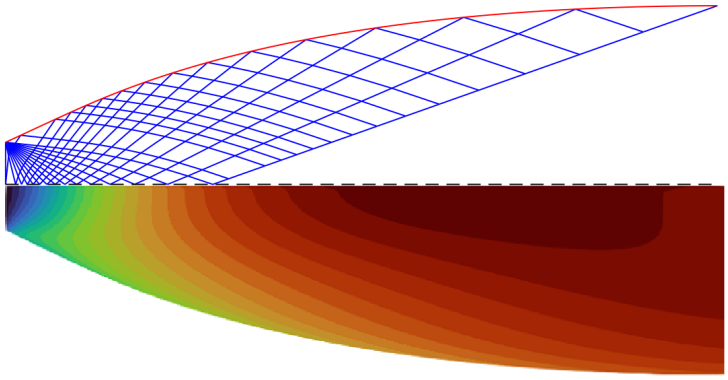
TÉCNICO LISBOA



On the design and optimization of rocket nozzles based on the method of characteristics

Tiago Bischoff Fernandes

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisors: Dr. Alain de Souza
Prof. Frederico José Prata Rente Reis Afonso

Examination Committee

Chairperson: Prof. Fernando José Parracho Lau
Supervisor: Dr. Alain de Souza
Member of the Committee: Prof. André Resende Rodrigues da Silva

28 November 2022

E PLURIBUS UNUM

"Out of many, One."

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

I want to express my deepest appreciation towards Dr. Frederico Afonso for his availability and constant helpfulness, welcoming every question with open arms. I am also grateful for Dr. Alain de Souza for providing me with the knowledge to implement during my dissertation. Thank you, Instituto Superior Técnico for these tough but enjoyable 5 years. A special acknowledgment goes to my parents, who worked hard to give me the best possible education and motivated me along the way. To my colleagues André Pereira, José Parreira, Tiago Martins e Tómas Cruz let us keep "Gang Gang" alive. Finally, I want to show my appreciation towards João Farinha, who despite constantly getting on my nerves always has my best interest at heart, and my girlfriend Mariana Malta Cruz, who always shows interest in my work and continuously supports me through tough times.

Resumo

A otimização do contorno aplicado a tubeiras de foguetões tem sido bem documentada desde os anos 50, crescendo em importância com o desenvolvimento do estudo da mecânica de fluidos computacional (MFC). No entanto, com o avanço tecnológico veio o aumento das exigências computacionais, que ainda não estão disponíveis.

O presente trabalho tem como objetivo desenvolver um método de baixa fidelidade capaz de otimizar tubeiras numa fase preliminar do desenvolvimento de um foguetão. Neste sentido, o Método das Características (MdC) bidimensional é implementado como a base para o desenvolvimento de um algoritmo de simulação de fluxo. A partir da automatização do desenho de várias geometrias de tubeira alcançado através da técnica de parametrização *Free Form Deformation*, é estabelecido um processo de otimização para maximizar o desempenho da tubeira.

De modo a verificar a fiabilidade do método desenvolvido, é executado um processo de otimização semelhante, utilizando as equações de Euler existentes no programa de código aberto SU^2, as quais são de uma fidelidade mais elevada do que o MdC. Para tal otimização é gerado um "modelo substituto" baseado nas simulações de Euler de forma a não só suprir alguns limitações do código SU^2 para tubeiras, mas também reduzir o tempo computacional.

É alcançada uma boa concordância entre os resultados dos dois métodos, havendo apenas um pequeno desalinhamento relativamente à largura ótima do contorno e ao coeficiente de impulso. Como tal o MdC provou ser uma ferramenta fiável para ser integrada na otimização preliminar da forma de tubeiras.

Palavras-chave: Otimização Contorno de Tubeiras, Método das Características, Técnica de Parametrização, Mecânica de Fluidos Computacional, Modelo Substituto

Abstract

Design optimization applied to rocket nozzles has been well-documented since the 1950s, gaining significant importance with the development of Computational Fluid Dynamics (CFD). However, with technological advancement comes an increase in computational requirements, which are not widely available.

The present work aims to develop a low-fidelity and computational-friendly method to conduct nozzle shape optimization at a preliminary design phase of a rocket nozzle. With this in mind, a two-dimensional Method of Characteristics (MoC) is implemented as the foundation for the development of a flow simulation algorithm. Automating the contouring of various nozzle geometries through the popular parameterization technique of Free Form Deformation, an optimization process is established to maximize the performance of the nozzle.

