the proposed optimal trajectory.

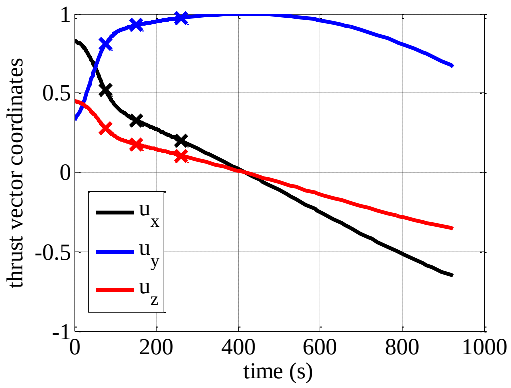


Figure 4.7 Delta III control time history

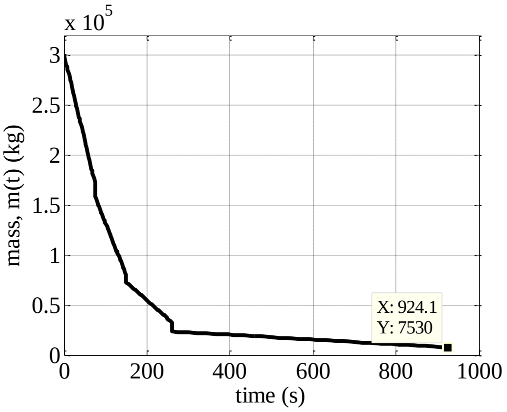


Figure 4.8 Delta III mass profile

Table 4.4 shows at what extent the terminal constraints regarding the target orbit were satisfied. It is easily seen that the desired orbital parameters are exactly the same. The important point here is that the value of free parameter, true anomaly (θ^*) was obtained as almost 6.8°, which indicates that the satellite is injected into the orbit after it passes the perigee point.

Table 4.4 Delta III target orbit parameters

Orbital element	**Desired**	**Result**
Semimajor axis (a)	24361140 m	24361139.9 m
Eccentricity (e)	0.7308	0.730799
Inclination (i)	28.5°	28.499°
Right ascension of the ascending node (Ω)	269.8°	269.800°
Argument of perigee (ω)	130.5°	130.500°
True anomaly (θ^*)	free	6.7834°

Time histories of altitude and speed given in Figure 4.9 and Figure 4.10 also support this conclusion such that the terminal altitude is around 200 km and the terminal inertial speed is around 10.24 km/s. Because if the satellite was injected into the orbit at its perigee point, the corresponding altitude and inertial speed should be around 180 km and 10.26 km/s, respectively according to Eqs. (4.77) and (4.78). This comparison shows that injection from perigee point is not optimal in terms of payload mass maximization, since the orbital velocity needed to get into the orbit increases with decreasing altitude.

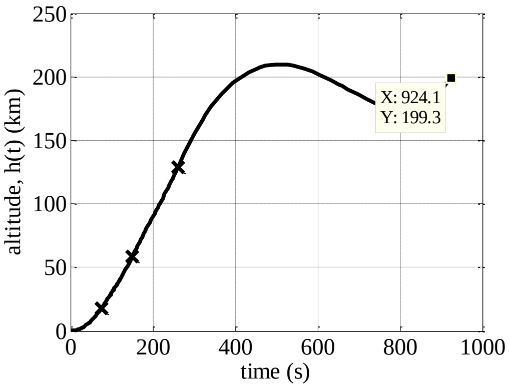


Figure 4.9 Delta III altitude profile

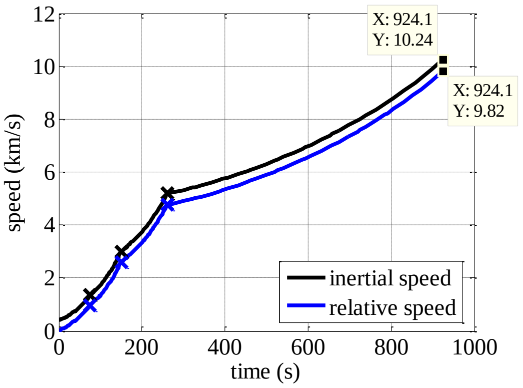


Figure 4.10 Delta III speed profile

Figure 4.11 illustrates the latitude versus longitude and Figure 4.12 illustrates the altitude versus range including the descent trajectories of the separated stages and the impact points.

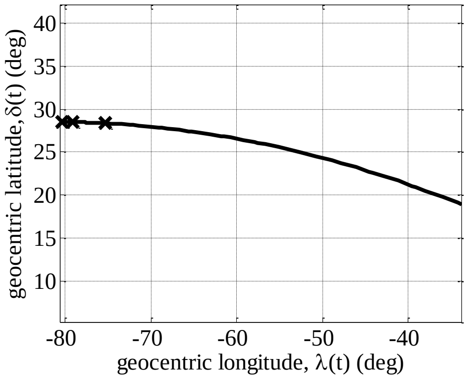


Figure 4.11 Delta III latitude vs. longitude

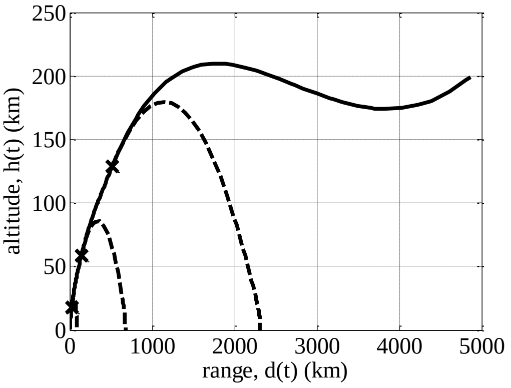


Figure 4.12 Delta III altitude versus range

Figure 4.9 also notifies an interesting oscillation in altitude during the last phase of flight. This type of trajectory is known as *lofted trajectory* and many launch vehicles today follow such kind of an optimal trajectory. In such a case, the vehicle climbs first to very high altitudes and then starts descending before ascending to the orbit while accelerating towards burn-out. Trajectory lofting is frequently associated with vehicles which have a low upper stage thrust to weight ratio as Patton and Hopkins (2006) emphasized. However, this kind of trajectory is unsafe for manned flights since it does not allow the passenger vehicle to continue on a ballistic trajectory at the time of the abort. Therefore, the upper stages of launch vehicles for manned flights are designed as more powerful, and consequently a more slanted and a softer trajectory is obtained.

Figure 4.13 and Figure 4.14 show the time histories of flight path angle and heading angle respectively. Although no mission constraints were imposed in this example, acceleration and dynamic pressure profiles are plotted in Figure 4.15 and Figure 4.16 for information.

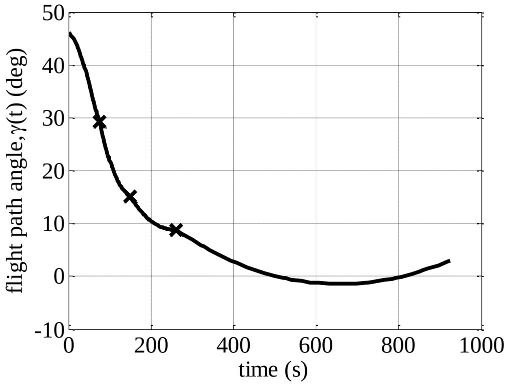


Figure 4.13 Delta III flight path angle

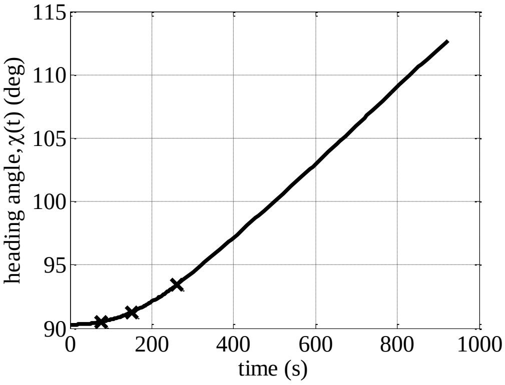


Figure 4.14 Delta III heading angle profile

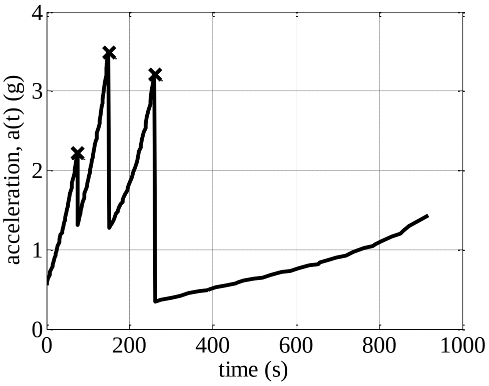


Figure 4.15 Delta III acceleration profile

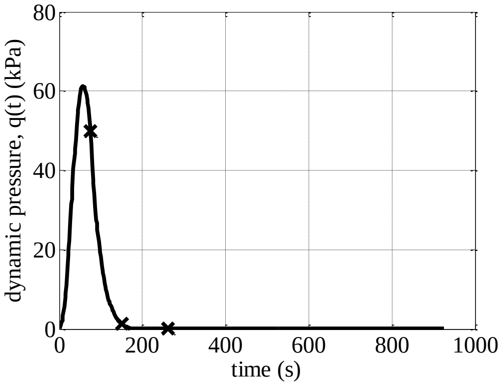


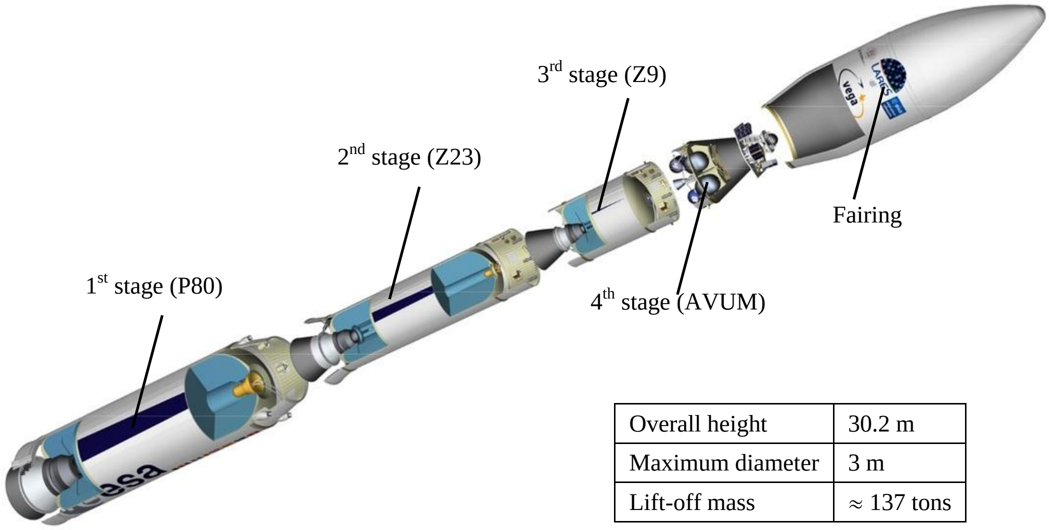
Figure 4.16 Delta III dynamic pressure profile

4.3 Vega Missions

After a simplified problem given in Section 4.2, different missions of the European Vega launch vehicle were solved, and the results were compared with the available data in the open literature. In these examples, higher fidelity engineering models were implemented, and the fine details of the mission scenarios were tried to be applied. The main objective is to verify and validate the trajectory optimization tool developed within the scope of this thesis and to assess its suitability for real-world applications.

4.3.1 Vehicle Properties

Vega is a new generation launch vehicle developed to provide highly efficient access to low Earth orbits for small payloads. It consists of three solid propellant lower stages and a restartable liquid propellant upper stage called AVUM – attitude and vernier upper module and it is illustrated in Figure 4.17.



Overall height	30.2 m
Maximum diameter	3 m
Lift-off mass	≈ 137 tons

Figure 4.17 Vega launch vehicle
(adapted from http://www.spaceflight101.com/vega-vv02-launch-updates.html)

The three solid propellant rocket motors (P80, Z23 and Z9) in the lower stages are responsible for executing the main ascent phase of the launch vehicle, while the upper stage is fired to reach an elliptical transfer orbit. The upper stage AVUM has restart capability up to five times, thus it provides mission flexibility by orbital maneuvering and ensures precise orbit injection. After the spacecraft separation, it is deorbited to comply with the debris mitigation policies. This module also houses the attitude control system and the avionics of the launch vehicle, and has an additional task of compensating the trajectory dispersions due to performance variations of the first three stages.

The physical and performance characteristics of the rocket stages used in Vega launch vehicle are summarized in Table 4.5 (Vega user's manual, 2014).

Table 4.5 Physical and performance characteristics of Vega

	1st stage	**2nd stage**	**3rd stage**	**AVUM**
Propellant type	Solid	Solid	Solid	Liquid
Propellant mass, m_p (kg)	87710	23814	10567	577
Structural mass, m_s (kg)	8533	2486	1433	688
Vacuum specific impulse, I_{sp} (sec)	280	287.5	295.9	314.6
Burn time, t_b (sec)	109.9	77.1	119.6	667 (max. for continuous burn)
Stage diameter (m)	3	1.9	1.9	1.9
Nozzle exit diameter (m)*	1.98	1.47	1.23	0.3
Vacuum thrust, T_{vac} (kN)	See Figure 4.18			2.45
Mass flow rate, $\dot{m}$ (kg/s)	See Figure 4.18			0.794
Payload fairing mass, m_{plf} (kg)	540			

* taken from the doctoral dissertation of Castellini (2012)

Thrust profiles of solid propellant motors P80, Z23 and Z9 were obtained from the paper of Fiorillo, Giliberti, Angelone, Milana and Serraglia (2013). Although the values are missing on the plots given in this paper, they were easily derived based on the values of maximum thrust and burn time of the motors with the help of a program for digitizing graphs called GetData graph digitizer. The mass profiles were also calculated with the assumption of constant specific impulse from Eq. (4.9). The resulting plots of variations of thrust and remaining propellant mass with respect to time are shown in Figure 4.18.

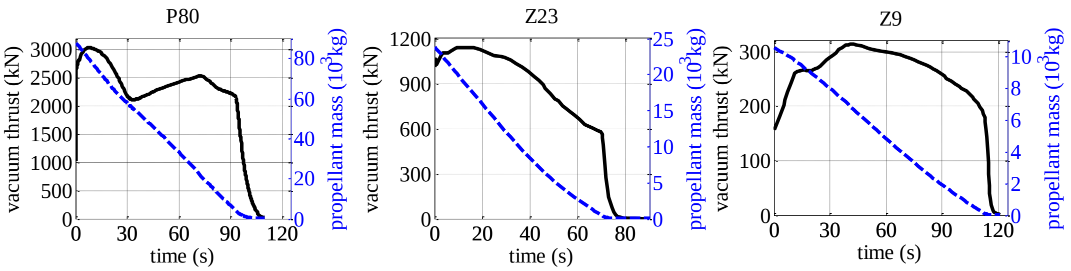


Figure 4.18 Thrust and propellant mass profiles of solid rocket motors of Vega

Thrust and mass profiles were loaded into the trajectory optimization code as tabulated data, and cubic interpolation was performed when a new point is needed to integrate the differential equations. The effective thrust in the atmosphere was also computed within the code using Eq. (4.7) in order to take into account the pressure losses. The corresponding nozzle exit areas were calculated based on the exit diameters given in Table 4.5.

The amount of propellant in the upper stage excluding the 5% unusable portion was completely allowed to be consumed. The maximum amount available for orbit insertion was determined considering the amounts needed for the trajectory corrections and the deorbiting maneuvering. Since these amounts depend upon the current mass of the launch vehicle, they were computed specific to each mission in the optimization code according to the approach described in Section 4.1.1.2. On the other hand, the propellant of attitude control system was included into the inert mass assuming as if it was going to be carried till the end of mission although it is partly ejected during flight.

For deorbiting, entry interface was specified such that the upper stage reenters the atmosphere establishing a -3° flight path angle at an altitude of 100 km representing the boundary between the Earth's atmosphere and the outer space. Descent trajectory of the upper stage was not solved since it was considered that these entry conditions ensure the disintegration and burning up of the remaining upper stage in the atmosphere.

The drag coefficients for five different configurations related to different phases of trajectory were calculated by Datcom. These configurations are the launch vehicle as a whole, launch vehicle excluding the first stage and the separated 1st, 2nd and 3rd stages. The aerodynamic forces were included in the equations of motion only in the first two phases of flight up to the second stage separation, because the dynamic pressure is very low in the rest of the flight.

The external geometry of the launch vehicle was constructed from the dimensions indicated on the drawings available in the presentation of Vega qualification mission (Lares, 2009). The coefficients were calculated for the reference area of $S = 7.07\ \text{m}^2$ for the ascent phase considering the 3 m diameter of the 1st stage, while the reference areas for the separated stages were simply obtained from the values of stage diameters given in Table 4.5. The input geometries and the coefficients predicted by Datcom are given in Figure 4.19 and Figure 4.20, respectively.

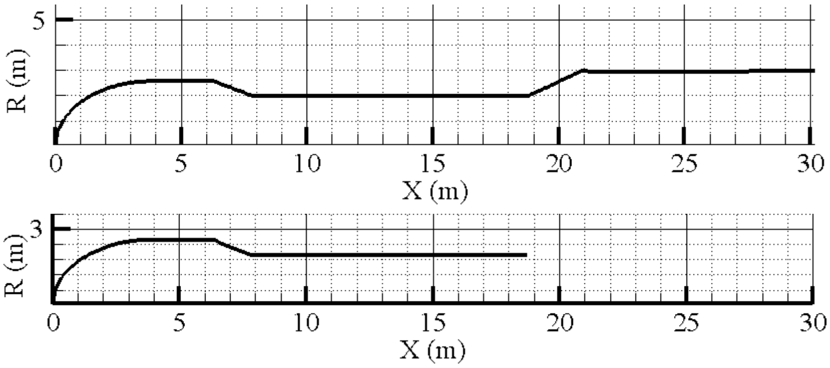


Figure 4.19 Aerodynamic configurations of Vega

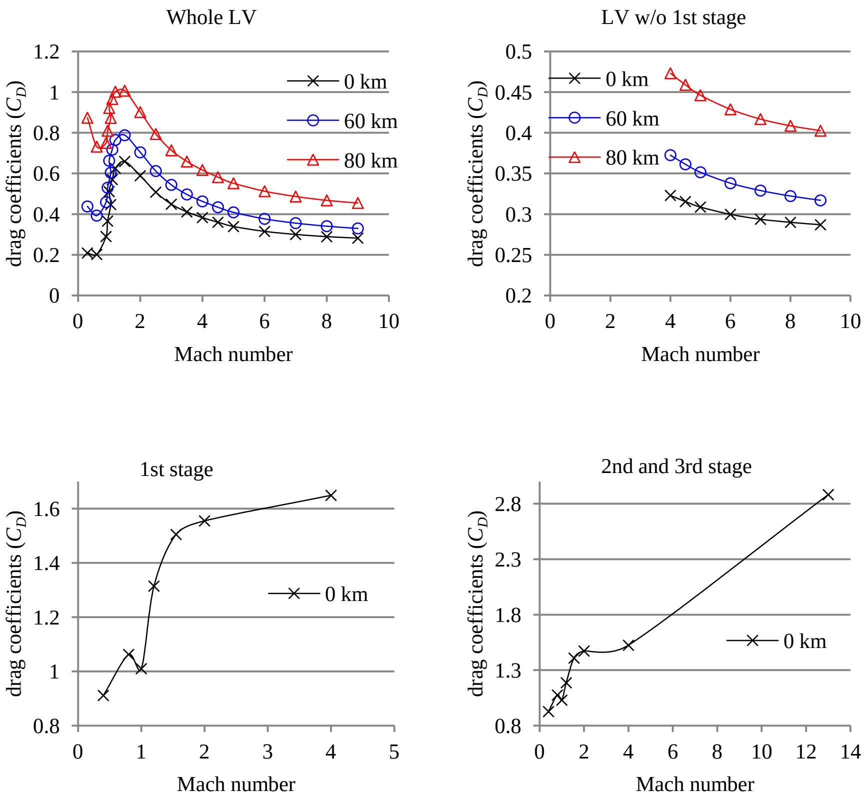


Figure 4.20 Drag coefficients of Vega

4.3.2 Mission Characteristics

Vega launch vehicle was designed primarily to carry about 1500 kg payload into polar low Earth orbit at an altitude of 700 km, but it is also capable of delivering satellites (from 300 kg to 2500 kg) into a wide variety of orbits from circular to elliptical and from equatorial to sunsynchronous. The first launch (maiden flight) of Vega took place in February 2012, and two more launches had successfully conducted till today. The following three mission examples are covered in the succeeding sections.

- Low Earth elliptical transfer orbit (LETO) with perigee altitude of 200 km, apogee altitude of 1500 km and 5.4° inclination
- Polar Earth orbit (PEO) at 700 km altitude (also known as reference mission)
- Sunsynchronous Earth orbit (SSO) at 750 km altitude

The ELA1 launch complex located in Kourou, French Guiana is used for Vega launches. Its geographic position is 5.236°N and 52.775°W (Wikipedia, 2014). After a transformation of latitude from geodetic to geocentric, the initial position coordinates are

$$\delta_0 = 5.201^\circ \quad ; \quad \lambda_0 = \text{-}52.775^\circ \tag{4.79}$$

The physical environment where the Vega launch vehicle operates is defined by the Earth and gravity models described in Section 4.1.1.4 and the atmosphere models described in Section 4.1.1.5.

Having a restartable liquid engine in the upper stage, multiburn orbit insertion scenarios become possible for Vega. As mentioned in Appendix A.4, a coasting phase can be inserted into the mission in order to increase the payload performance. In such kind of missions, the launch vehicle is first injected into an intermediate orbit by the first burn of the upper stage. And, after a certain period of time of coasting along this orbit, the second burn is performed to insert the spacecraft into the target orbit.

For circular low Earth orbit missions, the AVUM flight typically consists of two burns (burn-coast-burn trajectory arcs) to reach the target orbit. Whereas in case of elliptical orbits, a single AVUM boost can perform the orbit injection. The flight sequence varies according to the mission to be deployed, but may be summarized as in Table 4.6 and illustrated in Figure 4.21.

Table 4.6 Typical flight sequence of Vega

Event	**Time**
P80 ignition and lift-off	t_0 = 0 sec
P80 burn-out and separation Z23 ignition	$t_1 = t_0 + 106.7 = 106.7$ sec
Z23 burn-out and separation Start of coast	$t_2 = t_1 + 76 = 182.7$ sec
Payload fairing separation Z9 ignition	t_3^*
Z9 burn-out and separation Start of coast	$t_4 = t_3 + 116.8$ sec
AVUM 1st ignition (perigee firing and trajectory corrections)	$t_5 = t_4 + 15$ sec
AVUM 1st cut-off (injection into transfer orbit)	t_6^*
AVUM 2nd ignition (circularization burn)	t_7^*
AVUM 2nd cut-off (injection into target orbit)	t_8^*
Payload separation	$t_9^†$
AVUM 3rd ignition (deorbiting)	$t_{10}^†$

[*] to be optimized in order to achieve the maximum performance

[†] depend on pointing and collision avoidance requirements that are out of scope of this work

The durations of the solid motor powered phases given in Table 4.6 were specified according to the burn time values in Table 4.5. However, it is important to note that they were set slightly lower than the actual burn time. Because as seen from Figure 4.18, thrust level drops so much and almost diminishes to zero towards the end of burn, and this leads to significant problems in the optimization. For powered phases within the atmosphere, a certain level of thrust is always required; otherwise singularities may occur in the control variables especially when the thrust data are given as tabulated form. Therefore, few data points at the end of burn were removed from the propulsion input data. Although this is not a perfect solution, it is easy to apply, and as a matter of fact the very low thrust values in this region do not contribute to the ascent performance at all.

On the other hand, a short-time coasting period may exist between the burn-out of a stage and the ignition of the next one in order to provide a necessary clearance distance between the separated stages against undesirable effects due to plume reflection. In Vega missions, coasting is not allowed after 1st stage separation since the aerodynamic forces are still significant and may cause a considerable performance loss. Therefore, 2nd stage is ignited immediately after the separation, and the retro motors are used to pull the empty 1st stage away from the vehicle.

Vega missions have ten to thirty seconds duration of gaps after burn-outs of 2nd and 3rd stage, although five to fifteen seconds of coasting after separation is common in current launchers. A secondary purpose of coasting between 2nd and 3rd stage powered phases is to provide favorable conditions for payload fairing release without loss of performance. As mentioned in Section 4.1.4.3.5, the free molecular heat flux at fairing jettison should not exceed a certain value. For this reason, the first short coast after 2nd stage burn-out was optimized depending on the mission requirements, whereas a fixed duration of 15 seconds was added for safe separation before the 1st ignition of AVUM in all Vega mission examples.

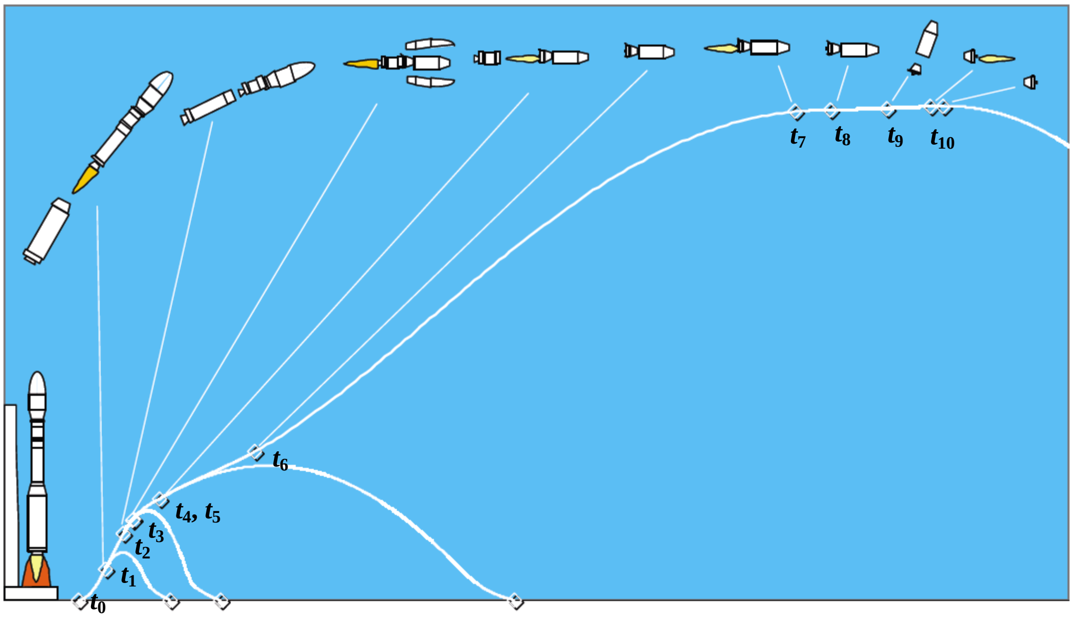


Figure 4.21 Vega flight sequence

(adapted from http://dma.dima.uniroma1.it:8080/users/lsa_eps/MATERIALE/Vega-20-11-2008.pdf)

As disclosed in the timeline of Vega qualification mission, the payload fairing is released at the beginning of the Z9 flight phase – a couple of seconds after the ignition – when the aerothermal heat flux becomes lower or equal to the typical value of 1135 W/m^2 (Lares, 2009). In the examples within this thesis, the payload fairing was jettisoned at a predefined time closer to the real mission scenario, but coinciding with any one of the phase endpoints. For example in Vega missions, fairing separation was assumed to be occurred at the end of coasting after 2nd stage separation and at the same time instant with the 3rd stage ignition. Additionally, terminal and path constraints were applied for the relevant and subsequent phases in order to ensure that the heat flux does not exceed 1135 W/m^2 after payload fairing jettisoning.

$$\dot{q}(t) \leq 1135 \frac{\mathrm{W}}{\mathrm{m}^2} \tag{4.80}$$

In the Vega mission examples solved in this thesis, the objective function was formulated as to maximize the payload mass. Therefore, both the trajectory parameters and the payload mass are design variables. Consequently, the initial mass of the launch vehicle is unknown, and thus this problem is more complex than the Delta III example even just because of this difference.

The target orbit was defined by the first three orbital elements, namely semimajor axis (a), eccentricity (e) and inclination (i), and only these three were imposed as terminal constraints in the optimization. The right ascension of the ascending node (Ω), argument of perigee (ω) and the true anomaly (θ^*) were left free in order to evaluate the optimal solution corresponding to the maximum payload mass. As a matter of fact, the orbit plane position can be adjusted as desired by specifying the sidereal time at launch.

The maximum angle of attack and the maximum bending moment indicator were constrained during the atmospheric portion of the trajectory corresponding to the first two powered phases of flight before the fairing jettisoning.

$$-5° \leq \alpha(t) \leq 5° \quad \text{(endoatmospheric flight)} \tag{4.81}$$

$$-60\ \mathrm{kPa}° \leq q(t) \cdot \alpha(t) \leq 60\ \mathrm{kPa}° \tag{4.82}$$

During the exoatmospheric flight, i.e. after the fairing jettisoning, only the angle of attack was constrained to 15° since the dynamic pressure decreases low enough.

$$-15° \leq \alpha(t) \leq 15° \quad \text{(exoatmospheric flight)} \tag{4.83}$$

The problem formulation was not described here in details, since and it was given in Section 4.1 and exemplified in Section 4.2.3 with Delta III mission. The differences from the Delta III example in terms of the disciplinary models and the optimization framework have already been elaborated herein before.

4.3.3 Bounds and Guesses

The bounds and guesses that are common in all three Vega mission examples are given in this section, while the differences will be discussed in the relevant sections specific to each mission example. The bounds introduced on position, velocity and controls are exactly the same as the previous Delta III example.

$$-2R_e \le r_x \le 2R_e \quad ; \quad -2R_e \le r_y \le 2R_e \quad ; \quad -2R_e \le r_z \le 2R_e \tag{4.84}$$

$$-10\,\tfrac{\text{km}}{\text{s}} \le v_x \le 10\,\tfrac{\text{km}}{\text{s}}\,; -10\,\tfrac{\text{km}}{\text{s}} \le v_y \le 10\,\tfrac{\text{km}}{\text{s}}\,; -10\,\tfrac{\text{km}}{\text{s}} \le v_z \le 10\,\tfrac{\text{km}}{\text{s}} \tag{4.85}$$

$$-1 \le u_x \le 1; -1 \le u_y \le 1; -1 \le u_z \le 1 \tag{4.86}$$

Until AVUM's first ignition, only the duration of the separation coast after 2nd stage burn-out is variable, and it was optimized by allowing it to vary between 15 and 50 seconds. The lower limit was assigned considering the required amount of time for safe separation, and the upper limit was not kept very flexible since it will result in loss of performance.

$$15\,\text{sec} \le t_f^{(3)} \le 50\,\text{sec} \tag{4.87}$$

$$t_f^{(5)} = 15\,\text{sec} \tag{4.88}$$

Estimated value of the payload mass was simply set to 1000 kg and the payload scaling factor (PLSF) being formulated as a static parameter was allowed to vary in the range between 0.01 and 3, which corresponds to 10 kg or 3000 kg of payload. Since the payload mass is variable in the optimization, the initial and final mass are also variable and unknown. Estimated values (guesses) for the initial and final mass of each phase were calculated based on the estimated value of the payload mass, and the upper and lower bounds of the mass were set to three times of the initial mass and one tenth of the final mass, respectively.

$$0.01 \le PLSF \le 3 \tag{4.89}$$

$$0.1m_f \le m \le 3m_0 \tag{4.90}$$

The initial guess of the static parameter PLSF was normally taken as 1.0, however this value is subject to change in order to improve the speed of convergence. The upper bound for maximum amount of propellant allowed to be consumed by the upper stage engine was constrained with the usable fraction (95%) of total loading.

$$m_p^{(6)}\left(+m_p^{(8)}\right) + m_{p,corrections} + m_{p,deorbit} \le 548.15\,\text{kg} \tag{4.91}$$

The estimated values of the free orbital parameters, the right ascension of the ascending node (Ω) and the argument of perigee (ω) were calculated using Eqs. (A.6) and (A.7), respectively. Although the input arguments δ and λ correspond to the target orbit's position, they were arbitrarily set to the latitude and the longitude of the launch site. The true anomaly (θ^*) was also taken as simply equal to zero in the estimation of terminal position and velocity, since the orbit injection usually occurs near perigee point.

The guesses for position and velocity in the last phase were taken as constant and equal to the terminal position and velocity, while they were simply set to the initial position ($\mathbf{r}_0$) and velocity ($\mathbf{v}_0$) in the other phases. On the other hand, the guesses for controls was arbitrarily set to a constant vector $\mathbf{u} = [0\ 1\ 0]^T$ in all phases.

All Vega mission examples were solved by GPOPS-II with NLP solver SNOPT in the first derivative mode. Derivatives required by SNOPT were approximated by GPOPS-II using sparse central finite-differencing of the optimal control functions. The results were obtained employing the implicit integration form of the Radau collocation method together with ‘hp1’ adaptive mesh refinement scheme. NLP tolerance was set to 10^{-5} with 2,000,000 maximum iterations, and mesh refinement accuracy tolerance was set to 10^{-3} in all examples. The number of collocation points in each problem was specified as giving better convergence, and automatic scaling procedure of GPOPS-II was implemented.

4.3.4 Example 2: Vega – LETO Mission

A direct ascent mission consisting of a single AVUM boost was chosen as the first example. The target orbit in this example is a low Earth elliptical transfer orbit (LETO) defined by

$$h_{a,f} = 1500\,\text{km} \tag{4.92}$$

$$h_{p,f} = 200\,\text{km} \tag{4.93}$$

$$i_f = 5.4° \tag{4.94}$$

The orbital elements a and e can easily be calculated from apogee and perigee heights h_a and h_p using Eqs. (A.1) - (A.4).

The flight trajectory is composed of a total of six phases, four of which are powered and the remaining two are the coasting periods after the separation of 2nd and 3rd stages. Mission ends after the first burn of AVUM when the target orbit is achieved. Therefore, the final time is free and the duration of AVUM burn is to be optimized together with the payload mass.

Duration of the last phase was restricted by the maximum continuous burn duration of AVUM given in Table 4.5. Let P be the number of the last phase, then the lower and upper bounds on final time will be

$$t_f^{(P-1)} < t_f^{(P)} < t_f^{(P-1)} + 667\,\text{sec} \tag{4.95}$$

The problem was solved with an initial mesh of consisting of 60 Legendre-Gauss-Radau points. The optimality conditions and the mesh error tolerance were satisfied on the initial mesh and thus, no mesh refinement was necessary. The NLP problem arising after collocation had 3295 nonlinear variables and 3112 nonlinear constraints. The computation time for solving this example on a computer whose specifications are given in Section 4.2.5 was around 535 seconds, 480 seconds of which was the NLP solution time. When the profiling option of Matlab was activated to track the execution time of the code pieces, the function consuming the most time was identified as `‘atmosnrlmsise00’`. It is important to note that the time spent for this function took approximately 242 seconds with a total of 90,000 calls, which highlights the fact that the run time performance can be improved by changing the atmosphere model in the future.

The state and control solutions obtained using GPOPS-II are shown in from Figure 4.22 to Figure 4.25. Cross markers existing in these graphs indicate the boundaries of the flight phases.

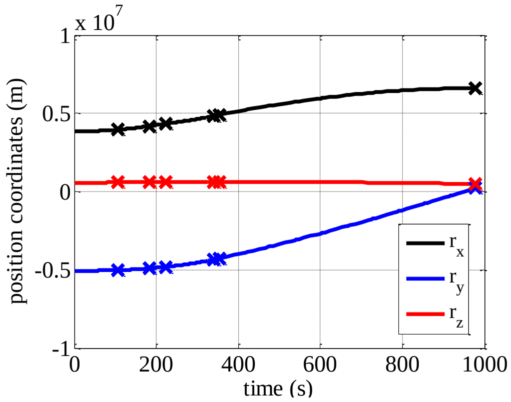


Figure 4.22 LETO position profile

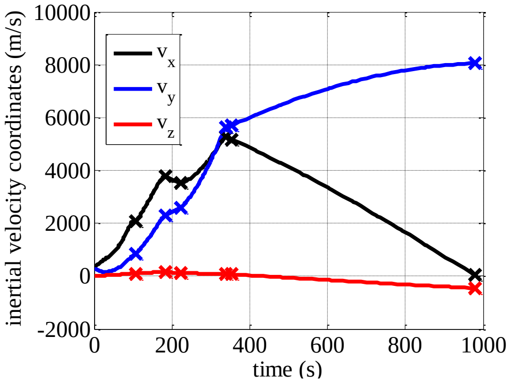


Figure 4.23 LETO inertial velocity profile

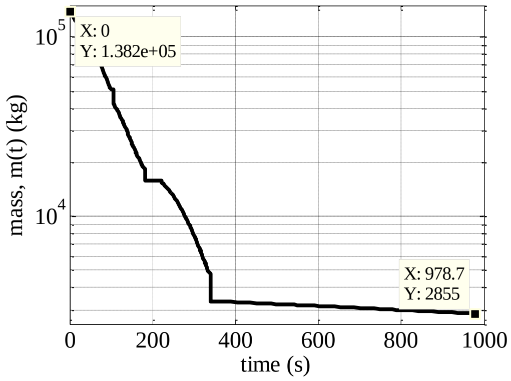


Figure 4.24 LETO mass time history

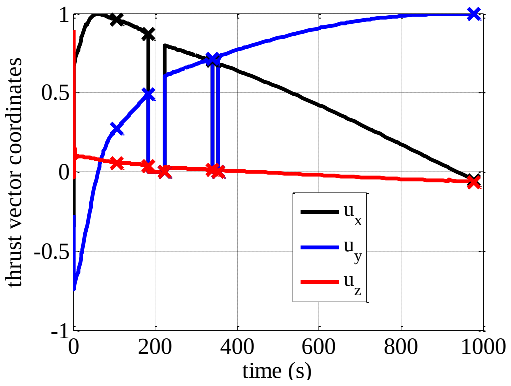


Figure 4.25 LETO control time history

Table 4.7 gives a brief summary of the optimized mission. The durations of powered and unpowered, i.e. coasting phases can also be comprehended from the mass time profile given in Figure 4.24. This profile clearly indicates the coasting phases with time invariant mass after 2nd and 3rd stage separations, and also the jettisoning of the empty stages and the payload fairing at the end of relevant phases. Coasting phases are also remarkable in Figure 4.25 with zero control values.

Table 4.7 Vega – LETO mission flight phases summary

#	**Duration (seconds)**	**Phase description**	**Active constraints**
1	0 – 106.7	Powered flight of P80	$-5° \leq \alpha \leq 5°$ -60 kPa° $\leq q\alpha \leq 60$ kPa°
2	106.7 – 182.7 (76)	Powered flight of Z23	
3	182.7 – 222.3 (39.6*)	Separation coast (fairing separation at end)	$\dot{q}_{end} \leq 1135$ W/m^2
4	222.3 – 339.1 (116.8)	Powered flight of Z9	-
5	339.1 – 354.1 (15)	Separation coast	-
6	354.1 – 978.7 (624.6*)	AVUM burn for orbit insertion	a_f, e_f, i_f ($h_{a,f}$, $h_{p,f}$, i_f)

* optimized in order to achieve the maximum performance

As seen from Table 4.7, the optimized durations of the 3rd and 6th phases corresponding to the free flight time and the AVUM burn are obtained as **39.6 sec** and **624.6 sec**, respectively. On the other hand, the orbital elements of the final orbit are given in Table 4.8 together with the comparison of initial guesses and the desired and optimized values.

Table 4.8 Vega – LETO mission target orbit parameters

Orbital element	**Desired**	**Guess**	**Result**
Semimajor axis (a)	7228137 m	7228137 m	7228137 m
Eccentricity (e)	0.0899	0.0899	0.0899
Inclination (i)	5.4°	5.4°	5.4000°
RAAN (Ω)	free	232.9066°	233.3630°
Argument of perigee (ω)	free	74.3796°	102.7518°
True anomaly (θ^*)	free	0°	25.8102°

Table 4.8 shows that the terminal position constraint is definitely satisfied, and the desired orbit is achieved with the true anomaly value of around 25° corresponds to an altitude considerably higher than its perigee height (see also the altitude profile given in Figure 4.30). Variations of perigee and apogee altitudes and the orbital inclination with respect to time during entire trajectory are given in Figure 4.26 and Figure 4.27, respectively.

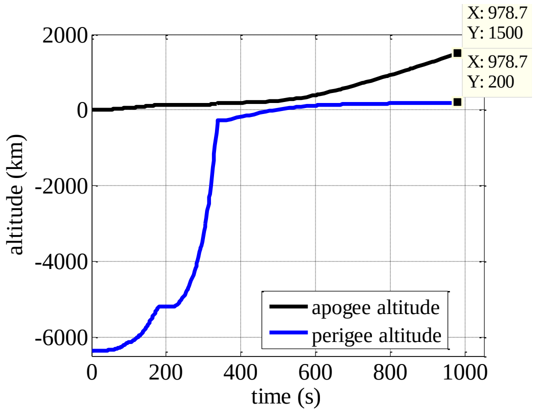


Figure 4.26 LETO perigee & apogee heights

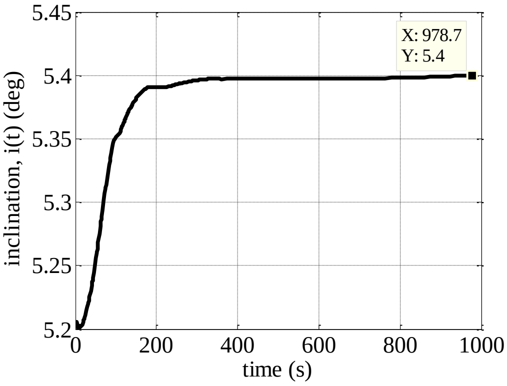


Figure 4.27 LETO orbit inclination profile

The initial guesses of the free parameters in Table 4.8 might give the impression that they are so close to the corresponding optimal solution; however it can be concluded that the problem is not sensitive to these guesses, because it was able to be solved by imposing a wide range of guesses without any distinguishable differences on convergence properties and computational time. Figure 4.28 depicts the inertial and relative speed profiles highlighting the latitude dependent initial (inertial) velocity gain due to Earth's rotation.

A summary of the propellant budget of AVUM in LETO mission is reported in Table 4.9. Excluding the unusable fraction and the amounts reserved for trajectory corrections and deorbiting, it is seen that all the available propellant has been utilized for orbit insertion. The duration of AVUM burn (624.6 sec) can also be cross checked from the consumed amount of propellant during orbit insertion and the mass flow rate given in Table 4.5.

Table 4.9 Vega – LETO mission AVUM propellant budget

Mission event	**Propellant mass (kg)**
Orbit insertion	495.8594
Trajectory corrections	38.9617
Deorbiting	13.3289
Unusable	28.8500
Remaining	0
Total	577

Figure 4.24 indicates the gross lift-off mass (including the payload mass) and the objective function that is final mass and equal to 2855 kg. Considering 688 kg structural mass of the

4th stage and the unusable and reserved propellant amounts given in Table 4.9, the optimized payload mass is obtained as **2086 kg**.