To verify the reliability of the proposed method, a similar optimization process is executed, established on high-fidelity CFD simulations using the open-source framework SU^2. An Euler solver is appointed instead of using RANS for the sake of similarity between both processes. This latter optimization process is established as a Surrogate-Based Optimization (SBO) not only to mitigate the SU^2 framework limitations in performing CFD-based shape optimization on nozzles but also as a way to reduce the computational power.

A good agreement between the results from both methods is achieved, displaying solely a small offset concerning the optimal contour width and the coefficient of thrust. Thus, proving the usefulness of developed aerodynamic shape optimization strategy based on the MoC for the preliminary design of nozzles.

Keywords: Nozzle Design Optimization, Method of Characteristics, Parameterization Techniques, Computational Fluid Dynamics, Surrogate-Based Optimization

Contents

List of Tables

List of Figures

Abbreviations

CAD	Computer-aided Design
CD	Convergent Divergent
CFD	Computational Fluid Dynamics
CFL	Courant-Friedrichs-Lewy
E-D	Expansion-deflection
EFFD	Extended Free Form Deformation
FDM	Finite Difference Method
FFD	Ordinary Differential Equation
FFD	Free Form Deformation
FSS	Free Shock Separation
FVM	Finite Volume Method
GA	Genetic Algorithm
LHS	Latin Hypercube Sampling
MDO	Multidisciplinary Optimization
MNG	Multi Nozzle Grid
MoC	Method of Characteristics
MSE	Mean Square Error
NS	Navier Stokes
NSGA	Nondominated Sorting Genetic Algorithm
NURBS	Non-uniform Rational B-splines
PDE	Partial Differential Equations
QP	Quadratic Programming

RANS	Reynold-Averaged-Navier Stokes
RMS	Root-Mean-Square
RMSD	Root-Mean-Square Deviation
RSS	Restricted Shock Separation
SBO	Surrogate-Based Optimization
SQP	Sequential Quadratic Programming
TIC	Truncated Ideal Contour
TOC	Thrust Optimized Contour
TOP	Thrust Optimized Parabola

Nomenclature

Greek symbols

α_{half}	Conical Nozzle Half Angle
Δ	Property Variation
ϕ	Velocity Potential
γ	Ratio of specific heats
λ_c	Conical momentum multiplier
μ	Mach Angle
ν	Prandtl-Meyer Function
$\hat{\Theta}, \Theta$	Estimator and estimated parameter
θ	Flow direction
θ_d	Deflection angle
θ_{ini}	Initiation angle vector
$\theta_{w_{max}}$	Sharp corner throat angle
ρ	Density
τ	Shear Stress
Ω	Domain
υ	Specific Volume

Other symbols

∇	Gradient
$\partial\Omega$	Boundary
ϖ	Specific Work
$d\theta$	Infinitesimal Deflection

Roman symbols

$\mathbf{A}, \mathbf{B}, \mathbf{T}_A, \mathbf{T}_B$ Ferguson Spline Coefficients

$\mathrm{A}, \mathrm{A}_{eq}, \mathrm{b}, \mathrm{b}_{eq}$ Linear inequality and equality constraints

A Area

a Speed of Sound

a_h Hicks coefficients

$\mathbf{B}$ Bézier Curve

B_i^n i th Bernstein basis polynomial functions of n degree

C CFL Number

C_+ Left-running characteristic

C_- Right-running characteristic

c_p Specific Heats at Constant Pressure

C_T Coefficient of Thrust

c_v Specific Heats at Constant Volume

$C_{k,n}$ Uniform B-spline of order k with n control points

$C_{T_{SSL}}$ Coefficient of Thrust at Sea Level

$C_{T_{vac}}$ Coefficient of Thrust in Vacuum

e Specific Internal Energy

$\mathbf{f}_b$ Body force

F Force

f Objective function

$f_{h=0}$ Richardson's Extrapolation Value

f_{Hicks} Hicks-Henne bump function

h Specific Enthalpy

h_s Grid element spacing

h_{nozzle} Nozzle exit height

I_{sp} Specific Impulse

K_+, K_-, KL, KR Prandtl-Meyer Constants

lb, ub	Lower and upper boundary constraints
L	Nozzle Length
L, U	Lower and Upper Riemann Sum
$\dot{m}$	Mass Flux
M	Mach Number
$m\mathbf{V}$	Momentum
$\mathcal{N}_i^k$	i th basis function of degree k
nlcon	Non-linear Constraints
n_f	Number of objectives
N_s	Number of cell in mesh
n_{char}	Number of Characteristics
$\mathbf{P}$	Control Point Vector
p	Pressure
$\dot{q}$	Heat Flux
q	Heat
R	Specific gas constant
r	Grid refinement ratio
R^2	Coefficient of Determination
R_{sum}	Centered Riemann Sum
R_{uni}	Universal gas constant
$\mathbf{S}$	Surface
s	Specific Entropy
S_{Fer}	Ferguson Spline
T	Temperature
t	Parameterization parameter
u, v, w	Velocity Cartesian components
$\mathbf{V}$	Velocity vector
$\mathcal{V}$	Volume

$\mathbf{v}_k$	Node Vector
V	Velocity magnitude
$\mathbf{x}$	Design variables
$\mathbf{x}_0$	Initial Design Variables
$\mathbf{x}_{\text{FFD}}$	FFD deformation parameters
$\mathbf{X}_{new}$	Updated contour after FFD
x, y, z	Coordinate system variables

Subscripts

0	Total Condition
adj	Adjacent condition
base	Base Contour
irrev	Irreversible
rev	Reversible
amb	Ambient conditions
avg	Average Condition
e	Exit conditions
i, j, k	Computational indexes
max	Maximal Value
n	Normal component
t	Throat conditions
x, y, z	Cartesian components
0	Total Condition

Chapter 1

Introduction

1.1 Motivation

The universe and the vastness of space have always been one of the greatest mysteries known to mankind. For millennia humanity perceived the Earth as the center of the universe. As times passed and technology evolved new secrets about the cosmos were revealed and humanity's interest in space grew exponentially.

It became apparent that the once empty and useless space had many more benefits than ever expected. However, as many might say, "to reach for the stars" is not an easy task. Many tried to achieve that feat and many failed as Greek mythology tells us about Icarus. Nevertheless, with the sudden evolution of technology and some of the most brilliant minds, mankind manages to send people to space aboard a rocket launcher. After this achievement, the floodgate to space and space travel opened up.

As of September 2021, more than 4500 satellites orbit the Earth simultaneously [1]. This number is only expected to grow higher and higher without any predictions to stop since most communication technologies of this day and age rely on satellites. SpaceX alone plans to build a 42 thousand satellite constellation (named Starlink) [2].

This comes with an enormous cost. In order to put a satellite in orbit, it is necessary to overcome the Earth's gravity which can only be achieved using a rocket launcher. Building and operating a rocket launcher requires a gigantic monetary fund. Even though the operational cost of rocket launchers has been declining in recent years due to improvements in their re-usability, space flight is still only available to governments, major companies, and hyper-wealthy persons.

Being made of many different components, one of the main parts of the rocket is its propulsive component, mainly the rocket nozzle. A rocket nozzle is single-handedly responsible for the creation of thrust allowing the rocket launcher to escape Earth's gravity.

Having a great impact on the overall performance of a rocket launcher, the proper assembly of a rocket nozzle weighs immensely on the amount of fuel needed for a specific mission [3], and consequently, the amount of payload supported, since the saved propellant weight can be replaced by more payload [4]. Therefore, the more optimal a rocket nozzle is the less money has to be invested into a space launch.

Proper nozzle contouring has been shown to have a very important impact on the nozzle's performance.

This triggered the now-known methods of nozzle design optimization, which first gained popularity back in the 1950s, being Rao one of its pioneers [5].

As technology advanced, more and more complex methods of optimization are developed (e.g., aerodynamic shape optimization based on high-fidelity Computational Fluid Dynamics). These high-fidelity methods are however not within reach of most computational budgets due to their extreme computational costs [6].

This is the motivation behind developing a low-fidelity and computationally friendly algorithm using the Method of Characteristics capable of being paired with optimization methods and together giving accurate results during the preliminary design of rocket nozzles.

1.2 Objectives and Deliverables

This dissertation comprises four main objectives, regarding the concepts of rocket propulsion and shape design optimization:

1. Develop a low-fidelity algorithm in MATLAB® based on the Method of Characteristics capable of simulating a flowfield around an arbitrary nozzle contour;
2. Apply various optimization methods to the previously developed algorithm in MATLAB®;
3. Apply Surrogate-based Optimization based on high-fidelity CFD simulations;
4. Compare results from low- and high-fidelity methods.

In order to achieve the first objective, a theoretical review regarding supersonic fluid mechanics is first conducted. The basis of Prandtl flows and the Method of Characteristics is utilized to design ideal nozzles, as well as, to describe the flowfield inside conical and bell nozzles. Developing a low-fidelity algorithm enables the optimization of nozzle flows without high computational expenses.

The second main objective derives from the motivation to study and optimize the contour of the nozzle for optimal thrust. This is achieved through the application of two unlike optimization algorithms, making sure that accurate numerical results are produced, by comparing the results of both methods. One of them being multi-objective gives a better perception on how optimizing a certain characteristic not always correlates well with an overall optimization.