The relevant performance data for the same mission is given as 1963 kg in the user's manual of Vega (2014). When these two values are compared, the performance estimate obtained through trajectory optimization method presented in this thesis is about 6% (123 kg) higher than the actual performance of the Vega launcher. This residual error should not be evaluated as just an overestimation of the performance due to inaccuracies in vehicle properties and the lack of several modeling features, because performance maps declared in user's manuals often have a certain amount of safety margins.

On the other hand, the nominal performance maps for the whole mission envelope given in user's manuals are generally computed by means of standard guidance laws although they provide near optimum solutions rather than the optimal ones. Standard guidance laws are specifically designed to handle the ascent trajectory optimization problems more quickly and effectively, and they are characterized by a small number of optimizable parameters representing the attitude controls. Their reduced complexity and robust performance speed up the optimization process considerably, and they offer similar performance results as full trajectory optimization.

As presented in the paper of Gallucci, Battie, Volpi, Fossati and Curti (2012), the ascent trajectory of Vega's maiden flight was designed by following a very common reference guidance program implemented and proven successfully in many mission planning studies (Edge and Powers, 1976; Markl, 2001; Well, 2003; Castellini, 2008).

This reference trajectory begins with lift-off and vertical ascent. The *lift-off* takes place when the thrust equals the vehicle's weight. Just after lift-off, a *vertical ascent* is required for a safe clearing of the launch pad, avoiding that the exhaust gases hit ground structures around the pad itself. Duration of this phase is desired to keep as short as possible, since it causes high gravity losses.

The *pitch-over* maneuver is necessary in order to deflect the trajectory out of the vertical and establish a well-defined kick angle and the required azimuth for the successive gravity turn maneuver. A typical pitch program is performed by means of a constant pitch rate and a constant heading for certain duration. Pitch rate, launch azimuth and maneuver duration are the optimization variables during this phase.

The *gravity turn* maneuver corresponds to a zero angle of attack turn due to the gravity force that allows the minimization of the drag losses and the structural loads. This guidance strategy is applied up to the end of 2nd stage flight of Vega, although it is nonoptimal from the performance point of view, but easy to implement.

During the 3rd stage flight and subsequent AVUM firing, the attitude program is optimized by means of a closed loop guidance in order to achieve the maximum performance as well as satisfying the mission requirements and constraints. The injection into the transfer or the target orbit is achieved by *target inclination* maneuvering such as linear or bilinear tangent laws or the velocity to-be-gained approach also known as Q-guidance.

The method proposed in this thesis allows control variables to be optimized at all discretization points. And, since they do not have to stick to any rule, it is possible to obtain the real optimum. The outputs of trajectory optimization tool developed in the scope of this thesis are plotted alongside the performance results given in the user's manual of Vega (2014) (Figure 4.29 - Figure 4.31). In these graphs, black lines labeled as 'result' refer to the outputs of the newly developed tool, while the blue lines labeled as 'reference' refer to the original data of Vega.

Comparing the overall altitude profiles in Figure 4.30 reveals remarkable differences during the second half of the trajectory which corresponds to the target inclination maneuvering. It can easily be concluded from the graphs that although the target orbits are same in terms of size, shape and inclination (200 km x 1500 km; 5.4°), the altitudes where the vehicle enters the orbit are different. The relative speed profiles in Figure 4.29 also assist to draw the same conclusion, since the point closer to the Earth on the same orbit has higher speed than the farther one.

It is well known that not all the vehicle's thrust power can be used for velocity increment, but there exist several loss mechanisms incurred mainly due to the external aerodynamic and gravitational forces. Figure 4.30 illustrates that the altitude rate in reference trajectory in high altitudes is much greater than the optimal result, and the vehicle climbs into the less dense atmosphere earlier than the optimal solution. This reveals that the steeper and slower trajectory of the guidance law solution incurs lower drag but higher gravity losses compared to the optimal solution. Since gravity is more dominant than drag at high altitudes, the guidance law trajectory is exposed to higher losses than the optimal trajectory in the entire flight and leads to less performance.

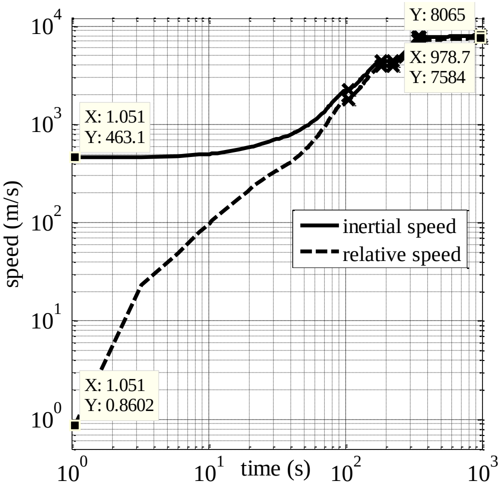


Figure 4.28 LETO speed profile

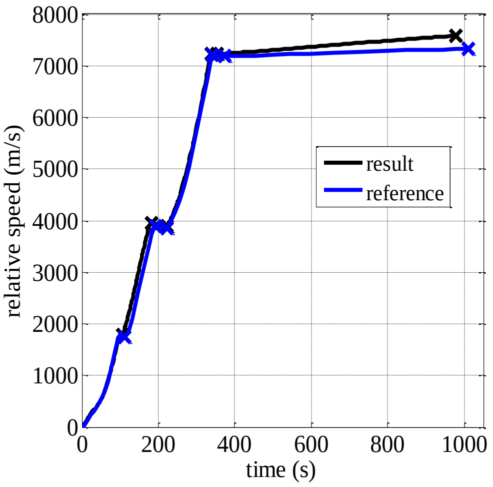


Figure 4.29 LETO relative speed comparison

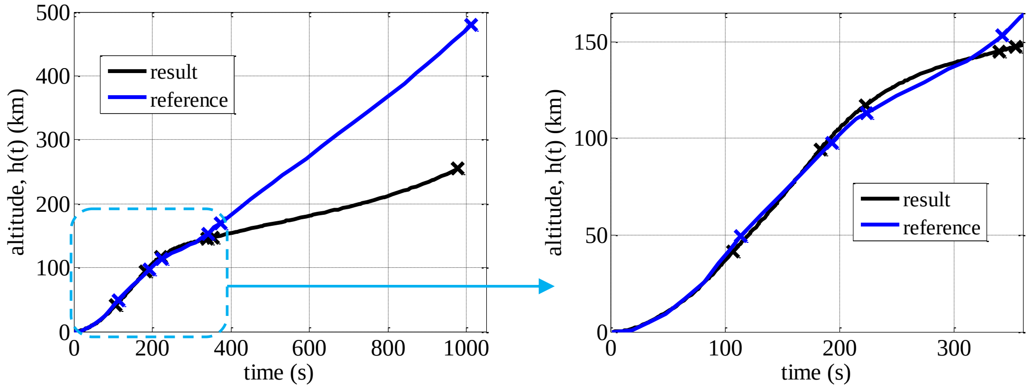


Figure 4.30 LETO altitude comparison

Figure 4.31 shows the ground track comparison of the launch vehicle's flight trajectory projected on the world map. It is seen that solutions are in close agreement excluding a small difference at the end of the mission. The vehicle flies over land in the optimal trajectory as labeled on the map; however it would not be a problem about the range safety since the flight altitude is sufficiently high.

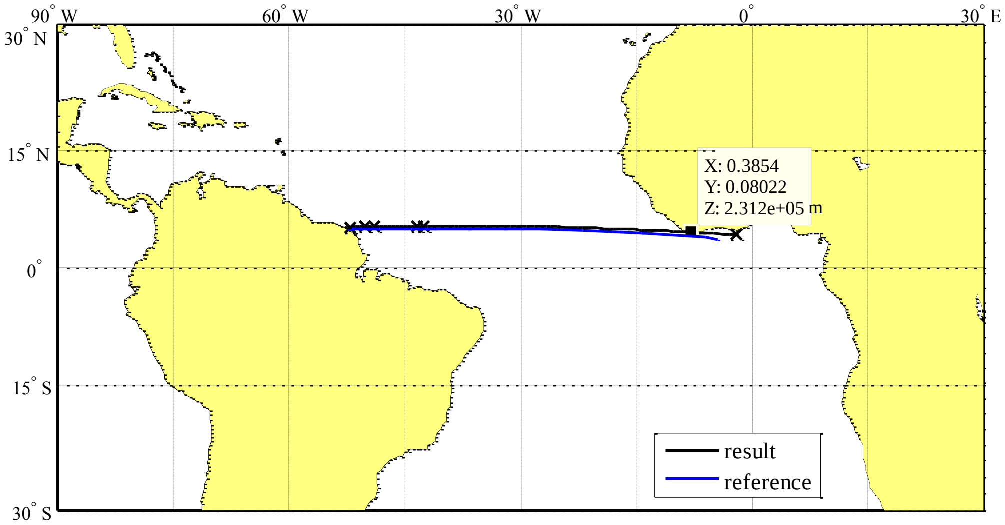


Figure 4.31 LETO launch vehicle's ground track comparison

Table 4.10 includes the geodetic position coordinates of the nominal impact points (IP) of the jettisoned empty stages, and Figure 4.32 illustrates the ground tracks and the nominal impact points on the world map. It is seen that all stages are dropped into the ocean sufficiently far from the coast without introducing any hazard. Impact point of the payload fairing was not solved, since it would exhibit a similar behavior to that of the separated 2nd stage. The similarity stems from the fact that the separated 2nd stage follows nearly the same trajectory with the vehicle during the separation coast phase; the only discrepancy would be due to the different aerodynamic properties. Figure 4.33 displays the altitude downrange profile. Altitude values are given with respect to Earth's equatorial radius, i.e. R_e = 6378.137 km.

Table 4.10 Vega – LETO mission separable parts impact points

	Longitude (°)	**Geodetic latitude (°)**	**Downrange (km)**
1st stage	-49.80	5.31	330.6
2nd stage	-40.05	5.43	1410.9
3rd stage	47.61	-0.15	11171.9

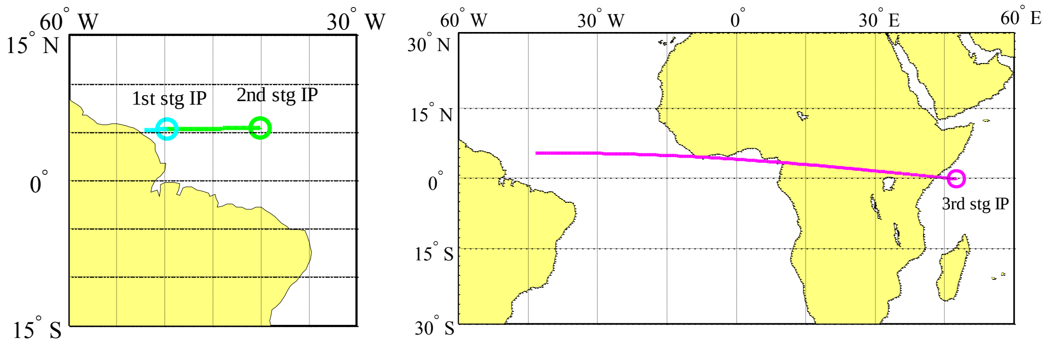


Figure 4.32 LETO separated stages impact points

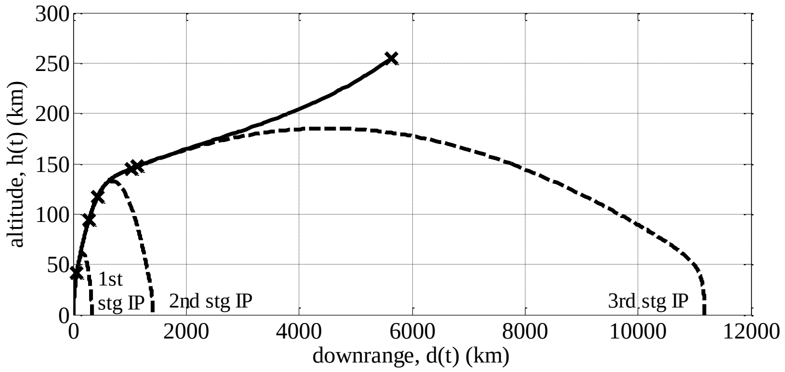


Figure 4.33 LETO altitude vs. downrange (incl. separated stages)

Other important trajectory parameters are presented in Figure 4.34 to Figure 4.39. Figure 4.34 shows the variation of thrust during the solid stage powered phases. Vacuum thrust is the interpolated value for each integration time step from the propulsion input data, and the corrected thrust profile is the effective thrust dependent on altitude, which is an input to the trajectory simulations. Figure 4.35 shows the acceleration profile with a maximum at around 4 g which is a quite moderate value for satellite launchers.

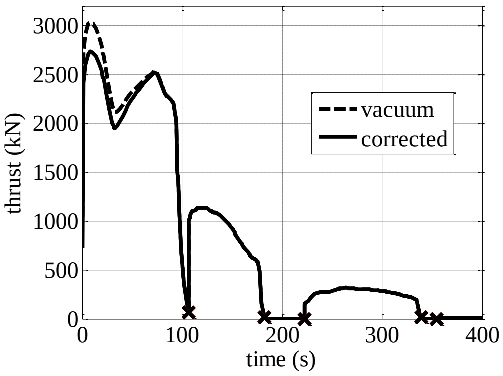


Figure 4.34 LETO thrust profile

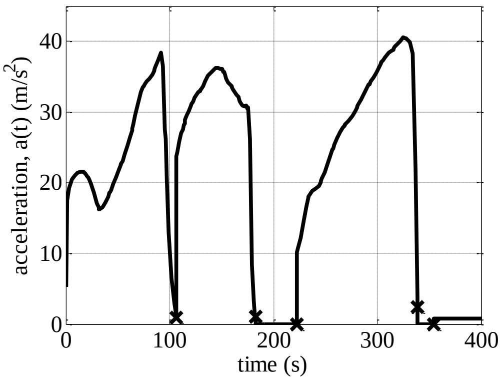


Figure 4.35 LETO acceleration profile

Figure 4.36 and Figure 4.37 shows the variations of the density and the Mach number, which are used to obtain aerodynamic loads. Density is extracted from the atmosphere model according to the flight altitude and the projected latitude over the oblate Earth, whereas Mach number is determined from the relative flight speed and the local sound speed.

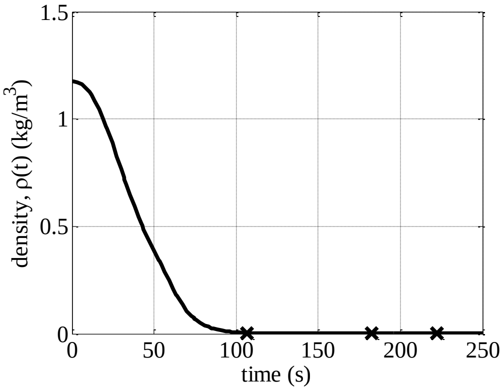


Figure 4.36 LETO density profile

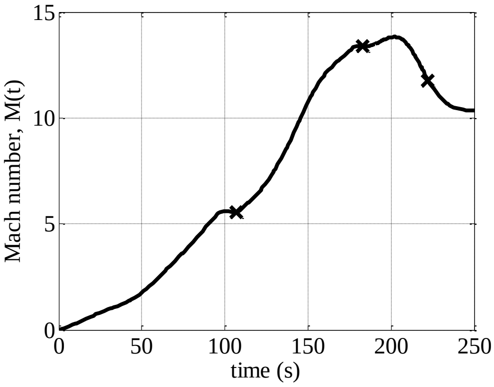


Figure 4.37 LETO Mach number profile

Figure 4.38 and Figure 4.39 demonstrates the variations of dynamic pressure and drag force with respect to time during the atmospheric flight phase. Although the dynamic pressure is not constrained within this problem, its maximum value is around 55 kPa which is acceptable for almost all existing launch vehicles in terms of structural strength.

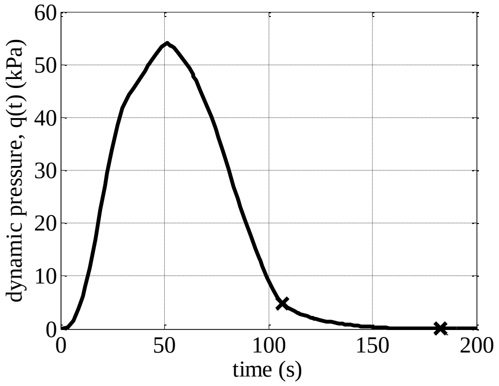


Figure 4.38 LETO dynamic pressure profile

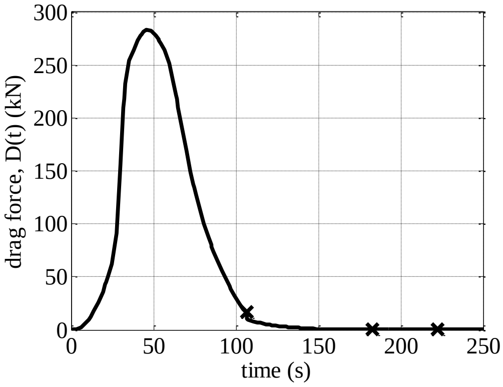


Figure 4.39 LETO drag force profile

Figure 4.40 and Figure 4.41 display the control profiles (pitch & yaw angles) and the velocity angles (flight path angle – FPA & heading angle) during entire trajectory. Control angles pitch and yaw are undefined during zero-thrust flight (3^{rd} and 5^{th} phase) under the assumption that the longitudinal axis of the vehicle coincides with the thrust vector.

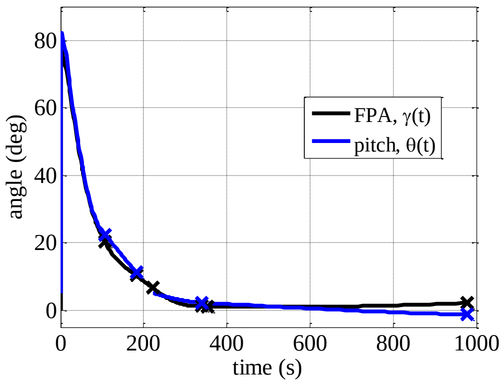


Figure 4.40 LETO FPA & pitch profile

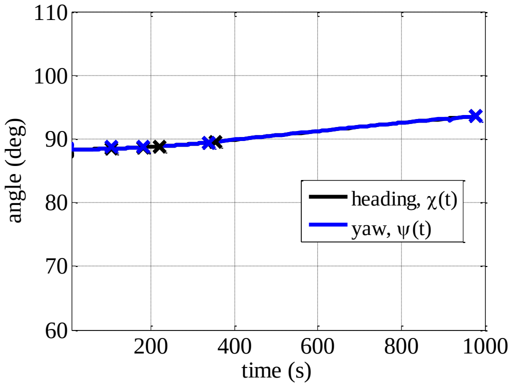


Figure 4.41 LETO heading & yaw profile

It is seen from Figure 4.40 that FPA is around 78° at time of lift-off, which is optimal in terms of payload performance. In a real launch, however, it should be equal to 90° due to the fact that the vehicle would be oriented vertically on the launch pad. When the same problem was solved such that FPA is enforced to be equal to 90°, it was observed that the effect of this change on payload performance is limited to just a few kilograms, because launch pad clearance is completed within a period of less than 5 seconds.

Figure 4.42 to Figure 4.44 show the results of the problem solved with and without trajectory constraints on the same graph. These figures point out that the maximum allowable values of 5° angle of attack and 60 kPa° bending moment indicator are both strictly satisfied during flight before fairing jettisoning, and heat flux is not exceeded 1135 W/m^2 after fairing jettisoning.

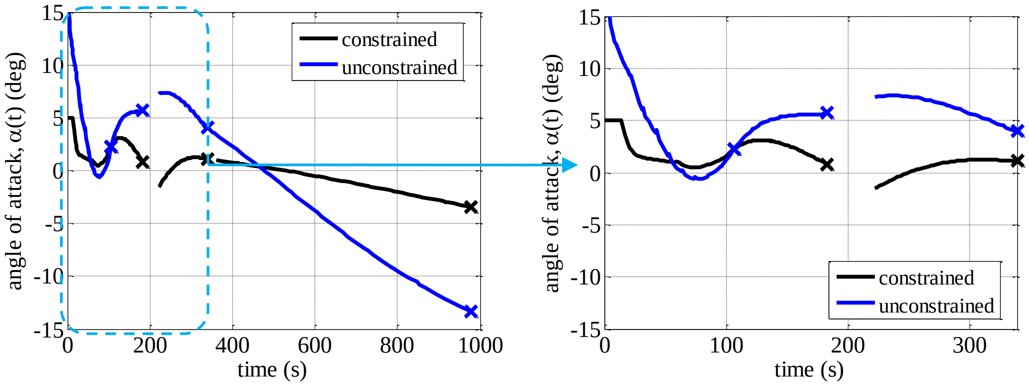
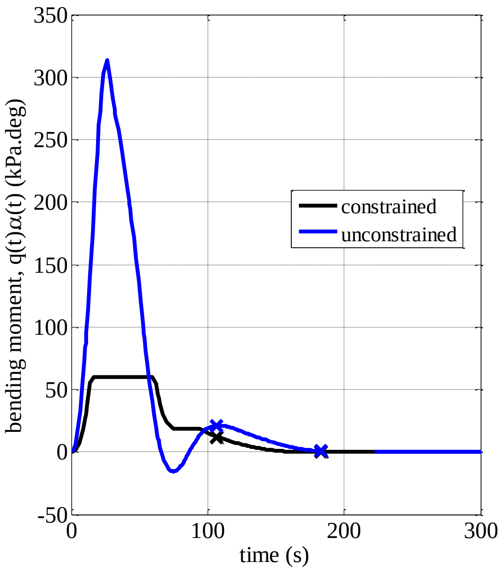


Figure 4.42 LETO angle of attack profile – constrained vs. unconstrained



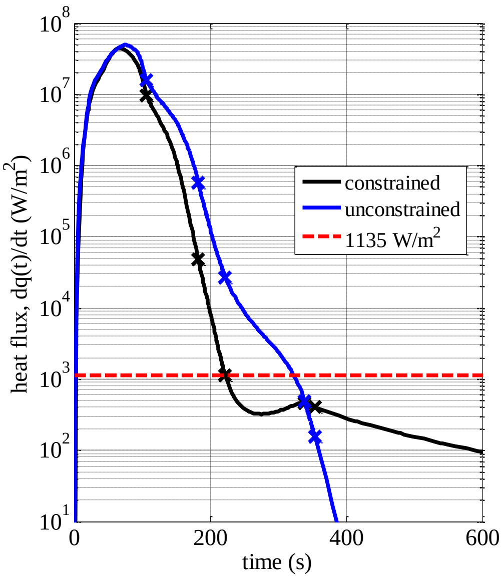


Figure 4.43 LETO bending moment profile

Figure 4.44 LETO heat flux profile

Additionally, the angle of attack was constrained to be lower than 15° during the exoatmospheric portion of the flight, but the limit values were not observed in this region even in the unconstrained trajectory as seen in Figure 4.42.

It is important to note that when the same problem was solved under the same conditions but without imposing any trajectory constraints, the optimized payload mass was obtained as **2138 kg**. Thus, it can be concluded that the trajectory constraints cause a performance loss of around 50 kg. It is also important to note that imposing constraints increases the complexity of the problem, but reduces the size of the feasible region, and improves the convergence properties.

The relatively quick run time and the optimized results presented in this section clearly demonstrate the success of the trajectory optimization tool developed within this thesis, and contribute to its validation through a real mission example.

4.3.5 Example 3: Vega – PEO Mission

Adding a level of complexity, multiple burn orbit injection mission consisting of two AVUM burns and a coasting between them was chosen as the second Vega example. The target orbit in this example is the polar circular Earth orbit (PEO) defined by

$$h_f = 700\,\text{km} \tag{4.96}$$

$$e_f = 0 \tag{4.97}$$

$$i_f = 90^\circ \tag{4.98}$$

The flight trajectory is composed of a total of eight phases, five of which are powered and the remaining three are the coasting periods. The first six phases are similar to the previous LETO mission, but at the end of AVUM's first burn the launch vehicle is injected into a transfer orbit whose apogee height is equal to the altitude of the circular target orbit. After a long coasting flight along this transfer orbit, AVUM is reignited and the circularization maneuvering is performed at the end of which the spacecraft is inserted into the target orbit.

In this mission, durations of AVUM burns and the coasting flights were optimized together with the payload mass. The perigee altitude of the transfer orbit was constrained to be higher than 150 km for safety reasons in accordance with the general practice. The reason is that in case the upper stage engine would not restart at the apogee, there will be at least a second full orbit for one more attempt. Otherwise, the vehicle starts to descent when it encounters the atmosphere at lower altitudes, and it is finally lost.

$$h_a^{(6)} = 700\,\text{km} \tag{4.99}$$

$$h_p^{(6)} \geq 150\,\text{km} \tag{4.100}$$

Since there are many phases in this problem including a quite long coasting period, the problem was first solved up to the transfer orbit injection, and then its results were given as an initial guess to the actual problem.

In the formulation of the first step, the propellant to be used during target orbit insertion maneuvering or so called circularization burn should have to be predicted and allocated for the second ignition of AVUM. Circularization maneuvering occurs in a short time interval compared to the entire time of flight; therefore impulsive orbit transfer assumption is valid. Accordingly, the required delta-V and consequently the required amount of propellant for circularization burn can be calculated analytically using Eqs. (A.14) and (A.15).

The optimal solution for the trajectory only up to the transfer orbit (TO) is presented in Figure 4.45 and Figure 4.46. According to this optimal solution, the vehicle enters an elliptical transfer orbit whose perigee is equal to 150 km and whose apogee is equal to 700 km at the end of the trajectory in compliance with the Eqs. (4.99) and (4.100). The flight altitude is slightly above 150 km with a corresponding true anomaly (θ^*) of around 6°. The maximized payload mass was obtained as 1482 kg when the estimated value of required propellant for circularization burn was considered.

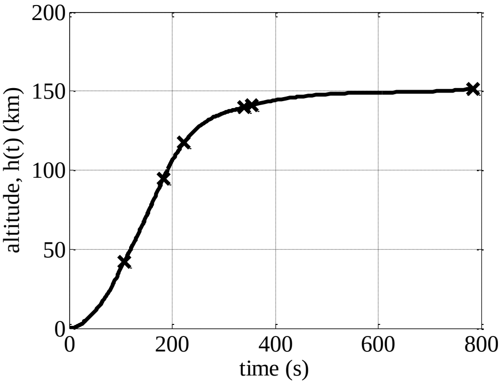


Figure 4.45 PEO TO altitude profile

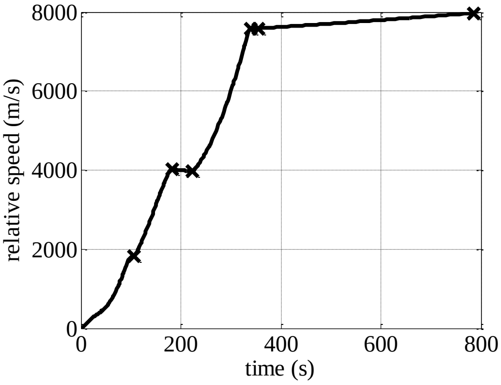


Figure 4.46 PEO TO speed profile

It is important to point out that the purpose of this initial guess is just to achieve easier convergence in the next step of the problem, in which the entire trajectory including all flight phases from ground to the final circular orbit will be optimized all at once. This initial guess was provided in the form of a time dependent state vector including the position, velocity and mass parameters related to the first six flight phase, and taking its advantage, the optimal solution regarding the entire mission was successfully deployed.

The results of the optimization of the whole trajectory into PEO confirms that the initial guess up to the transfer orbit is not stringent, and the optimal solution of the entire mission is distinctly different than the trajectory presented in Figure 4.45. As can be seen from

Table 4.12 and Figure 4.49, vehicle achieves a transfer orbit having a perigee altitude of 180 km at an altitude above 230 km ($\theta^* = 38°$), although it was around 150 km in transfer orbit solution.

Another important initial guess used for the optimization of the entire trajectory is related with the duration of coasting along the transfer orbit. This duration was estimated based on the half period (please refer to Appendix A.2) of the transfer orbit defined by Eqs. (4.99) and (4.100).

For circular orbit mission examples within this thesis, two different inequality constraints regarding the size and the shape of the target orbit were imposed. Although a circular orbit is mathematically the simplest orbit; it is difficult in practice to achieve an exact circular orbit due to launch vehicle dispersion errors. On the other hand, zero eccentricity introduces numerical difficulties such as spurious fluctuations and the nonconvergence of the solution although the trajectory models are fully general and any kind of orbit could in theory be reached. Therefore, the semimajor axis and the eccentricity were allowed to deviate from the desired values within predefined tolerances as follows.

$$\Delta a_f \leq \pm 2\ \text{km} \tag{4.101}$$

$$0 \leq e_f \leq 0.0001 \tag{4.102}$$

The first part of the problem, which was intended to provide a good initial guess, was solved within 740 seconds, and the whole trajectory was optimized in an additional time of almost 800 seconds with a computer whose specifications are given in Section 4.2.5. The optimized payload mass was obtained as **1488 kg**, which is almost 60 kg (4%) higher than the reference performance capability declared as 1430 kg in the user's manual of Vega (2014). Other important characteristics of the optimal solution are summarized in Table 4.11 to Table 4.13.

Table 4.11 Vega – PEO mission flight phases summary

#	**Duration (seconds)**	**Phase description**	**Active constraints**
1	0 – 106.7	Powered flight of P80	$-5° \leq \alpha \leq 5°$ $-60\ \text{kPa}° \leq q\alpha \leq 60\ \text{kPa}°$
2	106.7 – 182.7 (76)	Powered flight of Z23	
3	182.7 – 219.1 (36.4*)	Separation coast (fairing separation at end)	$\dot{q}_{end} \leq 1135\ \text{W/m}^2$
4	219.1 – 335.9 (116.8)	Powered flight of Z9	-
5	335.9 – 350.9 (15)	Separation coast	-
6	350.9 – 780.9 (430*)	AVUM burn for transfer orbit insertion	$h_{a,TO} = 700$ km $h_{p,TO} \geq 150$ km
7	780.9 – 3000.1 (2219.2*)	Coasting along the transfer orbit	-
8	3000.1 – 3147.1 (147*)	AVUM circularization burn	a_f, e_f, i_f ($h_{a,f}$, $h_{p,f}$, i_f)

* optimized in order to achieve the maximum performance

Table 4.12 Vega – PEO mission target orbit parameters

Orbital element	**Transfer orbit (start of coast)**	**Transfer orbit (end of coast)**	**Target orbit**
Perigee altitude, h_p (km)	179.04	179.07	698.30*
Apogee altitude, h_a (km)	700	700	699.72*
Inclination, i (°)	90	90	90
RAAN, Ω (°)	308.11	308.11	308.11
Argument of perigee, ω (°)	6.51	6.49	7.06
True anomaly, θ^* (°)	38.15	178.27	195.79

* deviates slightly from the circular orbit

Table 4.13 Vega – PEO mission AVUM propellant budget

Mission event	**Propellant mass (kg)**
Transfer orbit insertion	341.3554
Trajectory corrections	34.8430
Circularization burn	116.6616
Deorbiting	55.2907
Unusable	28.8500
Remaining	0
Total	577

The mass time history is given in Figure 4.47. The final mass includes the 4th stage structural mass (Table 4.5), the unusable propellant and the propellant reserved for deorbiting and trajectory corrections (Table 4.13) other than the payload. When these amounts are subtracted from the final mass, the payload mass is obtained as 1488 kg. Actually, the amount of propellant reserved for trajectory corrections could be discarded at the end of the first burn of AVUM, since it was considered to correct the trajectory deviations occur during solid motor powered phases. In any way, this approach ensures the optimized solution to remain on the safe side in terms of payload performance.

The comparison of the results with the reference data available in the open literature are given in Figure 4.48 to Figure 4.54 and Figure 4.57. The blue markers labeled by ‘ref.1’ in these graphs are originated from the presentation of Serraglia (2008) who is Integrated Project Team Leader of Vega Program. Whereas, green curves labeled by ‘ref.2’ are obtained from the doctoral dissertation of Castellini (2012) using GetData graph digitizer.

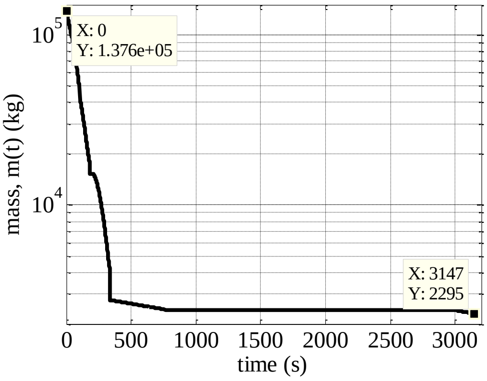


Figure 4.47 PEO mass time history

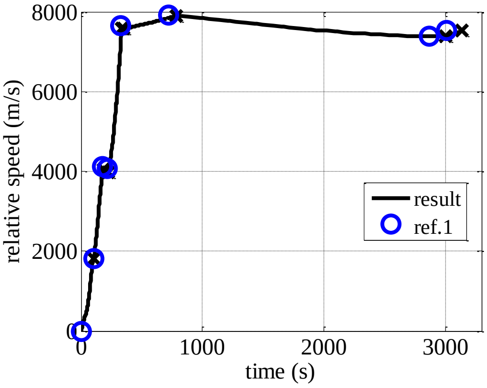


Figure 4.48 PEO relative speed comparison

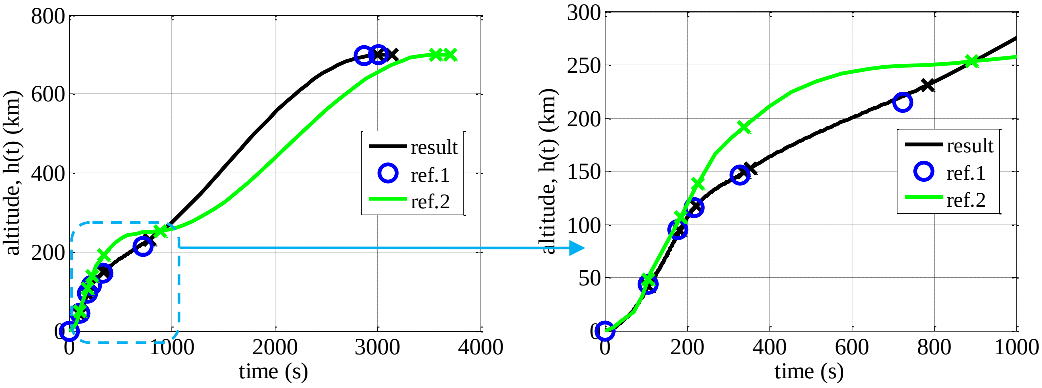


Figure 4.49 PEO altitude comparison

As seen from Figure 4.48 and Figure 4.49, results match fairly well with the performance estimation of Vega given in ‘ref.1’. A closer examination reveals that the key difference is the duration of AVUM’s first burn. This difference lies in the change of propellant capacity of AVUM main engine, which was declared as 550 kg in the old version of the user’s manual (Vega, 2006). Currently, it is able to contain 577 kg of propellant and the mass flow rate is almost exactly the same as announced in the updated version of the user’s manual (Vega, 2014).

Since ‘ref.1’ reflects the situation in the year 2008, the dissimilarity in burn time of AVUM can be clearly identified. It is important to point out that this capacity increase may have effect on the performance as well; however it was already compared with the up-to-date information given in the last issue of the manual of Vega (2014) as mentioned previously.

Another distinction observed between the result and ‘ref.1’ in Figure 4.48 and Figure 4.49 is the duration of the coasting phase. The coasting phase in the optimal solution is about 70 seconds longer than the reference performance estimation. This difference is probably explained by the differences between the preferred transfer orbits. As declared in Table 4.12, a transfer orbit having a perigee altitude of 179 km and apogee altitude of 700 km was obtained as optimal solution within the constraints imposed in this study. However, the perigee and apogee heights of the transfer orbit followed in the performance analyses of Vega might be different but lead up to the same target orbit by a slightly different orbit insertion maneuvering.

Reference solution represented by green curves labeled as ‘ref.2’ follows a longer trajectory showing a bit more horizontal tendency. It is important to point out that the configurational properties of the vehicle in ‘ref.2’ substantially match with the ones considered in this study and the manual of Vega (2014).

Figure 4.50 shows the variation of flight path angle along the trajectory, and Figure 4.51 and Figure 4.52 shows the control profiles and the resulting angle of attack in comparison with ‘ref.2’.

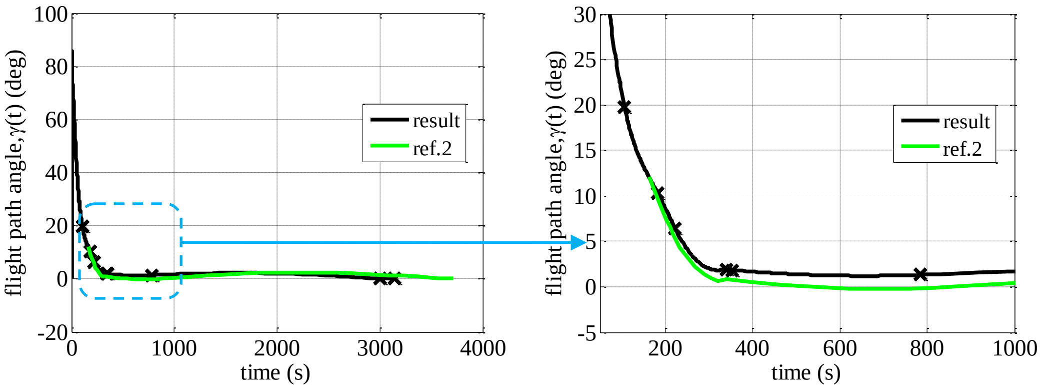


Figure 4.50 PEO flight path angle profile comparison

Figure 4.50 highlights the fact that trajectory in ‘ref.2’ inclines to the local horizontal earlier than the optimal solution. Figure 4.51 indicates the distinctive nose-up maneuvering of the launch vehicle between the time interval between 200 and 400 sec, which can also be observed from Figure 4.49. Figure 4.52 points out that the trajectory in ‘ref.2’ completes its maneuvering around 850 seconds after lift-off and it enters into the orbit by assuming tangential thrust coinciding with the velocity direction.

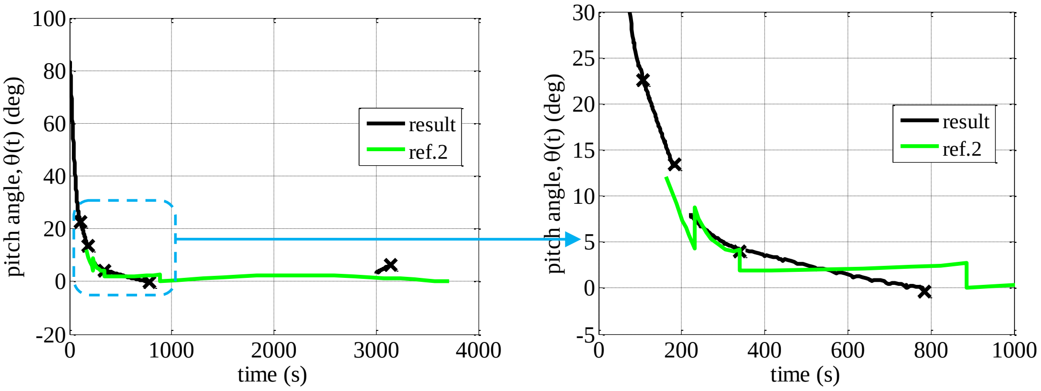


Figure 4.51 PEO pitch angle profile comparison

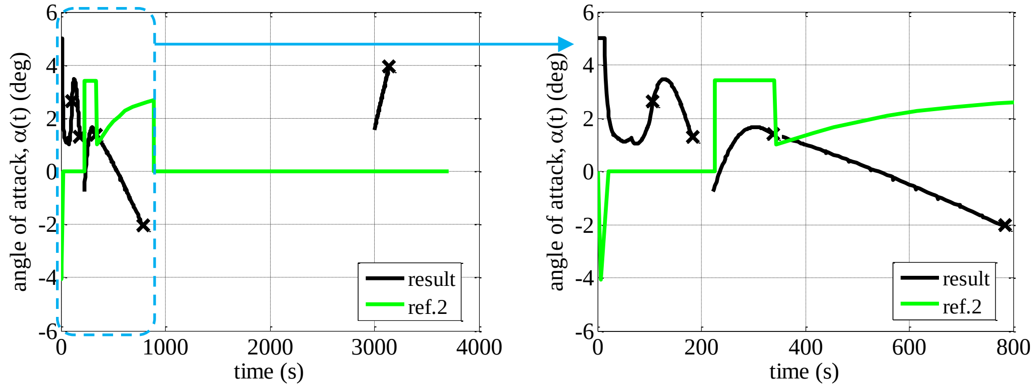


Figure 4.52 PEO angle of attack profile comparison

Figure 4.53 to Figure 4.56 shows the variation of the orbital elements over the whole flight. Figure 4.53 confirms that both trajectories represented by solid and dotted lines reach the desired circular orbit such that semimajor axis length, apogee radius and perigee radius are all converged to the same value at the end points of the trajectories. Figure 4.54 demonstrates the change of orbit's shape from parabola ($e = 1$) to circle ($e \approx 0$) during flight.

Figure 4.55 implies that inclination change maneuvering is completed within the first 350 seconds of flight, since it would be more costly when the vehicle speed increases. Figure 4.56 indicates the values of true anomaly at the beginning and end of the coasting phase, which represents the location at which vehicle enters the transfer orbit and the location at which circularization burn starts respectively.

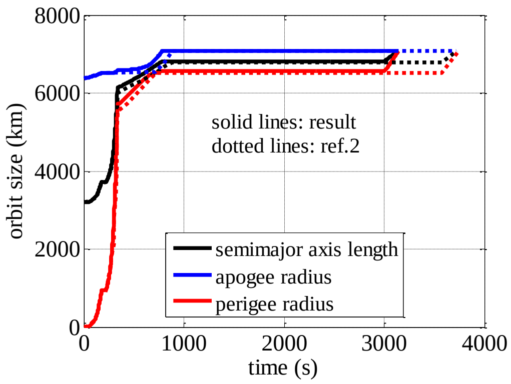


Figure 4.53 PEO a, r_a, r_p comparison

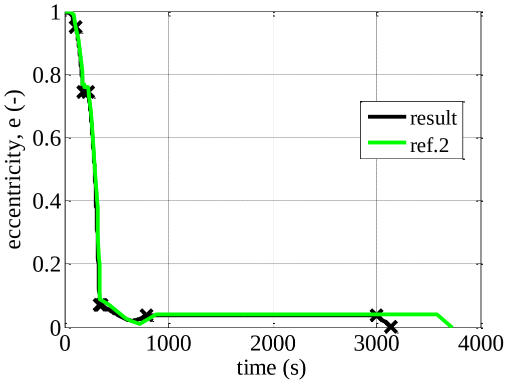


Figure 4.54 PEO eccentricity comparison

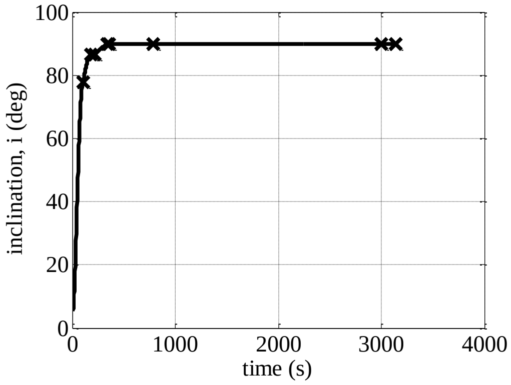


Figure 4.55 PEO inclination comparison

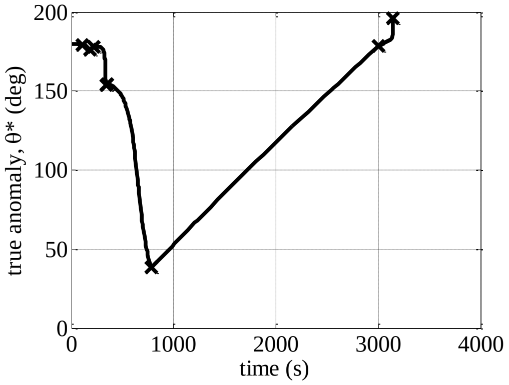


Figure 4.56 PEO true anomaly comparison

Figure 4.57 shows the comparison of ground tracks which match closely, only difference results from the difference of phase durations previously discussed and consequently the whole mission duration. Table 4.14 lists the geodetic position coordinates of the nominal impact points and Figure 4.58 illustrates the ground tracks and the nominal impact points on the world map. Additionally, Figure 4.59 displays the altitude downrange profile. It is important to point out that altitude values are given with respect to Earth's equatorial radius, i.e. R_e = 6378.137 km.

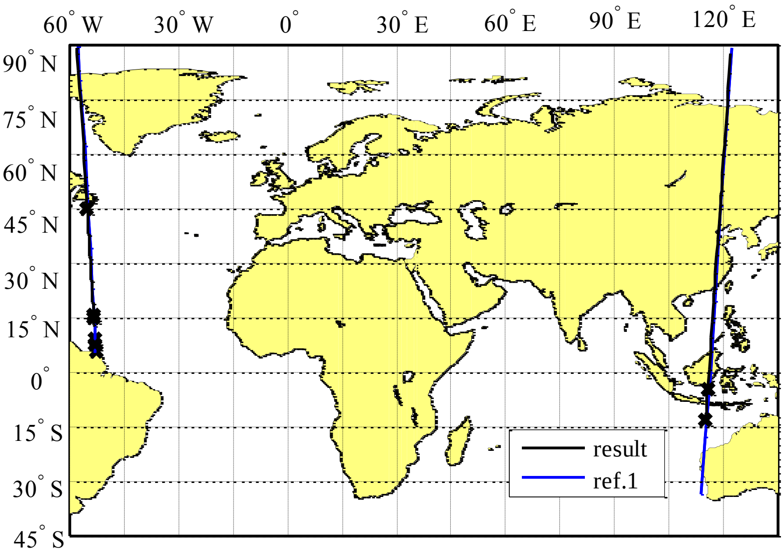


Figure 4.57 PEO launch vehicle's ground track comparison

Table 4.14 Vega – PEO mission separable parts impact points

	Longitude (°)	**Geodetic latitude (°)**	**Downrange (km)**
1st stage	-52.95	8.20	332.2
2nd stage	-53.50	17.77	1392.6
3rd stage	-50.60	88.85	9300.5

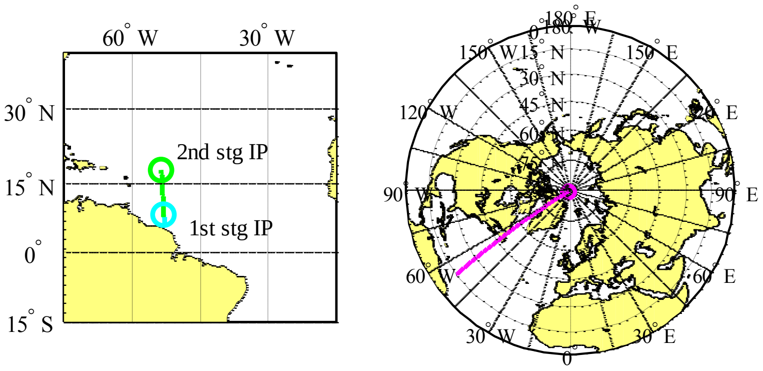


Figure 4.58 PEO separated stages impact points

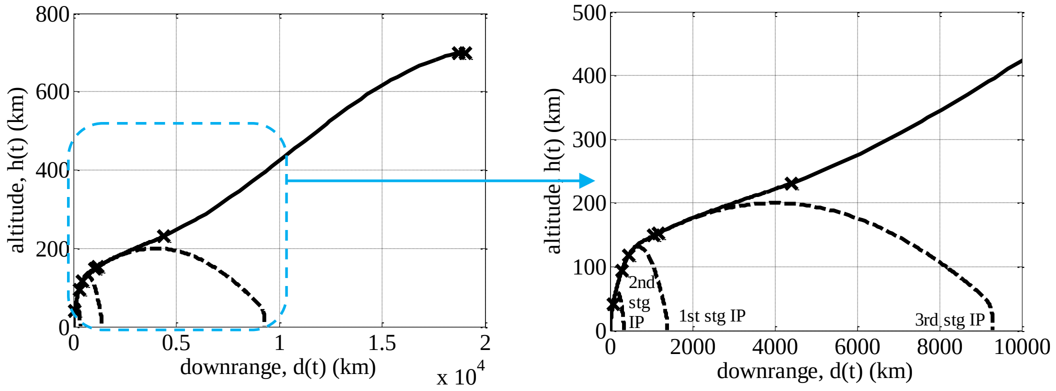


Figure 4.59 PEO altitude vs. downrange (incl. separated stages)

Other important trajectory parameters are presented in Figure 4.60 to Figure 4.63. Figure 4.60 and Figure 4.61 confirm that the trajectory constraints related with bending moment indicator and heat flux are successfully satisfied. Whereas, Figure 4.62 and Figure 4.63 demonstrates the acceleration and dynamic pressure profiles which are acceptable for Vega launcher. It is useful to remind that acceleration and dynamic pressure are driven by the thrust time profile of motors, which is given and fixed in the examples of this chapter.

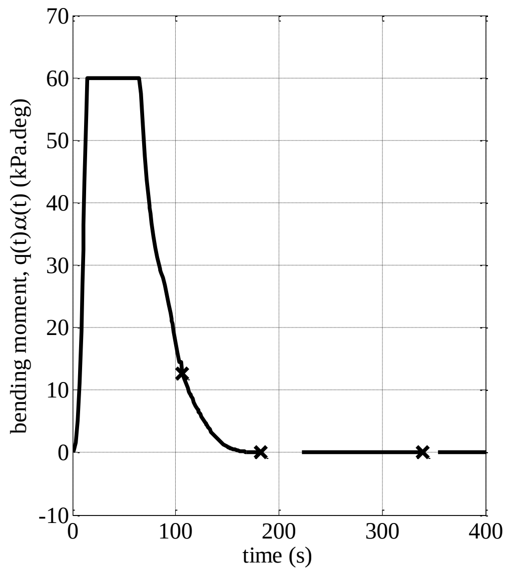


Figure 4.60 PEO bending moment profile

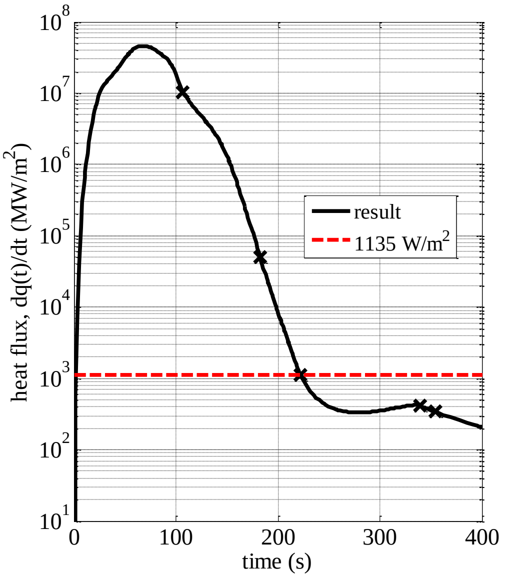


Figure 4.61 PEO heat flux profile

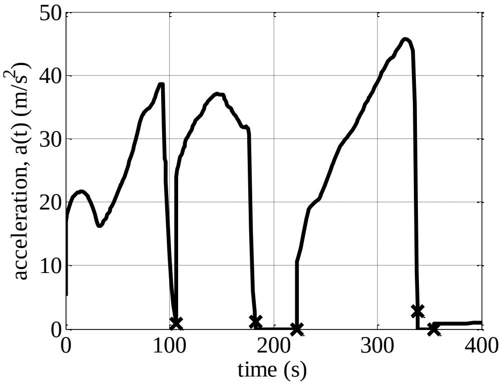


Figure 4.62 PEO acceleration profile

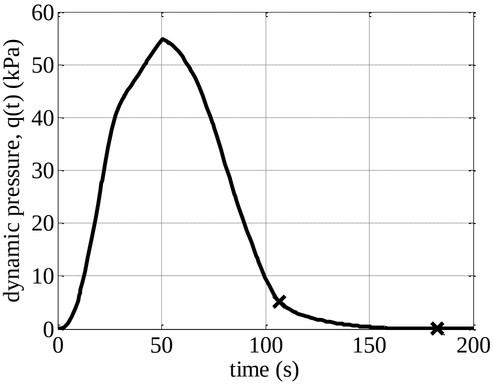


Figure 4.63 PEO dynamic pressure profile

4.3.6 Example 4: Vega – SSO Mission

The last example from Vega is its Sunsynchronous Earth orbit (SSO) mission, whose orbit injection scenario is similar to PEO mission consisting of two AVUM burns. The target orbit is defined by

$$h_f = 750\,\text{km} \tag{4.103}$$

$$e_f = 0 \tag{4.104}$$

$$i_f = 98.5° \tag{4.105}$$

Since the same methodology discussed extensively in Section 4.3.5 was applied for the trajectory design of SSO mission, only a summary of the trajectory optimization results is given here in this section.

The solution to provide initial guess regarding the trajectory up to the transfer orbit having an apogee altitude of 750 km was solved within 510 seconds, and the whole trajectory was optimized in an additional time of 575 seconds with a computer whose specifications are given in Section 4.2.5. The optimized payload mass was obtained as **1354 kg**, which is 54 kg (around 4%) higher than the reference performance capability declared as 1300 kg in the user's manual of Vega (2014).

Other important characteristics of the optimal solution are summarized in Table 4.15 to Table 4.17. The comparison of the results with the reference data available in the open literature are given in Figure 4.65, Figure 4.66 and Figure 4.72. The reference data were obtained from the user's manual of Vega (2014) using GetData graph digitizer.

The relative speed and altitude profiles (Figure 4.65 and Figure 4.66) have the same trend and in line with the discussion is Section 4.3.5. The key difference in the acceleration profile (Figure 4.72) originates from the slight differences of the burn time of motors as explained in Section 4.3.2.

Table 4.15 Vega – SSO mission flight phases summary

#	**Duration (seconds)**	**Phase description**	**Active constraints**
1	0 – 106.7	Powered flight of P80	$-5° \leq \alpha \leq 5°$ -60 kPa° $\leq q\alpha \leq 60$ kPa°
2	106.7 – 182.7 (76)	Powered flight of Z23	
3	182.7 – 219.8 (37.1[*])	Separation coast (fairing separation at end)	$\dot{q}_{end} \leq 1135$ W/m^2
4	219.1 – 336.6 (116.8)	Powered flight of Z9	-
5	336.6 – 351.6 (15)	Separation coast	-
6	351.6 – 787.6 (436[*])	AVUM burn for transfer orbit insertion	$h_{a,TO} = 750$ km $h_{p,TO} \geq 150$ km
7	787.6 – 2945.2 (2157.6[*])	Coasting along the transfer orbit	-
8	2945.2 – 3087 (141.8[*])	AVUM circularization burn	a_f, e_f, i_f ($h_{a,f}$, $h_{p,f}$, i_f)

[*] optimized in order to achieve the maximum performance

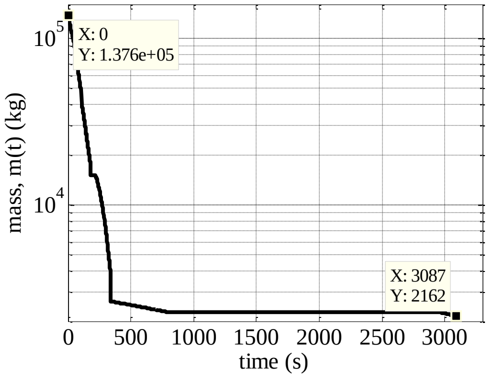
Table 4.16 Vega – SSO mission target orbit parameters

Orbital element	**Transfer orbit (start of coast)**	**Transfer orbit (end of coast)**	**Target orbit**
Perigee altitude, h_p (km)	200.01	200.16	748.29[*]
Apogee altitude, h_a (km)	750.00	750.95	749.71[*]
Inclination, i (°)	98.5	98.5	98.5
RAAN, Ω (°)	308.91	308.91	308.91
Argument of perigee, ω (°)	11.08	11.16	52.88
True anomaly, θ^* (°)	33.96	169.77	136.43

[*] deviates slightly from the circular orbit

Table 4.17 Vega – SSO mission AVUM propellant budget

Mission event	**Propellant mass (kg)**
Transfer orbit insertion	346.1185
Trajectory corrections	32.5851
Circularization burn	111.8061
Deorbiting	57.6403
Unusable	28.8500
Remaining	0
Total	577



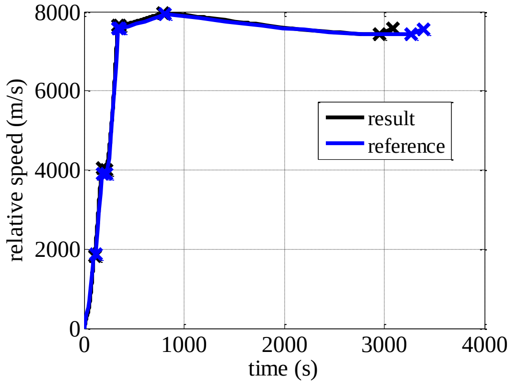


Figure 4.64 SSO mass time history

Figure 4.65 SSO relative speed comparison

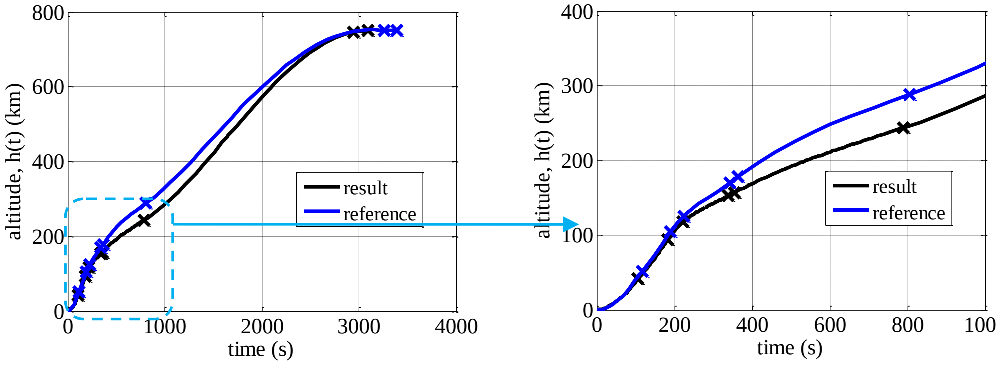
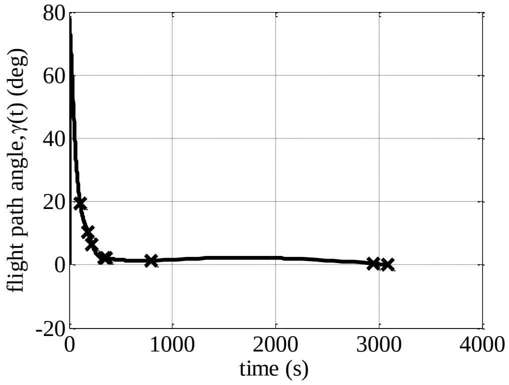


Figure 4.66 SSO altitude comparison



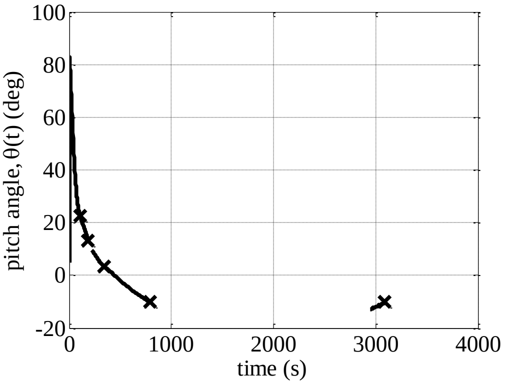


Figure 4.67 SSO FPA profile

Figure 4.68 SSO pitch angle profile

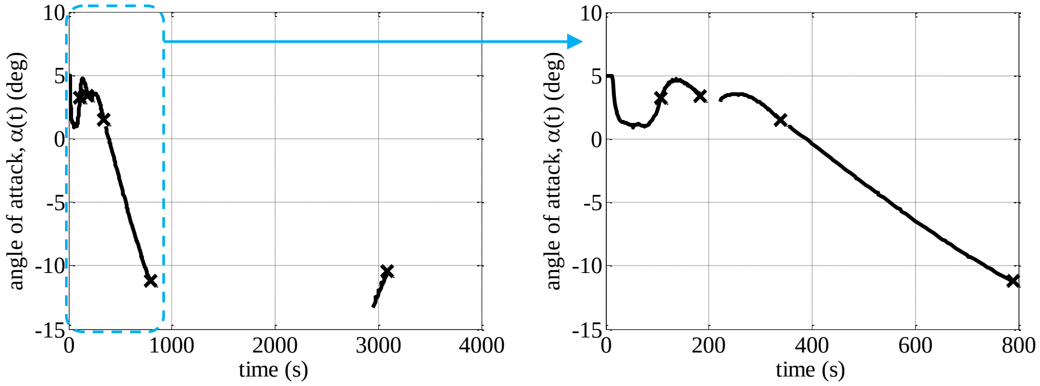


Figure 4.69 SSO angle of attack profile

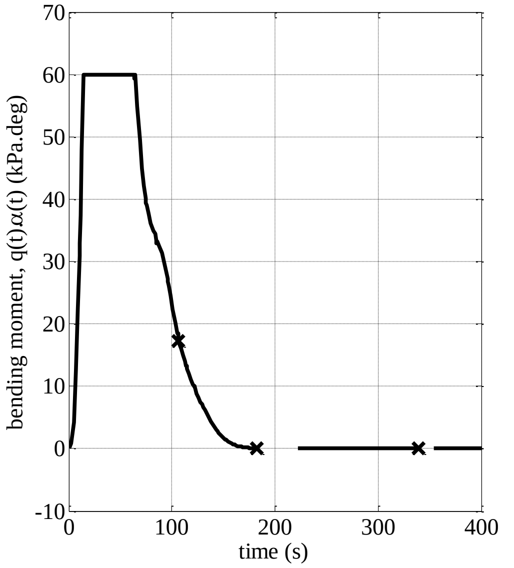


Figure 4.70 SSO bending moment profile

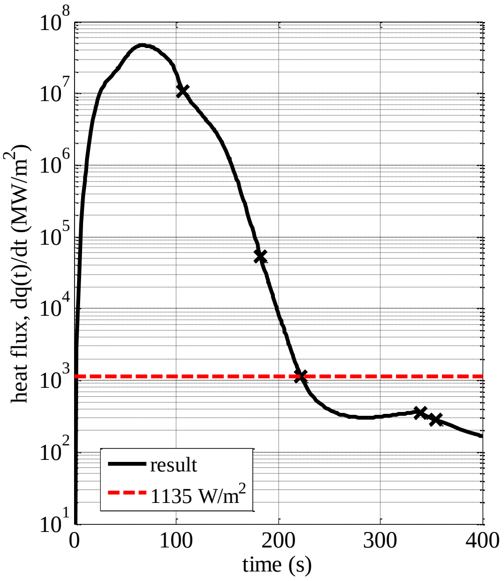


Figure 4.71 SSO heat flux profile

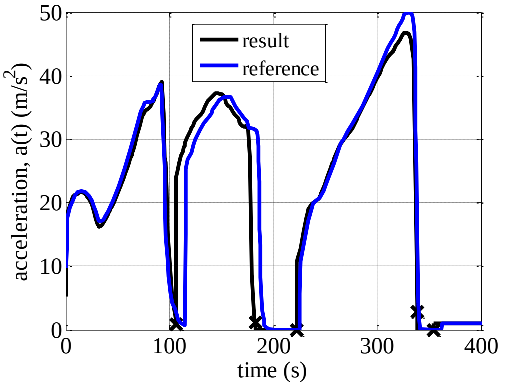


Figure 4.72 SSO acceleration comparison

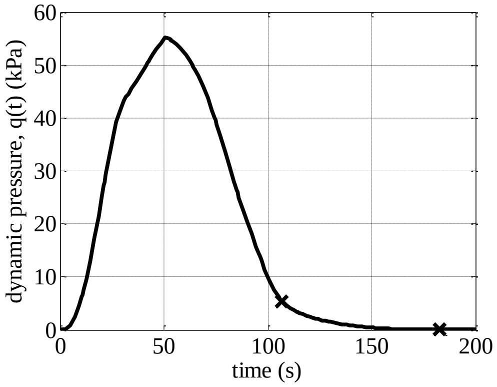


Figure 4.73 SSO dynamic pressure profile

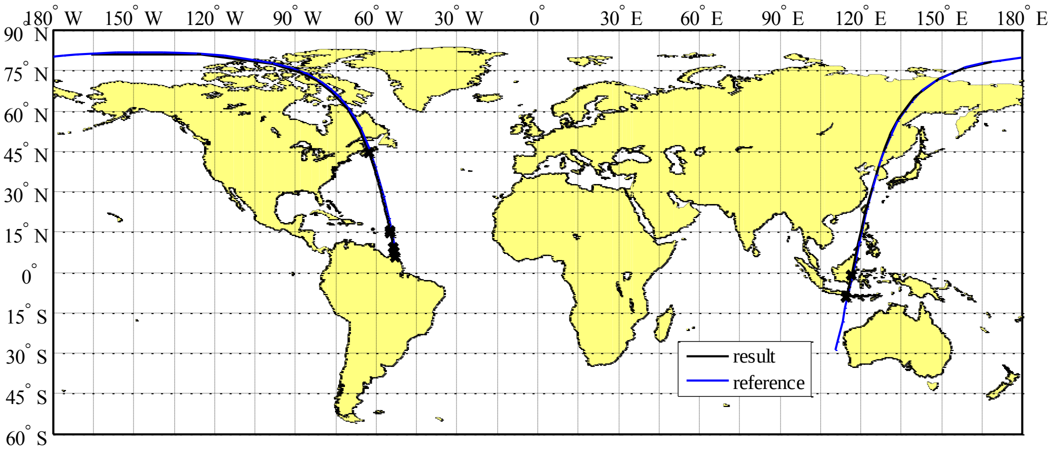


Figure 4.74 SSO launch vehicle's ground track comparison

Table 4.18 Vega – SSO mission separable parts impact points

	Longitude (°)	Geodetic latitude (°)	Downrange (km)
1st stage	-53.37	8.11	327.8
2nd stage	-55.39	17.57	1397.4
3rd stage	-165.63	81.29	9811.5

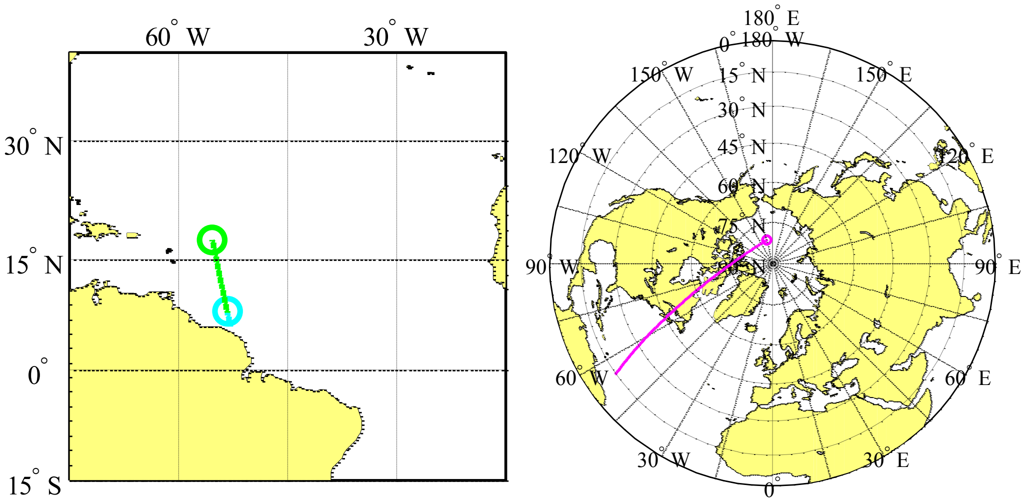


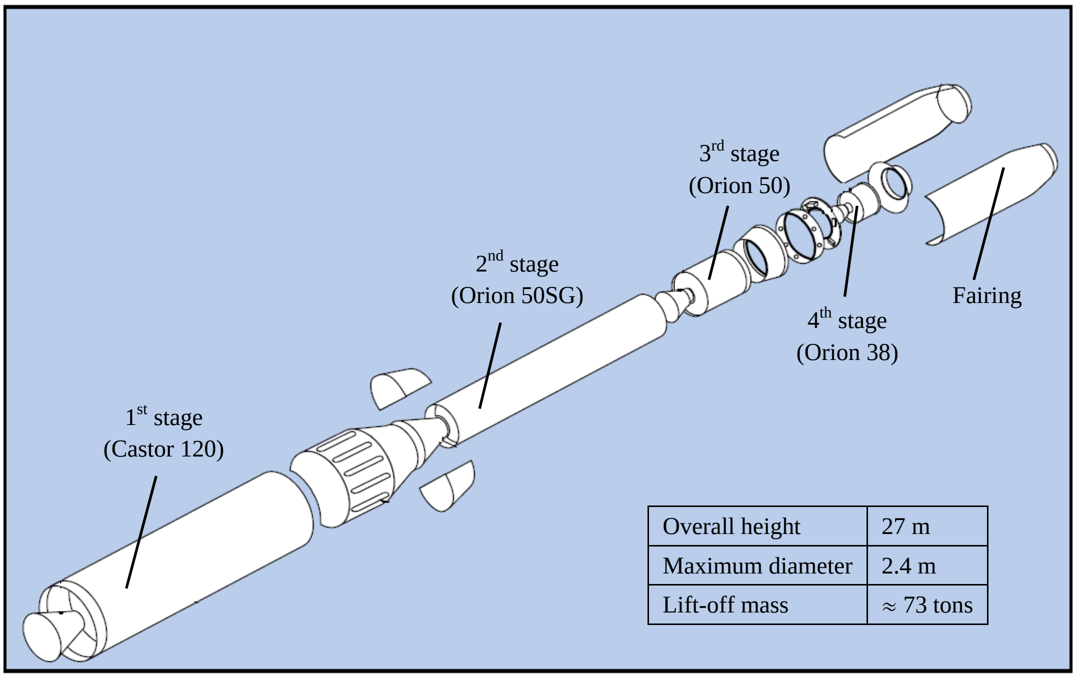
Figure 4.75 SSO separated stages impact points

4.4 Example 5: Taurus – SSO Mission

The fifth and the last example related with the verification and validation of the trajectory optimization code developed within the scope of this thesis is the SSO mission of the Taurus launch vehicle. The details of the problem formulation are not described herein, since they are exactly the same as the previous Vega mission examples in terms of applicable aspects. The major differences arose from the fact that the last stage of Taurus uses a solid rocket motor whereas the upper stage of Vega comprises a liquid engine. Therefore, the applied scenarios for orbit insertion are different, and the durations of powered flight phases in Taurus mission are fixed and known in the optimization.

4.4.1 Vehicle Properties

Taurus is a four stage solid propellant launch vehicle developed by Orbital Sciences Corporation in the United States. It has different configuration variants, and the one designated by 2210 is the subject of this verification example (Figure 4.76).



Overall height	27 m
Maximum diameter	2.4 m
Lift-off mass	≈ 73 tons

Figure 4.76 Taurus launch vehicle
(adapted from Taurus payload user's guide, 1999)

All four stages of the Taurus launch vehicle use solid propellant rocket motors developed by ATK (formerly known as Thiokol). The first stage (also known as stage 0) uses the Castor 120 motor, and the upper stages use the Orion 50SG, Orion 50 and Orion 38 motors, respectively. Table 4.19 summarizes the physical and performance characteristics of the Taurus launch vehicle, and Figure 4.77 shows the thrust and mass profiles of the solid rocket motors. All these data were gathered together from three distinct sources (Taurus payload user's guide, 1999; ATK product catalog, 2008; The space launch report, 2014).

Table 4.19 Physical and performance characteristics of Taurus

	1st stage	**2nd stage**	**3rd stage**	**4th stage**
Propellant type	Solid	Solid	Solid	Solid
Propellant mass, m_p (kg)	49005.2	12156.7	3025	770.7
Structural mass, m_s (kg)*	4922.7	2029.7	470.8	356.5
Vacuum specific impulse, I_{sp} (sec)	278.98	286.44	292.25	288.99
Burn time, t_b (sec)	84.6	77.9	75.8	68.6
Stage diameter (m)	2.36	1.27	1.27	1.665
Nozzle exit diameter (m)	1.52	1.21	0.86	0.53
Vacuum thrust, T_{vac} (kN)	See Figure 4.77			
Mass flow rate, $\dot{m}$ (kg/s)	See Figure 4.77			
Fairing mass m_{plf} (kg)	300			

* structural masses except the inert masses of stage motors were estimated using the interstage masses of a similar launch vehicle Vega so as to be consistent with the lift-off mass of Taurus

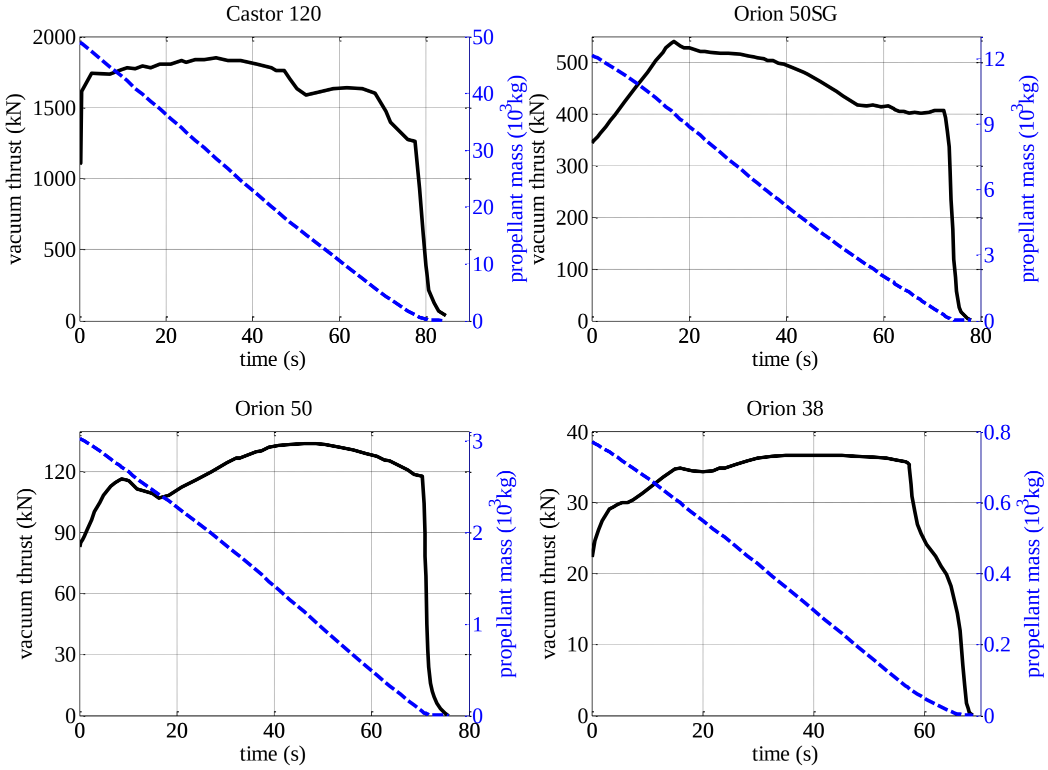


Figure 4.77 Thrust and propellant mass profiles of solid rocket motors of Taurus

4.4.2 Mission Characteristics

Taurus is able to carry payloads around 1000 kg to low Earth orbits of a variety of altitudes and geosynchronous transfer orbit. The target orbit in this example is one of the typical mission profiles of Taurus which is 300 nmi (556 km) sunsynchronous orbit defined by

$$h_f = 556\ \text{km} \tag{4.106}$$

$$e_f = 0 \tag{4.107}$$

$$i_f = 97.6° \tag{4.108}$$

Launch is from South Vandenberg Air Force Base whose geocentric position is

$$\delta_0 = 34.73° \quad ; \quad \lambda_0 = \text{-}120.57° \tag{4.109}$$

Taurus flight trajectory is composed of five flight phases as depicted in Figure 4.78. Since all stages of Taurus uses solid rocket motors, the increments of velocity change are relatively fixed, and the trajectory strongly depends on the target orbit. Mission starts with the successive burn of the first three stages, and at burn-out of the 3rd stage the vehicle is reached a suborbital trajectory whose peak is slightly above the desired orbit's altitude. After a ballistic (coasting) phase along this suborbital trajectory up to its apogee (peak) point, the 4th stage is fired and its burn adds the velocity to insert the payload into the target orbit.

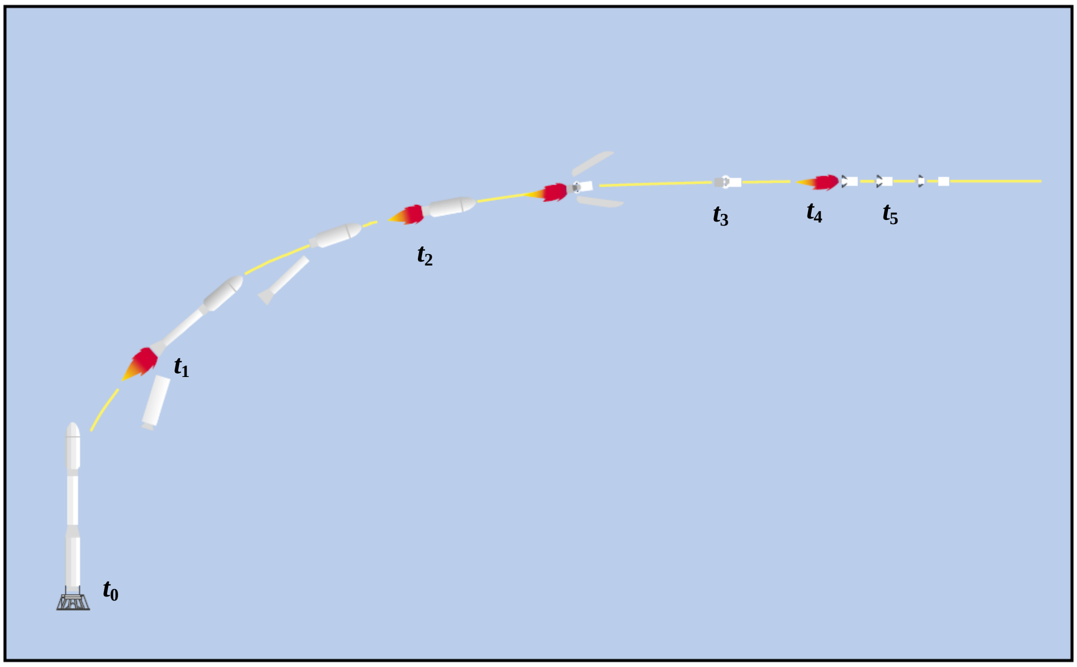


Figure 4.78 Taurus flight sequence
(adapted from Taurus payload user's guide, 1999)

As can be noticed promptly, Vega and Taurus operate different orbit insertion scenarios for circular Earth orbits with different coasting characteristics to the final burn point. As explained in 4.3.5, Vega uses the conventional approach by first inserting the upper stage vehicle into a transfer/parking orbit whose apogee height coincides with the target orbit's altitude and perigee is above the ground. In this scenario, when the upper stage vehicle reaches the desired apogee location, it is ignited and places the payload into its target orbit. Whereas, Taurus uses another technique so called *nonapsidal injection* such that the upper stage coasts along a suborbital trajectory whose perigee altitude is lower than zero (intersecting with the ground), and then it is fired at a point neither apogee nor perigee. This technique allows a heavier mass since suborbital trajectories require lower energy, however it has poor flexibility, and hard constraints have to be satisfied to insert the payload into the desired orbit. This difference in orbit insertion patterns can easily be observed from the altitude downrange profiles of circular orbit missions of Vega and Taurus illustrated respectively in Figure 4.59 and Figure 4.81.

Table 4.20 summarizes the mission timeline of Taurus. Although, there exists a short coast time between 2nd stage burn-out and 3rd stage ignition, which is typically 12 seconds, this phase was not required to be modeled since heat flux constraint for fairing separation can easily be satisfied. As experienced from the fixed time coast of 15 seconds after 3rd stage separation of Vega, no significant difference is expected on performance even if it is modeled.

Table 4.20 Taurus flight sequence

Event	**Time**
Castor 120 ignition and lift-off	$t_0 = 0$ sec
Castor 120 burn-out and separation Orion 50SG ignition	$t_1 = t_0 + 79.5 = 79.5$ sec
Orion 50SG burn-out and separation Payload fairing separation Orion 50 ignition	$t_2 = t_1 + 75 = 154.5$ sec
Orion 50 burn-out and separation Start of coast	$t_3 = t_2 + 75.6 = 230.1$ sec
Orion 38 ignition	t_4^*
Orion 38 burn-out (injection into transfer orbit)	$t_5 = t_4^* + 67.7$ sec

* to be optimized in order to achieve the maximum performance

4.4.3 Results and Discussion

The problem was solved with an initial mesh of consisting of 40 Legendre-Gauss-Radau points. The optimality conditions and the mesh error tolerance were satisfied on the third mesh, that is, two mesh refinement iterations were performed. The NLP problem arising after collocation had 2046 nonlinear variables and 1793 nonlinear constraints. The computation time for solving this mission example was around 340 seconds on a computer whose specifications are given in Section 4.2.5.

The optimized payload mass was obtained as **787 kg**, which is 117 kg higher than the reference performance capability declared as 670 kg in Taurus payload user's guide (1999). This difference of around 17% may result from the unavailability of the actual structural mass data of the stages. As mentioned in the footnote under Table 4.19, only the inert masses of stage motors are available in the published data sources cited in Section 4.4.1, and therefore the other structural masses such as the mass of interstage structures, auxiliary propulsion systems and the avionic units and components had to be estimated based on the data of similar launch vehicle Vega. Although the value of gross lift-off mass of Taurus is conserved, it appears to be that the distribution of structural mass between stages could not have been determined accurately.

Table 4.21 gives a brief summary of the optimized mission, and Table 4.22 presents the orbital elements of the target orbit. In this mission example, the entire trajectory up to the circular target orbit was optimized at once different from the circular orbit mission examples of Vega, in which the vehicle first achieves an intermediate elliptical transfer orbit. Other important characteristics of the optimal solution are illustrated in from Figure 4.79 to Figure 4.95.