The third objective consists in running CFD simulation using an Euler-based numerical method for an array of data points to compute an accurate surrogate-model. The open-source framework SU^2 is used for this purpose.

Following the implementation, an optimization process similar to the previously described is applied to the surrogate-model. This results in a computationally cheaper optimization, since the computational cost of applications involving complex simulations can be greatly reduced with the help of surrogate models [7].

Finally, the fourth main objective one matches both methods and concludes if the low-fidelity method is accurate and whether it may be introduced in real life application in the future.

1.3 Thesis Outline

This thesis is divided into six chapters. Each chapter is subsequently divided into sections and subsections aiming for a clear organization and smooth reading. Following the main body, appendixes are included where some of the MATLAB® scripts developed throughout this thesis are presented.

Chapter 1 - Introduction - describes the objectives and motivation behind the completion of this dissertation.

Chapter 2 - Literature Review - presents a concise revision regarding the topics of Rocket Propulsion Systems and Rocket Optimization. Different articles and bibliographies are reviewed for the reader to get familiarized with the subject at hand and stimulate the reader's critical thinking. The chapter is divided into two sections, starting with a review on nozzle propulsion systems followed by a study of nozzle optimization.

Chapter 3 - Theoretical Background - comprises the most relevant flow equations and other relevant mathematical formulations related to the field of fluid mechanics. Additionally, various parameterization techniques are displayed and suggested regarding their applicability to nozzle contours. Finally, the third chapter introduces the basis behind nozzle optimization, as well as, surrogate-modeling.

Chapter 4 - Implementation - employs the mathematical foundations reviewed in the previous chapter and describes the steps and strategies towards achieving the results. Divided into three sections, each section carries out the first three objective enumerated in section 1.2, respectively.

Chapter 5 - Results and Discussion - displays the results regarding both low- and high-fidelity based optimizations. Moreover, this chapter provides comparisons between both methods and enumerates their limitations.

Chapter 6 - Conclusions - draws the main conclusions regarding the obtained results along with challenges endured throughout the dissertation. The thesis finalizes with an outlook towards future work and possible improvements to apply on the MoC algorithm.

Chapter 2

Literature Review

This chapter will present a concise literature review regarding the topic of Rocket Propulsion Systems and Rocket Optimization.

A recommended article on which the literature review is based since it describes the development and research done on rocket nozzles along the history is "Rocket nozzles: 75 years of research and development" [8].

Nozzles were first introduced to build a device capable of changing a flow's characteristics, such as its velocity and pressure distribution. Carl Gustaf Patrik de Laval developed the first convergent-divergent nozzle capable of increasing a jet flow into a supersonic state in 1890, which later became known as a "de Laval" nozzle [9]. Robert Goddard is credited with the first flight using a "de Laval" nozzle with a combustion chamber [10].

Rocket nozzles form a large segment of the rocket engine structure. The principal parameters regarding its performance are the shape of the nozzle contour and the nozzle area expansion ratio. Rocket engines generate thrust by accelerating the exhaust gases to the desired speed and direction using the pressure generated inside the combustion chamber to enhance the magnitude of thrust by accelerating the combustion products to a high supersonic velocity [8, 11].

Rocket nozzles come in a wide variety of configurations, like an ideal, conical, bell, plug, expansion-deflection, and dual bell nozzles. Recently, a multi-nozzle grid was also developed.

2.1 Nozzle Configurations

A short review will be conducted on the most usual shapes these being the ideal, conic, and bell nozzles.

2.1.1 Ideal Nozzle

An ideal nozzle is characterized by producing an isentropic flow (i.e., without internal shocks) and having uniform properties at the exit, meaning constant pressure, temperature, and velocity over the whole exit plane. The contour of such a nozzle can be designed with the Method of Characteristics [12] (described in chapter 3). In an ideal nozzle, the thrust becomes maximum when the parallel uniform flow exit pressure matches its

ambient pressure [8, 13].

However, the ideal nozzle that gives maximum thrust performance is heavy and lengthy. In order to expand a flow to match near vacuum conditions one would need a near infinite nozzle [14].

The length of an ideal nozzle can be decreased by permitting all the expansion to occur just at the throat (amid a sharp corner) and then constructing the nozzle contour to turn the flow such that it can attain an axial uniform flow at the exit [15]. This nozzle is referred to as a minimum-length nozzle.

In the following subsections, one will read about nozzle configuration with reduced length, however without appreciable loss in thrust performance.

2.1.2 Conical Nozzle

The application of conical nozzles was very popular during the early stages of rocket propulsion due to its simplistic design and ease of manufacture. However, its main downfall compared to an ideal nozzle comes from the fact that the exit flow is divergent, meaning that the conical nozzle leads to a reduction in thrust as the flow direction is not completely axial at the exit.