Table 4.21 Taurus – SSO mission flight phases summary

#	**Duration (seconds)**	**Phase description**	**Active constraints**
1	0 – 79.5	Powered flight of Castor 120	$-5° \leq \alpha \leq 5°$ $-60\ \text{kPa}° \leq q\alpha \leq 60\ \text{kPa}°$ $\dot{q}_{end} \leq 1135\ \text{W/m}^2$
2	79.5 – 154.5 (75)	Powered flight of Orion 50SG	
3	154.5 – 230.1 (75.6)	Powered flight of Orion 50	-
4	230.1 – 622.1 (392[*])	Coasting along suborbital trajectory	-
5	622.1 – 689.8 (67.7)	Powered flight of Orion 38 (orbit insertion at end)	a_f, e_f, i_f ($h_{a,f}$, $h_{p,f}$, i_f)

[*] optimized in order to achieve the maximum performance

Table 4.22 Taurus – SSO mission target orbit parameters

Orbital element	**Desired**	**Guess**	**Result**
Semimajor axis (a)	6934137 m	6934137 m	6933998 m
Eccentricity (e)	0	0	0.0001[*]
Inclination (i)	97.6°	97.6°	97.6°
RAAN (Ω)	free	244.7371°	245.4822°
Argument of perigee (ω)	free	35.0827°	50.7827°
True anomaly (θ^*)	free	0°	14.7699°

[*] deviates slightly from the circular orbit

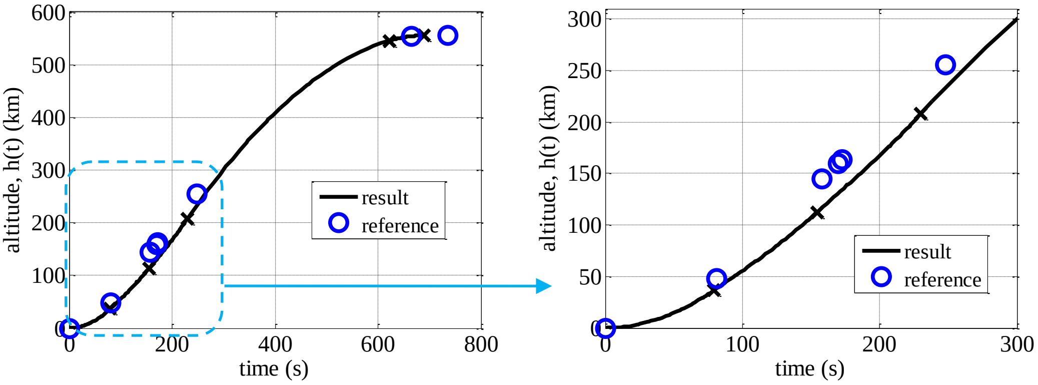


Figure 4.79 Taurus altitude comparison

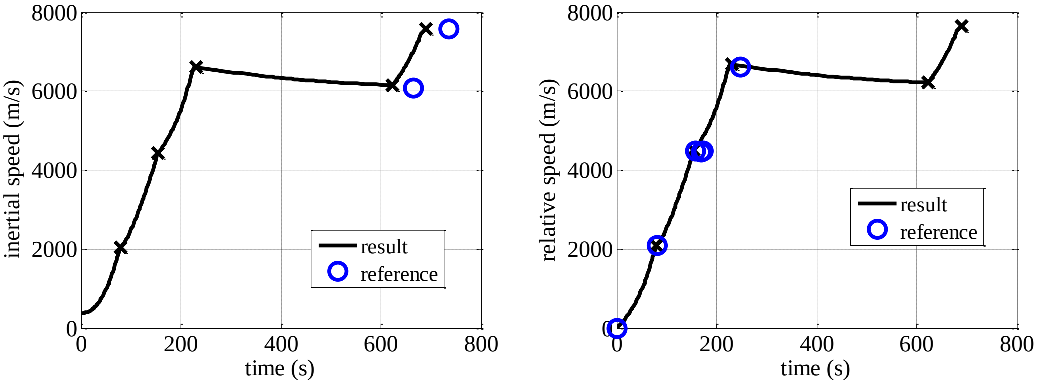


Figure 4.80 Taurus speed comparison

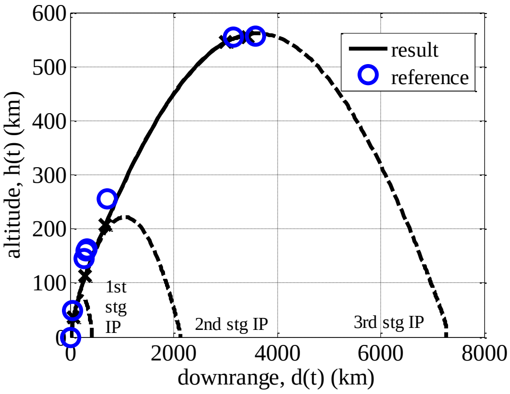


Figure 4.81 Taurus altitude-range comparison

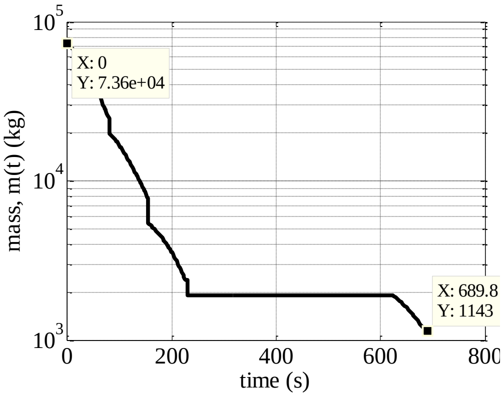


Figure 4.82 Taurus mass profile

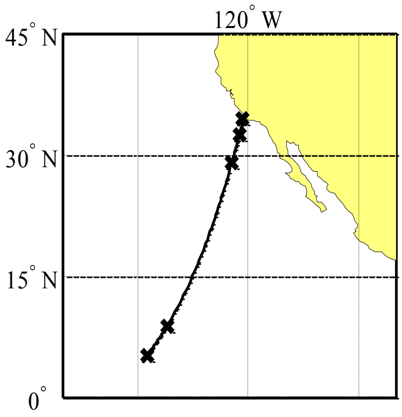


Figure 4.83 Taurus ground track comparison

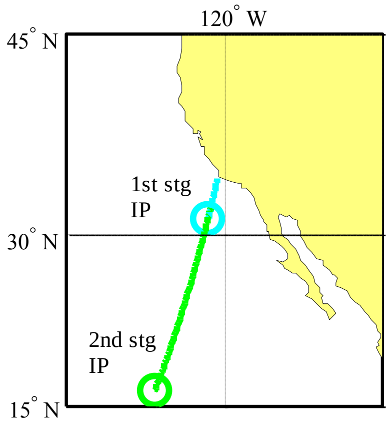


Figure 4.84 Taurus 1st & 2nd impact points

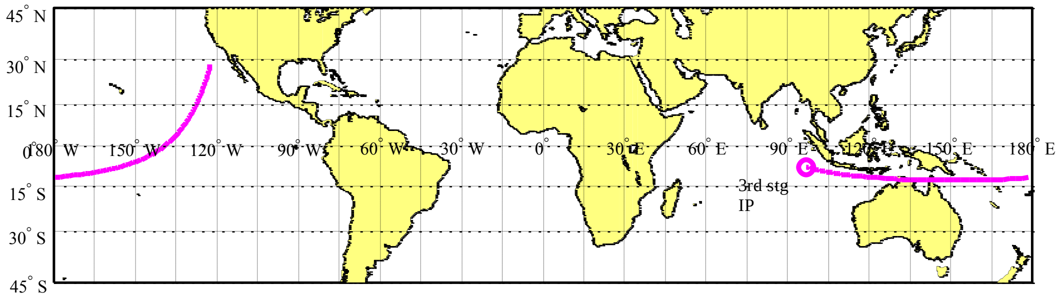


Figure 4.85 Taurus separated 3rd stage ground track and impact point

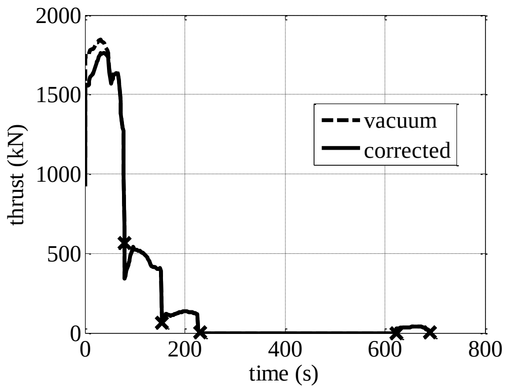


Figure 4.86 Taurus thrust profile

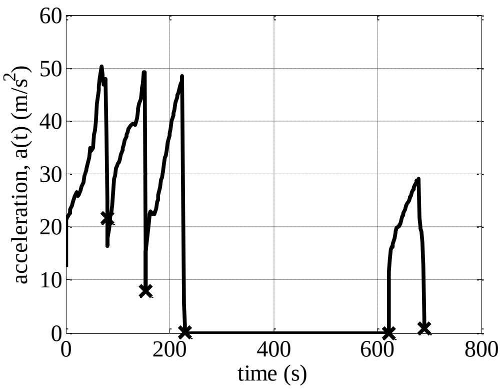


Figure 4.87 Taurus acceleration profile

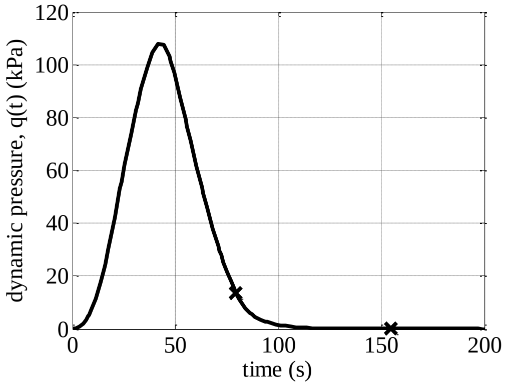


Figure 4.88 Taurus dynamic pressure profile

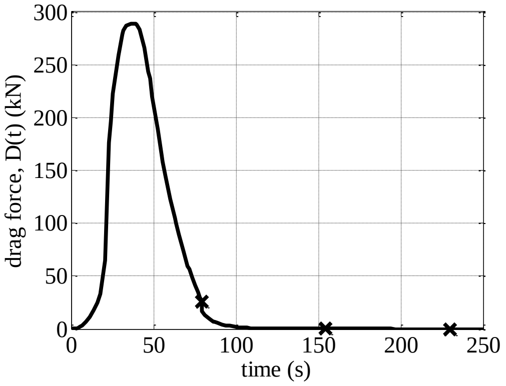


Figure 4.89 Taurus drag force profile

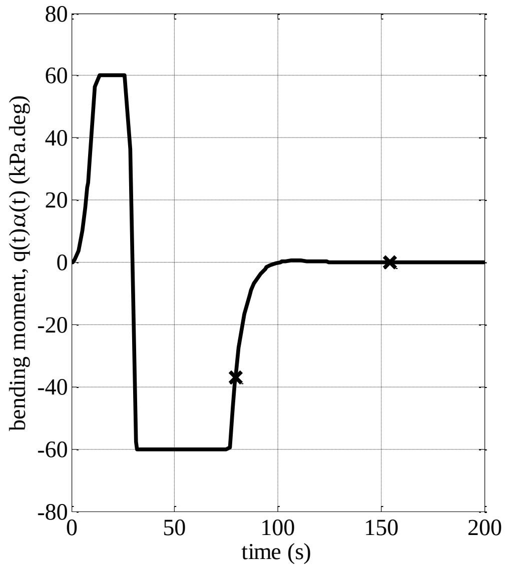


Figure 4.90 Taurus bending moment profile

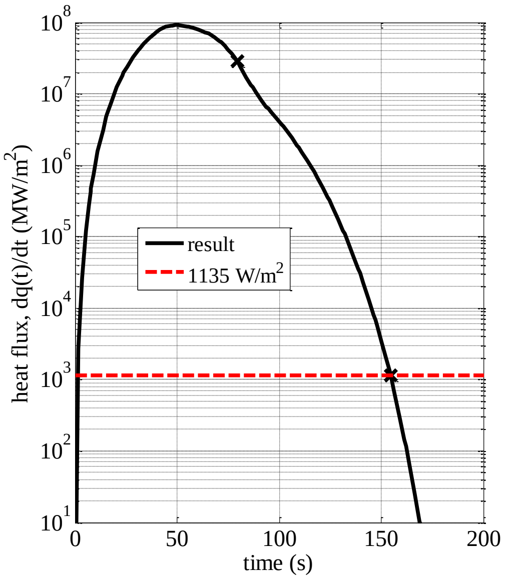


Figure 4.91 Taurus heat flux profile

Launch vehicle’s design dependent parameters such as acceleration and dynamic pressure are illustrated in Figure 4.87 and Figure 4.88, respectively. As seen from these figures, maximum acceleration approaches to 5 g for the first three stages which can be considered as an average value among the launchers in the market, whereas the dynamic pressure rises nearly up to 120 kPa, which is a quite high value. This situation might be come up as a consequence of the fact that the first stage motor of Taurus launch vehicle, namely Castor

120 is derived from the first stage of Peacekeeper intercontinental ballistic missile. It is also seen from Figure 4.90 that one of the constrained trajectory parameters, the bending moment indicator leans to the limit value of 60 kPa° or -60 kPa° during the significant portion of the flight. Figure 4.91 confirms that the separation of payload fairing at the end of second flight phase is appropriate since the corresponding value of heat flux drops under 1135 W/m^2 during flight beginning with 3rd stage ignition. Finally, Figure 4.94 highlights that the vehicle follows a suborbital trajectory with a negative perigee altitude until the last stage burn which provides the required velocity change for circularization.

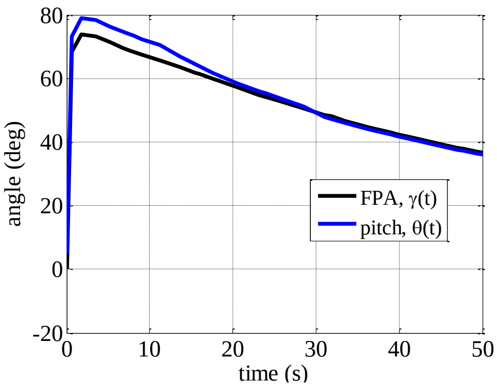


Figure 4.92 Taurus FPA & pitch profile

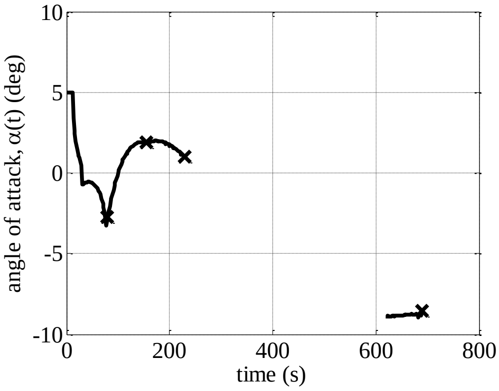


Figure 4.93 Taurus angle of attack profile

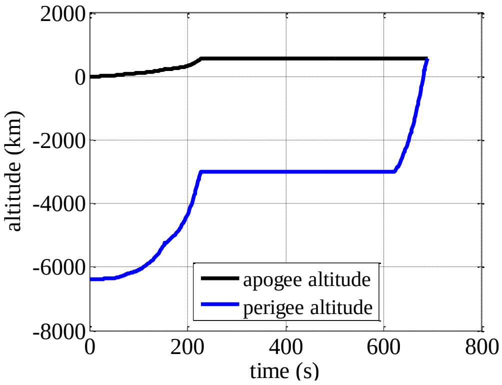


Figure 4.94 Taurus perigee-apogee heights

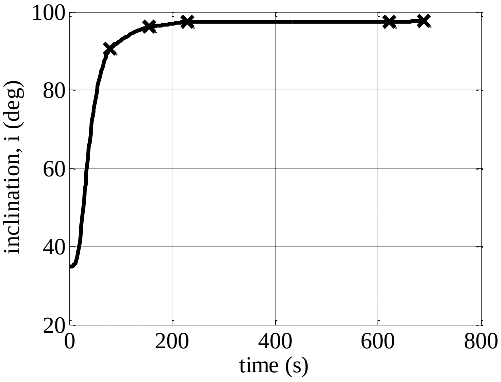


Figure 4.95 Taurus inclination profile

CHAPTER 5

LAUNCH VEHICLE DESIGN

The conceptual design of a launch vehicle is an iterative process and requires strong interaction of multiple disciplines. The major disciplines involved in the design process include configuration and layout, mass properties and sizing, aerodynamics, propulsion, flight dynamics and sometimes cost and reliability analyses. Rowell, Olds and Unal (1996) describes the data exchange between various disciplines involved in launch vehicle conceptual design process within a flow diagram.

Geethaikrishnan, Mujumdar, Sudhakar and Adimurthy (2010) define the traditional launch vehicle conceptual design methodology in two steps. In the first step, propellant and structural mass of each stage that yield optimum stage configuration for each concept are determined. Then, a number of configurations are selected based on a performance index. The selected stage configurations pass through the second step, and remaining configuration variables are then specified through a sequential process of disciplinary designs to arrive at optimum vehicle configurations. Finally, one or more of the vehicle configurations are selected to proceed with preliminary design.

In this traditional approach, mass split between the stages is not free to change, and sequential process does not allow the synergetic effects of mutual coupling between the disciplines which may lead to suboptimal designs. Alternatively, there is a fully coupled design approach in which the propulsion, trajectory and weights and sizing parameters are iteratively refined in an integrated manner to obtain an optimum vehicle configuration.

Such an interactive design approach is now being applied during the conceptual design phase of launch vehicles owing to the considerable increase in computational power. It offers a significant time saving since the design process is automated to some extent, and it provides improved understanding of complex optimization problems by taking into account interactions among design variables.

The objective of this chapter is to find optimal vehicle configurations for a given mission by simultaneous optimization of the thrust profile and the trajectory. Thus, the sizing (propellant and structural mass), thrust time history and burn time of each stage are able to be obtained together with the optimal flight trajectory. The payload mass is given and fixed, and the minimization of the gross lift-off mass (GLOM) is considered as the objective function of this optimization problem since it is the key parameter having a significant impact on both performance and cost. The details of the problem formulation were constructed to achieve an appropriate level of accuracy and precision required for conceptual design while offering affordable computation times.

Both approaches known as decoupled and coupled approaches mentioned above were applied in this study. The proposed methodologies are presented in the following sections together with the application examples, and finally solutions are discussed and analyzed. It is important to point out that the application examples cover only the optimization of stages with solid propellant rocket motors (SRM).

5.1 Decoupled Approach

In the decoupled approach, the staging and the trajectory optimization codes developed to serve for two different objectives and described extensively in CHAPTER 3 and CHAPTER 4 were linked together by an overhead program as illustrated in Figure 5.1. In this program, the difference between the desired payload mass and the maximized payload mass (output of the trajectory optimization subprogram) is compared, and a new ΔV requirement (input of the staging optimization subprogram) is estimated to achieve the desired payload mass. Thus, the vehicle's GLOM is minimized iteratively.

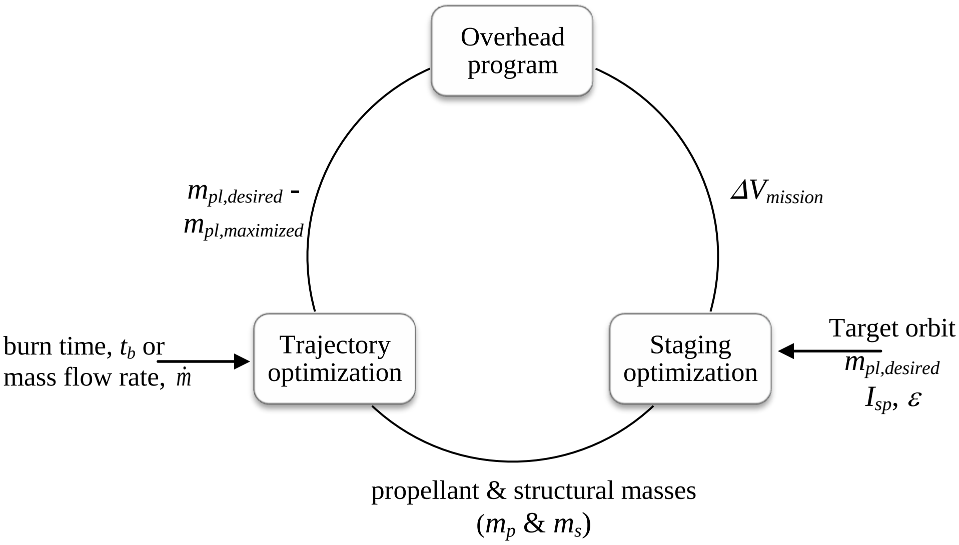


Figure 5.1 Schematic for the decoupled approach

5.1.1 Problem Formulation

Staging optimization method presented in CHAPTER 3 is based on the estimation of $\Delta V_{mission}$, and on the other hand, the basic performance requirements namely the specific impulse (I_{sp}) and the structural ratio (ε) of the launch vehicle's stages are determined by design team considering the technology options. These technology dependent characteristics and the mission requirements such as target orbit parameters and the desired payload mass are input to the staging optimization code given in Appendix C.2, and its output is the optimal mass ratios of stages. Given the structural ratios, one can find the propellant (m_p) and structural mass (m_s) of each stage easily.

Trajectory optimization code was written based on the formulation described in Section 4.1, and the related code to solve for Delta III example presented in Section 4.2 is given in Appendix C.3. The inputs to this code are the mass and performance characteristics of the vehicle and trajectory related parameters such as initial and terminal constraints and path constraints. When these two codes, so called the staging optimization and the trajectory optimization subprograms are executed sequentially and iteratively, the optimal stage weight distribution is first obtained via staging optimization, and either the burn time (t_b) or the mass flow rate ($\dot{m}$) values of the stages are then given as input to the trajectory optimization code with constant $\dot{m}$ and thrust assumption.

Since the mass flow rate was first assumed as constant in this approach, the thrust level can easily be calculated from the known values of I_{sp} and t_b (or $\dot{m}$). During the conceptual design phase, burn time (t_b) can be determined based on the technology options and similar existing systems (Figure D.1).

$$\dot{m} = \frac{m_p}{t_b} \tag{5.1}$$

$$T_{const} = I_{sp} \cdot g_0 \cdot \dot{m} \tag{5.2}$$

In the decoupled approach, burn time is not an optimization parameter; however it can be adjusted if it is desired to be longer or smaller due to the drop zone restrictions of separated stages if any. But, this time thrust and consequently $\dot{m}$ should not be kept as constant in order to conserve the propellant mass and fulfill the mission by the launch vehicle having the same GLOM. In order to achieve this aim, thrust will be variable over time but the total impulse must be the same as the predetermined value by imposing I_{sp} and $\dot{m}$ given at the beginning of the problem. By this way, the resulting launch vehicle configuration will comply with the optimal staging.

To summarize, the staging optimization code finds optimal staging based on estimated $\Delta V_{mission}$, and the trajectory optimization code finds the optimal trajectory and maximum payload mass based on optimal staging. Then, the overhead program compares the values of maximized and desired payload mass, and estimates a new value of $\Delta V_{mission}$. This loop is repeated until the maximized payload mass becomes equal to the desired payload mass.

As it can easily be concluded that the accuracy of the estimated $\Delta V_{mission}$ is not so important since it is corrected through a few iterations using the outputs of trajectory optimization including more realistic gravity and drag calculations.

5.1.2 Example 1: Delta III

In order to evaluate the applicability and accuracy of the decoupled approach, design of a new launch vehicle for a mission specified in Section 4.2 was selected as the first application example.

Technology dependent inputs required for the launch vehicle design are presented in Table 5.1. The structural ratios (ε) were calculated simply from Eq. (3.12) using the data given in Table 4.2. Similarly, the other inputs such as the thrust (T) and the mass flow rate ($\dot{m}$) were also calculated from the data given in Table 4.2 using the Eqs. (5.1) and (5.2).

As it was indicated in Section 3.3, the staging optimization method generates a launch vehicle configuration in serial staging, and then it is converted to its parallel equivalent through the equations presented in Section 3.1.2.2. Therefore, before the execution of the staging and trajectory optimization codes, the original Delta III launch vehicle was first converted to an equivalent hypothetical launch vehicle with four stages in serial (4SLV) stage configuration. The basic performance data derived for 4SLV are also given in Table 5.1 for comparison. Since the boosters and the 1st stage main engine burn together during first and second flight phases, I_{sp} of the 1st and 2nd stages of 4SLV take values between $I_{sp,b}$ and $I_{sp,c}$.

Table 5.1 Comparison of Delta III and 4SLV

Delta III	**Booster**	**1st stage ME**	**2nd stage**	
Specific impulse, I_{sp} (sec)	283.3	301.7	467.2	
Structural ratio, ε (-)	0.118	0.085	0.129	
Burn time, t_b (sec)	75.2	261	700	
4SLV	**1st stage**	**2nd stage**	**3rd stage**	**4th stage**
Specific impulse, I_{sp} (sec)	287.2	289.8	301.7	467.2
Structural ratio, ε (-)	0.096	0.080	0.179	0.129
Burn time, t_b (sec)	75.2	75.2	110.6	700

Then, the trajectory of 4SLV was optimized and compared with the optimal trajectory of Delta III presented in 4.2.5. Figure 5.2 and Figure 5.3 show that results are exactly the same as expected, thus the conversion from parallel to serial staging was confirmed. Overhead program was then executed to obtain a Delta III equivalent launch vehicle configuration designated with 2SLV, which is capable of accomplishing the same mission but with minimum GLOM and without imposing any trajectory constraints.

Table 5.2 shows the comparison of 2SLV and Delta III. The first four-row block in Table 5.2 belongs to the serial equivalent of 2SLV, which is an intermediate solution. The second four-row block in Table 5.2 gives the properties of the optimized launch vehicle for minimum GLOM called 2SLV, while the last four-row block contains the original Delta III launch vehicle's data.

Table 5.2 points out that the optimal staging brings almost 90 tons reduction in GLOM without leaving any excess fuel mass in the tanks. On the other hand, staging optimization proposes to scale up 2nd stage, whose specific impulse is the largest and the structural ratio is moderate as given in Table 5.1. Whereas, it proposes to decrease the size of the 1st stage main engine because its values of specific impulse and structural ratio are not very good.

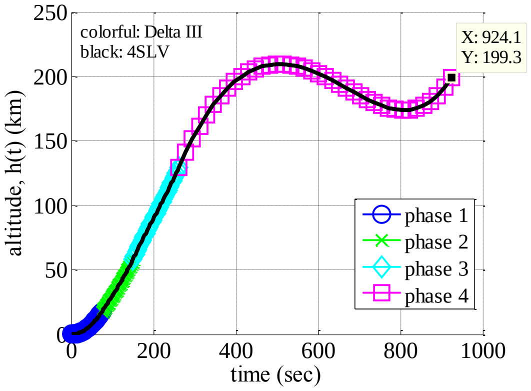


Figure 5.2 Delta-4SLV altitude comparison

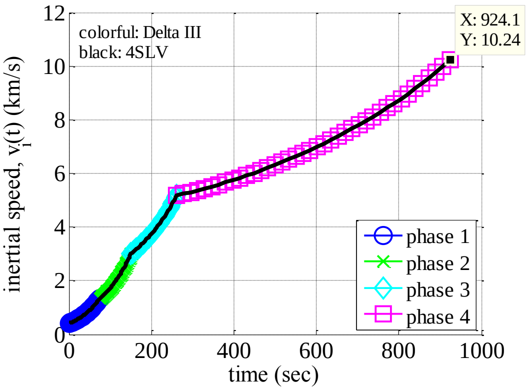


Figure 5.3 Delta-4SLV speed comparison

Table 5.2 Comparison of 2SLV and Delta III

4SLV in serial - optimized	**1st stage**	**2nd stage**	**3rd stage**	**4th stage**	Overall
Structural mass, m_s (kg)	10832	5059	1586	3069	209328 kg + 4164 kg payload
Propellant mass, m_p (kg)	102592	58094	7276	20811	
Burn time, t_b (sec)	59.5	55.6	20	866	
2SLV	**Booster**	**1st stage ME**	**2nd stage**	Overall	
Structural mass, m_s (kg)	9 x 1766	1586	3069	209328 kg + 4164 kg payload (all fuel burned)	
Propellant mass, m_p (kg)	9 x 13200	49150	20811		
Thrust, T (N)	9 x 616345	1077174	110103		
Burn time, t_b (sec)	59.5	135	866		
Delta III	**Booster**	**1st stage ME**	**2nd stage**	Overall	
Structural mass, m_s (kg)	9 x 2280	8830	2480	296404 kg + 4164 kg payload + 886 kg excess fuel	
Propellant mass, m_p (kg)	9 x 17010	95550	16820		
Thrust, T (N)	9 x 628500	1083100	110094		
Burn time, t_b (sec)	75.2	261	700		

Almost 90 tons reduction in GLOM is mainly because of the reduction in propellant mass whereas the values of thrust *T* were converged to almost the same levels. Thus, it can be concluded that the vehicle climbs faster by resulting larger values of axial acceleration and dynamic pressure which are plotted in Figure 5.4 and Figure 5.5, respectively.

However, as mentioned previously these parameters should be kept below certain values in order not to damage the payload or the launch vehicle itself. When these parameters are constrained, the launch vehicle has to climb slower, and thus needs more propellant to achieve the same mission. That's why Delta III is heavier than 2SLV, but still it might not be optimal.

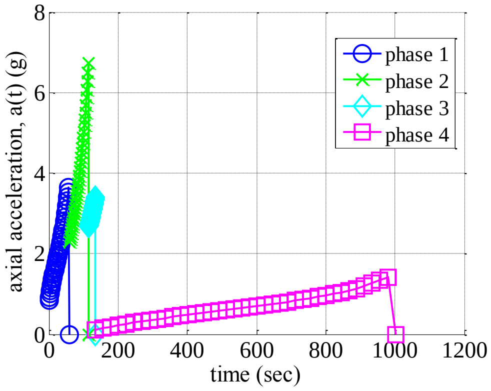


Figure 5.4 2SLV acceleration profile

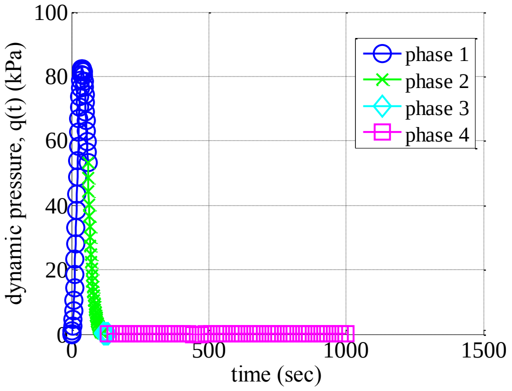


Figure 5.5 2SLV dynamic pressure profile

5.1.3 Example 2: Vega

Design of a new launch vehicle for PEO mission specified in Section 4.3.5 was selected as the second example for the application and assessment of the decoupled approach. Similar to the previous Delta III example, the problem was formulated relatively simple considering the following assumptions and conditions.

1. The vehicle concept and the technology options were considered same as actual Vega configuration.
2. Only the first three solid stages (SRMs) were optimized.
3. AVUM was used in the upper stage as a ready product.
4. The structural ratios and specific impulses of stages were identified based on Vega.
5. The overall mission characteristics such as the launch site location, target orbit, orbit insertion scenarios and trajectory design considerations comply with Section 4.3.5.

6. The short term coast phases after 2nd and 3rd stage separations were not modeled since they do not have a significant impact on the overall performance, and also they cause longer computation times.
7. Trajectory was optimized up to the transfer orbit defined by 150 km x 700 km, 90° inclination orbit. Since it was demonstrated in Section 4.3.5 that there is not a considerable difference in terms of payload mass that can be delivered to target PEO or its transfer orbit.
8. In addition to the trajectory design variables, the thrusts of each SRM were defined as design variables and assumed as constant.
9. Burn times of each SRM were defined by the user as equal to 90 sec which is an average value of similar SRMs given in Figure D.1.
10. No trajectory constraints were imposed.

Thrust levels and the mass properties of the first three solid stages were optimized so as to minimize GLOM for a desired payload mass of 1488 kg, which is the maximized value for Vega launch vehicle for the same mission. The optimal vehicle configuration obtained via decoupled approach for PEO mission are summarized in Table 5.3, its optimal trajectory is given in Figure 5.6 and Figure 5.7, and the most important trajectory parameters are presented in Figure 5.8 to Figure 5.11.

Table 5.3 Optimal vehicle configuration for PEO mission (decoupled)

	1st stage	**2nd stage**	**3rd stage**
Propellant mass, m_p (kg)	87765	19579	6016
Thrust, T (kN)	2678	613.3	194
Burn time, t_b (sec)	90	90	90
GLOM (kg)	128013		

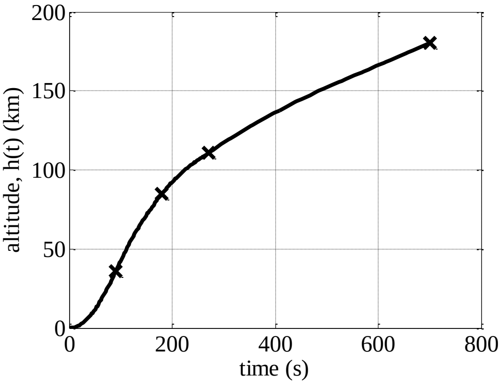


Figure 5.6 Altitude profile (Vega decoupled)

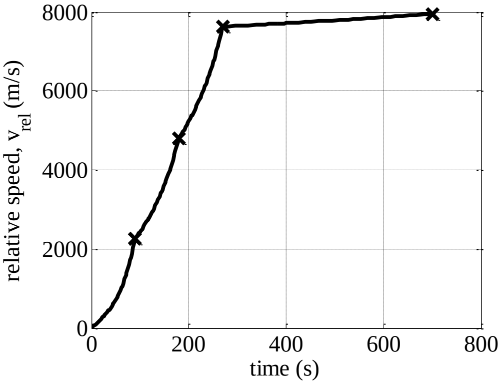
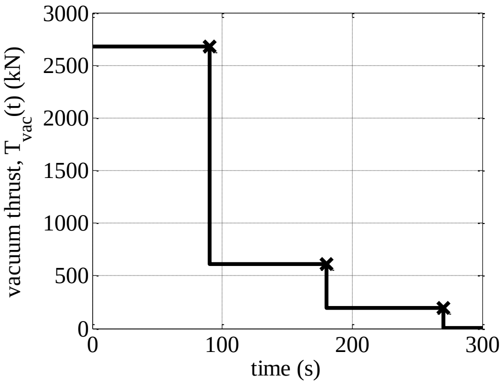


Figure 5.7 Speed profile (Vega decoupled)

Since no constraints were imposed on acceleration, dynamic pressure and bending moment in this problem for simplicity, the maximum values of these parameters increase up to 65.1 m/s^2, 81.8 kPa and 333.3 kPa°, respectively and are definitely unacceptable for most of the existing satellite launch vehicles. These values are also much higher than the values regarding the optimal trajectory of PEO mission of Vega presented in Section 4.3.5.



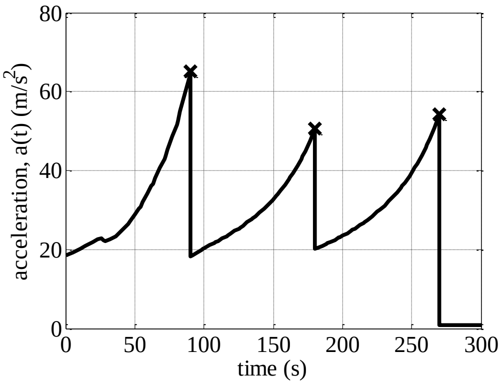
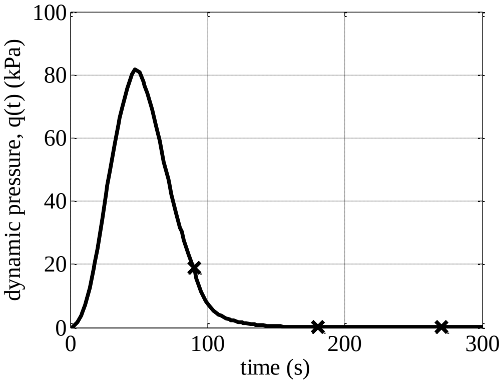


Figure 5.8 Thrust profile (Vega decoupled) Figure 5.9 Acceleration (Vega decoupled)



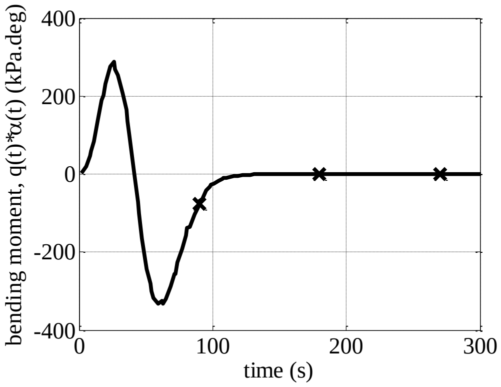


Figure 5.10 Dyn. pressure (Vega decoupled) Figure 5.11 Bend. moment (Vega decoupled)

5.1.4 Discussion

Applicability of the decoupled approach for launch vehicle design was successfully proven with the two examples above. The problem solved in Section 5.1.3 was then solved by the coupled approach as well, and the results are given in Section 5.2.2.1 in details. However, a short comparison is given here in Table 5.4.

Table 5.4 Decoupled vs. coupled approach for PEO mission (T_{const} & t_b: fixed)

	Decoupled	**Coupled**
1st stage mass (kg)	96304	99205
2nd stage mass (kg)	22124	19181
3rd stage mass (kg)	6832	6105
Payload mass (kg)	1488	1488
GLOM (kg)	128013	127245
1st stage thrust (kN)	2678	2758
2nd stage thrust (kN)	613.3	531.7
3rd stage thrust (kN)	194	173.3
Acceleration (m/s^2)	65.1	73.4
Dynamic pressure (kPa)	81.8	86.1
Bending moment (kPa°)	333.3	418.1
Run time (sec)	369	28

As noted in Table 5.4, minimized GLOM obtained through the decoupled approach is 768 kg more than GLOM obtained as a result of the coupled approach. This difference can be explained as an error margin of 0.6%, which is totally acceptable in conceptual design level studies. However, since the ultimate optimal is searched, the output of decoupled approach seems to be a little bit suspicious. As described previously, in the decoupled approach, the staging and the trajectory are solved separately and the stage mass distribution is specified before the trajectory optimization. Therefore, the velocity losses dependent on the flight path and flight time have a general effect on staging ignoring their split among flight phases.

In the coupled approach, however, the staging and the trajectory are optimized simultaneously. Therefore, ΔV generated by each stage (consequently the stage masses) are determined considering the actual velocity losses corresponding to the relevant flight phase.

Table 5.4 also indicates that the optimal trajectory obtained as a result of the coupled approach with constant thrust assumption and without trajectory constraints exposes higher structural loads than the trajectory optimized by the decoupled approach. This excess amount results from the fact that the coupled approach’s solution follows a steeper trajectory with less GLOM compared to the solution of the decoupled approach.

Although both approaches answered to conceptual design needs of launch vehicles in a reasonable accuracy, the coupled approach was found more reliable and practical for thrust profile optimization. As a conclusion, the decoupled approach was not applied on more realistic scenarios such as optimization under constraints and allowing thrust shaping.

5.2 Coupled Approach

Performance requirements of solid propellant rocket motors (SRMs) used in launch vehicle stages are driven by trajectory constraints especially the maximum allowable acceleration and dynamic pressure. SRM performance requirements include parameters like specific impulse, burn time and thrust. Therefore, the need for adaptation of thrust profile specifically to the mission requirements and constraints is extremely important for optimal launch vehicle design.

A systematic design procedure to find optimal SRMs for the predefined launch vehicle concept and its specified mission was proposed within the scope of this thesis. An overview of this procedure is depicted in Figure 5.12.

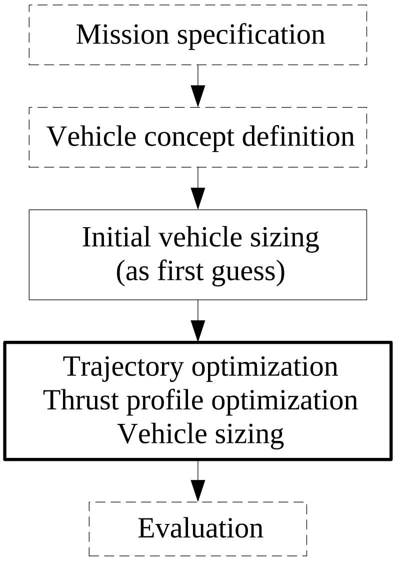


Figure 5.12 Schematic for the coupled approach

As seen from Figure 5.12, first of all, the vehicle concept (staging type, number of stages, technological preferences on propellant composition and structural materials, etc.) is chosen by taking into consideration the mission requirements. Then, basic performance requirements are determined by design team according to the technology options. In the next step, initial vehicle sizing is performed by the staging optimization tool described in CHAPTER 3. The output of this step was utilized as a reference vehicle configuration just to provide an initial guess for the design variables to be optimized.

Afterwards, the trajectory optimization tool described in CHAPTER 4 takes over the task of simultaneous optimization of trajectory, thrust profile and vehicle sizing parameters. Since the general formulation of the trajectory optimization problem was elaborated in Section 4.1, only the details regarding the modeling of the thrust profile is described in the next section.

5.2.1 Problem Formulation

The variation of thrust with respect to time can be modeled as a constant or a variable profile. When the thrust is assumed as constant, the same principles applied on the examples covered in Section 5.1 are exactly valid. To summarize, the burn time is fixed and predefined by the user taking into account the similar vehicles (Figure D.1), and the corresponding thrust level and propellant mass are then optimized.

When it is required to model the thrust as a varying quantity with respect to time, thrust time curve modeling is performed considering the following principles.

- The thrust profiles at vacuum conditions are defined as design variables. The effective thrust is calculated in the dynamic equations according to the flight altitude.
- Thrust shapes are represented by *n* number of piecewise continuous linear functions (Figure 5.13). The initial thrust (T_i) and the *n* number of slopes with evenly spaced time intervals are treated as parameters in the optimization. Increasing the number of linear segments, more accurate approximation of the thrust profile can be achieved.

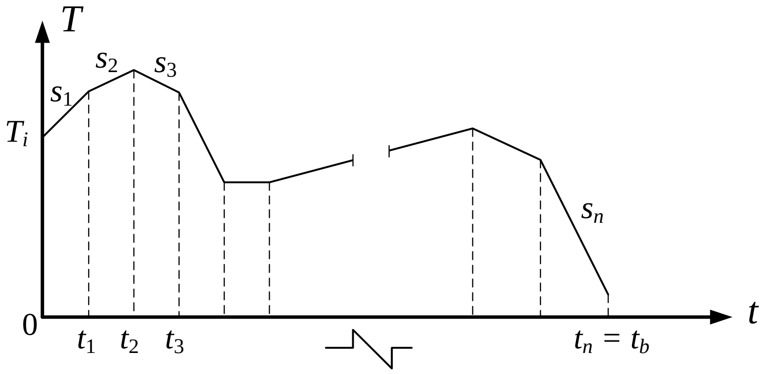


Figure 5.13 Thrust time curve modeling

- Burn times are defined as design variables and optimized within a range defined by the user considering the application area and the state of the art in technology.

Thrust profile modeling has to be done that it finally can be realized by reasonable grain design; therefore some limitations should be imposed on typical SRM thrust parameters. These are listed in the following.

- Maximum value of thrust (T_{max})
- Maximum over minimum thrust ratio (*TR*)

$$\frac{T_{\max}}{T_{\min}} \leq TR \tag{5.3}$$

- Maximum thrust rate: derivatives of the functions, the rates of change (slope) are restricted.

$$-s_{lower}\ \frac{T}{\sec} \leq \frac{dT}{dt} \leq s_{upper}\ \frac{T}{\sec} \tag{5.4}$$

It is important to point out that the upper and the lower bounds for thrust values must be determined by the user to ensure realistic results if the thrust profile is considered as constant. It is clear that the other thrust shaping constraints such as maximum over minimum thrust ratio and the maximum thrust rate are not applicable to problems with constant thrust assumption.

5.2.2 Example 1: Vega

The trajectory optimization of Vega for its reference mission of 700 km PEO was performed and its results were given in Section 4.3.5. An optimal vehicle configuration was then generated for this reference mission using the decoupled approach and its results were given in Section 5.1.3. The same reference mission was also chosen as design objective for the first example of the coupled approach. The problem was formulated according to the methodology described in Section 5.2.1 and solved by the trajectory optimization tool developed within this thesis.

5.2.2.1 Constant Thrust

The launch vehicle design problem specific to PEO mission was first solved under the constant thrust assumption in order to demonstrate the applicability of the coupled approach. The assumptions and conditions stated in Section 5.1.3 are also valid for this example. The only difference lies in the method of solution. The resulting optimal vehicle configuration is summarized in Table 5.5 and the important trajectory parameters are presented in Figure 5.14 to Figure 5.19.

Table 5.5 Optimal vehicle configuration for PEO mission (coupled – T_{const})

	1st stage	**2nd stage**	**3rd stage**
Propellant mass, m_p (kg)	90409	16974	5375.7
Thrust, T (kN)	2758	531.7	173.3
Burn time, t_b (sec)	90	90	90
GLOM (kg)	127245		

When the resulting optimal trajectories are compared with the results of the decoupled approach, it can be concluded that the trajectories are almost the same. As also discussed in Section 5.1.4, the decoupled approach gives a slightly heavier vehicle, and the coupled approach proposes to use a larger and more powerful motor in the first stage, and lighter and smaller motors in the upper stages in order to decrease the gross lift-off mass.

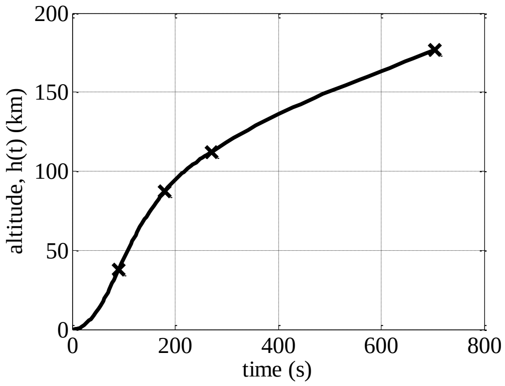


Figure 5.14 Altitude profile (Vega coupled)

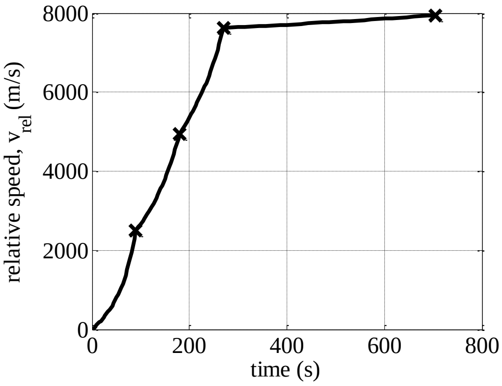


Figure 5.15 Speed profile (Vega coupled)

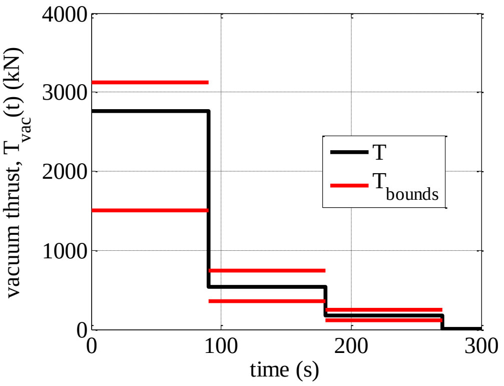


Figure 5.16 Thrust profile (Vega coupled)

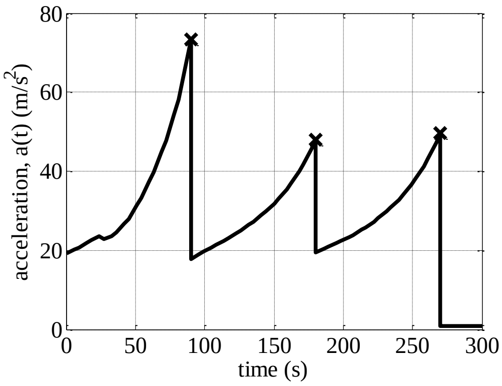


Figure 5.17 Acceleration (Vega coupled)

Figure 5.16 shows the optimal constant thrust profile for the first three stages together with the user defined upper and lower bounds. It is seen that the optimized thrust levels remain within the defined region and thrust constraints are satisfied. However, resulting acceleration levels are quite high since no constraints are imposed on the trajectory. Similarly, the time history of the bending moment indicator exhibits considerably high values as given in Figure 5.19.

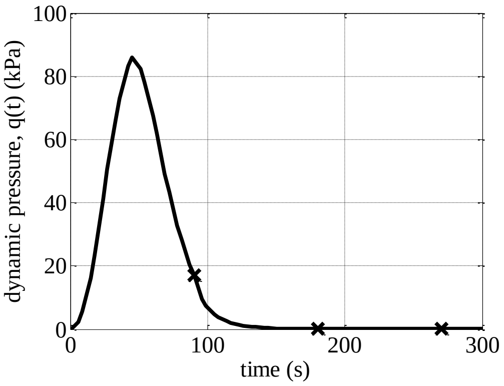


Figure 5.18 Dyn. pressure (Vega coupled)

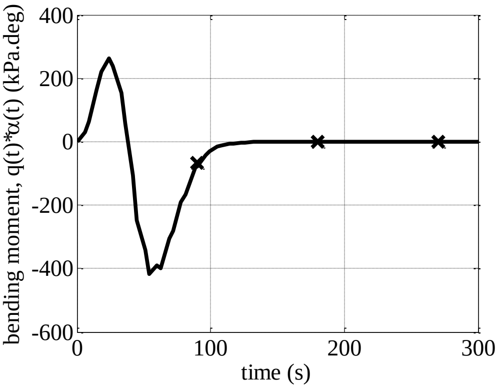


Figure 5.19 Bending moment (Vega coupled)

5.2.2.2 Variable Thrust

The first seven assumptions stated in Section 5.1.3 were applied in this problem, whereas the last three were replaced by the following ones.

1. Thrust shape of the 1st stage SRM was represented by 15 piecewise continuous linear functions, whereas 2nd and 3rd stage SRMs were modeled by 3 linear functions.

2. Thrust time profiles of each SRM were optimized as satisfying the following constraints.

   2.1. Maximum value of thrust (estimated based on relevant Vega SRMs)

$$T_{\max,1} = 3500\,\text{kN} \qquad T_{\max,2} = 1200\,\text{kN} \qquad T_{\max,3} = 250\,\text{kN} \tag{5.5}$$

   2.2. Maximum over minimum thrust ratio (taken similar to relevant Vega SRMs)

$$TR_1 = 1.5 \qquad TR_2 = TR_3 = 2.2 \tag{5.6}$$

   2.3. Maximum thrust rate as proposed by Johnson (1975)

$$-2\%\text{ of } T_{\max} \le \dot{T} \le 0.5\%\text{ of } T_{\max} \text{ for all stages} \tag{5.7}$$

3. Burn times of each SRM were optimized within the range of 60 sec - 120 sec. This range was determined based on the burn time values of SRMs used in similar launch vehicles (Figure D.1). The initial guesses were simply set to 90 sec which is an average value.

4. The following trajectory constraints were imposed.

    4.1. Acceleration, $a_{max} \leq 4.5$ g
    4.2. Dynamic pressure, $q_{max} \leq 55$ kPa
    4.3. Angle of attack, $-5° \leq \alpha^{(1),(2)} \leq 5°$ & $-15° \leq \alpha^{(3),(4)} \leq 15°$
    4.4. Bending moment indicator, -60 kPa° $\leq q\alpha^{(1)} \leq 60$ kPa°

Trajectory optimization was accomplished in two successive steps. In the first step, thrust levels were optimized without any trajectory constraints and assuming constant thrust and fixed burn time, which is equal to 90 sec which is an average value of similar SRMs. A new vehicle configuration was then obtained by performing staging optimization again considering the amount of velocity change attained by the vehicle at previous trajectory optimization run. Finally, the resulting vehicle and the constant thrust levels corresponding to fixed burn time of 90 sec were given as an initial guess to the actual problem defined in the previous paragraphs.

The problem was successfully solved within 530 seconds of CPU time on a computer with 2.67 GHz i7 processor and 12 GB of memory. All trajectory constraints were met and the desired transfer orbit was exactly achieved. The optimal solution is summarized in Table 5.6, and the important trajectory parameters are presented in Figure 5.20 to Figure 5.25 alongside with the unconstrained solution of the problem in order to interpret the effect of trajectory constraints. The mass and performance characteristics of the minimized GLOM vehicle without any trajectory constraints are tabulated in Table 5.7.

Table 5.6 Optimal solution for PEO mission (constrained coupled – *T(t)*)

	1st stage	**2nd stage**	**3rd stage**
Propellant mass, m_p (kg)	82428	22759	6488
Average thrust, T (kN)	2337	953.7	156.9
Burn time, t_b (sec)	96.85	67.28	120
GLOM (kg)	126285		

Table 5.7 Optimal solution for PEO mission (unconstrained coupled – *T(t)*)

	1st stage	**2nd stage**	**3rd stage**
Propellant mass, m_p (kg)	79705	20275	5928
Average thrust, T (kN)	2925	952.7	143.3
Burn time, t_b (sec)	74.83	60	120
GLOM (kg)	119860		

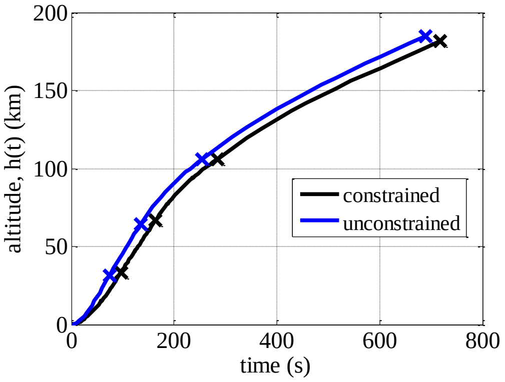


Figure 5.20 Altitude (cons. vs. unc.)

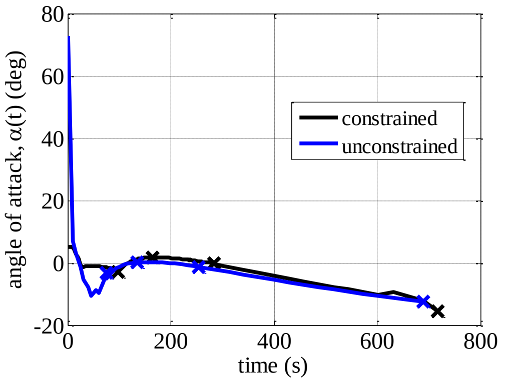


Figure 5.21 AOA (cons. vs. unc.)

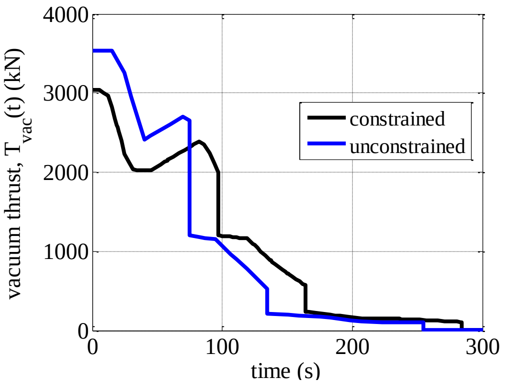


Figure 5.22 Thrust (cons. vs. unc.)

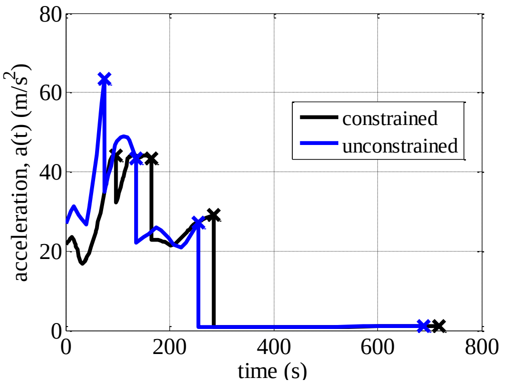


Figure 5.23 Acceleration (cons. vs. unc.)

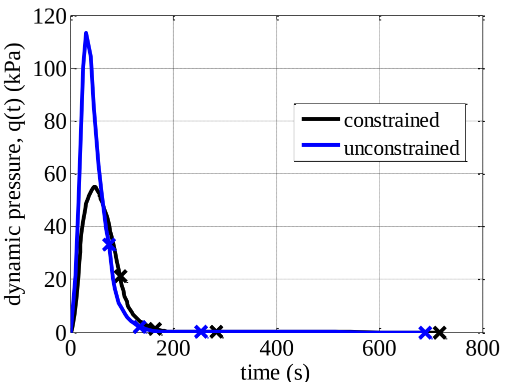


Figure 5.24 Dyn. pressure (cons. vs. unc.)

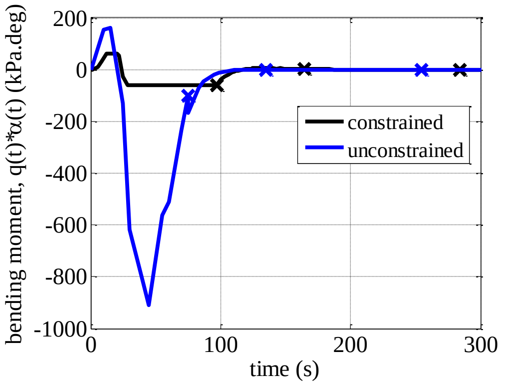


Figure 5.25 Bending moment (cons. vs. unc.)

The mass distribution between stages is presented in Table 5.8 together with the values of actual Vega launch vehicle. This table points out that the desired mission can be accomplished by the optimal vehicle whose mass is 8.3% less compared to Vega.

Table 5.8 Mass comparison between optimal vehicle and actual Vega

	Vega (kg)	**Optimal (kg)**	**Difference (%)**
1st stage	96243	90447	-6.0
2nd stage	26300	25178	-4.3
3rd stage	12000	7367	-38.6
4th stage	1265	1265	0.0
Payload fairing	540	540	0.0
Payload	1430	1488	+4.1
GLOM	137778	126285	-8.3

Table 5.9 shows the comparison of the velocity change (Δv) generated by the first three stages and the resulting inertial speed (v) of the vehicle at the end of burn-out of the corresponding stages. According to Table 5.9, both vehicles reach almost the same speed at the end of 3rd stage's burn-out as expected. However, it seems that Vega was forced to produce more delta-V to overcome velocity losses incurred during flight due to the nonoptimality of its trajectory and its heaviness.

Table 5.9 Delta-V and speed comparison between optimal vehicle and Vega

	Vega (m/s)		**Optimal (m/s)**		
	Δv	v	Δv	v	Δv **Difference (m/s)**
1st stage	2779.5	1856	2904.0	2015	-124.5
2nd stage	2401.5	4036	2841.9	4718	-440.3
3rd stage	3684.4	7625	2973.5	7616	710.9
Total	8865.4		8719.4		145.4

The comparisons of optimized thrust profiles with the reference thrust profiles of Vega are illustrated in Figure 5.26, Figure 5.27 and Figure 5.28. The thrust profile of the optimal vehicle's 1st stage is in a close agreement with P80. This demonstrates that the evenly spaced 16 points appear to be sufficient to represent the actual thrust profile. The maximum and the local minimum of the curves match quite well, but the local maximum around 75 seconds after lift-off is slightly shifted with a decrease in magnitude in the optimal solution. The reason for this decrease is mainly due to the higher thrust in the early periods of operation and the incomplete modeling of the thrust tail-off.

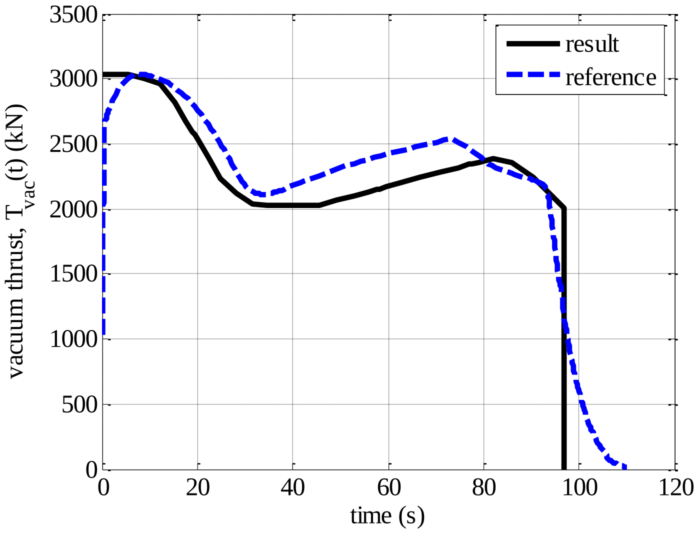


Figure 5.26 Thrust profile comparison of 1st stage SRM

The thrust profile of the 2nd stage which was represented by three linear functions given in Figure 5.27 coincides to a great extent with Z23, but the 3rd stage's thrust time curve is considerably different than Z9. The similar drastic difference is also observed in the values of mass and delta-V presented in Table 5.8 and Table 5.9, respectively. These results imply that the thrust shapes of the first two stages are driven by the trajectory constraints within the endoatmospheric portion of the flight, whereas the impact of the trajectory on the 3rd stage's thrust profile is relatively insignificant since aerodynamic forces are no more effective in this region.

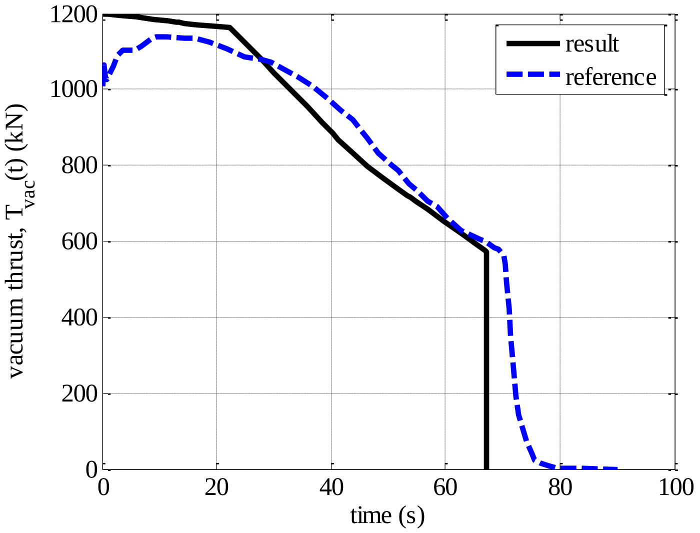


Figure 5.27 Thrust profile comparison of 2nd stage SRM

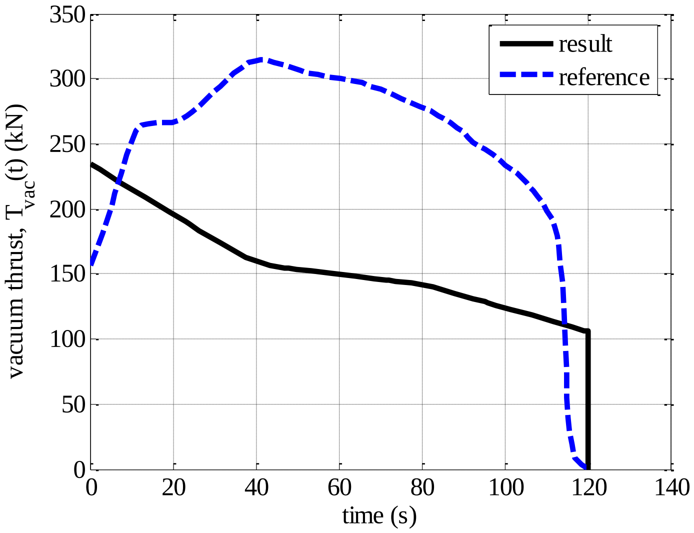


Figure 5.28 Thrust profile comparison of 3rd stage SRM

In order to examine the differences in mass distribution more rigorously, ideal rocket equation was utilized. Trajectory-independent optimal staging providing the same delta-V given in Table 5.9 was obtained for both Vega and the optimal vehicle and the results are compared in terms of delta-V and mass split and presented in Table 5.10 and Table 5.11, respectively. The percentage differences also confirm that optimal solution under the given constraints is closer to the ideal solution compared to Vega.

Table 5.10 Delta-V comparison between Vega, optimal vehicle and ideal vehicle

	Vega (m/s)	**Ideal (m/s)**	**%Dev. from Ideal**
1st stage	2779.5	3217.6	13.62
2nd stage	2401.5	2746.7	12.57
3rd stage	3684.4	2901.1	-27.00
Total	8865.4	8865.4	
	Optimal (m/s)	**Ideal (m/s)**	**%Dev. from Ideal**
1st stage	2904.0	3165.8	8.27
2nd stage	2841.9	2698.1	-5.33
3rd stage	2973.5	2855.5	-4.13
Total	8719.4	8719.4	

Table 5.11 Mass comparison between Vega, optimal vehicle and ideal vehicle

	Vega (kg)	**Ideal (kg)**	**%Dev. from Ideal**
1st stage	96243	102645	6.24
2nd stage	26300	23136.4	-13.67
3rd stage	12000	7000.6	-71.41
GLOM	137778	135477	
	Optimal (m/s)	**Ideal (m/s)**	**%Dev. from Ideal**
1st stage	90447	94491.6	4.28
2nd stage	25178	21822.7	-15.38
3rd stage	7367	6777.2	-8.7
GLOM	126285	125844	

5.2.2.3 Discussion

For the reference mission of PEO four different vehicle configurations were generated using the decoupled and the coupled approaches and with constant and variable thrust options. Results are presented together with the optimized trajectory of Vega for the same mission according to the solution given in Section 4.3.5.

Table 5.12 Comparison of optimal vehicle configurations and Vega for PEO mission

	Decoupled Constant thrust Unconstrained	**Coupled Constant thrust Unconstrained**	**Coupled Variable thrust Constrained**	**Coupled Variable thrust Unconstrained**	**Vega Optimal trajectory**
m_1 (kg)	96304	99205	90447	87459	96243
m_2 (kg)	22124	19181	25718	22911	26300
m_3 (kg)	6832	6105	7367	6732	12000
m_{pl} (kg)	1488	1488	1488	1488	1488
GLOM (kg)	128013	127245	126285	119860	137836
$T_{ave,1}$ (kN)	2678	2758	2337	2925	2253.7
$T_{ave,2}$ (kN)	613.3	531.7	953.7	952.7	883.2
$T_{ave,3}$ (kN)	194	173.3	156.9	143.3	262.6
$t_{b,1}$ (sec)	90	90	96.85	74.83	106.7
$t_{b,2}$ (sec)	90	90	67.28	60	76
$t_{b,3}$ (sec)	90	90	120	120	116.8
a_{max} (m/s^2)	65.1	73.4	44.13	63.4	45.73
q_{max} (kPa)	81.8	86.1	55	113.3	54.8
$q\alpha_{max}$ (kPa°)	333.3	418.1	60	907.9	60
run time (sec)	369	28	530	132	800

If Table 5.12 is examined in the sense of GLOM, the first remarkable point is that there is at least 10 tons difference between the optimal vehicle configurations and the existing launch vehicle Vega. This difference points out that Vega is not optimum in terms of mass minimization.

Although the problems with constant thrust assumptions were formulated to provide relatively good initial guesses to the variable thrust optimization problems, they yield reasonable average values in terms of GLOM. Actually, keeping the thrust constant within a flight phase while the mass of the launch vehicle continuously decreases, adversely affect the performance and mission flexibility. In accordance with this argument, vehicle configurations modeled with variable thrust are lighter than the vehicles with constant thrust. The vehicle with minimum GLOM is the one having variable thrust and is not subject to any trajectory constraints. Therefore, the acceleration, dynamic pressure and bending moment indicator reach peak values during flight.

5.2.3 Example 2: Taurus

The second example used to assess the applicability of the developed trajectory optimization tool to the simultaneous optimization of thrust and mass profile of the stages together with the trajectory of the launch vehicle. The coupled approach was employed to obtain the optimal vehicle configuration for the SSO mission of the Taurus launch vehicle solved in Section 4.4. Following assumptions and conditions were applied in the formulation of the problem.

1. The vehicle concept and the technology options were considered same as actual Taurus configuration. Thus, the structural ratios and specific impulses of stages were identified based on Taurus. All four stages were optimized to generate minimum GLOM vehicle.
2. The overall mission characteristics were specified in accordance with Section 4.4. The entire trajectory from ground up to the circular target orbit was optimized.
3. Thrust shape of the first two stages were represented by 5 piecewise continuous linear functions, 3rd stage were modeled by 3 linear functions and finally the thrust of the fourth and the last stage was assumed as constant.
4. Thrust time profiles of each SRM were optimized as satisfying the following constraints.

$$T_{\max,1} = 2000\ \text{kN} \quad T_{\max,2} = 500\ \text{kN} \quad T_{\max,3} = 150\ \text{kN} \quad T_{\max,4} = 50\ \text{kN} \tag{5.8}$$

$$TR_1 = 1.5 \qquad TR_2 = TR_3 = 1.3 \qquad TR_4 = 2.2 \tag{5.9}$$

$$-2\%\ \text{of}\ T_{\max} \le \dot{T} \le 0.5\%\ \text{of}\ T_{\max}\ \text{for all stages} \tag{5.10}$$

5. Burn times of the 2nd and 3rd stages were constrained to be equal to each other based on the fact that their diameters are same. The range of 60 sec - 120 sec was specified as lower and upper bounds, respectively.

6. The following trajectory constraints were imposed.

   6.1. Acceleration, $a_{max} \leq 50$ m/s^2
   6.2. Dynamic pressure, $q_{max} \leq 120$ kPa
   6.3. Angle of attack, $-5° \leq \alpha^{(1),(2)} \leq 5°$ & $-15° \leq \alpha^{(3),(4)} \leq 15°$
   6.4. Bending moment indicator, -60 kPa° $\leq q\alpha^{(1)} \leq 60$ kPa°

The mass and performance characteristics of the optimal vehicle configuration are summarized in Table 5.13 and Table 5.14, and the thrust time profile comparison of stages are illustrated in from Figure 5.29 to Figure 5.32. According to these figures, it can be concluded that the thrust profile of the 1st stage is almost exactly the same as the actual reference vehicle, thrust profiles of the 2nd and 3rd stages can be considered as reasonably compatible, whereas the last stage's thrust profile has the maximum difference.

Table 5.13 Optimal vehicle configuration for SSO mission of Taurus

	1st stage	**2nd stage**	**3rd stage**	**4th stage**
Propellant mass, m_p (kg)	48133	10572	3580	885
Average thrust, T (kN)	1719.1	423	146.1	27.9
Burn time, t_b (sec)	76.6	70.22	70.22	90
GLOM (kg)	71728			

Table 5.14 Mass comparison between optimal vehicle and actual Taurus

	Taurus (kg)	**Optimal (kg)**	**Difference (%)**
1st stage	53928	52968	-1.78
2nd stage	14186	12298	-13.31
3rd stage	3496	4137	18.33
4th stage	1127	1238	9.81
Payload fairing	300	300	0.0
Payload	670	787	17.46
GLOM	73707	71728	-2.69

The most important trajectory parameters are presented in from Figure 5.33 to Figure 5.36. It can be seen from these figures that the vehicle successfully achieves the target orbit and the trajectory constraints are satisfied.

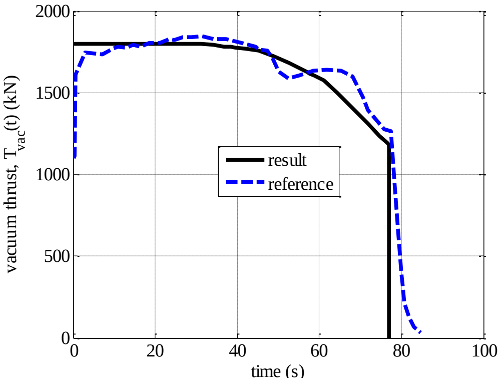


Figure 5.29 Thrust comparison of 1st stage

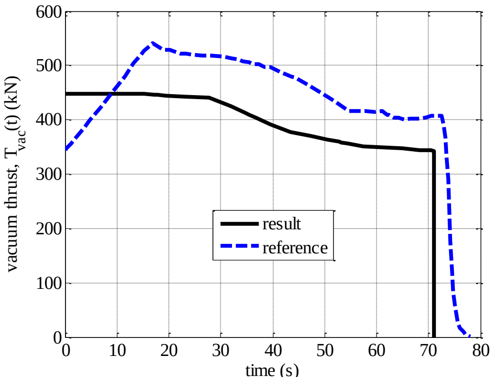


Figure 5.30 Thrust comparison of 2nd stage

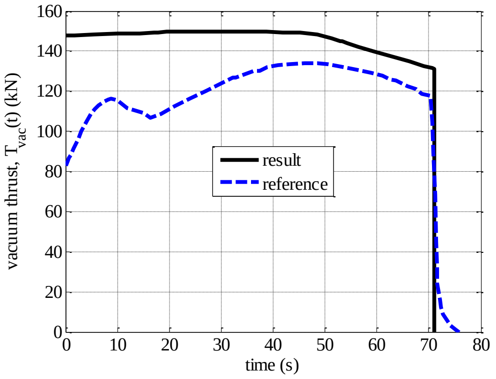


Figure 5.31 Thrust comparison of 3rd stage

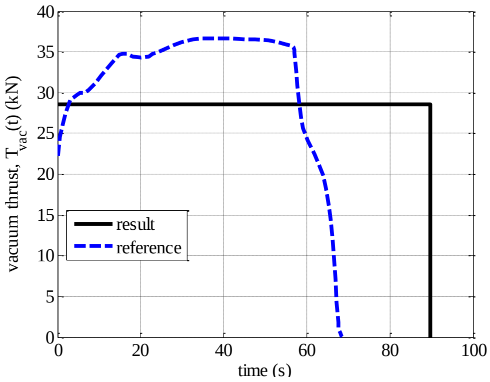


Figure 5.32 Thrust comparison of 4th stage

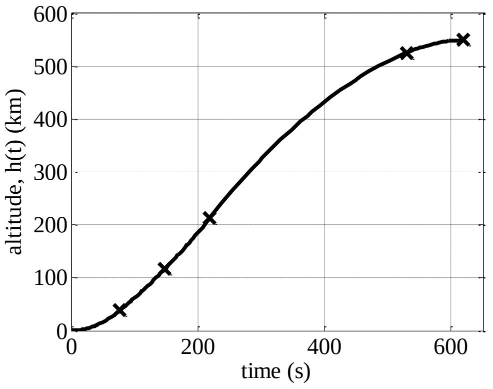


Figure 5.33 Altitude (Taurus coupled)

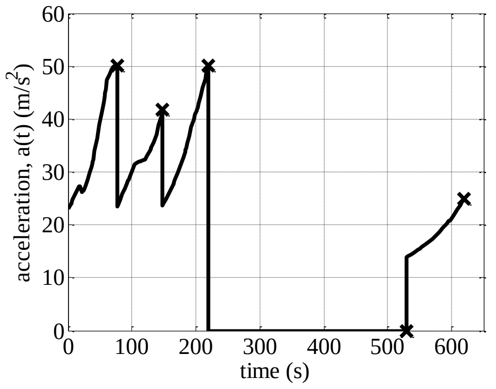
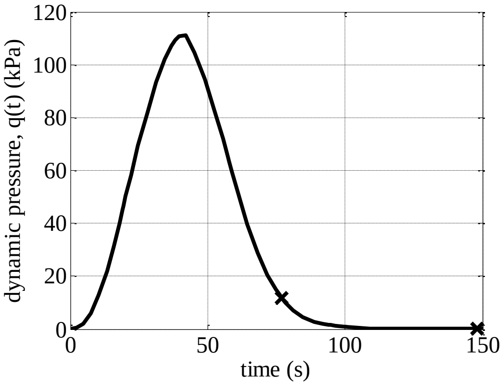


Figure 5.34 Acceleration (Taurus coupled)



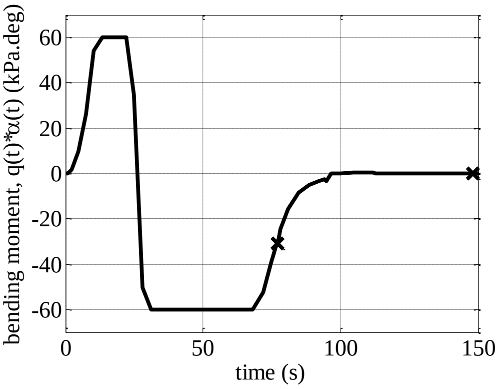


Figure 5.35 Dyn. pressure (Taurus coupled) Figure 5.36 Bend. moment (Taurus coupled)

CHAPTER 6

CONCLUSIONS

6.1 Thesis Summary and Conclusions

Throughout this thesis work, the development and verification of a quick and effective tool to find optimal vehicle configurations in the early design phases was achieved. Within this framework, first, a trajectory optimization code was developed to determine the optimal trajectory in terms of position and velocity, together with the control functions corresponding to that trajectory while maximizing the payload mass. This code utilized a general purpose optimal control solver GPOPS-II®. Next, a staging optimization code was developed to determine the staging parameters (number of stages, mass distribution between stages, and the propellant and structural mass of each individual stage) which minimized the gross lift-off mass. Finally, these two codes were integrated for simultaneous optimization of the thrust and mass profile of the stages together with the trajectory of the launch vehicle.

This newly developed tool can be utilized for both of the following two types of problems.

1. Preliminary mission design of an existing launch vehicle, i.e. for the maximization of the payload mass for a given vehicle and target orbit
2. Conceptual design of a new launch vehicle, i.e., for the minimization of vehicle mass for a given payload and target orbit

With reference to the objectives primarily set in the introduction chapter of this research, the major contribution and conclusions of this thesis are as follows.

- What has been accomplished in this thesis is not necessarily something totally new for the whole world. In other words, there already exist some software packages which offer simultaneous optimization of ascent trajectory and launch vehicle design. However, these software packages have been developed by governmental agencies or universities for national space programs and their sale to other countries is restricted due to export control regulations. Therefore, there was still a demand, at least in Turkey, for such an integrated and efficient system design and trajectory optimization tool for performance calculations and simulations of launch vehicles. Finally, it is important to point out that only the open source information, data and documents in the world’s literature have been used in this thesis work.

- It is well known that relatively simplified engineering models and associated physical laws can be utilized even in the field of space rocket design in its early phases. As an example, the preliminary trajectories of the launch vehicles are generally computed by means of standard guidance laws (e.g. pitch-over, gravity turn, bilinear tangent law, etc.) mentioned in Section 4.3.4 even though they provide near optimum solutions. These so-called standard guidance laws are particularly preferred since they offer reasonably similar performance results as those obtainable by full trajectory optimization tools owing to their reduced complexity and robust performance.
- However, the tool developed within this thesis can be used to achieve full trajectory optimization, because it allows the control variables to be optimized at all discretization points generated by the optimal control software GPOPS-II. Thus, it offers a so-called "real optimal trajectory".
- Moreover, this thesis has not only achieved the above-mentioned aims, but has also offered a practical, quick, and reliable design tool to find optimal vehicle configurations with an appropriate level of accuracy required for the early design phases. The following are simultaneously optimized to obtain the so called "optimal vehicle configuration".
    - Propellant and structural mass distribution between stages
    - Thrust profile of the stages
    - Flight trajectory of the launch vehicle
- Two different launch vehicle design methodologies were applied in this work, which are designated as the decoupled and coupled approaches. These methodologies were exemplified by two reference missions belonging to Vega and Taurus launch vehicles, and both are evaluated to be acceptable for conceptual design of launch vehicles with an appropriate level of accuracy and affordable computation times.
- It is also important to highlight that the quality of the thrust profile modeling is bound to the user-specified limitations imposed on the maximum thrust, thrust ratio and thrust rate. For the present research, these limitations were determined based on the qualified reference launch vehicles. As a consequence, they may not be applicable to significantly different solid rocket motors, and therefore specialized expertise is always beneficial and required.
- In any case, after the optimal thrust profile is found, the feasibility of the ballistic performance and the mass characteristics of solid rocket motor should be confirmed considering the burn rate of the propellant, operating pressure, grain geometry and other solid rocket motor related design parameters. These parameters are independent design variables and can be adjusted so as to obtain the desired thrust profile by subsystem designers.
- The last as a conclusion is that, as mentioned before, the developed tool can be considered to be quite fast compared to its similar counterparts. For instance, the present work offers a solution in the order of minutes (less than 15 minutes for the examples covered in this work) utilizing a gradient-based algorithm. On the other

hand, the other known up-to-date computer tools give results in the order of hours, especially the ones that make use of the evolutionary optimization algorithms.

6.2 Future Developments

Although this thesis work deals with a large number of topics, suggestions for short terms developments are listed in the following.

- Although the developed tool was verified by several mission examples based on real launch scenarios, the accessible data in the open literature was limited. Therefore, few more verifications and validations to be organized in a systematic approach will be very much desirable whenever such opportunities becomes available.
- If it is going to be employed as it is, the resulting optimal vehicle configurations should necessarily be re-evaluated at least by the trajectory optimization tool developed within this thesis in order to ensure that the desired payload mass is satisfied. This time trajectory design considerations should be modeled in more details with fixed mass and thrust parameters. For instance, it becomes possible to include wind effects, controllability issues, etc.
- Although the trajectory models implemented in this thesis are fully general, and any kind of trajectory of an expendable launch vehicle into Earth’s orbit could in theory be computed, the developed tool was verified only with ascent trajectory optimization into orbits like sun-synchronous and polar Earth orbits and geostationary transfer orbits. Therefore, the applicability of this tool had better be verified for other kind of applications ranging from rendezvous vehicles to reentry vehicles for human spaceflight and interplanetary or deep space transfer vehicles.
- It is a well known fact that gradient-based optimization algorithms are sensitive to initial guesses. As it is commonly applied in the literature, an initial guess generator with much less optimization variables utilizing standard guidance laws can be developed in order to minimize the convergence issues.
- Regarding the above-mentioned issue about the feasibility of the solid rocket motor, the more favorable solution is to consider the solid rocket motor parameters with a simplified internal ballistics model and to optimize also the rocket motors together with the trajectory. In order to achieve this goal, a large-scale mixed-integer nonlinear programming solver might be utilized, or GPOPS-II can be employed after it is integrated with an appropriate heuristic algorithm.
- If the developed tool is upgraded so as to be worked with an evolutionary algorithm, then the fidelity of the subsystem models involving discrete optimization variables can be increased easily. The following subsystem models are strongly recommended to be included as a top priority.
    - Solid rocket motor ballistic performance prediction, geometrical sizing, and mass estimation
    - Interstage structures geometrical sizing and mass estimation

- An aerodynamic prediction code such as Missile Datcom can be integrated into the optimization cycle.
- A 6-DOF trajectory model with appropriate guidance and control algorithms can be implemented for more detailed trajectory design.

REFERENCES

Addison, E. (2009). Elliptical orbit node passage time. *SelectedWorks of Eric Addison*. Utah State University. Retrieved July 12, 2014, from http://works.bepress.com/eric_addison/1

Adkins, C. N. (1970). Optimization of multistage rockets including drag. *Journal of Spacecraft, 7*(6), 751-755.

Akhtar, S., & Linshu, H. (2005, December). Simulation-based optimization strategy for liquid fueled multi-stage space launch vehicle. *Proceedings of the Sixth International Conference on Parallel and Distributed Computing, Applications and Technologies.* Dalian, China: IEEE.

ANSI. (1992, February). *Recommended practice for atmospheric and space flight vehicle coordinate systems.* ANSI/AIAA R-004-1992.

Arslantaş, Y. E. (2012). *Conceptual design optimization of a nano-satellite launcher.* MSc Thesis, Middle East Technical University, Aerospace Engineering, Ankara, Turkey.

ASTOS. (2011). *ASTOS® AeroSpace Trajectory Optimization Software*. Retrieved May 13, 2011, from http://www.astos.de/products/astos

ATK. (2008, May). Space propulsion products catalog. Elkton, MD: Alliant Techsystems Inc.

Balesdent, M., Bérend, N., Dépincé, P., & Chriette, A. (2012). A survey of multidisciplinary design optimization methods in launch vehicle design. *Structural and Multidisciplinary Optimization, 45*, 619–642.

Bayley, D. J., Hartfield, R. J., Burkhalter, J. E., & Jenkins, R. M. (2008). Design optimization of a space launch vehicle using a genetic algorithm. *Journal of Spacecraft and Rockets, 45*(4), 733-740.

Benson, D. (2004). *A gauss pseudospectral transcription for optimal control.* PhD Thesis, Massachusetts Institute of Technology, Aeronautics and Astronautics, Cambridge, MA.

Benson, D. A., Huntington, G. T., Thorvaldsen, T. P., & Rao, A. V. (2006). Direct trajectory optimization and costate estimation via an orthogonal collocation method. *Journal of Guidance, Control, and Dynamics, 29*(6), 1435-1440.

Betts, J. T. (1998). Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics, 21*(2), 193-207.

Betts, J. T. (2010). *Practical methods for optimal control and estimation using nonlinear programming* (2nd ed.). USA: SIAM.

Betts, J. T., & Huffman, W. P. (1990). Application of sparse nonlinear programming to trajectory optimization. *Journal of Guidance, 15*(1), 198-206.

Blake, W. B. (1998). *Missile Datcom: User's manual - 1997 FORTRAN 90 revision.* Wright-Patterson Air Force Base. Ohio: AFRL-VA-WP-TR-1998-3009.

Bradt, J. E., Jessick, M., & Hardtla, W. (1987). Optimal guidance for future space applications. *AIAA Guidance, Navigation and Control Conference*, (pp. 745-754). Monterey, CA.

Brauer, G. L., Cornick, D. E., & Stevenson, R. (1977). *Capabilities and applications of the program to optimize simulated trajectories.* NASA CR-2770.

Braun, R. D., Moore, A. A., & Kroo, I. M. (1997). Collaborative approach to launch vehicle design. *Journal of Spacecraft and Rockets, 34*(4), 478-486.

Brown, K. R., & Johnson, G. W. (1967). Real-time optimal guidance. *IEEE Transactions on Automatic Control, AC-12*(5), 501-506.

Brusch, R. G. (1974). A nonlinear programming approach to space shuttle trajectory optimization. *Journal of Optimization Theory and Applications, 13*(1), 94-118.

Brusch, R. G. (1977). Trajectory optimization for the Atlas/Centaur launch vehicle. *Journal of Spacecraft and Rockets, 14*(9), 550-555.

Brusch, R. G., & Peltier, J. P. (1974). Gradient generation for parametric control models. *Acta Astronautica, 1*(11-12), 1453-1466.

Builder, C. H. (1959). General solution for optimization of staging of multistaged boost vehicles. *ARS Journal*, 497-499.

Burghes, D. N. (1974). Optimum staging of multistage rockets. *International Journal of Mathematical Education in Science and Technology, 5*(1), 3-10.

Calise, A. J., Melamed, N., & Lee, S. (1998). Design and evaluation of a three-dimensional optimal ascent guidance algorithm. *Journal of Guidance, Control, and Dynamics, 21*(6), 867-875.

Castellini, F. (2008). *Global optimization techniques in space missions design.* MSc Thesis, Politecnico di Milano, Aerospace Engineering, Italy.

Castellini, F. (2012). *Multidisciplinary design optimization for expendable launch vehicles.* PhD Thesis, Politecnico di Milano, Aerospace Engineering, Italy.

Castellini, F., & Lavagna, M. R. (2012). Comparative analysis of global techniques for performance and design optimization of launchers. *Journal of Spacecrafts and Rockets, 49*(2), 274-285.

Castellini, F., Lavagna, M., & Erb, S. (2008, September). Concurrent optimization of launcher architectures and ascent trajectories with global optimization algorithms. *59th International Astronautical Congress.* Glasgow, Scotland.

Chenglong, H., Xin, C., & Leni, W. (2008, December). Optimizing RLV ascent trajectory using PSO algorithms. *2nd International Symposium on Systems and Control in Aerospace and Astronautics.* Shenzhen, China.

Chobotov, V. A. (2002). *Orbital mechanics* (3rd ed.). Reston, VA: AIAA.

Cobb, E. R. (1961). Optimum staging technique to maximize payload total energy. *ARS Journal*, 342-344.

COESA. (1976). *US Standard Atmosphere.* US Government Printing Office. Washington, DC: NASA TM-X-74335.

Conway, B. A. (2012). A survey of methods available for the numerical optimization of continuous dynamic systems. *Journal of Optimization Theory and Applications, 152*(2), 271-306.

Cremaschi, F. (2013). Trajectory optimization for launchers and re-entry vehicles. In G. Fasano, & J. D. Pinter, *Modeling and optimization in space engineering* (pp. 159-185). New York: Springer Science+Business Media.

Curtis, H. D. (2005). *Orbital mechanics for engineering students.* Burlington, MA: Elsevier Butterworth-Heinemann.

DeltaIII. (1997, March). *Payload Planners Guide.* Huntington Beach, CA: McDonnell Douglas Corporation.

Dukeman, G. A. (2002). Atmospheric ascent guidance for rocket-powered launch vehicles. *AIAA Guidance Navigation and Control Conference*, (pp. 486-496).