Malina [16] has shown that the exit momentum of a conical nozzle is equal to the value computed from the one-dimensional theory (ideal nozzle) multiplied by a factor:

$$\lambda_c = \frac{1 + \cos(\alpha_{half})}{2}. \tag{2.1}$$

On the other hand, by increasing the nozzle's divergent angle α_{half} both the weight and size of the nozzle can be reduced.

This results in a trade-off between the divergence angle and the nozzle length. It has been concluded that the best compromise for the diverging angle of a conical nozzle is 15°, resulting only in the coefficient of pressure dropping 1.7% [8, 17].

A downside to a conical nozzle is the possible formation of a shock wave near the throat. This shock occurrence can, however, be eliminated by smoothing out the wall contour near the junction of the throat profile and the cone [18].

2.1.3 Bell/Contoured Nozzle

Bell nozzles are the most commonly used shapes in rocket engines. They are comprised of a high-angle expansion section (20° to 50°) downstream of the nozzle throat followed by a gradual reversal of the nozzle contour slope so that at the nozzle exit the divergence angle is small, avoiding major divergence losses [11].

Contoured nozzles, however, demonstrate a high dependency of the exit flow on the nozzle contour, meaning no simple relation can be derived for the coefficient of thrust, contrarily to the ideal and conical nozzles [17].

Rao [5] developed in 1958 a method by using the calculus of variations to design the wall contour of the optimum thrust nozzle by using a simple parabolic approximation. Later in 1981, Allman and Hoffman [19] examined a technique for the design of maximum thrust nozzle contours using direct optimization methods. Both methods predicted similar maximum thrust.

Finally, Frey *et al.* [15] presented a revolutionary nozzle design called TICTOP, consisting of Truncated Ideal Contour (TIC) and Thrust Optimized Parabolic (TOP) designs.

2.2 The TICTOP Nozzle

A deeper review will be administered regarding the three previous types of nozzles. To judge the different nozzle contouring methods, it is necessary to enumerate some design rules, which lay the foundation for appropriate nozzle design: i) The specific impulse I_{sp} as well as the coefficient of thrust C_T shall be maximized; ii) The start-up side-loads shall be minimized; and iii) flow separation during steady state operations shall be avoided [15].

Points ii) and iii) stem from the fact that whenever the flow inside the nozzle expands to a pressure lower than 30% of the ambient pressure p_{amb}, the boundary layer momentum is not sufficient to overcome the adverse pressure differential and as a consequence, the flow separates from the nozzle wall. During take-offs, this is always the case, since the chamber pressure is far below its design value [15]. More details on flow separation phenomena will be presented later.

2.2.1 Truncated Ideal Contour (TIC) Nozzles

As mentioned before, ideal nozzles are too long and thus too heavy. Furthermore, the nozzle exit section contributes little to the thrust, because the flow is already almost parallel to the nozzle's centerline a long way before the exit plane [20]. This results, in an ideal nozzle being truncated. Often a geometrical envelope is given as a boundary condition during the contouring process [15].

2.2.2 Thrust-Optimized Parabola (TOP) Nozzles

The TOP contouring method was proposed by Rao in 1962 [21] as a simplification of his aforementioned thrust optimization contour approach (TOC) [5].

In opposition to the TIC nozzle, a TOP contour is characterized by an internal shock originating from the throat region [22]. However, TOP nozzles usually show a higher exit pressure compared to TIC nozzles (21% higher), meaning a TOP nozzle has a higher margin against flow separation than the TIC through the same exit point, indicating a better behavior regarding requirement iii) [15].

On the other hand, the occurrence of an internal shock may cause high side loads when combined with flow separation (discussed in detail in section 2.3). Consequently, a TIC nozzle fulfills requirement ii) better than a TOP nozzle. Regarding requirement i) both nozzles have shown to reach almost identical values, not justifying the obvious selection of a method.

2.2.3 TICTOP Concept

The TICTOP contouring method consists in combining the good side-load properties of a TIC and the high nozzle exit pressure and better separation margin of a TOP [15]. This is achieved by applying the ideal contour

near the nozzle's throat ensuring a shock-free flow followed by a smooth transition into a parabola in order to guarantee that the exit pressure is sufficiently high.

2.3 Flow Separation inside a Rocket Nozzle

Flow separation in supersonic nozzles is a phenomenon intrinsically connected to shock waves, meaning that one influences the other and vice-versa [23]. All flow separations in supersonic flows are accompanied by an oblique shock necessary to permit the flow from deviating from the wall. As aforementioned, separation in rocket nozzle flows is undesirable because of its unsteady and nonsymmetric nature, which leads to dangerous side-loads [24].

Two distinctive shock separation patterns can take place inside a rocket nozzle: i) Free Shock Separation (FSS), in which the boundary layer never reattaches; and ii) Restricted Shock Separation (RSS) characterized by a closed recirculation bubble.

Experimental and numerical studies have led to believe that the highest side loads take place when the flow structure inside the nozzle passes from a condition with separated flow (FSS) to a condition of flow separation and reattachment (RSS) and contrariwise [23, 25]. To understand the fluid mechanics behind this phenomenon, both shock separation processes are presented.

2.3.1 Free Shock Separation

The flow structure of an FSS is easily recognizable by the three main shock branches: i) the Mach disk; ii) the reflected shock; and iii) the separation shock, where the connecting point is referred to as Triple Point [25].

The separation and subsequent formation of a recirculation zone induce the formation of an oblique shock, which consequently interacts with the Mach disk forming a reflected shock. The separated jet continues as a free jet confined by a "fluidic wall".

2.3.2 Restricted Shock Separation

First and foremost, this type of separated flow only exits in nozzles that contain an internal shock [24]. This is the reason why the TICTOP concept is practical, since it eliminates the possibility of this flow condition occurring, meaning no high side loads are generated by the transition of shock separation.

RSS is described by the interaction of the internal shock with the normal shock, previously referred to as Mach disk, far away from the throat. Consequently, a reflected shock forms from this common point of interaction which in turn interacts with the incident oblique shock, forming two reflected shocks. Downstream of this interaction, the boundary layer reflects back towards the nozzle wall and reattaches, forming a small recirculation zone attached to the wall. The reattached supersonic jet undergoes a series of shocks and expansions to adapt to the ambient pressure [24, 25].

A comparison between both shock separations can be visualized in figure 2.1.

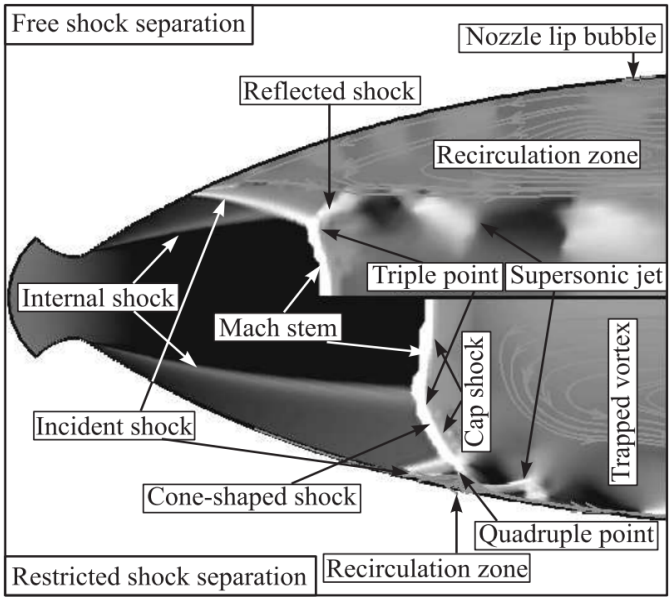


Figure 2.1: Free shock and restricted shock separation flow regimes [24].

2.3.3 Transition During Up- and Down-Ramping

During the up-ramping (e.g., in take-off), the transition from FSS to RSS occurs when the internal shock intersects the separation shock below the triple point. If the intersection is above, the internal and the separation shocks coalesce [25] as shown in figure 2.1.

During the down-ramping, the transition occurs similarly. However, it does not occur for the same flow condition as the transition. A type of hysteresis phenomenon takes place [22, 26].

2.3.4 Generation of Side-Loads

When the engine is started, during the transition from one separation pattern to another, a phase might exist, during which one side of the nozzle experiences a free-shock separation while on the other side, the flow reattaches. As the separation point is located further downstream for RSS and the wall pressure behavior shows completely different distributions, there would be severe lateral forces acting on the nozzle. These side forces would be described as being strong and short.

For the same nozzle, a second side-load peak could be expected during engine start as soon as the reattachment point reaches the nozzle exit and ambient air can flow into the previously closed recirculation zone. At this moment the RSS would be converted into an FSS and with it, a short and strong side force occurs [27].

2.4 Advanced/Innovative Nozzle Configurations

After analyzing the more usual nozzle configuration a short review will be conducted on the innovative, but noteworthy contouring methods.

2.4.1 Plug Nozzle

The main characteristic of a plug nozzle is its capacity to interact with the external ambient, resulting in having a free jet boundary that acts as a virtual outer wall and expands and compresses to match the freestream ambient pressure. The separation of flow can be avoided, implying that plug nozzles are altitude compensating [28].