Eagle, C. D. (n.d.). *Single impulse de-orbit from circular and elliptical orbits*. Retrieved July 22, 2014, from The Orbital and Celestial Mechanics Website: http://www.cdeagle.com/pdf/deorbit.pdf

Ebrahimi, M., Farmani, M. R., & Roshanian, J. (2011). Multidisciplinary design of a small satellite launch vehicle using particle swarm optimization. *Structural and Multidisciplinary Optimization, 44*, 773-784.

Edge, E. R., & Powers, W. F. (1976). Shuttle ascent trajectory optimization with function space quasi-newton techniques. *AIAA Journal, 14*(10), 1369-1376.

Encyclopedia Astronautica. (n.d.). Retrieved January 12, 2012, from http://www.astronautix.com/lvs/gslv.htm

Enright, P. J., & Conway, B. A. (1992). Discrete approximations to optimal trajectories using direct transcription and nonlinear programming. *Journal of Guidance, Control and Dynamics, 15*(4), 994-1002.

Fahroo, F., & Ross, M. I. (2002). Direct trajectory optimization by a chebyshev pseudospectral method. *Journal of Guidance, Navigation, and Control, 25*(1), 160-166.

Fiorillo, M., Giliberti, F., Angelone, M., Milana, C., & Serraglia, F. (2013). Vega solid rocket motors flight performances results. *49th AIAA/ASME/SAE/ASEE Joint Propulsion Conference.* San Jose, CA.

Fischer, H. M. (2005). Trajectories and orbital mechanics: Ascent of launchers, orbits and orbital transfers. Retrieved May 5, 2011, from http://www.scribd.com

Gallucci, S., Battie, F., Volpi, M., Fossati, T., & Curti, G. (2012, October). Vega launch vehicle first flight mission analysis – VV01. *IEEE First AESS European Conference on Satellite Telecommunications (ESTEL)* (pp. 1-5). Rome, Italy: IEEE.

Gath, P. F., & Well, K. H. (2002). Initial guess generation for rocket ascent trajectory optimization using indirect methods. *Journal of Spacecraft and Rockets, 39*(4), 515-521.

Geethaikrishnan, C., Mujumdar, P. M., Sudhakar, K., & Adimurthy, V. (2008, June). Genetic algorithm guided gradient search for launch vehicle trajectory optimization. *Proceedings of the International Conference on Aerospace Science and Technology.* Bangalore, India.

Geethaikrishnan, C., Mujumdar, P. M., Sudhakar, K., & Adimurthy, V. (2010, April). A hybrid MDO architecture for launch vehicle conceptual design. *51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference.* Orlando, FL: AIAA-2010-2757.

Gill, P. E., Murray, W., & Saunders, M. A. (2005). SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Journal on Optimization, 47*(1), 99-131.

Goldsmith, M. (1957). On the optimization of two stage rockets. *Jet Propulsion, 27*, 415-416.

Gray, J. S., & Alexander, R. V. (1965). Cost and weight optimization for multistage rockets. *Journal of Spacecraft, 2*(1), 80-86.

Griffin, M. D., & French, J. R. (2004). *Space vehicle design* (2nd ed.). Reston, VA: AIAA.

Hall, H. H., & Zambelli, E. D. (1958). On the optimization of multistage rockets. *Jet Propulsion, 28*, 463-465.

Hammond, W. E. (1999). *Space transportation: A systems approach to analysis and design.* Reston, VA: AIAA.

Hammond, W. E. (2001). *Design methodologies for space transportation systems.* Reston, VA: AIAA.

Hanson, J. M., Shrader, M. W., & Cruzen, C. A. (1995). Ascent guidance comparisons. *The Journal of the Astronautical Sciences, 43*(3), 307-326.

Hargraves, C. R., & Paris, S. W. (1987). Direct trajectory optimization using nonlinear programming and collocation. *Journal of Guidance, Control, and Dynamics, 10*(4), 338-342.

Hill, P. R. (1967, May 25-27). Cost optimization of multistage rockets. *Conference on the use of Space Systems for Planetary Geology and Geophysics.* Boston, MA.

Huntington, G. T. (2007). *Advancement and analysis of a gauss pseudospectral transcription for optimal control problems.* PhD Thesis, Massachusetts Institute of Technology, Aeronautics and Astronautics, Cambridge, MA.

Jänsch, C., Schnepper, K., & Well, K. H. (1990). Ascent and descent trajectory optimization of Ariane V-Hermes. *AGARD Conference Proceedings No. 489 on Space Vehicle Flight Mechanics,* (pp. 4.1-4.27).

Johnson, I. L. (1975). Optimization of the solid-rocket assisted space shuttle ascent trajectory. *Journal of Spacecraft, 12*(12), 765-769.

Jorris, T. R., & Cobb, R. G. (2009). Three-dimensional trajectory optimization satisfying waypoint and no-fly zone constraints. *Journal of Guidance, Control, and Dynamics, 32*(2), 551-572.

Karslı, G., & Tekinalp, O. (2005, June). Trajectory optimization of advanced launch system. *Proceedings of 2nd International Conference on Recent Advances in Space Technologies,* (pp. 374-378). İstanbul, Turkey.

Lares. (2009). Vega launch vehicle and qualification mission characteristics. *The First International Lares Workshop*. Retrieved July 24, 2014, from http://www.lares-mission.com/2009_Workshop_Rome_talks/de_pasquale_spaziani.pdf

Lawden, D. F. (1963). *Optimal trajectories for space navigation.* London: Butterworths.

Ley, W., Wittmann, K., & Hallmann, W. (2009). Chapter 3: Space tansportation systems. In *Handbook of space technology* (pp. 115-200). Sussex, UK: John Wiley & Sons.

Loftus, J. P., & Teixeira, C. (1999). Chapter 18: Launch systems. In W. J. Larson, & J. R. Wertz (Eds.), *Space mission analysis and design* (3rd ed., pp. 719-744). El Segundo, CA & New York, NY: Microcosm Press & Springer.

Lu, P., Zhang, L., & Sun, H. (2005, August). Ascent guidance for responsive launch: A fixed-point approach. *AIAA Guidance, Navigation, and Control Conference and Exhibit,* (pp. 1-18). San Francisco, CA.

Lubowe, A. G. (1965). Application of dynamic programing to the optimum staging of rockets. *Journal of Spacecraft, 2*(1), 110-111.

Maini, A. K., & Agraval, V. (2011). *Satellite technology: Principles and applications* (2nd ed.). John Wiley & Sons.

Malina, F. J., & Summerfield, M. (1947). The problem of escape from the earth by rocket. *Journal of the Aeronautical Sciences, 14,* 471.

Markl, A. W. (2001). *An initial guess generator for launch and reentry vehicle trajectory optimization.* Dr.-Ing Thesis, University of Stuttgart, Institute of Flight Mechanics and Control, Germany.

Matlab. (2012). Version R2012b (8.0.0). Natick, MA, USA: The MathWorks Inc.

Moss, J. B. (1995). Chapter 7: Launch vehicles. In P. Fortescue, & J. Stark (Eds.), *Spacecraft systems engineering* (2nd ed., pp. 181-211). UK: John Wiley & Sons.

Ndefo, E., Encalada, J., Hallman, W., Wang, W., & Abbott, P. (2004). Launch vehicle design and development. In L. B. Rainey (Ed.), *Space modeling and simulation* (pp. 289-368). El Segundo, CA & Reston, VA: The Aerospace Press & AIAA.

NIMA. (2000). *World geodetic system 1984, its definition and relationships with local geodetic systems.* National Imagery and Mapping Agency, US Department of Defense. TR8350.2.

OTIS. (2011). *Optimal Trajectories by Implicit Simulation*. Retrieved May 13, 2011, from http://otis.grc.nasa.gov

Patterson, M. A., & Rao, A. V. (2013a). *GPOPS-II version 1.0: A general-purpose Matlab toolbox for solving optimal control problems using the radau pseudospectral method.* Gainesville, FL: University of Florida.

Patterson, M. A., & Rao, A. V. (2013b). GPOPS-II: A Matlab software for solving multiple-phase optimal control problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming. *ACM Transactions on Mathematical Software, 39*(3), 1: 1-41.

Patton, J. A., & Hopkins, J. B. (2006). Atlas V for commercial passenger transportation. *AIAA Space 2006 Conference and Exposition.* San Jose, CA.

Peters, J. F. (2004). *Spacecraft systems design and operations.* USA: Kendall/Hunt Publishing Company.

Picone, J. M., Hedin, A. E., Drob, D. P., & Aikin, A. C. (2002). NRLMSISE-00 empirical model of the atmosphere: Statistical comparisons and scientific issues. *Journal of Geophysical Research, 107*(A12), 1468.

Pisacane, V. L. (2005). Chapter 4: Spacecraft propulsion, launch systems, and launch mechanics. In V. L. Pisacane (Ed.), *Fundamentals of space systems* (2nd ed., pp. 171-235). New York: Oxford University Press.

Pontani, M. (2014). Particle swarm optimization of ascent trajectories of multistage launch vehicles. *Acta Astronautica, 94,* 852–864.

Rafique, A. F., He, L. S., Zeeshan, Q., Kamran, A., & Nisar, K. (2011). Multidisciplinary design and optimization of an air launched satellite launch vehicle using a hybrid heuristic search algorithm. *Engineering Optimization, 43*(3), 305-328.

Rao, A. V. (2009, August 10-13). A survey of numerical methods for optimal control. *AAS/AIAA Astrodynamics Specialist Conference.* Pittsburgh, PA: AAS Paper 09-334.

Rao, A. V., Benson, D. A., Darby, C., Patterson, M. A., Francolin, C., Sanders, I., & Huntington, G. T. (2010). Algorithm 902: GPOPS, A Matlab software for solving multiple-phase optimal control problems using the gauss pseudospectral method. *ACM Transactions on Mathematical Software, 37*(2), 22: 1-39.

Rea, J. (2003). Launch vehicle trajectory optimization using a legendre pseudospectral method. *AIAA Guidance, Navigation, and Control Conference and Exhibit*, (pp. 2030-2040).

Ross, I. M., D'Souza, C., Fahroo, F., & Ross, J. B. (2003, August). A fast approach to multi-stage launch vehicle trajectory optimization. *AIAA Guidance, Navigation and Control Conference and Exhibit.* Austin, TX: AIAA-2003-5639.

Rowell, L. F., & Korte, J. J. (2003). *Launch vehicle design and optimization methods and priority for the advanced engineering environment.* NASA/TM-2003-212654.

Rowell, L. F., Olds, J. R., & Unal, R. (1996, September). Recent experiences in multidisciplinary conceptual design optimization for launch vehicles. *6th NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.* Bellevue, WA: AIAA-96-4050-CP.

Serraglia, F. (2008). Vega Launcher. ESA-ESRIN. Retrieved March 9, 2014, from http://dma.dima.uniroma1.it:8080/users/lsa_eps/MATERIALE/Vega-20-11-2008.pdf

Srivastava, T. N. (1966). Optimum staging with varying thrust attitude angle. *Defence Science Journal, 16*, 153-164.

Stanley, D. O., Engelund, W. C., Lepsch, R. A., McMillin, M., Wurster, K. E., Powell, R. W., . . . Unal, R. (1994). Rocket-powered single-stage vehicle configuration selection and design. *Journal of Spacecraft and Rockets, 31*(5), 792-798.

Stengel, R. (2008). Lecture 2: Launch vehicle design. *MAE 342: Space System Design course lecture notes, Princeton University*. Retrieved March 24, 2011, from http://www.princeton.edu/~stengel/MAE342Lecture2.pdf

Styrk, O. v. (1993). Numerical solution of optimal control problems by direct collocation. In R. Bulirsch, A. Miele, J. Stoer, & K. H. Well (Eds.), *Optimal Control - Calculus of Variation, Optimal Control Theory and Numerical Methods* (Vol. 111, pp. 129-143). Basel: Birkhäuser.

Subotowicz, M. (1958). The optimization of the n-step rocket with different construction parameters and propellant specific impulses in each stage. *Jet Propulsion, 28*, 460-463.

Sutton, G. P., & Biblarz, O. (2001). *Rocket propulsion elements* (7th ed.). Canada: John Wiley & Sons.

Taurus. (1999, September). Payload User's Guide. (release 3.0). Dulles, VI: Orbital Sciences Corporation.

Tawakley, V. B. (1967). On the calculation of optimum mass distribution of a multi-stage rocket vehicle. *Defence Science Journal, 17*, 53-66.

Tewari, A. (2007). *Atmospheric and space flight dynamics: Modeling and simulation with Matlab and Simulink.* Boston: Birkhaeuser.

Tewari, A. (2011). *Advanced control of aircraft, spacecraft and rockets.* UK: John Wiley & Sons.

The Space Launch Report. (n.d.). Retrieved July 23, 2014, from http://www.spacelaunchreport.com/taurus.html

Todd, D., & Angelo, J. A. (2005). *A to Z of scientists in space and astronomy.* New York: Facts on File, Inc.

Turner, M. J. (2009). *Rocket and spacecraft propulsion: Principles, practice and new developments* (3rd ed.). Germany: Springer-Praxis.

Vega. (2006, March). User's Manual. (issue 3, revision 0). Retrieved February 17, 2011, from www.arianespace.com

Vega. (2014, April). User's Manual. (issue 4, revision 0). Retrieved May 14, 2014, from http://www.arianespace.com/launch-services-vega/Vega-Users-Manual_Issue-04_April-2014.pdf

Vertregt, M. (1956). A method of calculating the mass ratio of step rockets. *Journal of the British Interplanetary Society, 15*, 95.

Vincenty, T. (1975). Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. *Survey Review, XXII*(176), 88-93.

Weisbord, L. (1958). A generalized optimization procedure for n-staged missiles. *Jet Propulsion, 28*, 164-167.

Well, K. H. (2003). Neighboring vehicle design for a two-stage launch vehicle. In A. Miele, & A. Frediani (Eds.), *Advanced design problems in aerospace engineering* (Vol. 1, pp. 131-153). New York, NY: Kluwer Academic & Plenum Publishers.

Wikipedia. (n.d.). *Guiana Space Centre*. Retrieved May 15, 2014, from http://en.wikipedia.org/wiki/Guiana_Space_Centre

Xuan, Y., Zhang, W., & Zhang, Y. (2009). Pseudospectral method based trajectory optimization and fairing rejection time analysis of solid launch vehicle. *Science in China Series E: Technological Sciences, 52*(11), 3198-3206.

Yokoyama, N., & Suzuki, S. (2005). Modified genetic algorithm for constrained trajectory optimization. *Journal of Guidance, Control, and Dynamics, 28*(1), 139-144.

Zhang, D., Lu, X., Liu, L., & Wang, Y. (2012, June 24-26). An online ascent phase trajectory reconstruction algorithm using gauss pseudospectral method. *Proceedings of 2012 International Conference on Modelling, Identification and Control,* (pp. 1184-1189). Wuhan, China.

Zhang, L., & Lu, P. (2008). Fixed-point algorithms for optimal ascent trajectories of launch vehicles. *Engineering Optimization, 40*(4), 361-381.

Zipfel, P. H. (2007). *Modeling and simulation of aerospace vehicle dynamics* (2nd ed.). Reston, VA: AIAA.

Zotes, F. A., & Peñas, M. S. (2010). Multi-criteria genetic optimisation of the manoeuvres of a two-stage launcher. *Information Sciences, 180,* 896–910.

APPENDIX A

SATELLITE ORBITS

Satellites travel around the Earth along predetermined repetitive paths called orbits. Figure A.1 represents an elliptical orbit with one focus at the Earth's center.

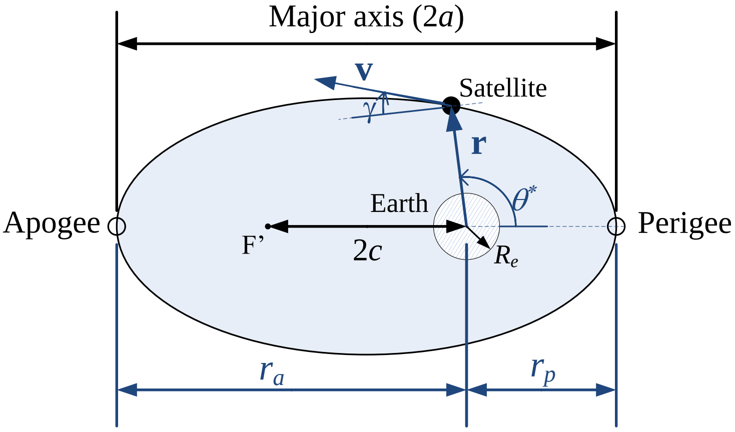


Figure A.1 Elliptical orbit

The *apogee* is the point on the orbit that is farthest from the Earth's center; whereas the *perigee* is the point closest to the Earth's center. Distance from the apogee to the Earth's center is called the a*pogee radius* (r_a) and the distance from the perigee to the Earth's center is called the *perigee radius* (r_p). The *apogee altitude* (h_a) and the *perigee altitude* (h_p) are the heights above the Earth's surface and expressed as

$$h_a = r_a - R_e \tag{A.1}$$

$$h_p = r_p - R_e \tag{A.2}$$

In Eqs. (A.1) and (A.2),
R_e is the Earth's equatorial radius, i.e. R_e = 6378.137 km.

A.1 Orbital Elements

There are six parameters required to uniquely identify a specific orbit and they are called classical orbital elements, also known as Keplerian parameters. The two main elements, which are the semimajor axis (a) and the eccentricity (e), respectively define the size and the shape of the ellipse.

1. ***Semimajor axis*** **(*a*)** defines the size of the orbit and it is the half length of the major axis of the ellipse.

$$a = \frac{r_a + r_p}{2} \tag{A.3}$$

2. ***Eccentricity*** **(*e*)** defines the shape of the orbit and it is the ratio of the distance between the two foci to the length of the major axis.

$$e = \frac{c}{a} = \frac{r_a - r_p}{r_a + r_p} \tag{A.4}$$

For a circular orbit $e = 0$ ($a = r$); whereas for an elliptical orbit $0 < e < 1$ ($a > 0$). Trajectory is parabolic when $e = 1$ ($a = \infty$) and hyperbolic when $e > 1$ ($a < 0$).

Two elements, namely the inclination (i) and the right ascension of the ascending node (Ω) define the orientation of the orbital plane in space (Figure A.2).

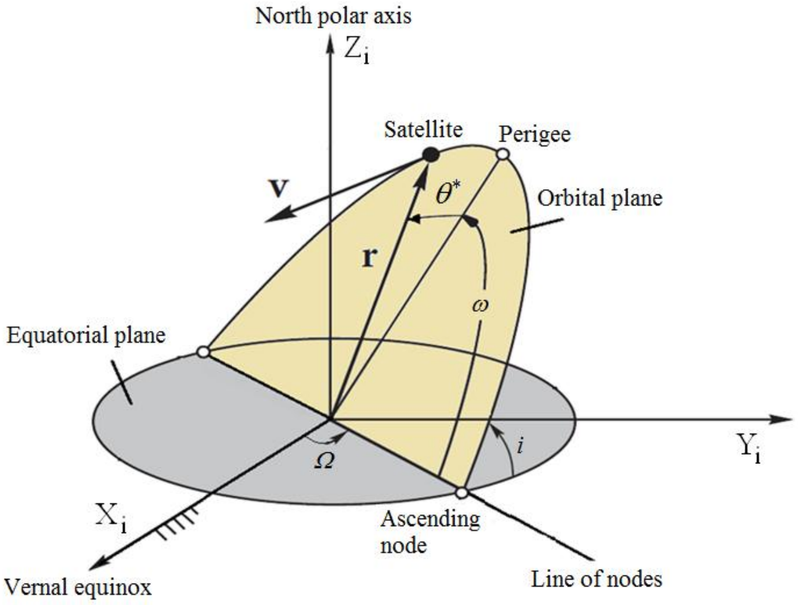


Figure A.2 Orbital elements (i, Ω, ω, θ^*)
(adapted from the book of Curtis, 2005)

3. ***Inclination* (*i*)** is the angle between the orbital plane and the equatorial plane and takes values in the range of 0-180°. It is equal to zero for equatorial orbits and 90° for polar orbits. Inclination can be determined by the following well known relation.

$$\cos i = \sin A_0 \cdot \cos \delta_0 \tag{A.5}$$

   In Eq. (A.5),
   A_0 is the inertial launch azimuth (angle measured clockwise from north),
   δ_0 is the geocentric latitude of the launch site.

   Eq. (A.5) can be satisfied if and only if $i \geq \delta_0$. Therefore, achievable inclinations are constrained by the latitude of the launch site. It is impossible to launch directly into an inclination lower than the launch site's latitude without orbit plane transfer which requires a large amount of velocity change (ΔV). Therefore, launch sites at or near the Equator are highly desirable, because launch into any inclination from them are possible.

4. ***Right ascension of the ascending node – RAAN* (Ω)** is the angle between the vernal equinox direction and the ascending node. The *ascending node* is the point where the orbit crosses the equatorial plane when the satellite passes from the southern hemisphere to the northern hemisphere, and the *vernal equinox* is the vector pointing the fixed stars in the constellation of Aries (Figure A.2). On the first day of spring, line joining the Earth's center and the Sun's center points in vernal equinox direction.

   RAAN takes values in the range of 0-360° and is undefined when i = 0° or i = 180°. Specified RAAN can be achieved by choosing an appropriate injection time depending upon the longitude of the injection point as Maini & Agraval (2011) emphasized. Tewari (2007) derived a useful relationship to determine RAAN utilizing the spherical trigonometry.

$$\Omega = \lambda - \arcsin\left(\frac{\tan \delta}{\tan i}\right) \tag{A.6}$$

   In Eq. (A.6),
   λ is the geodetic longitude,
   δ is the geocentric latitude.

5. ***Argument of perigee* (*ω*)** defines the orientation of the ellipse (in which direction it is flattened compared to a circle) in the orbital plane, as an angle measured from the ascending node to the perigee. Argument of perigee takes values in the range of 0-360° and is undefined when i = 0° or i = 180° or e = 0. As Maini & Agraval (2011) noted that ω can directly be calculated from the following relation when the injection point is the same as the perigee point.

$$\omega = \arcsin\left(\frac{\sin\delta}{\sin i}\right) \tag{A.7}$$

6. ***True anomaly* (*θ^**)** defines the position of the satellite along the orbit at a specific time and it is the angle between the perigee and the satellite location. True anomaly takes values in the range of 0-360° and is undefined when $e = 0$. At perigee and apogee points, $\theta^* = 0°$ and $\theta^* = 180°$, respectively.

A real orbit (and its elements) changes over time due to gravitational perturbations by other objects and the effects of the relativity. A Keplerian orbit is an idealized mathematical approximation and true anomaly is assumed as the only orbital element that changes with time. As mentioned above, some of the orbital elements become undefined for certain special cases (Table A.1).

Table A.1 Special cases of orbits

Type of orbit	**Undefined parameter**
circular ($e = 0$)	ω & θ^*: undefined
equatorial ($i = 0$)	Ω & ω: undefined
circular and equatorial ($e = 0$ & $i = 0$)	Ω, ω & θ^*: undefined

A.2 Orbital Position, Velocity and Period

As depicted in Figure A.1, **r** is the satellite's position vector measured from the Earth's center and **v** is the satellite's velocity vector. *Radial distance from the Earth's center to the satellite* (r) can be calculated by using the *orbit equation* defining the shape of the orbit in polar coordinates.

$$r = \frac{a\cdot\left(1-e^2\right)}{1+e\cdot\cos\theta^*} \tag{A.8}$$

As illustrated in Figure A.1, *flight path angle* (γ) is the angle measured from local horizontal to the velocity vector (**v**). Local horizontal is a line perpendicular to the position vector (**r**). At apogee and perigee points, $\gamma = 0°$ since **v** is perpendicular to **r**. Flight path angle (γ) can be calculated as

$$\gamma = \arctan\left(\frac{e\cdot\sin\theta^*}{1+e\cdot\cos\theta^*}\right) \tag{A.9}$$

Velocity of the satellite at any point in an orbit can be calculated using the following *vis viva equation* derived from the orbital energy conservation equation.

$$V_{orbit} = \sqrt{\mu\left(\frac{2}{r} - \frac{1}{a}\right)} \tag{A.10}$$

In Eq. (A.10),

V_{orbit} is the orbital velocity at radial distance r,

μ is the Earth's gravitational parameter, i.e. μ = 398600.4 km^3/s^2,

r is the radial distance from the Earth's center to the satellite,

a is the semimajor axis.

Satellite reaches its maximum speed at perigee ($r = r_{min}$) and minimum speed at apogee ($r = r_{max}$) as Eq. (A.10) implies. For circular orbits r is constant and equals to a, thus Eq. (A.10) reduces to a constant speed of circular orbit.

$$V_{circ} = \sqrt{\frac{\mu}{r}} \tag{A.11}$$

Time taken for a satellite to make one complete revolution along an elliptical orbit is called *orbital period* (τ) and can be calculated by

$$\tau = \frac{2\pi a^{3/2}}{\sqrt{\mu}} \tag{A.12}$$

The time elapsed from perigee point along an elliptical orbit can be calculated as a fraction of orbital period using Kepler's second law. Or, more generally, it can be expressed for an arbitrary θ^* as derived by Addison (2009).

$$t\left(\theta^*\right) = \begin{cases} \sqrt{\dfrac{a^3}{\mu}}\left\{2\arctan\left(\sqrt{\dfrac{1-e}{1+e}}\tan\left(\dfrac{\theta^*}{2}\right)\right) - \sqrt{1-e^2}\,\dfrac{e\sin\theta^*}{1+e\cos\theta^*}\right\}, & 0 \le \theta^* < \pi \\ \sqrt{\dfrac{a^3}{\mu}}\left\{2\pi + 2\arctan\left(\sqrt{\dfrac{1-e}{1+e}}\tan\left(\dfrac{\theta^*}{2}\right)\right) - \sqrt{1-e^2}\,\dfrac{e\sin\theta^*}{1+e\cos\theta^*}\right\}, & \pi \le \theta^* < 2\pi \end{cases} \tag{A.13}$$

A.3 Orbit Types

The orbital elements of a particular satellite depend upon its intended application and mission. Thus, given a particular mission, orbital elements are specified so as to satisfy the mission requirements such as coverage area, ground track, surface orientation relative to the sun, photographic resolution, satellite visibility, rendezvous considerations, etc. Table A.2 summarizes the characteristic orbital elements for specific types of satellite orbits.

Table A.2 Types of satellite orbits

Classification	**Orbit type**	**Characteristic parameter**
Inclination	Equatorial	$i = 0°$
	Polar	$i = 90°$
	Prograde	$0° < i < 90°$
	Retrograde	$90° < i < 180°$
Eccentricity	Elliptical	$0 < e < 1$
	Circular	$e = 0$
	Molniya (highly elliptical)	$e = 0.7$; $i = 63.4°$; $\omega = 270°$; $a = 26571$ km
Distance from Earth	Low Earth orbit (LEO)	$h < 1500$ km
	Medium Earth orbit (MEO)	10000 km $< h <$ 20000 km
	Geostationary orbit (GEO)	$h = 35786$ km; $e = 0$; $i = 0°$
Particular mission	Geostationary transfer orbit (GTO)	$h_a = 35786$ km; $e \neq 0$; $i \neq 0°$
	Sunsynchronous orbit (SSO)	$i \approx 98°$, $e = 0$, LEO

In general, satellites are placed in one of the three types of orbits around the Earth: geostationary (GEO), polar (PEO) or sunsynchronous (SSO) whose characteristic orbital elements are listed in Table A.2. The type of orbit determines the sensor type, its altitude and the instantaneous field of view.

GEO is an orbit matching the Earth's sidereal rotation period. These orbits enable a satellite to always view the same area on the Earth; therefore they are commonly used by the communication and meteorological satellites. Polar orbiting satellites pass over the Earth poles and the equator at a different longitude at each revolution. A PEO satellite eventually sees every part of the Earth's surface, which is highly desirable for remote sensing applications.

A nearly polar orbit whose altitude is such that the satellite will always pass over a location at given latitude at the same local solar time is called SSO. These orbits are useful for Earth observation satellites which can always view a part of the Earth under the same illumination conditions ensuring the lengths of shadows are same on every pass. Imaging satellites having active sensors such as radar or lidar may not require sun light, but they do rely on solar energy as a power source.

A.4 Orbit Insertion

Once the orbital altitude (injection point) is reached, launch vehicle is accelerated to attain the necessary orbital velocity. If the satellite is released when the velocity is equal to V_{circ} at that altitude calculated from Eq. (A.11), then a circular orbit is achieved. If the velocity is higher than V_{circ}, then a variety of elliptical orbits with the injection point as perigee can be generated. If the velocity is lower than V_{circ}, then elliptical orbits with the injection point as apogee are possible. In this case, if the perigee exists virtually below the surface of the Earth, satellite enters a ballistic flight and falls back to the Earth.

Usually the intended injection point is chosen to be the perigee point but provided that is high enough to prevent atmospheric drag from causing rapid decay. Orbits of having injection points as apogee are therefore usually short lived and the result of an injection error or motor failure (Turner, 2009).

It is generally more energy efficient to first insert the satellite into a transfer/parking orbit and after a coasting (unpowered flight) phase where it gains altitude, an additional thrust maneuver is performed to attain the final orbit as illustrated in Figure A.3.

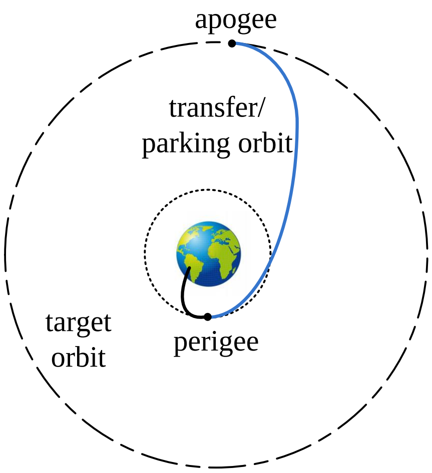


Figure A.3 Orbital injection

LEO satellites are almost always directly injected into their mission orbits by the launch vehicles. However, for higher orbits such as GEO, launch vehicles usually deliver the payload to GTO and then onboard propulsion of the satellite is employed to reach GEO. For circular LEO insertions, restartable upper stage motors of launch vehicles are utilized to circularize the elliptical transfer orbit after the coasting period. This operation is called the *circularization burn*, which can be modeled as a single impulse maneuver performed at the apogee point. The required delta-V (ΔV_{CB}) for this maneuvering can be calculated easily from the difference between the velocity of the desired circular orbit and the transfer orbit's velocity at apogee.

$$\Delta V_{CB} = \sqrt{\frac{\mu}{r_{circ}}} - \sqrt{\mu\left(\frac{2}{r_{a,TO}} - \frac{1}{a_{TO}}\right)} \tag{A.14}$$

In Eq. (A.14),

ΔV_{CB} is the required delta-V for circularization burn,

r_{circ} is the radius of desired circular orbit,

$r_{a,TO}$ is the apogee radius of transfer orbit,
a_{TO} is the semimajor axis length of transfer orbit.

The amount of propellant required for circularization burn can then easily be calculated by using the famous rocket equation given in Eq. (3.1) and can be expressed as

$$m_{p,CB} = m_0 \cdot \left(1 - e^{\left(-\frac{\Delta V_{CB}}{I_{sp} \cdot g_0} \right)} \right) \tag{A.15}$$

APPENDIX B

COORDINATE SYSTEMS & TRANSFORMATIONS

Three dimensional modeling of launch vehicle ascent trajectories from the ground to the Earth orbit requires definition of a number of reference frames and their associated coordinate systems and coordinate transformations. Reference frames and coordinate systems are two distinct entities according to Zipfel (2007). *Reference frames* are models of physical references, whereas *coordinate systems* establish the association with the Euclidean space. In line with this definition, coordinate systems are used to express the motion of a system within a particular reference frame.

This appendix presents the various reference frames, coordinate systems and coordinate transformations used in this thesis. The descriptions of the reference frames and the definitions of trajectory variables are originated from the Recommended Practice for Atmospheric and Space Flight Vehicle Coordinate Systems outlined by American National Standards Institute (ANSI, 1992).

B.1 Coordinate Systems

A reference frame is defined by an origin and a set of three orthogonal axes with specified positive directions. A coordinate system consists of a reference frame and a number of vectorial variables (or parameters) as needed. Particular frames of reference used in this thesis to specify the position variables are inertial, relative and orbital reference frames located on the Earth, whereas a local frame attached to the vehicle in its current location is used to specify the velocity variables. As being vectors, position and velocity can be resolved into a variety of components. Most common are rectangular and spherical coordinates.

B.1.1 Rectangular ECI Coordinate System

Earth-centered-inertial (ECI) frame denoted by $OX_iY_iZ_i$ has the origin at the center of the Earth. The X_i axis is the continuation of the line from the center of the Earth through the vernal equinox, the Z_i axis is coincident with the mean spin axis of the Earth positive to the North Pole, and the Y_i axis completes the right hand triad (Figure B.1). The axes of ECI frame moves very slightly over time due to precession of vernal equinox and assumed as fixed with respect to the distant stars. Trajectory variables in the rectangular ECI coordinate system are three rectangular components of position and velocity vectors. They are expressed as $\{r_x, r_y, r_z, v_x, v_y, v_z\}$ as illustrated in Figure B.1.

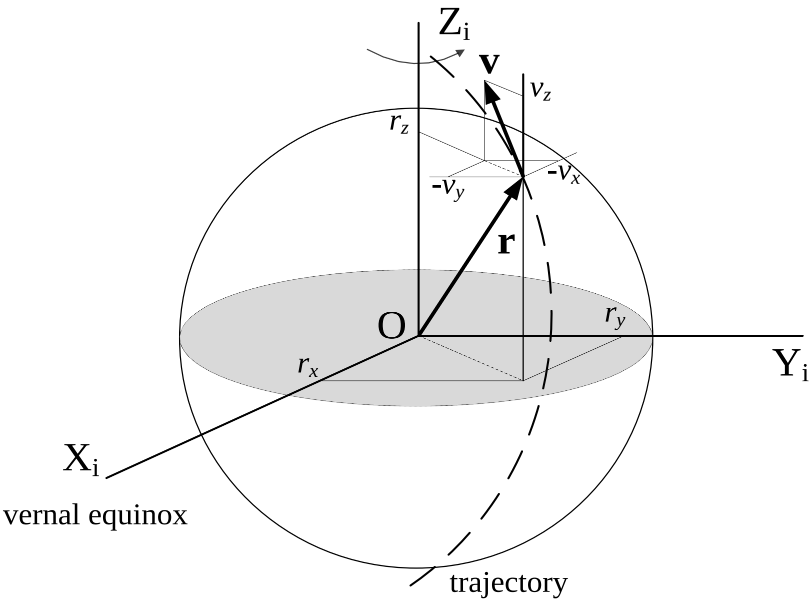


Figure B.1 Rectangular ECI coordinate system

A special form of ECI frame, which is more convenient to initialize launch vehicle inertial measurement units, is called the *go-inertial* ECI frame. In this frame, X_i axis is coincident with the Greenwich meridian at time very near or at lift-off.

B.1.2 Spherical ECI Coordinate System

Apart from ECI frame defined in Section B.1.1, vehicle carried East-North-Up (ENU) frame denoted by $ox_Ly_Lz_L$ is required to define the trajectory in spherical coordinates which is more convenient for physical interpretation (Figure B.2). ENU frame has the origin at the vehicle's center of gravity. The x_L axis is along the position vector **r** pointing upwards, the y_L and z_L axes point towards East and North, respectively while they span the local horizontal frame.

Trajectory variables in the spherical ECI coordinate system are one radial and two angular components of position and velocity vectors. They are expressed in terms of the following six quantities $\{r, \lambda_i, \delta, v_i, A, \gamma_i\}$ as illustrated in Figure B.2.

r : geocentric radius,
λ_i : right ascension (inertial longitude),
δ : declination (geocentric latitude),
v_i : inertial speed,
A : velocity azimuth,
γ_i : inertial flight path angle.

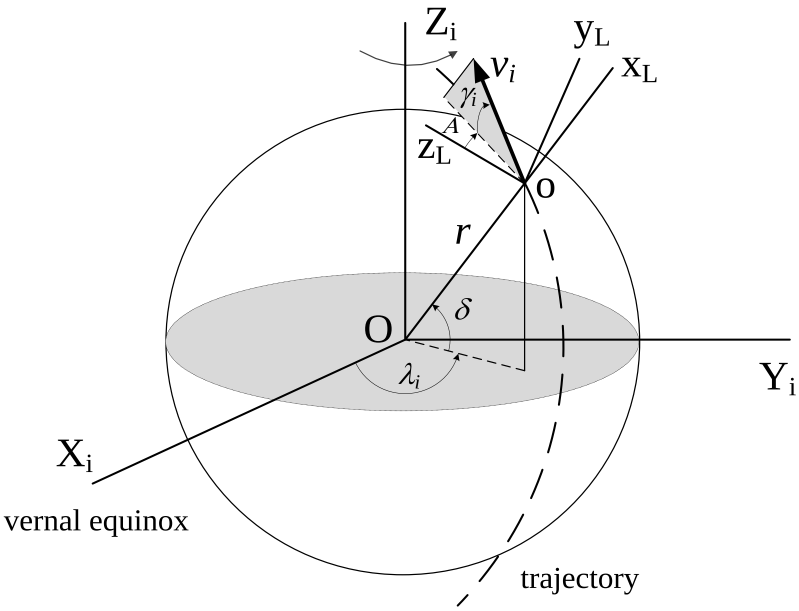


Figure B.2 Spherical ECI coordinate system

In Figure B.2, *right ascension* i.e. *inertial longitude* (λ_i) is the angle made by the projection of **r** on OX_iY_i with the vernal equinox direction OX_i, while *declination* i.e. *geocentric latitude* (δ) is the angle between the position vector **r** with the equatorial plane OX_iY_i. Longitude is divided into ±180° with the positive direction starting at the Greenwich meridian in an easterly direction (counterclockwise). Whereas, latitude is measured above the equatorial plane positive to the North from 0 to 90° and negative to the South from 0 to -90°.

In order to specify the velocity vector **v**, vehicle carried ENU frame $ox_Ly_Lz_L$ is employed. The horizontal projection of **v** makes an angle A with North oz_L called the *velocity azimuth* and defined positive toward the East. The velocity vector **v** makes an angle γ_i called the *inertial flight path angle* with the local horizontal plane oy_Lz_L and defined positive above the plane.

B.1.3 Spherical ECEF Coordinate System

Earth-centered Earth-fixed (ECEF) frame denoted by OXYZ has the origin at the center of the Earth. The X axis is the continuation of the line from the center of the Earth through the intersection of the Greenwich meridian and the equator, the Z axis is the mean spin axis of the Earth, positive to the North, and the Y axis completes the right hand triad. (Figure B.3). The axes of ECEF frame are fixed relative to the Earth and rotate with the Earth at a constant rate as time progresses.

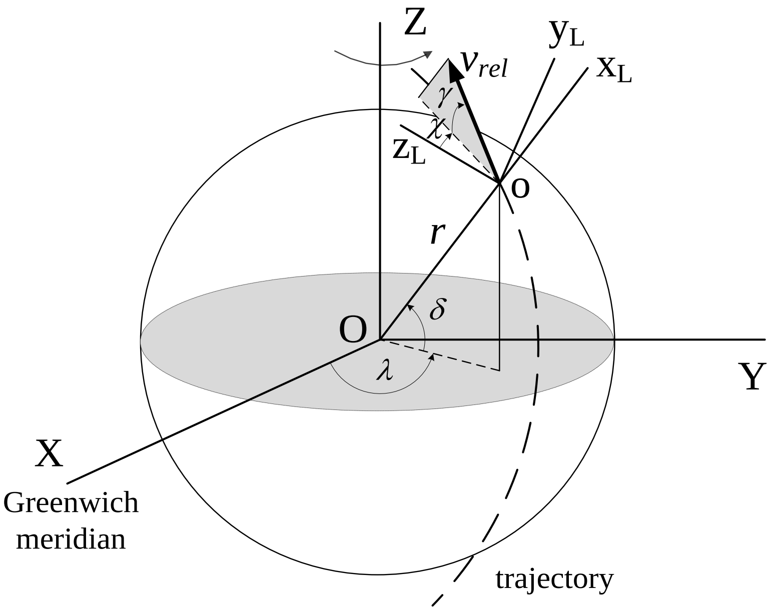


Figure B.3 Spherical ECEF coordinate system

Spherical ECEF coordinate system is similar to the spherical ECI coordinate system, the only difference is that it is based on ECEF frame rather than ECI frame as it is evident from its name. A set of spherical coordinates in the ECEF frame are $\{r, \lambda, \delta, v_{rel}, \chi, \gamma\}$ as illustrated in Figure B.3.

r : geocentric radius,
λ : geodetic longitude,
δ : geocentric latitude,
v_{rel} : relative speed with respect to Earth,
χ : heading angle,
γ : flight path angle.

B.1.4 Geographic Coordinate System

Geographic coordinate system is to define the position of objects relative to the oblate Earth. In the preceding sections, Earth was assumed as sphere although it is an oblate spheroid which can be represented by a reference ellipsoid as illustrated exaggeratedly in Figure B.4.

Geographical coordinates are geodetic latitude (ϕ), geodetic longitude (λ) and geodetic altitude (h). The relationships between geodetic and geocentric coordinates are given in Section B.2.3. Definition of the longitude does not change due to oblateness, since it is defined in the equatorial frame.

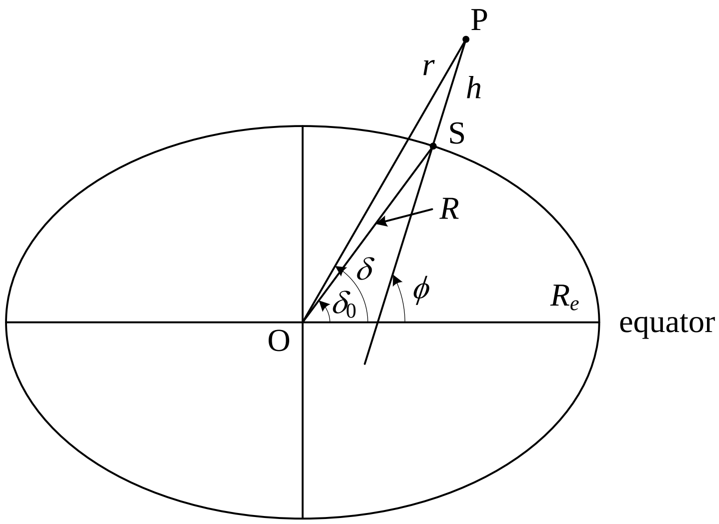


Figure B.4 Geographic coordinate system

B.1.5 Orbital Coordinate System

Orbital frame also known as perifocal frame denoted by OPWQ has the origin at the center of the Earth which is at the same time at one of the focuses of the elliptical orbit. The P axis is directed towards the perigee of the orbit and W axis is perpendicular to orbital plane i.e. direction of angular momentum vector **h**, which is the radius vector crossed the velocity vector. Q axis completes the right hand triad being in the orbital plane (Figure B.5).

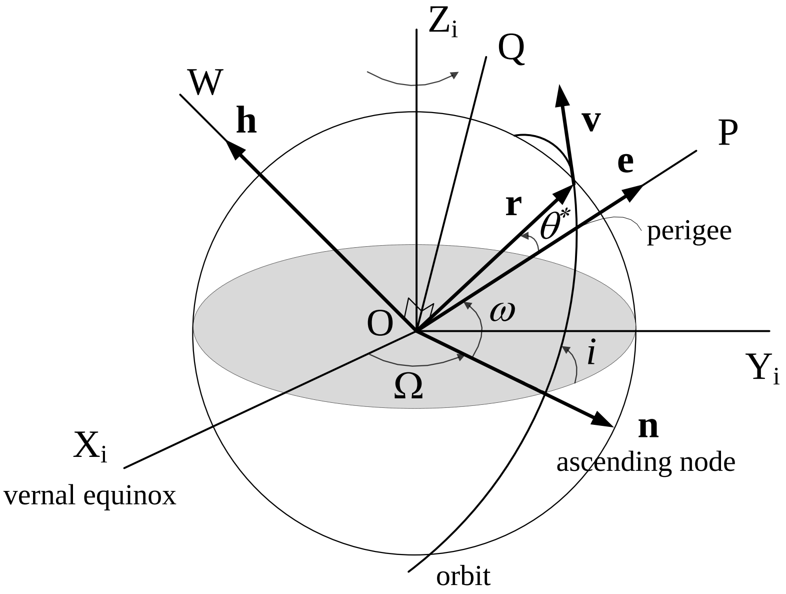


Figure B.5 Orbital frame

The orbital elements {a, e, i, Ω, ω, θ^*} defined in Section A.1 form the perifocal coordinate system, which is commonly used to express orbital trajectories.

B.1.6 Summary

Different reference frames and coordinate systems used for describing atmospheric and orbital trajectories were introduced in the preceding sections. Table B.1 summarizes these reference frames, while Table B.2 and Table B.3 present the trajectory variables and their ranges, respectively.

Table B.1 Summary of reference frames

Reference frame	**Origin**	**Axes**	**1st direction**	**3rd direction**
ECI	Earth's center	$X_iY_iZ_i$	Vernal equinox	Earth's spin axis
ECEF	Earth's center	XYZ	Greenwich	Earth's spin axis
Vehicle carried ENU	Vehicle's center of gravity	$x_Ly_Lz_L$	Position vector	North
Orbital	Earth's center	PQW	Perigee point	Satellite motion

Table B.2 Summary of trajectory variables

Coordinate system	**1**	**2**	**3**	**4**	**5**	**6**
Rectangular ECI	r_x	r_y	r_z	v_x	v_y	v_z
Spherical ECI	r	λ_i	δ	v_i	A	γ_i
Spherical ECEF	r	λ	δ	v_{rel}	χ	γ
Geographic	h	λ	ϕ	-	-	-
Orbital	a	e	i	Ω	ω	θ

Table B.3 Range of trajectory variables

Variables	**Range**
r_x, r_y, r_z, v_x, v_y, v_z	$-\infty, +\infty$
r, v_i, v_{rel}, h	≥ 0
λ_i, λ, A, χ, Ω, ω, θ^*	0°, 360°
δ, ϕ, γ_i, γ	-90°, 90°
i	0°, 180°
a	> 0 (for circular or elliptical orbits)
e	$0 \leq e < 1$ (for circular or elliptical orbits)

B.2 Coordinate Transformations

Trajectory variables defined in any one of the coordinate systems summarized in Table B.2 can easily be converted to other coordinate systems via appropriate coordinate transformations. Coordinate transformations used in this thesis are formulated based on the derivations done by Chobotov (2002) and Tewari (2007).

B.2.1 Rectangular ECI to Spherical ECI and vice versa

Rectangular to spherical: Radial components of position and velocity vectors can easily be calculated from the rectangular components by taking their magnitude.

$$r = \sqrt{r_x^2 + r_y^2 + r_z^2} \tag{B.1}$$

$$v_i = \sqrt{v_x^2 + v_y^2 + v_z^2} \tag{B.2}$$

The angular components of position vector can be obtained using the basic trigonometric identities.

$$\lambda_i = \arctan 2\left(r_y, r_x\right) \tag{B.3}$$

$$\delta = \arcsin\left(\frac{r_z}{r}\right) \tag{B.4}$$

Since the spherical coordinates of velocity are expressed in local horizon frame, the orientation of this frame relative to the ECI frame should first be defined. As seen from Figure B.2, rotations of ECI frame first about the Z_i axis by λ_i and then about the Y_i axis by $-\delta$ will cause the X_i axis become collinear with the radius vector. One can write the transformation matrix from ECI frame to local horizon frame using the elementary rotation matrices as follows.

$$\mathbf{R_{i2L}} = \mathbf{R_2}\left(-\delta\right)\cdot\mathbf{R_3}\left(\lambda_i\right) = \begin{bmatrix} \cos\delta\cos\lambda_i & \cos\delta\sin\lambda_i & \sin\delta \\ -\sin\lambda_i & \cos\lambda_i & 0 \\ -\sin\delta\cos\lambda_i & -\sin\delta\sin\lambda_i & \cos\delta \end{bmatrix} \tag{B.5}$$

Velocity vector in local horizon frame with respect to ECI frame can now be expressed as

$$\begin{Bmatrix} v_{L,x} \\ v_{L,y} \\ v_{L,z} \end{Bmatrix} = \mathbf{R_{i2L}} \cdot \begin{Bmatrix} v_x \\ v_y \\ v_z \end{Bmatrix} \tag{B.6}$$

The angular components of velocity can be derived by using Eqs. (B.5) and (B.6) as follows.

$$\gamma_i = \arcsin \frac{v_{L,x}}{v_i} \tag{B.7}$$

$$\mathrm{A} = \arctan 2\left(v_{L,y}, v_{L,z}\right) \tag{B.8}$$

Spherical to rectangular: From Figure B.2, one can immediately write

$$r_x = r\cos\delta\cos\lambda_i \tag{B.9}$$

$$r_y = r\cos\delta\sin\lambda_i \tag{B.10}$$

$$r_z = r\sin\delta \tag{B.11}$$

Velocity vector can be resolved in local horizon frame as

$$v_{L,x} = v_i \sin\gamma_i \tag{B.12}$$

$$v_{L,y} = v_i \cos\gamma_i \sin\mathrm{A} \tag{B.13}$$

$$v_{L,z} = v_i \cos\gamma_i \cos\mathrm{A} \tag{B.14}$$

Transformation matrix from local horizon frame to ECI frame will be the inverse of the transformation matrix defined in Eq. (B.5).

$$\mathbf{R_{L2i}} = \mathbf{R_{i2L}}^{\mathbf{-1}} = \mathbf{R_3}\left(-\lambda_i\right)\cdot\mathbf{R_2}\left(\delta\right) = \begin{bmatrix} \cos\lambda_i\cos\delta & -\sin\lambda_i & -\cos\lambda_i\sin\delta \\ \sin\lambda_i\cos\delta & \cos\lambda_i & -\sin\lambda_i\sin\delta \\ \sin\delta & 0 & \cos\delta \end{bmatrix} \tag{B.15}$$

$$\begin{Bmatrix} v_x \\ v_y \\ v_z \end{Bmatrix} = \mathbf{R_{L2i}} \cdot \begin{Bmatrix} v_{L,x} \\ v_{L,y} \\ v_{L,z} \end{Bmatrix} \tag{B.16}$$

Matlab scripts written for transformations between rectangular and spherical coordinates in ECI frame are given in Appendix C.1.1.

B.2.2 Spherical ECI to Spherical ECEF and vice versa

A set of spherical coordinates in ECI frame is $\{r, \lambda_i, \delta, v_i, A, \gamma_i\}$, while it is $\{r, \lambda, \delta, v_{rel}, \chi, \gamma\}$ in ECEF frame. Figure B.6 illustrates the difference between ECI and ECEF frames which share the common OZ axis. As it is clearly seen, definitions of geocentric radius r and latitude δ are same both for ECI and ECEF frames.

The relationship between the inertial velocity and the velocity relative to rotating frame is given by

$$\mathbf{v_i} = \mathbf{v_{rel}} + \boldsymbol{\omega}_\mathbf{e} \times \mathbf{r} \tag{B.17}$$

In Eq. (B.17),

$\boldsymbol{\omega_e} = \omega_e\mathbf{k}$ is the angular velocity of the Earth, i.e. $\omega_e = 7.29211585 \times 10^{-5}$ rad/s

$\mathbf{r} = r\mathbf{i}$.

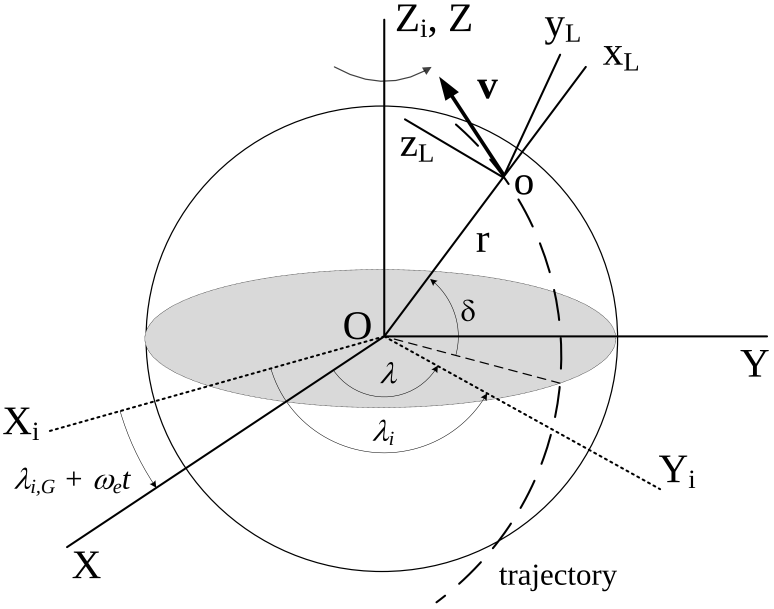


Figure B.6 Difference in longitude definitions in ECI and ECEF frames

Expressing the inertial and relative velocity vectors as

$$\mathbf{v_i} = v_i\left(\sin\gamma_i\mathbf{i} + \cos\gamma_i\sin\mathrm{A}\mathbf{j} + \cos\gamma_i\cos\mathrm{A}\mathbf{k}\right) \tag{B.18}$$

$$\mathbf{v_{rel}} = v_{rel}\left(\sin\gamma\mathbf{i} + \cos\gamma\sin\chi\mathbf{j} + \cos\gamma\cos\chi\mathbf{k}\right) \tag{B.19}$$

And substituting them into Eq. (B.17) to yield

$$\begin{aligned} & v_i\left(\sin\gamma_i\mathbf{i} + \cos\gamma_i\sin\mathrm{A}\mathbf{j} + \cos\gamma_i\cos\mathrm{A}\mathbf{k}\right) \\ & = v_{rel}\left(\sin\gamma\mathbf{i} + \cos\gamma\sin\chi\mathbf{j} + \cos\gamma\cos\chi\mathbf{k}\right) + \omega_e r\cos\delta\mathbf{j} \end{aligned} \tag{B.20}$$

Equating the respective vector components on both sides of Eq. (B.20) gives

$$v_i\sin\gamma_i = v_{rel}\sin\gamma \tag{B.21}$$

$$v_i\cos\gamma_i\sin\mathrm{A} = v_{rel}\cos\gamma\sin\chi + \omega_e r\cos\delta \tag{B.22}$$

$$v_i\cos\gamma_i\cos\mathrm{A} = v_{rel}\cos\gamma\cos\chi \tag{B.23}$$

<u>ECI to ECEF:</u> Dividing Eq. (B.22) by Eq. (B.23), one can find

$$\chi = \arctan\left(\tan \mathrm{A} - \frac{\omega_e r \cos\delta}{v_i \cos\gamma_i \cos \mathrm{A}} \right) \tag{B.24}$$

Dividing Eq. (B.21) by Eq. (B.23), one can find

$$\gamma = \arctan\left(\tan\gamma_i \frac{\cos\chi}{\cos \mathrm{A}} \right) \tag{B.25}$$

Finally, from Eq. (B.21)

$$v_{rel} = v_i \frac{\sin\gamma_i}{\sin\gamma} \tag{B.26}$$

Thus, the explicit relationships between the inertial and relative velocity components are obtained.

The geodetic longitude (λ) can be calculated at a given time t from the current right ascension (λ_i) and the right ascension of the zero longitude line OX (Greenwich meridian on Earth) ($\lambda_{i,G}$) known at some previous time t = 0 as follows.

$$\lambda = \lambda_i - \left(\lambda_{i,G} + \omega_e t \right) \tag{B.27}$$

In Eq. (B.27), $\lambda_{i,G} + \omega_e t$ refers to the *sidereal time* which is the time scale that is based on the Earth's rate of rotation measured relative to the fixed stars. $\lambda_{i,G}$ is usually available from periodically published astronomical (or ephemeris) charts such as the American Ephemeris and National Almanac for the right ascension of the Greenwich meridian. Or alternatively, $\lambda_{i,G}$ is assumed as 0 at time of lift-off and relative longitude values are calculated for different time instants of the trajectory for simple evaluations independent of time.

<u>ECEF to ECI:</u> Rearranging Eqs. (B.24) - (B.27), spherical coordinates in ECI frame can be obtained from spherical coordinates in ECEF frame as follows.

$$\mathrm{A} = \arctan\left(\tan\chi + \frac{\omega_e r \cos\delta}{v_{rel} \cos\gamma \cos\chi} \right) \tag{B.28}$$

$$\gamma_i = \arctan\left(\tan\gamma \frac{\cos \mathrm{A}}{\cos\chi} \right) \tag{B.29}$$

$$v_i = v_{rel} \frac{\sin\gamma}{\sin\gamma_i} \tag{B.30}$$

$$\lambda_i = \lambda + \left(\lambda_{i,G} + \omega_e t \right) \tag{B.31}$$

Matlab scripts written for transformations between ECI and ECEF frames in spherical coordinates are given in Appendix C.1.2.

B.2.3 Geocentric to Geodetic and vice versa

Figure B.4 shows the relationships of the positions and angular measurements for the triangle represented by the vectors *R*, *h* and *r*, bounded by points O, S and P. It is easy to obtain the *geodetic latitude* (ϕ) and the *Earth radius* (*R*) for a given geocentric latitude (δ_0) using spherical trigonometric relations.

$$\tan\left(\phi\right)=\frac{\tan\left(\delta_0\right)}{\left(1-e_e^{\,2}\right)} \tag{B.32}$$

$$R=\frac{R_e\sqrt{1-e_e^{\,2}}}{\sqrt{1-e_e^{\,2}\cos^2\left(\delta_0\right)}} \tag{B.33}$$

In Eqs. (B.32) and (B.33),
R_e is the equatorial radius of the Earth,
e_e is the first eccentricity of the reference ellipsoid for Earth.

According to WGS-84 Earth's model defined in the report of NIMA (2000), values for these physical constants are

R_e = 6378.137 km
e_e = 0.08181919084

The *geodetic altitude* (*h*) can be calculated from the law of cosines considering the spherical triangle OSP in Figure B.4.

$$h^2=r^2-R^2+2Rh\cos(\phi-\delta_0) \tag{B.34}$$

Finally, by applying the law of sines to the triangle OSP, it can be shown that the *geocentric latitude* or the *declination angle* (δ) of point P relative to equatorial plane is

$$\delta=\phi-\arcsin\left[\frac{R}{r}\sin(\phi-\delta_0)\right] \tag{B.35}$$

Moreover, the *geodetic ground distance* (*d*) or so called *downrange* can be computed from the known values of geodetic latitude and longitude through iterative formulas developed by Vincenty (1975). Using these formulas, arc length of the great circle arcs connecting pairs of points on the surface of the reference ellipsoid and the azimuth, which is the angle at which the arc crosses the meridian containing the first point, can be calculated.

User written and built-in Matlab functions used for transformations between geocentric and geodetic coordinates in ECEF frame are given in Appendix C.1.3.

B.2.4 Orbital to Rectangular ECI and vice versa

Orbital to rectangular: From Figure B.5, $X_iY_iZ_i$ axes become PQW axes by three successive rotations. First is about the Z_i axis by Ω, second is about the X_i axis by *i*, the last is about the Z_i axis by *ω*. Transformation matrix from inertial to orbital frame can be obtained using the elementary rotation matrices as follows.

$$\mathbf{R}_{\mathbf{i2oe}} = \mathbf{R}_3(\omega)\cdot\mathbf{R}_1(i)\cdot\mathbf{R}_3(\Omega) \tag{B.36}$$

Inverse transformation is required to convert orbital elements to inertial rectangular coordinates.

$$\mathbf{R}_{\mathbf{oe2i}} = \mathbf{R}_3(-\Omega)\cdot\mathbf{R}_1(-i)\cdot\mathbf{R}_3(-\omega) = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \tag{B.37}$$

Position and velocity vectors in orbital frame with respect to ECI frame can now be expressed as

$$\begin{Bmatrix} r_x \\ r_y \\ r_z \end{Bmatrix} = \mathbf{R}_{\mathbf{oe2i}}\cdot r \begin{Bmatrix} \cos\theta^* \\ \sin\theta^* \\ 0 \end{Bmatrix} \tag{B.38}$$

$$\begin{Bmatrix} v_x \\ v_y \\ v_z \end{Bmatrix} = \mathbf{R}_{\mathbf{oe2i}}\cdot \sqrt{\mu/K} \begin{Bmatrix} -\sin\theta^* \\ e+\cos\theta^* \\ 0 \end{Bmatrix} \tag{B.39}$$

In Eq. (B.37),

$$R_{11} = \cos\Omega\cos\omega - \sin\Omega\cos i\sin\omega \tag{B.40}$$

$$R_{12} = -\cos\Omega\sin\omega - \sin\Omega\cos i\cos\omega \tag{B.41}$$

$$R_{13} = \sin\Omega\sin i \tag{B.42}$$

$$R_{21} = \sin\Omega\cos\omega + \cos\Omega\cos i\sin\omega \tag{B.43}$$

$$R_{22} = -\sin\Omega\sin\omega + \cos\Omega\cos i\cos\omega \tag{B.44}$$

$$R_{23} = -\cos\Omega\sin i \tag{B.45}$$

$$R_{31} = \sin i\sin\omega \tag{B.46}$$

$$R_{32} = \sin i\cos\omega \tag{B.47}$$

$$R_{33} = \cos i \tag{B.48}$$

Rectangular to orbital: In this transformation, a set of inertial rectangular coordinates {r_x, r_y, r_z, v_x, v_y, v_z} is to be expressed as orbital elements {a, e, i, Ω, ω, θ^*}. Computational steps for this transformation are as follows.

$$r = \sqrt{r_x^2 + r_y^2 + r_z^2} \tag{B.49}$$

$$v = \sqrt{v_x^2 + v_y^2 + v_z^2} \tag{B.50}$$

The *semimajor axis* (a) can be obtained directly from the definition of orbital velocity in Eq. (A.10)

$$\frac{v^2}{\mu} = \frac{2}{r} - \frac{1}{a} \quad \rightarrow \quad a = \frac{r\mu}{2\mu - v_i^2 r} \tag{B.51}$$

The *eccentricity vector* (**e**) can be calculated by the known position and velocity vectors in inertial space (Figure B.5).

$$\mathbf{e} = \frac{1}{\mu}\left[\left(v^2 - \frac{\mu}{r}\right)\mathbf{r} - (\mathbf{r}\cdot\mathbf{v})\mathbf{v}\right] \tag{B.52}$$

whose magnitude is equal to *orbit eccentricity* (e)

$$e = |\mathbf{e}| \tag{B.53}$$

The *unit angular momentum vector* (**h**) is defined by

$$\mathbf{h} = \frac{\mathbf{r}\times\mathbf{v}}{|\mathbf{r}\times\mathbf{v}|} \tag{B.54}$$

From Figure B.5, *orbit inclination* (i) can be obtained from Eq.(B.54)

$$i = \arccos\left(\frac{h_3}{|\mathbf{h}|}\right) \tag{B.55}$$

The *ascending node vector* (**n**) can be calculated by

$$\mathbf{n} = \frac{\mathbf{K}\times\mathbf{h}}{|\mathbf{K}\times\mathbf{h}|} \tag{B.56}$$

From Figure B.5,

$$\mathbf{n} = \cos\Omega\mathbf{I} + \sin\Omega\mathbf{J} \tag{B.57}$$

From Eq. (B.57), *right ascension of the ascending node* (Ω) can be obtained without quadrant ambiguity.

$$\Omega = \arctan 2\left(n_2, n_1\right) \tag{B.58}$$

The *argument of perigee* (ω), which is the angle between the vectors of ascending node and the eccentricity, can be calculated by

$$\omega = \arccos\left(\frac{\mathbf{n}\cdot\mathbf{e}}{|\mathbf{n}|\cdot|\mathbf{e}|}\right) \tag{B.59}$$

Finally, the *true anomaly* (θ^*) is found by

$$\theta^* = \arccos\left(\frac{\mathbf{e}\cdot\mathbf{r}}{|\mathbf{e}|\cdot|\mathbf{r}|}\right) \tag{B.60}$$

Matlab scripts written for transformations between rectangular and perifocal coordinates are given in Appendix C.1.4.

APPENDIX C

MATLAB IMPLEMENTATION

Computer codes utilized in this thesis work were developed in Matlab® (Version R2012b, 8.0.0)

C.1 Coordinate Transformations

C.1.1 Rectangular ECI to Spherical ECI and vice versa

```
% Transformation from rectangular ECI to spherical ECI
% Ref. Tewari'2007, pp. 124.

function sph = rec2sph(rec)

N = size(rec,1);

% Input argument "rec" is a vector of size N x 6. Rows are inertial position
% and velocity vectors of size 6 in cartesian coordinates.
% [rx, ry, rz, vx, vy, vz] [m . . m/s . .]

% Output "sph" is a vector of size N x 6. Rows are inertial position and
% velocity vectors of size 6 in spherical coordinates. [radius, celestial
% longitude, geocentric latitude, celestial speed, velocity azimuth in
% celestial frame , flight path angle in celestial frame.
% [r alpha delta v_i A beta] [m deg deg m/s deg deg]

% CONSTANTS
% ========================================================================
r2d = 180/pi;                     % unit conversion from radian to degree

% USER INPUTS
% ========================================================================
% Position and velocity in rectangular celestial frame (ECI, cartesian)
r_ijk = rec(:,1:3);         % [m]
v_ijk = rec(:,4:6);         % [m/s]

% Transformation from ECI to spherical celestial frame
r = sqrt(sum(r_ijk.^2,2));                  % [m] geocentric radius

alpha = atan2(r_ijk(:,2),r_ijk(:,1))*r2d;   % [deg] celestial longitude
                                            % (right ascension)
                                            % within -180 and +180 deg

delta = asind(r_ijk(:,3)./r);               % [deg] celestial latitude
                                            % (declination)
                                            % = geocentric latitude
                                            % within -90 and +90 deg

v_i = sqrt(sum(v_ijk.^2,2));                % [m/s] speed in celestial fr.


% Transformation matrix from ECI to local horizon
sa = sind(alpha); ca = cosd(alpha);
```

```
sd = sind(delta); cd = cosd(delta);

C_r2lh1 = [cd.*ca cd.*sa sd];
C_r2lh2 = [-sa ca zeros(N,1)];
C_r2lh3 = [-sd.*ca -sd.*sa cd];

C_r2lh = zeros(3,3,N);
v_lh = zeros(N,3);

for i =1:N
    C_r2lh(:,:,i) = cat(1, C_r2lh1(i,:), C_r2lh2(i,:), C_r2lh3(i,:));
    % Velocity in local horizon frame wrt celestial frame
    v_lh(i,:) = (C_r2lh(:,:,i)*v_ijk(i,:)')';
end

A = atan2(v_lh(:,2),v_lh(:,3))*r2d;     % [deg] azimuth in celestial fr.
                                        % within 0 and 360 deg

% make all A positive
for i = 1:N
    if A(i) < 0
        A(i) = A(i) + 360;
    else
        A(i) = A(i);
    end
end

beta = asind(v_lh(:,1)./v_i);           % [deg] fpa in celestial fr.
                                        % within -90 and +90 deg

sph = [r alpha delta v_i A beta];
```

```
% Transformation from spherical to rectangular (cartesian) coordinates in
% Earth Centered Inertial frame Ref. Tewari, 2007, p.123.

function r = sph2rec(sp)

% Input argument "fp" is a row vector containing six elements (r lambda
% delta v chi gamma) in flight path coordinate system. Units must be in
% meters and degrees.
% sp = [11866107 269.8817 74.2467 4166 210.8382 -8.4259];

% USER INPUTS
% ===================================================================
r = sp(1);                  % [m] radial distance
lambda = sp(2);             % [deg] geocentric longitude
delta = sp(3);              % [deg] geocentric latitude
v = sp(4);                  % [m/s] speed wrt Earth
chi = sp(5);                % [deg] heading angle
gamma = sp(6);              % [deg] flight path angle

sl = sind(lambda); cl = cosd(lambda);
sd = sind(delta); cd = cosd(delta);
sc = sind(chi); cc = cosd(chi);
sg = sind(gamma); cg = cosd(gamma);

% position vector in ECEF frame
x = r*cd*cl;
y = r*cd*sl;
z = r*sd;
r_ijk = [x; y; z];

% Resolve velocity vector in local horizon frame
vx = v*sg;
vy = v*cg*sc;
vz = v*cg*cc;
```

```
v_lh = [vx; vy; vz];

% Transformation from local horizon frame to rectangular (ECI, cartesian)
C_lh2r = R3d(-lambda)*R2d(delta-90)*R2d(90);

v_ijk = C_lh2r*v_lh;

r = [r_ijk; v_ijk];
```

C.1.2 Spherical ECI to Spherical ECEF and vice versa

```
% Transformation from spherical ECI to spherical ECEF (velocity components
% only) (flight path coordinate system) Ref. Tewari'2007, pp. 128.

function fp = sph2fp(sph)

N = size(sph,1);

% Input argument "sph" is a vector of size N x 6. Rows are inertial
% position and velocity vectors of size 6 in spherical coordinates.
% [radius, geographic longitude, geocentric latitude, celestial speed,
% velocity azimuth in celestial frame , flight path angle in celestial
% frame. [r alpha delta v_i A beta] [m deg deg m/s deg deg]

% Output "fp" is a vector of size N x 3. Rows are relative velocity vectors
% of size 3 in spherical coordinates. [relative speed, velocity azimuth
% (heading angle), flight path angle]. [v_rel chi gamma] [m/s deg deg]

% CONSTANTS
% ==========================================================================
w_E = 7.29211585275553e-5;      % [rad/s] Earth's angular velocity

% USER INPUTS
% ==========================================================================
r = sph(:,1);                   % [m] geocentric radius
delta = sph(:,3);               % [deg] celestial latitude
                                % (declination)
                                % = geocentric latitude

v_i = sph(:,4);                 % [m/s] speed in celestial fr.

A = sph(:,5);                   % [deg] azimuth in celestial fr.

beta = sph(:,6);                % [deg] fpa in celestial fr.

% Transformation from spherical celestial frame (ECI) to flight path
% coordinate system (ECEF)

% heading angle in local horizon wrt ECEF (deg)
% within 0 and 360 deg
chi = atand(tand(A) - w_E*r.*cosd(delta)./(v_i.*cosd(beta).*cosd(A)));

% Fix the quadrant
for i = 1:N
    if sign(cosd(A(i))) == sign(cosd(chi(i)))
        chi(i) = chi(i);
    else
        chi(i) = chi(i) + 180;
    end
end

% make all chi positive
for i = 1:N
    if chi(i) < 0
        chi(i) = chi(i) + 360;
    else
        chi(i) = chi(i);
    end
```

```
end

gamma = atand(tand(beta).*cosd(chi)./cosd(A));   % [deg] fpa in ECEF
                                                  % within -90 and +90 deg

v_rel = v_i.*sind(beta)./sind(gamma);             % [m/s] velocity in ECEF

fp = [v_rel chi gamma];  % [deg]
```

```
% Transformation from celestial (inertial) longitude to relative longitude
% (Transformation from spherical ECI to spherical ECEF)
% Ref. Tewari'2007, pp. 129.

function lambda = i2rlong(alpha,t)

N = size(alpha,1);

% Input argument "alpha" is the inertial longitude with respect to
% constellation Aries (or Vernal equinox) in degrees. It is a vector of size
% N. "t" is a vector of size N containing time instants corresponding to
% alpha positions.

% Output argument "lambda" is the relative longitude with respect to
% Greenwich prime meridian in degrees. It is a vector of size N.

% CONSTANTS
% =========================================================================
w_E = 7.29211585275553e-5;      % [rad/s] Earth's angular velocity
r2d = 180/pi;                   % unit conversion from radian to degree

l_G = zeros(N,1);    % right ascension of the Greenwich meridian at time t0
lambda = alpha - (l_G + w_E*t*r2d );    % [deg] longitude in ECEF
                                        % = geodetic (geographic) longitude
                                        % within -180 and +180 deg

for i = 1:N
    if lambda(i) < -180
        lambda(i) = lambda(i) + 360;
    else if lambda(i) > 180
        lambda(i) = lambda(i) - 360;
        else
            lambda(i) = lambda(i);
        end
    end
end
```

```
% Transformation from spherical ECEF to spherical ECI coordinate system
% Ref. Tewari, 2007, p.126.

function spheci = sph_ecef2eci(fp,t)

N = size(fp,1);
l_G = zeros(N,1);    % right ascension of the Greenwich meridian at time t0

% Input argument "fp" is a row vector containing six elements (r lambda
% delta v chi gamma) in flight path coordinate system. Input argument "t"
% is the time instants corresponding to position and velocity vectors.
% Units must be in meters, degrees and seconds

% CONSTANTS
% =========================================================================
w_E = 7.29211585e-5;                % [rad/s] Earth's rotation rate

% USER INPUTS
% ==========================================================================
% coordinates in ECEF spherical frame
r = fp(:,1);                    % [m] radial distance
lambda = fp(:,2);               % [deg] geocentric longitude
delta = fp(:,3);                % [deg] geocentric latitude
v = fp(:,4);                    % [m/s] speed wrt Earth
chi = fp(:,5);                  % [deg] heading angle
gamma = fp(:,6);                % [deg] flight path angle

% velocity components in local horizon wrt ECI spherical frame
A = atand(tand(chi) + w_E*r.*cosd(delta)./(v.*cosd(gamma).*cosd(chi)));
beta = atand(tand(gamma).*cosd(A)./cosd(chi));
vi = v.*sind(gamma)./sind(beta);

alpha = lambda + l_G + wE*t;     % rad
spheci = [r alpha delta vi A beta];
```

C.1.3 Spherical ECEF to Geographic and vice versa

```
% Conversion from geocentric latitude to geodetic latitude
ecc = 0.08181919;    % Earth's eccentricity

% geodetic latitude
lat_geod = atan2(sind(lat_geoc), (1-ecc^2)*cosd(lat_geoc))*180/pi;

% Earth's radius wrt latitude
Re = 6378137;           % Earth's equatorial radius (semi-major axis)
R_geoc = Re*sqrt(1-ecc^2)./(1-ecc^2*cosd(lat_geoc).^2).^(1/2);
```

The following Matlab built-in functions can easily be utilized for conversions between geodetic to geocentric coordinates.

```
% Conversion of latitude from geodetic to geocentric and from geocentric to
geodetic
delta = geodetic2geocentricLat(ecc,phi)
phi = geocentric2geodeticLat(ecc,delta)
% (ecc is the first eccentricity of the reference ellipsoid)

% Conversion of rectangular position coordinates in ECEF frame to geodetic
coordinates (latitude, longitude and altitude).
lla = ecef2lla(r,'WGS84')

% Calculation of rectangular geodetic arclength (range) and azimuth from
specified latitude and longitude coordinates and assuming that the points lie
on the reference ellipsoid.
[range,az] = distance(lat0,long0,phi,lambda,referenceEllipsoid('wgs84'));

% 'WGS84' is the reference Earth model (oblate spheroid) supported by Matlab.
```

C.1.4 Orbital to Rectangular ECI and vice versa

```
% Transformation from perifocal (kepler elements) coordinate system to
% rectangular coordinate system Ref. Chobotov, 2002.

function [r v] = oe2rv(oe)

% Input argument "oe" is a row vector containing six elements
% Distances must be in m and angles must be in degrees

% CONSTANTS
% ==========================================================================
mu = 3.986004418e14;     % [m^3/s^2] Earth's gravitational parameter

% USER INPUTS
% ==========================================================================
% Classical orbital elements
a = oe(1);              % [m] semimajor axis
e = oe(2);              % [-] eccentricity
i = oe(3);              % [deg] inclination
Om = oe(4);             % [deg] right ascension of the ascending node
om = oe(5);             % [deg] argument of perigee
nu = oe(6);             % [deg] true anomaly

% CALCULATIONS
% ==========================================================================
% Transformation matrix
C = R3d(-Om)*R1d(-i)*R3d(-om);

p = a*(1-e^2);                           % [m] semilatus rectum
r = p/(1+e*cosd(nu));                    % [m] eqn. of conic section

% Position and velocity vector in rectangular coordinates
r = C*r*[cosd(nu); sind(nu); 0];                 % [m]
v = C*sqrt(mu/p)*[-sind(nu); (e+cosd(nu)); 0];   % [m/s]

% ROTATION MATRICES
% ==================
% Rotation matrix about 3rd axis
% input: x (deg)
function R = R3d(x)
R = [cosd(x)  sind(x) 0;
     -sind(x) cosd(x) 0;
     0        0       1];

% Rotation matrix about 1st axis
% input: x (deg)
function R = R1d(x)
R = [1  0        0;
     0  cosd(x)  sind(x);
     0  -sind(x) cosd(x)];
```

```
% Transformation from inertial coordinate system to perifocal coordinate
% system, REF: Tewari, 2007
% input argument 'rv' is a 1x6 row vector. [m]
% output argument 'oe' is a 1x6 row vector. [m, rad]

function oe = rv2oe(rv,mu)
ri = rv(1:3)';      % inertial position (column vector)
vi = rv(4:6)';      % inertial velocity (column vector)
K  = [0;0;1];
hv = cross(ri,vi);  % angular momentum vector
nv = cross(K,hv);   % ascending node vector
n  = sqrt(nv.'*nv);
h2 = (hv.'*hv);
v2 = (vi.'*vi);
r  = sqrt(ri.'*ri);
% eccentricity vector
ev = 1/mu*((v2-mu/r)*ri - (ri.'*vi)*vi);
p  = h2/mu;         % semilatus rectum
e  = sqrt(ev.'*ev);
a  = p/(1-e*e);     % semimajor axis
i  = acos(hv(3)/sqrt(h2));  % inclination
Om = acos(nv(1)/n);         % RAAN
if (nv(2)<0-eps),
  Om = 2*pi-Om;
end;
om = acos(nv.'*ev/n/e);     % argument of perigee
if (ev(3)<0),
  om = 2*pi-om;
end;
```

```
theta = acos(ev.'*ri/e/r);  % true anomaly
if (ri.'*vi<0),
  theta = 2*pi-theta;
end;
oe = [a e i Om om theta];   % orbital elements vector
```

C.2 Staging Optimization

```
clc
clear all

% USER INPUTS
% =========================================================================
% Mission definition
h_inj = 200;                    % [km] injection altitude
h_a = 200;                      % [km] apogee altitude
h_p = 200;                      % [km] perigee altitude
i = 28.5;                       % [deg] orbit inclination
mpl = 150;                      % [kg] payload mass

% Launch conditions
h_0 = 9.1;                      % [km] initial altitude
phi_0 = 28.5;                   % [deg] initial latitude
V_0 = 152.4;                    % [m/s] initial speed wrt ground

% Stage configuration
Isp = [305; 325; 350];          % [sec] specific impulses for each stage
epsilon = [0.15; 0.14; 0.12];   % [-] structural ratios for each stage
TW = 1.8;                       % [-] lift-off thrust-to-weight ratio

% CONSTANTS
% =========================================================================
GM = 398600.4418;    % [km^3/s^2] Earth's gravitational parameter
R = 6378.137;        % [km] Earth's mean radius
g0 = 9.80665;        % [m/s^2] gravitational acceleration @ Earth's surface
Omega = 7.292115e-5; % [rad/s] Earth's angular velocity
d2r = pi/180;        % unit conversion from degree to radian

% CALCULATIONS
% =========================================================================
% Target orbit parameters
r_inj = R + h_inj;                       % [km] injection radius
r_a = R + h_a;                           % [km] apogee radius
r_p = R + h_p;                           % [km] perigee radius
a = (r_a + r_p)/2;                       % [km] semimajor axis

% Launch parameters
r_0 = R + h_0;                           % [km] initial radius (@launch)
A = asin(cos(i*d2r)/cos(phi_0*d2r));     % [rad] launch azimuth
V_phi = 1e3*Omega*r_0*cos(phi_0*d2r);    % [m/s] Earth's speed wrt latitude

% Vehicle configuration
N = length(Isp);                % number of stages
C = g0*Isp;                     % [m/s] exhaust velocities for each stage

% DeltaV calculations
% ------------------------------------------------------------------------
% velocity required to keep payload in a specified orbit [m/s]
V_orbit = 1e3*sqrt(GM*(2/r_inj - 1/a));

% velocity gain due to Earth's rotation [m/s]
V_rot = V_orbit - sqrt((V_orbit*sin(A) - V_phi)^2 + (V_orbit*cos(A))^2);

% velocity loss due to gravity [m/s]
V_g = 81.006*TW^2 - 667.62*TW + 1505.4;

% velocity loss due to aerodynamic drag [m/s]
V_d = -32.692*TW^2 + 258.86*TW - 226.57;
```

```
% velocity loss due to steering and pressure change [m/s]
V_p = 100;

% initial absolute velocity [m/s]
V_i = V_rot + V_0;

% margin for unexpected disturbances and inaccuracies [m/s]
V_m = 100;

% mission deltaV [m/s]
Vmission = V_orbit + V_g + V_d + V_p - V_i + V_m;

% STAGING OPTIMIZATION (minimizing gross lift-off weight, m0 while
% satisfying mission deltaV)
% ======================================================================
p = Lagr_NR(Vmission, C, epsilon);  % Lagrange multiplier
if isnan (p) == 0                   % checking p is a defined number
L = (1 + p*C)./(p*C.*epsilon);      % [-] mass ratios for each stage
lambda = (L.*epsilon - 1)./(1 - L); % [-] payload ratios for each stage
m = zeros(N,1);
m_pl = mpl;
for k = N:-1:1                      % calculation of step masses of each
    m(k) = m_pl/lambda(k);          % stage beginning with stage N
    m_pl = m_pl + m(k);
end

%optimum stage configuration
dV = C.*log(L);                     % [m/s] optimal deltaV split of stages
m_s = epsilon.*m;                   % [kg] structural masses for each stage
m_p = m - m_s;                      % [kg] propellant masses for each stage
m01 = sum(m) + mpl;                 % [kg] gross lift-off mass
lambda_t = mpl/m01;                 % [-] overall payload fraction

% print results to the command window
fprintf('optimal deltaV split of stages:\n')
for j = 1:N
    fprintf('%d\t %g\n', j, dV(j))
end
fprintf('\noptimal stage masses:\n')
for j = 1:N
    fprintf('%d\t %g\n', j, m(j))
end
fprintf('\noptimal structural masses:\n')
for j = 1:N
    fprintf('%d\t %g\n', j, m_s(j))
end
fprintf('\noptimal propellant masses:\n')
for j = 1:N
    fprintf('%d\t %g\n', j, m_p(j))
end
fprintf('\ngross lift-off mass:\t')
fprintf('%g', m01)
fprintf('\ntotal payload ratio:\t')
fprintf('%g', lambda_t)
else
fprintf('Failed to find a solution.')
fprintf('\nPlease increase Isp, or decrease epsilon or payload mass.\n')
end


function p = Lagr_NR(Vmission, C, epsilon)
% Solution of Lagrange Multiplier, p by Newton-Raphson method.
p_0 = -1/(min(C.*(1 - epsilon)));
p = p_0;         % initial guess
tol = 10;        % error tolerance
V = Vmission - sum(C.*log((1 + p*C)./(p*C.*epsilon)));
while abs(V)>tol
V = Vmission - sum(C.*log((1 + p*C)./(p*C.*epsilon)));
```

```
dV = sum(C./p./(1 + p*C));
d = -V./dV;
p = p + d;
end
```