For underexpanded or optimal conditions, the plug nozzle behaves like a conventional convergent-divergent nozzle. However, for overexpanded conditions, the jet boundary is pulled nearer to the plug by a series of compression waves and expansion fans that occur naturally in an attempt to match the ambient pressure, as demonstrated in figure 2.2. A series of compressions and expansions often indicates a loss of efficiency, but in this case, it improves the nozzle performance [29]. Berman *et al.* [30] showed that the plug nozzle shows a thrust benefit over a conventional nozzle when it is working at below design pressure ratio $p_r = p_0/p_{amb}$.

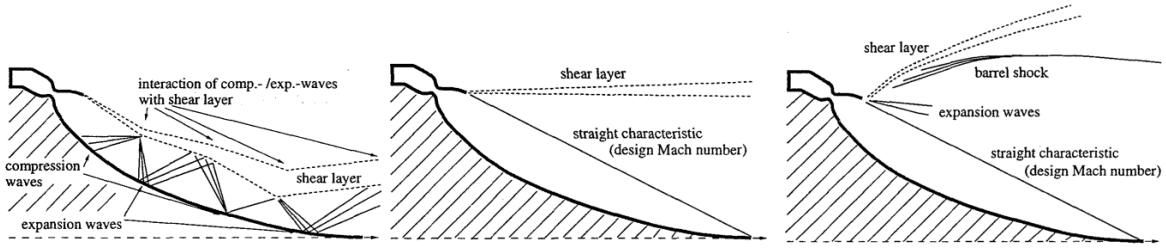


Figure 2.2: Flow phenomena of a plug nozzle with full length at different pressure rations [31].

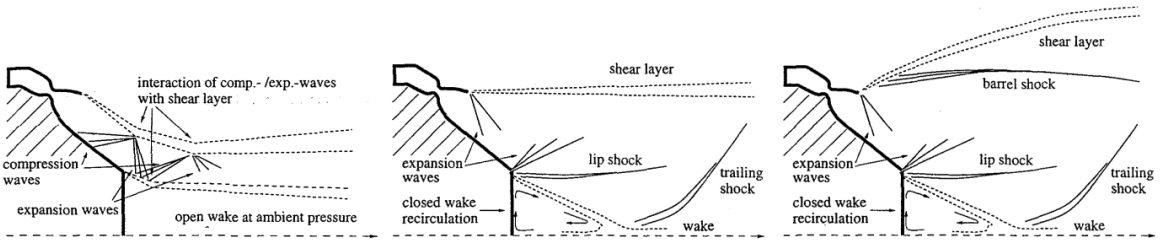


Figure 2.3: Flow phenomena of a plug nozzle with truncated central body at different pressure rations [31].

Similarly to the ideal nozzle, the plug nozzle can also be truncated, because it eliminates the huge ideal length and heavy structural mass of the well-contoured body. However, for low-pressure ratios, an open wake flow is established with a pressure level equal to the ambient pressure, which eventually closes when the design pressure ratio of the full-length plug nozzle is achieved. At the point of transition, the pressure inside the wake reaches a value below the ambient pressure, and the full base area generates a negative thrust. On the contrary, for high-pressure ratios, a close wake forms, with a constant pressure higher than the ambient one, resulting in a positive thrust contribution [31]. Such developments are presented in figure 2.3.

The major disadvantage of plug nozzles lies in their relatively high cooling requirements [8].

2.4.2 Expansion-Deflection (E-D) Nozzle

Like the previous nozzle configuration, the E-D nozzle is altitude-compensating. Although looking similar to the bell nozzle, the flow is turned by a "center body" onto the outer diverging nozzle wall [11]. This results

in the creation of a viscous wake region within the nozzle.

For low altitudes during over-expanded conditions, the wake region is open to the atmosphere and will vary the effective area ratio similar to a plug nozzle. As the atmospheric pressure decreases with increasing altitude and the flow conditions near the design conditions, the required area ratio will increase until the physical maximum is reached, meaning that the shear layers will intersect, resulting in a closed wake, as shown in figure 2.4 [32].

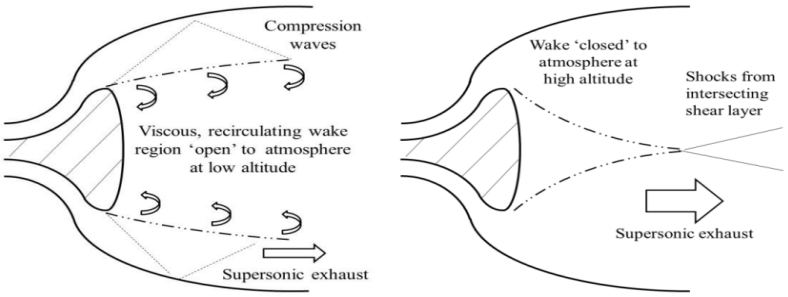


Figure 2.4: Flow behavior in the ED nozzle in open mode and closed mode [32].