C.3 Trajectory Optimization

The following code belongs to the Delta III GTO mission presented in Section 4.2 and it is also included as an example in the software GPOPS-II developed by Patterson and Rao (2013b).

```
clc; clear all; close all
%% USER INPUTS
% Physical data
earthRadius         = 6378137;
gravParam           = 3.986012e14;
initialMass         = 301454;
earthRotRate        = 7.29211585e-5;
seaLevelDensity     = 1.225;
densityScaleHeight  = 7200;
g0                  = 9.80665;
eccentricity        = 0.08181919;

scales.length       = 1;
scales.speed        = 1;
scales.time         = 1;
scales.acceleration = 1;
scales.mass         = 1;
scales.force        = 1;
scales.area         = 1;
scales.volume       = 1;
scales.density      = 1;
scales.gravparam    = 1;

if 0,
scales.length       = earthRadius;
scales.speed        = sqrt(gravParam/scales.length);
scales.time         = scales.length/scales.speed;
scales.acceleration = scales.speed/scales.time;
scales.mass         = initialMass;
scales.force        = scales.mass*scales.acceleration;
scales.area         = scales.length^2;
scales.volume       = scales.area.*scales.length;
scales.density      = scales.mass/scales.volume;
scales.gravparam    = scales.acceleration*scales.length^2;
end

omega               = earthRotRate*scales.time;
auxdata.omegaMatrix = omega*[0 -1 0;1 0 0;0 0 0];
auxdata.mu          = gravParam/scales.gravparam;
auxdata.cd          = 0.5;
auxdata.sa          = 4*pi/scales.area;
auxdata.rho0        = seaLevelDensity/scales.density;
auxdata.H           = densityScaleHeight/scales.length;
auxdata.Re          = earthRadius/scales.length;
auxdata.g0          = g0/scales.acceleration;
auxdata.ecc         = eccentricity;

% Boundary conditions
delta0              = 28.5*pi/180;   % geodetic latitude (rad)
long0               = -80.6;         % geodetic longtude (lambda)
lambda0             = 0;             % ECI-ECEF coincident @lift-off ass.

% geocentric latitude of launch point (due to oblateness)
phi0 = atan2((1-auxdata.ecc^2)*sin(delta0), cos(delta0));
```

```
% Earth's geoncentric radius wrt geocentric latitude (due to oblateness)
R0 = auxdata.Re*sqrt(1-auxdata.ecc^2)/(1-auxdata.ecc^2*cos(phi0)^2)^(1/2);

% initial position vector (1x3)
r0 = R0*[cos(phi0)*cos(lambda0) cos(phi0)*sin(lambda0) sin(phi0)];

% initial velocity vector (1x3)
v0 = r0*auxdata.omegaMatrix.';

% initial position-velocity vector (1x6)
rv0 = [r0 v0];

af = 24361140/scales.length;
ef = 0.7308;
incf = 28.5*pi/180;
Omf = 269.8*pi/180;
omf = 130.5*pi/180;
nuguess = 0;

oef = [af ef incf Omf omf nuguess];
rvf = oe2rv(oef,auxdata.mu);

% Vehicle Properties
btSrb = 75.2/scales.time;
btFirst = 261/scales.time;
btSecond = 700/scales.time;

mTotSrb      = 19290/scales.mass;
mPropSrb     = 17010/scales.mass;
mDrySrb      = mTotSrb-mPropSrb;
mTotFirst    = 104380/scales.mass;
mPropFirst   = 95550/scales.mass;
mDryFirst    = mTotFirst-mPropFirst;
mTotSecond   = 19300/scales.mass;
mPropSecond  = 16820/scales.mass;
mDrySecond   = mTotSecond-mPropSecond;
mPayload     = 4164/scales.mass;
thrustSrb    = 628500/scales.force;
thrustFirst  = 1083100/scales.force;
thrustSecond = 110094/scales.force;
mdotSrb      = mPropSrb/btSrb;
ispSrb       = thrustSrb/(auxdata.g0*mdotSrb);
mdotFirst    = mPropFirst/btFirst;
ispFirst     = thrustFirst/(auxdata.g0*mdotFirst);
mdotSecond   = mPropSecond/btSecond;
ispSecond    = thrustSecond/(auxdata.g0*mdotSecond);

auxdata.thrustSrb    = thrustSrb;
auxdata.thrustFirst  = thrustFirst;
auxdata.thrustSecond = thrustSecond;
auxdata.ispSrb       = ispSrb;
auxdata.ispFirst     = ispFirst;
auxdata.ispSecond    = ispSecond;

%% TRAJECTORY OPTIMIZATION
% Time events
t0 = 0/scales.time;
t1 = 75.2/scales.time;
t2 = 150.4/scales.time;
t3 = 261/scales.time;
t4 = 961/scales.time;

% Mass change at time events (kg)
m10 = mPayload+mTotSecond+mTotFirst+9*mTotSrb;
m1f = m10-(6*mdotSrb+mdotFirst)*t1;
m20 = m1f-6*mDrySrb;
m2f = m20-(3*mdotSrb+mdotFirst)*(t2-t1);
m30 = m2f-3*mDrySrb;
m3f = m30-mdotFirst*(t3-t2);
```

```
m40 = m3f-mDryFirst;
m4f = mPayload;

rmin = -2*auxdata.Re;
rmax = -rmin;
vmin = -10000/scales.speed;
vmax = -vmin;
rvmin = [rmin*ones(1,3),vmin*ones(1,3)];
rvmax = [rmax*ones(1,3),vmax*ones(1,3)];

% Bounds and guesses in each phase of problem
iphase = 1;
bounds.phase(iphase).initialtime.lower = t0;
bounds.phase(iphase).initialtime.upper = t0;
bounds.phase(iphase).finaltime.lower = t1;
bounds.phase(iphase).finaltime.upper = t1;
bounds.phase(iphase).initialstate.lower = [rv0,m10];
bounds.phase(iphase).initialstate.upper = [rv0,m10];
bounds.phase(iphase).state.lower = [rvmin,m1f];
bounds.phase(iphase).state.upper = [rvmax,m10];
bounds.phase(iphase).finalstate.lower = [rvmin,m1f];
bounds.phase(iphase).finalstate.upper = [rvmax,m10];
bounds.phase(iphase).control.lower = -ones(1,3);
bounds.phase(iphase).control.upper = +ones(1,3);
bounds.phase(iphase).path.lower  = 1;
bounds.phase(iphase).path.upper  = 1;
guess.phase(iphase).time = [t0; t1];
guess.phase(iphase).state(:,1) = [rv0(1); rv0(1)];
guess.phase(iphase).state(:,2) = [rv0(2); rv0(2)];
guess.phase(iphase).state(:,3) = [rv0(3); rv0(3)];
guess.phase(iphase).state(:,4) = [rv0(4); rv0(4)];
guess.phase(iphase).state(:,5) = [rv0(5); rv0(5)];
guess.phase(iphase).state(:,6) = [rv0(6); rv0(6)];
guess.phase(iphase).state(:,7) = [m10; m1f];
guess.phase(iphase).control(:,1) = [0; 0];
guess.phase(iphase).control(:,2) = [1; 1];
guess.phase(iphase).control(:,3) = [0; 0];

iphase = 2;
bounds.phase(iphase).initialtime.lower = t1;
bounds.phase(iphase).initialtime.upper = t1;
bounds.phase(iphase).finaltime.lower = t2;
bounds.phase(iphase).finaltime.upper = t2;
bounds.phase(iphase).initialstate.lower = [rvmin,m2f];
bounds.phase(iphase).initialstate.upper = [rvmax,m20];
bounds.phase(iphase).state.lower = [rvmin,m2f];
bounds.phase(iphase).state.upper = [rvmax,m20];
bounds.phase(iphase).finalstate.lower = [rvmin,m2f];
bounds.phase(iphase).finalstate.upper = [rvmax,m20];
bounds.phase(iphase).control.lower = -ones(1,3);
bounds.phase(iphase).control.upper = +ones(1,3);
bounds.phase(iphase).path.lower  = 1;
bounds.phase(iphase).path.upper  = 1;
guess.phase(iphase).time = [t1; t2];
guess.phase(iphase).state(:,1) = [rv0(1); rv0(1)];
guess.phase(iphase).state(:,2) = [rv0(2); rv0(2)];
guess.phase(iphase).state(:,3) = [rv0(3); rv0(3)];
guess.phase(iphase).state(:,4) = [rv0(4); rv0(4)];
guess.phase(iphase).state(:,5) = [rv0(5); rv0(5)];
guess.phase(iphase).state(:,6) = [rv0(6); rv0(6)];
guess.phase(iphase).state(:,7) = [m10; m1f];
guess.phase(iphase).control(:,1) = [0; 0];
guess.phase(iphase).control(:,2) = [1; 1];
guess.phase(iphase).control(:,3) = [0; 0];

iphase = 3;
bounds.phase(iphase).initialtime.lower = t2;
bounds.phase(iphase).initialtime.upper = t2;
bounds.phase(iphase).finaltime.lower = t3;
bounds.phase(iphase).finaltime.upper = t3;
bounds.phase(iphase).initialstate.lower = [rvmin,m3f];
bounds.phase(iphase).initialstate.upper = [rvmax,m30];
bounds.phase(iphase).state.lower = [rvmin,m3f];
bounds.phase(iphase).state.upper = [rvmax,m30];
bounds.phase(iphase).finalstate.lower = [rvmin,m3f];
bounds.phase(iphase).finalstate.upper = [rvmax,m30];
bounds.phase(iphase).control.lower = -ones(1,3);
bounds.phase(iphase).control.upper = +ones(1,3);
bounds.phase(iphase).path.lower  = 1;
bounds.phase(iphase).path.upper  = 1;
guess.phase(iphase).time       = [t2; t3];
guess.phase(iphase).state(:,1) = [rvf(1); rvf(1)];
guess.phase(iphase).state(:,2) = [rvf(2); rvf(2)];
guess.phase(iphase).state(:,3) = [rvf(3); rvf(3)];
guess.phase(iphase).state(:,4) = [rvf(4); rvf(4)];
guess.phase(iphase).state(:,5) = [rvf(5); rvf(5)];
guess.phase(iphase).state(:,6) = [rvf(6); rvf(6)];
guess.phase(iphase).state(:,7) = [m30; m3f];
guess.phase(iphase).control(:,1) = [0; 0];
guess.phase(iphase).control(:,2) = [1; 1];
guess.phase(iphase).control(:,3) = [0; 0];

iphase = 4;
bounds.phase(iphase).initialtime.lower = t3;
bounds.phase(iphase).initialtime.upper = t3;
bounds.phase(iphase).finaltime.lower = t3;
bounds.phase(iphase).finaltime.upper = t4;
bounds.phase(iphase).initialstate.lower = [rvmin,m4f];
bounds.phase(iphase).initialstate.upper = [rvmax,m40];
bounds.phase(iphase).state.lower = [rvmin,m4f];
bounds.phase(iphase).state.upper = [rvmax,m40];
bounds.phase(iphase).finalstate.lower = [rvmin,m4f];
bounds.phase(iphase).finalstate.upper = [rvmax,m40];
bounds.phase(iphase).control.lower = -ones(1,3);
bounds.phase(iphase).control.upper = +ones(1,3);
bounds.phase(iphase).path.lower  = 1;
bounds.phase(iphase).path.upper  = 1;
guess.phase(iphase).time    = [t3; t4];
guess.phase(iphase).state(:,1) = [rvf(1); rvf(1)];
guess.phase(iphase).state(:,2) = [rvf(2); rvf(2)];
guess.phase(iphase).state(:,3) = [rvf(3); rvf(3)];
guess.phase(iphase).state(:,4) = [rvf(4); rvf(4)];
guess.phase(iphase).state(:,5) = [rvf(5); rvf(5)];
guess.phase(iphase).state(:,6) = [rvf(6); rvf(6)];
guess.phase(iphase).state(:,7) = [m40; m4f];
guess.phase(iphase).control(:,1) = [0; 0];
guess.phase(iphase).control(:,2) = [1; 1];
guess.phase(iphase).control(:,3) = [0; 0];

% Phase continuity constraints
bounds.eventgroup(1).lower = [zeros(1,6), -6*mDrySrb, 0];
bounds.eventgroup(1).upper = [zeros(1,6), -6*mDrySrb, 0];
bounds.eventgroup(2).lower = [zeros(1,6), -3*mDrySrb, 0];
bounds.eventgroup(2).upper = [zeros(1,6), -3*mDrySrb, 0];
bounds.eventgroup(3).lower = [zeros(1,6), -mDryFirst, 0];
bounds.eventgroup(3).upper = [zeros(1,6), -mDryFirst, 0];

% Terminal constraint (desired orbit parameters)
bounds.eventgroup(4).lower = oef(1:5);
bounds.eventgroup(4).upper = oef(1:5);

% Optimization problem setup
setup.name = 'ECivek-PhD_TrajOpt';
setup.functions.continuous = @continuous;
setup.functions.endpoint = @endpoint;
setup.bounds = bounds;
setup.guess = guess;
setup.auxdata = auxdata;
setup.derivatives.supplier = 'sparseFD';
setup.derivatives.derivativelevel = 'first';
setup.derivatives.dependencies = 'sparseNaN';
setup.scales.method = 'automatic-bounds';
setup.mesh.method = 'hp1';
setup.mesh.tolerance = 1e-6;
setup.mesh.maxiteration = 3;
% provide an initial mesh for each phase
meshphase(4).colpoints(10) = 0;
meshphase(4).fraction(10) = 0;
for i=1:4
  meshphase(i).colpoints = 4*ones(1,10);
  meshphase(i).fraction = 0.1*ones(1,10);
end
setup.mesh.phase = meshphase;
setup.nlp.solver = 'snopt';
setup.nlp.options.tolerance = 1e-7;
setup.method = 'RPMintegration';

% solve problem using GPOPS2
tic
output = gpops2(setup);
toc

%% SEPARATED STAGES
sol = output.result.solution;
% solve trajectories of separated stages
rv1s = sol.phase(1).state(end,1:6);
rv2s = sol.phase(2).state(end,1:6);
rv3s = sol.phase(3).state(end,1:6);
ms1 = 6*mDrySrb;
ms2 = 3*mDrySrb;
ms3 = mDryFirst;
x10s = [rv1s ms1];
x20s = [rv2s ms2];
x30s = [rv3s ms3];

options = odeset('Events', @descentStopCond);
[t1s, x1s, t1imp, x1imp] = ode45(@descent,[t1, 1000], x10s', options);
[t2s, x2s, t2imp, x2imp] = ode45(@descent,[t2, 1000], x20s', options);
[t3s, x3s, t3imp, x3imp] = ode45(@descent,[t3, 1000], x30s', options);

%% POSTPROCESSING & PLOTS
% extract data and endpoint values from solution
% --------------------------------------------------------------------------
% time
t = [sol.phase(1).time; sol.phase(2).time; sol.phase(3).time;
sol.phase(4).time];
Pt = [t1 t2 t3];
i1 = find(t == t1);
i2 = find(t == t2);
i3 = find(t == t3);
t4f = sol.phase(4).time(end);
i4 = find(t == t4f);

% states
x = [sol.phase(1).state; sol.phase(2).state; sol.phase(3).state;
sol.phase(4).state];

% position (m)
rv = x(:,1:3);
r1s = x1s(:,1:3); r2s = x2s(:,1:3); r3s = x3s(:,1:3);
Px = [x(i1(1),1) x(i2(1),1) x(i3(1),1)];
Py = [x(i1(1),2) x(i2(1),2) x(i3(1),2)];
Pz = [x(i1(1),3) x(i2(1),3) x(i3(1),3)];
figure
plot(t,x(:,1),'k',t,x(:,2),'b',t,x(:,3),'r',...
    Pt,Px,'kx',Pt,Py,'bx',Pt,Pz,'rx','markers',12,'LineWidth',2)
xlabel('time (s)'); ylabel('position (m)')
```

```
legend('x','y','z')
grid on

% velocity (m/s)
vv = x(:,4:6);
Pvx = [x(i1(1),4) x(i2(1),4) x(i3(1),4)];
Pvy = [x(i1(1),5) x(i2(1),5) x(i3(1),5)];
Pvz = [x(i1(1),6) x(i2(1),6) x(i3(1),6)];
figure
plot(t,x(:,4),'k',t,x(:,5),'b',t,x(:,6),'r',...
    Pt,Pvx,'kx',Pt,Pvy,'bx',Pt,Pvz,'rx','markers',12,'LineWidth',2)
xlabel('time (s)'); ylabel('inertial velocity (m/s)')
legend('v_x','v_y','v_z')
grid on

% mass (kg)
figure
plot (t,x(:,7),'k','LineWidth',2)
xlabel('time (s)'); ylabel('mass (kg)')
grid on

% controls
u = [sol.phase(1).control; sol.phase(2).control; sol.phase(3).control;
sol.phase(4).control];
Pu1 = [u(i1(1),1) u(i2(1),1) u(i3(1),1)];
Pu2 = [u(i1(1),2) u(i2(1),2) u(i3(1),2)];
Pu3 = [u(i1(1),3) u(i2(1),3) u(i3(1),3)];
figure
plot(t,u(:,1),'k',t,u(:,2),'b',t,u(:,3),'r',...
    Pt,Pu1,'kx',Pt,Pu2,'bx',Pt,Pu3,'rx','markers',12,'LineWidth',2)
xlabel('time (s)'); ylabel('thrust direction cosines')
legend('u_x','u_y','u_z')
grid on

% PLOTS
% data postprocessing, extract endpoint values for plotting purposes
% -------------------------------------------------------------------------
% Conversion from ECI cartesian to flight path coordinate system [m & deg]
fp = rv2fp(x(:,1:6),t);
fp1s = rv2fp(x1s(:,1:6),t1s);
fp2s = rv2fp(x2s(:,1:6),t2s);
fp3s = rv2fp(x3s(:,1:6),t3s);

% geocentric radius (m)
r = fp(:,1);

% altitude (m)
h = r - auxdata.Re;
Ph = [h(i1(1)) h(i2(1)) h(i3(1))]*1e-3;
h1s = fp1s(:,1) - auxdata.Re;
h2s = fp2s(:,1) - auxdata.Re;
h3s = fp3s(:,1) - auxdata.Re;
figure
plot(t,h*1e-3,'k',Pt,Ph,'kx',t1s,h1s*1e-3,'--k',...
    t2s,h2s*1e-3,'--k',t3s,h3s*1e-3,'--k','markers',12,'LineWidth',2)
xlabel('time (s)'); ylabel('altitude, h(t) (km)')
grid on

% inertial speed (m/s)
vi = sqrt(sum(vv.^2,2));
Pvi = [vi(i1(1)) vi(i2(1)) vi(i3(1))];

% relative speed (m/s)
vrel = fp(:,4);
Pvrel = [vrel(i1(1)) vrel(i2(1)) vrel(i3(1))];

figure
plot(t,vi,'k',t,vrel,'b',Pt,Pvi,'kx',Pt,Pvrel,'bx',...
    'markers',12,'LineWidth',2)
```

```
xlabel('time (s)'); ylabel('speed (m/s)')
legend('inertial speed','relative speed')
grid on

% heading angle (deg)
chi = fp(:,5);
Pchi = [chi(i1(1)) chi(i2(1)) chi(i3(1))];
figure
plot(t,chi,'k',Pt,Pchi,'kx','markers',12,'LineWidth',2)
xlabel('time (s)'); ylabel('heading angle, \chi(t)(deg)')
grid on

% relative flight path angle
gamma = fp(:,6);
Pgamma = [gamma(i1(1)) gamma(i2(1)) gamma(i3(1))];
figure
plot(t,gamma,'k',Pt,Pgamma,'kx','markers',12,'LineWidth',2)
xlabel('time (s)'); ylabel('flight path angle, \gamma(t) (deg)')
grid on

% latitude & longitude
dlong = fp(:,2);          % change in longitude
long = long0 + dlong;     % geographic longitude
Plong = [long(i1(1)) long(i2(1)) long(i3(1))];
delta = fp(:,3);          % geocentric latitude
Plat = [delta(i1(1)) delta(i2(1)) delta(i3(1))];
lat1s = fp1s(:,3); long1s = long0 + fp1s(:,2);
lat2s = fp2s(:,3); long2s = long0 + fp2s(:,2);
lat3s = fp3s(:,3); long3s = long0 + fp3s(:,2);

figure
plot(long,delta,'k',Plong,Plat,'kx',long1s,lat1s,'--k',...
    long2s,lat2s,'--k',long3s,lat3s,'--k','markers',12,'LineWidth',2)
xlabel('longitude (deg)'); ylabel('latitude (deg)')
grid on

% downrange (m)
ellipsoid = [auxdata.Re auxdata.ecc];
[range,az] = distance(delta0*180/pi,long0,delta,long, ellipsoid);
Prange = [range(i1(1)) range(i2(1)) range(i3(1))]*1e-3;
[drange1s,az1s] = distance(lat1s(1),long1s(1),lat1s,long1s, ellipsoid);
[drange2s,az2s] = distance(lat2s(1),long2s(1),lat2s,long2s, ellipsoid);
[drange3s,az3s] = distance(lat3s(1),long3s(1),lat3s,long3s, ellipsoid);
range1s = range(i1(1)) + drange1s;
range2s = range(i2(1)) + drange2s;
range3s = range(i3(1)) + drange3s;

figure
plot(range*1e-3,h*1e-3,'k',Prange,Ph,'kx',...
    range1s*1e-3,h1s*1e-3,'--k',range2s*1e-3,h2s*1e-3,'--k',...
    range3s*1e-3,h3s*1e-3,'--k','markers',12,'LineWidth',2)
xlabel('range (km)'); ylabel('altitude, h(t) (km)')
grid on

% dt
dt = diff(t);
dt(dt == 0) = [];  % prevent division by zero
% dv
dv = diff(vi);
dv(dv == 0) = [];

% acceleration (g)
a = dv./dt/auxdata.g0;
Pa = [a(i1(1)-1) a(i2(1)-2) a(i3(1)-3)];
ta = [t(1:i1(1)-1); t(i1(1)+1:i2(1)-1); t(i2(1)+1:i3(1)-1); t(i3(1)+1:i4-1)];
figure
plot(ta,a,'k',Pt,Pa,'kx','markers',12,'LineWidth',2)
xlabel('time (s)'); ylabel('acceleration, a(t) (g)');
grid on
```

```
% density (kg/m^3)
rho = auxdata.rho0*exp(-h/auxdata.H);
Prho = [rho(i1(1)) rho(i2(1)) rho(i3(1))];
figure
plot(t,rho,'k',Pt,Prho,'kx','markers',12,'LineWidth',2)
xlabel('time (s)'); ylabel('density, \rho(t) (kg/m^3)')
grid on;

% dynamic pressure (Pa)
q = 1/2*rho.*vrel.^2;
Pq = [q(i1(1)) q(i2(1)) q(i3(1))]*1e-3;
figure
plot(t,q*1e-3,'k',Pt,Pq,'kx','markers',12,'LineWidth',2)
xlabel('time (s)'); ylabel('dynamic pressure, q(t) (kPa)')
grid on

% aerodynamic heat flux (W/m^2)
Q = q.*vrel;
PQ = [Q(i1(1)) Q(i2(1)) Q(i3(1))]*1e-6;
figure
plot(t,Q*1e-6,'k',Pt,PQ,'kx','markers',12,'LineWidth',2)
xlabel('time (s)'); ylabel('aerodynamic heat flux, Q(t) (MW/m^2)')
grid on

% Thrust angle wrt the local horizon frame [deg]
Tdircos = [x(:,1:3), u];
Ta = rec2sph(Tdircos);
Ta1 = Ta(:,5);
Ta2 = Ta(:,6);

% Thrust direction angle - lateral [deg]
psi1 = Ta1 - chi;
% Thrust direction angle - vertical [deg]
psi2 = Ta2 - gamma;

Ppsi1 = [psi1(i1(1)) psi1(i2(1)) psi1(i3(1))];
Ppsi2 = [psi2(i1(1)) psi2(i2(1)) psi2(i3(1))];
figure
plot(t,psi1,'k',t,psi2,'b',Pt,Ppsi1,'kx',Pt,Ppsi2,'bx','markers',12,'LineWidth
',2)
xlabel('time (s)'); ylabel('thrust direction angle, \psi(t)(deg)')
legend('lateral','vertical')
grid on

% orbital elements at terminal point (m, deg)
rvf_result = x(end,1:6);
oef_result = rv2oe(rvf_result,auxdata.mu);
orbit = [oef_result(1:2) oef_result(3:6)*180/pi];

% payload mass (kg)
payload = -output.result.objective - mDrySecond;

% Write outputs to an excel file
header1 = {'t (sec)' 'rx (m)' 'ry (m)' 'rz (m)' 'vx (m/s)' 'vy (m/s)'...
    'vz (m/s)' 'mass (kg)' 'u1' 'u2' 'u3'  'r (m)' 'dlong (deg)'...
    'delta (deg)' 'vrel (m/s)' 'chi (deg)' 'gamma (deg)' 'alt. (m)'...
    'v_i(m/s)' 'long (deg)' 'range (m)' 'rho (kg/m3)' 'q (Pa)' 'Q (W/m2)'...
    'Ta1 (deg)' 'Ta2 (deg)' 'psi1 (deg)' 'psi2 (deg)' 'ta(s)' 'accel (g)'};
results1 = [t x u fp h vi long range rho q Q Ta1 Ta2 psi1 psi2];
results2 = [ta a];
xlswrite('results', header1, 'Sheet1')
xlswrite('results', results1, 'Sheet1', 'A2')
xlswrite('results', results2, 'Sheet1', 'AC2')

header2 = {'a (m)' 'e (-)' 'i (deg)' 'Om (deg)' 'om (deg)' 'theta (deg)'...
    'mpl (kg)'};
xlswrite('results', header2, 'Sheet1', 'AE1')
xlswrite('results', orbit, 'Sheet1', 'AE2')
xlswrite('results', payload, 'Sheet1', 'AK2')

header3 = {'t (sec)' 'rx (m)' 'ry (m)' 'rz (m)' 'vx (m/s)' 'vy (m/s)'...
    'vz (m/s)' 'mass (kg)' 'r (m)' 'dlong (deg)' 'delta (deg)'...
    'vrel (m/s)' 'chi (deg)' 'gamma (deg)' 'alt. (m)' 'long (deg)'...
    'range (m)'};
results1s = [t1s x1s fp1s h1s long1s range1s];
results2s = [t2s x2s fp2s h2s long2s range2s];
results3s = [t3s x3s fp3s h3s long3s range3s];
xlswrite('results', header3, 'Sheet2')
xlswrite('results', results1s, 'Sheet2', 'A2')
xlswrite('results', header3, 'Sheet3')
xlswrite('results', results2s, 'Sheet3', 'A2')
xlswrite('results', header3, 'Sheet4')
xlswrite('results', results3s, 'Sheet4', 'A2')
```

C.3.1 Dynamic Equations

```
function phaseout = continuous(input)
%---------------------%
% Dynamics in Phase 1 %
%---------------------%
t1 = input.phase(1).time;
x1 = input.phase(1).state;
u1 = input.phase(1).control;
r1 = x1(:,1:3);
v1 = x1(:,4:6);
m1 = x1(:,7);

rad1 = sqrt(sum(r1.^2,2));
rad1(rad1 < input.auxdata.Re) = input.auxdata.Re;
omegaMatrix = input.auxdata.omegaMatrix;
omegacrossr = r1*omegaMatrix.';
vrel1 = v1-omegacrossr;
speedrel1 = sqrt(sum(vrel1.^2,2));
h1 = rad1-input.auxdata.Re;
rho1 = input.auxdata.rho0*exp(-h1/input.auxdata.H);
bc1  = (rho1./(2*m1)).*input.auxdata.sa*input.auxdata.cd;
bcspeed1 = bc1.*speedrel1;
bcspeedmat1 = repmat(bcspeed1,1,3);
Drag1 = -bcspeedmat1.*vrel1;
muoverradcubed1 = input.auxdata.mu./rad1.^3;
muoverradcubedmat1 = repmat(muoverradcubed1,1,3);
grav1 = -muoverradcubedmat1.*r1;

TSrb1   = 6*input.auxdata.thrustSrb*ones(size(t1));
TFirst1 = input.auxdata.thrustFirst*ones(size(t1));
TTot1   = TSrb1+TFirst1;
m1dot1  = -TSrb1./(input.auxdata.g0*input.auxdata.ispSrb);
m2dot1  = -TFirst1./(input.auxdata.g0*input.auxdata.ispFirst);
mdot1   = m1dot1+m2dot1;

path1 = sum(u1.^2,2);
Toverm1 = TTot1./m1;
Tovermmat1 = repmat(Toverm1,1,3);
thrust1 = Tovermmat1.*u1;
rdot1 = v1;
vdot1 = thrust1+Drag1+grav1;
phaseout(1).dynamics = [rdot1 vdot1 mdot1];
phaseout(1).path = path1;

%---------------------%
% Dynamics in Phase 2 %
%---------------------%
t2 = input.phase(2).time;
x2 = input.phase(2).state;
u2 = input.phase(2).control;
r2 = x2(:,1:3);
```

```
v2 = x2(:,4:6);
m2 = x2(:,7);

rad2 = sqrt(sum(r2.^2,2));
rad2(rad2 < input.auxdata.Re) = input.auxdata.Re;
omegaMatrix = input.auxdata.omegaMatrix;
omegacrossr = r2*omegaMatrix.';
vrel2 = v2-omegacrossr;
speedrel2 = sqrt(sum(vrel2.^2,2));
h2 = rad2-input.auxdata.Re;
rho2 = input.auxdata.rho0*exp(-h2/input.auxdata.H);
bc2  = (rho2./(2*m2)).*input.auxdata.sa*input.auxdata.cd;
bcspeed2 = bc2.*speedrel2;
bcspeedmat2 = repmat(bcspeed2,1,3);
Drag2 = -bcspeedmat2.*vrel2;
muoverradcubed2 = input.auxdata.mu./rad2.^3;
muoverradcubedmat2 = repmat(muoverradcubed2,1,3);
grav2 = -muoverradcubedmat2.*r2;
TSrb2 = 3*input.auxdata.thrustSrb*ones(size(t2));
TFirst2 = input.auxdata.thrustFirst*ones(size(t2));
TTot2 = TSrb2+TFirst2;
m1dot2 = -TSrb2./(input.auxdata.g0*input.auxdata.ispSrb);
m2dot2 = -TFirst2./(input.auxdata.g0*input.auxdata.ispFirst);
mdot2 = m1dot2+m2dot2;
path2 = sum(u2.^2,2);
Toverm2 = TTot2./m2;
Tovermmat2 = repmat(Toverm2,1,3);
thrust2 = Tovermmat2.*u2;
rdot2 = v2;
vdot2 = thrust2+Drag2+grav2;
phaseout(2).dynamics = [rdot2 vdot2 mdot2];
phaseout(2).path = path2;

%---------------------%
% Dynamics in Phase 3 %
%---------------------%
t3 = input.phase(3).time;
x3 = input.phase(3).state;
u3 = input.phase(3).control;
r3 = x3(:,1:3);
v3 = x3(:,4:6);
m3 = x3(:,7);

rad3 = sqrt(sum(r3.^2,2));
rad3(rad3 < input.auxdata.Re) = input.auxdata.Re;
omegaMatrix = input.auxdata.omegaMatrix;
omegacrossr = r3*omegaMatrix.';
vrel3 = v3-omegacrossr;
speedrel3 = sqrt(sum(vrel3.^2,2));
h3 = rad3-input.auxdata.Re;
rho3 = input.auxdata.rho0*exp(-h3/input.auxdata.H);
bc3  = (rho3./(2*m3)).*input.auxdata.sa*input.auxdata.cd;
bcspeed3 = bc3.*speedrel3;
bcspeedmat3 = repmat(bcspeed3,1,3);
Drag3 = -bcspeedmat3.*vrel3;
muoverradcubed3 = input.auxdata.mu./rad3.^3;
muoverradcubedmat3 = repmat(muoverradcubed3,1,3);
grav3 = -muoverradcubedmat3.*r3;
TTot3 = input.auxdata.thrustFirst*ones(size(t3));
mdot3 = -TTot3./(input.auxdata.g0*input.auxdata.ispFirst);
path3 = sum(u3.^2,2);
Toverm3 = TTot3./m3;
Tovermmat3 = repmat(Toverm3,1,3);
thrust3 = Tovermmat3.*u3;
rdot3 = v3;
vdot3 = thrust3+Drag3+grav3;
phaseout(3).dynamics = [rdot3 vdot3 mdot3];
phaseout(3).path = path3;
```

```
%---------------------%
% Dynamics in Phase 4 %
%---------------------%
t4 = input.phase(4).time;
x4 = input.phase(4).state;
u4 = input.phase(4).control;
r4 = x4(:,1:3);
v4 = x4(:,4:6);
m4 = x4(:,7);
rad4 = sqrt(sum(r4.^2,2));
rad4(rad4 < input.auxdata.Re) = input.auxdata.Re;
omegacrossr = r4*input.auxdata.omegaMatrix.';
vrel4 = v4-omegacrossr;
speedrel4 = sqrt(sum(vrel4.^2,2));
h4 = rad4-input.auxdata.Re;
rho4 = input.auxdata.rho0*exp(-h4/input.auxdata.H);
bc4  = (rho4./(2*m4)).*input.auxdata.sa*input.auxdata.cd;
bcspeed4 = bc4.*speedrel4;
bcspeedmat4 = repmat(bcspeed4,1,3);
Drag4 = -bcspeedmat4.*vrel4;
muoverradcubed4 = input.auxdata.mu./rad4.^3;
muoverradcubedmat4 = repmat(muoverradcubed4,1,3);
grav4 = -muoverradcubedmat4.*r4;
TTot4 = input.auxdata.thrustSecond*ones(size(t4));
mdot4 = -TTot4/(input.auxdata.g0*input.auxdata.ispSecond);
path4 = sum(u4.^2,2);
Toverm4 = TTot4./m4;
Tovermmat4 = repmat(Toverm4,1,3);
thrust4 = Tovermmat4.*u4;
rdot4 = v4;
vdot4 = thrust4+Drag4+grav4;
phaseout(4).dynamics = [rdot4 vdot4 mdot4];
phaseout(4).path = path4;
```

C.3.2 Terminal Constraint and Objective Function

```
function output = event(input)
% Variables at Start and Terminus of Phase 1
t0{1} = input.phase(1).initialtime;
tf{1} = input.phase(1).finaltime;
x0{1} = input.phase(1).initialstate;
xf{1} = input.phase(1).finalstate;
% Variables at Start and Terminus of Phase 2
t0{2} = input.phase(2).initialtime;
tf{2} = input.phase(2).finaltime;
x0{2} = input.phase(2).initialstate;
xf{2} = input.phase(2).finalstate;
% Variables at Start and Terminus of Phase 3
t0{3} = input.phase(3).initialtime;
tf{3} = input.phase(3).finaltime;
x0{3} = input.phase(3).initialstate;
xf{3} = input.phase(3).finalstate;
% Variables at Start and Terminus of Phase 2
t0{4} = input.phase(4).initialtime;
tf{4} = input.phase(4).finaltime;
x0{4} = input.phase(4).initialstate;
xf{4} = input.phase(4).finalstate;

% Event Group 1:  Linkage Constraints Between Phases 1 and 2
output.eventgroup(1).event = [x0{2}(1:7)-xf{1}(1:7), t0{2}-tf{1}];
% Event Group 2:  Linkage Constraints Between Phases 2 and 3
output.eventgroup(2).event = [x0{3}(1:7)-xf{2}(1:7), t0{3}-tf{2}];
% Event Group 3:  Linkage Constraints Between Phases 3 and 4
output.eventgroup(3).event = [x0{4}(1:7)-xf{3}(1:7), t0{4}-tf{3}];
% Event Group 4:  Constraints on Terminal Orbit
orbitalElements = rv2oe(xf{4}(1:6),input.auxdata.mu);
output.eventgroup(4).event = orbitalElements(1:5);
output.objective = -xf{4}(7);
```

C.3.3 Descent Trajectories of Separated Stages

```
function deriv = descent(t,x)
% Descent trajectories of separated stages
omega = 7.29211585e-5;
omegaMatrix = omega*[0 -1 0;1 0 0;0 0 0];
Re = 6378137;
rho0 = 1.225;
H = 7200;
sa = 4*pi;
cd = 0.5;
mu = 3.986012e14;

r = x(1:3);          % position @separation instant
v = x(4:6);          % velocity @separation instant
m = x(7);            % mass of separated stage

rad = norm(r);
omegacrossr = omegaMatrix*r;
vrel = v - omegacrossr;
speedrel = norm(vrel);
h = rad - Re;
% calculation of drag
rho = rho0*exp(-h/H);
bc  = (rho/(2*m))*sa*cd;
bcspeed = bc*speedrel;
Drag = -bcspeed*vrel;
% calculation of gravity
muoverradcubed = mu/rad^3;
grav = -muoverradcubed*r;
% equations of motion
rdot = v;
vdot = grav+Drag;
mdot = 0;

deriv = [rdot; vdot; mdot];


function [value,isterminal,direction] = descentStopCond(t,x)
Re = 6378145;
value = norm(x(1:3)) - Re;  % when value = 0, an event is triggered
isterminal = 1;             % terminate after the first event
direction = 0;              % get all the zeros
```

APPENDIX D

SOLID PROPELLANT ROCKET MOTORS

Rocket motors are used to impart a desired velocity to a flight vehicle which requires high thrust in order to transport its payload. Solid propellant rocket motors (SRM) consists of the fuel and oxidizer in solid state and thus it can operate in all environmental conditions.

Solid propellant is generally cast directly into the motor case in a special geometric form that is called the propellant grain. The thrust time history of SRM is dependent on the grain geometry and is proportional to the instantaneous burning area. The most common grain shapes and the corresponding thrust profiles are given in Figure D.1. Different thrust time curves can be accomplished by an appropriate combination of particular grain shapes.

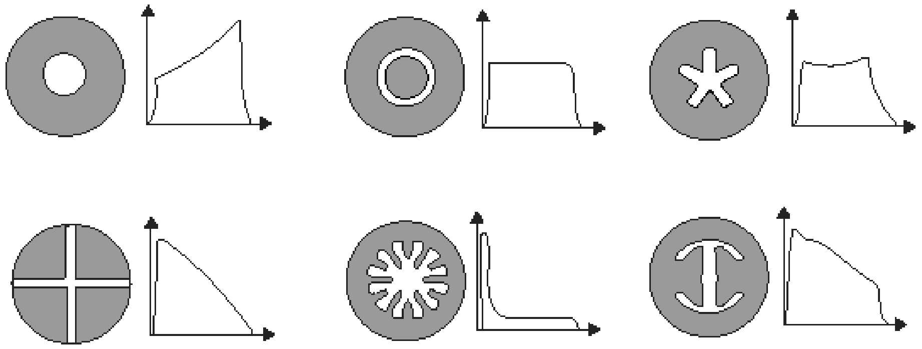


Figure D.1 SRM grain cross sections and influence on thrust profile
(adapted from www.allstar.fiu.edu/aero/rocket2.htm)

The burn time (operation time) of SRM is also determined by the grain shape, especially the web thickness of the grain. The burn time of SRMs used in space launch applications are illustrated in Figure D.1 according to a classification based on the relevant stage number. This chart is generated by using the available data of existing small class launch vehicles those are Vega (Europe), Taurus, Minotaur I, Minotaur IV, Athena I, Athena II (United States), Start (Russia), PSLV (India), M-V (Japan) and VLS (Brazil).

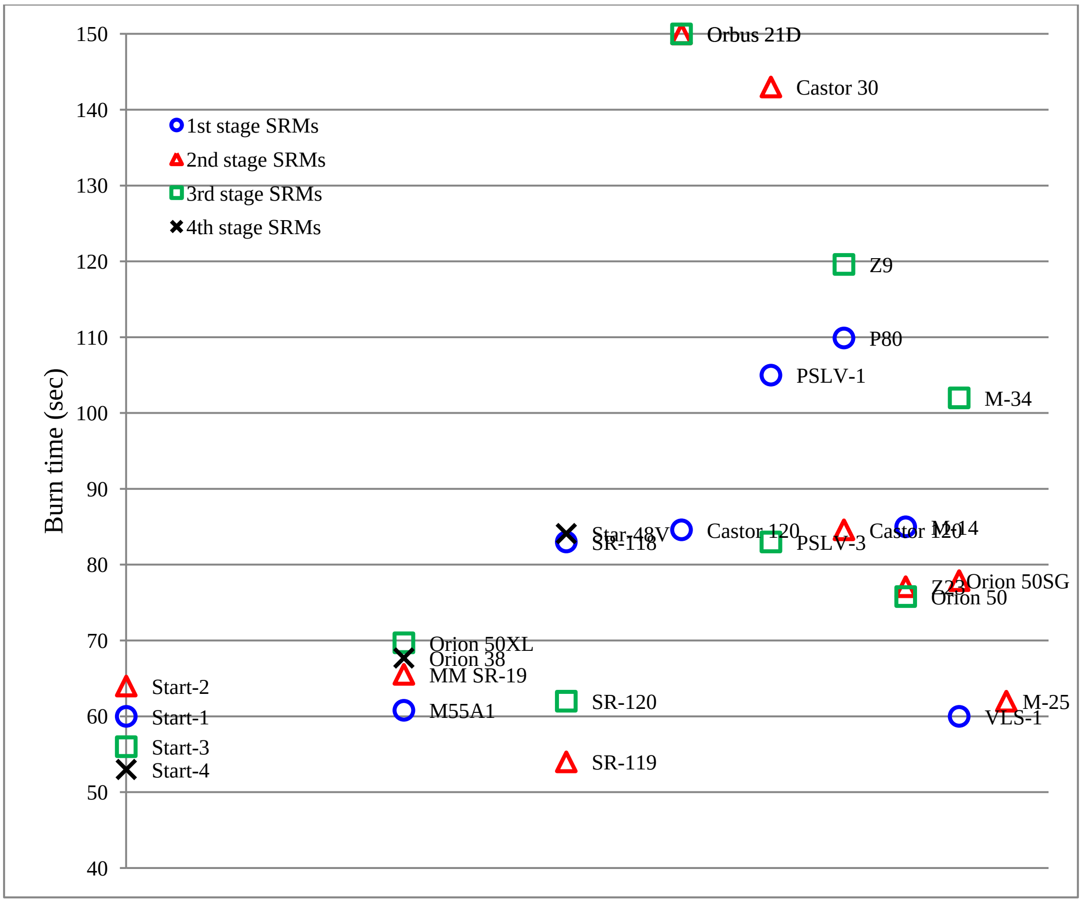


Figure D.1 Burn time of SRMs used in launch vehicles

From Figure D.1, it can be concluded that burn time of SRMs used in space launch applications vary between 60 sec – 120 sec. When the burn times of SRMs are examined according to the stage on which they are used, the same distribution applies without any distinctive mark.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Civek Coşkun, Ezgi
Nationality: Turkish
Date and Place of Birth: 23 July 1981, Ankara
Marital Status: Married
Phone: +90 555 217 85 81
Fax: +90 312 863 42 08
E-mail: ecivek@roketsan.com.tr / ezgicivek@gmail.com

EDUCATION

Degree	**Institution**	**Year of Graduation**
MS	METU Mechanical Engineering	2006
BS	METU Mechanical Engineering	2003
High School	Çankaya Milli Piyango Anatolian High School	1999

WORK EXPERIENCE

Year	**Place**	**Enrollment**
2007-Present	ROKETSAN Inc.	Senior Design Engineer
2006-2008	TÜBİTAK 1001, YAHTS Project	Scholarship Holder
2004-2007	National Productivity Centre of Turkey	Junior Expert
2004	Ankara Patent Bureau	Patent Engineer

PUBLICATIONS

1. Civek-Coşkun, E., & Özgören, K. (in progress). Space launch vehicle design with simultaneous optimization of thrust profile and trajectory. *Journal of Spacecraft and Rockets.*

2. Civek-Coşkun, E., & Özgören, K. (2013, June 12-14). A generalized staging optimization program for space launch vehicles. *Proceedings of 6th International Conference on Recent Advances in Space Technologies – RAST 2013*, (pp. 857-862). İstanbul, Turkey: IEEE.

3. Civek-Coşkun, E., Yaman, İ., & Seçkin, B. (2012, Haziran 20-22). Uydu fırlatma sistemleri için kavramsal tasarım metodolojisi. *6. Savunma Teknolojileri Kongresi Bildiriler Kitabı – SAVTEK 2012*. ODTÜ, Ankara.

4. Civek, E., Seçkin, B., & Tınaztepe, T. (2009, June 11-13). An industrialization model for satellite propulsion system manufacturing, assembly, integration & test. *Proceedings of 4th International Conference on Recent Advances in Space Technologies – RAST 2009*. İstanbul, Turkey: IEEE.

5. Civek, E., Tönük, E., Yavuzer, G., & Tümer, S.T. (2008, September 10-13). Comparison of kinematic results of Middle East Technical University custom made KISS and Ankara University commercial VICON gait analysis systems. Poster presented at *17th Annual Meeting of European Society of Movement Analysis for Adults and Children*. Antalya, Turkey. *Gait and Posture, 28*(S2), S99-S100.

6. Şen-Köktaş, N., Yalabık, N., Yavuzer, G., Atalay, V., & Civek, E. (2007, April 30-May 2). Combining decision trees and neural networks for grading knee osteoarthritis. Poster presented at *International Symposium on Health Informatics and Bioinformatics – HIBIT 2007*. Antalya, Turkey.

7. Civek, E. (2006). *Comparison of kinematic results between METU-KISS & Ankara University-Vicon gait analysis systems*. MSc. Thesis, Middle East Technical University, Mechanical Engineering, Ankara, Turkey.

8. Civek, E., Karakaya, Ş., & Belgin, Ö. (2005, November 23-24). Multi-criteria comparison of logistics outsourcing providers using fuzzy AHP. *Proceedings of 3rd International Logistics and Supply Chain Congress*. İstanbul, Turkey.

RESEARCH INTERESTS

System level design and trajectory optimization of space launch vehicles, flight dynamics and control, systems engineering, space propulsion, gait analysis.