Rao [33] described a method of designing the optimum nozzle wall contour, showing that the required length of an E-D nozzle was only 50% of that of a bell nozzle to achieve the same performance.

2.4.3 Dual-Bell Nozzle

The dual-bell nozzle, as mentioned in the name, consists of two distinct contours between the throat and the exit [8].

Considered an altitude-adaptive nozzle concept, it combines a small nozzle area ratio at low altitude with a large one at high altitude. Its design is typically based on an inner base nozzle followed by a wall inflection region and an outer nozzle extension. The contour inflection forces the flow to a controlled and symmetrical flow separation under high ambient pressure conditions [34].

For higher altitudes, the nozzle flow is attached to the wall until the exit plane, and the full geometrical area ratio is used. Because of the higher area ratio, an improved vacuum performance is achieved [31]. Both modes are presented in figure 2.5.

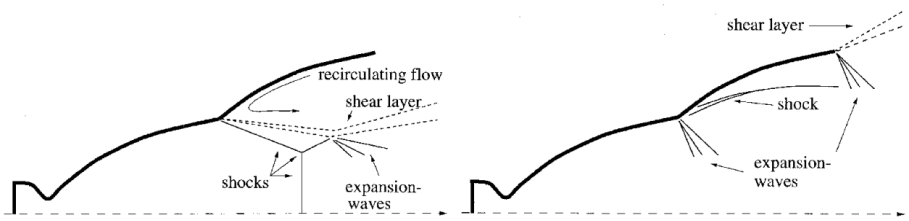


Figure 2.5: Flow field phenomena in dual-bell nozzles [31].

Performance losses come, however, with the flow separation in sea-level operations. The pressure within the separated flow region of the dual-bell nozzle is slightly lower than the ambient pressure, inducing a thrust loss referred to as "aspiration drag". Additionally, this aspiration effect also triggers transition before the optimum performance cross-over point [35].

2.4.4 Multi Nozzle Grid (MNG)

The multi-nozzle grid configuration is the most recent and the only nozzle configuration where the nozzle length (i.e., plate thickness) to throat diameter can be less than one, yet is capable of providing an extremely high area ratio. The multidisciplinary optimization presented in [36] created a thin and lightweight MNG plate, concluding that the length saved is proportional to the square root of the number of the "nozzlettes" (small nozzles) in the MNG, resulting in a performance improvement by more than 11% due to its mass savings [37].

2.5 Nozzle Optimization

Traditionally, rocket nozzles have been designed exclusively for propulsive performance. The optimization of nozzle profiles for maximum thrust was pursued long before computational fluid mechanics (CFD) or multidisciplinary optimization (MDO) became widely-available tools [38]. A short review will be conducted on the subject of nozzle optimization.

2.5.1 Nozzle Configuration Applied to Optimization Purposes

In order to conduct a design optimization, it is necessary to adapt a base geometry through various parameterization techniques. As a starting point for every optimizer, a simple design is desired. The ultimate goal is for engineers to modify initial simple prototypes and analyze the modified configurations to improve their performance [39]. Typical base nozzle configurations to replicate real rocket nozzles are TIC, TOP, and TOC nozzles, being the latter the most widely used design, developed by Rao [5, 40]. Although not so popular, conical nozzles are also widely used for optimization purposes, due to their simplicity [41].

An up-and-coming nozzle configuration that has become popular and is constantly being optimized due to its high potential is the aerospike nozzle, previously mentioned as plug nozzle, which is said to have overall better performance than the conventional bell-shaped nozzle during 90% of its flight configurations [42].

Yumusak *et al.* [39] concluded that the definition of the objective function has major importance during optimization studies. For thrust optimization, the amount of thrust gained was minimal, while for a study of nozzle length minimization the overall performance of the rocket propulsion system had a bigger effect.

Sun *et al.* [40] proposed an innovative nozzle extension section comprised of an outward bent curve capable of increasing the vacuum-specific impulse by 1.5 seconds.

With their conical nozzle study, Hoffman *et al.* [41] deduced that for real systems subjected to non-isentropic flow effects, maximum performance requires some amount of under-expansion and consequently a smaller area ratio than isentropic flows.

2.5.2 Numerical Methods

Classical optimization procedures usually begin with an inviscid design (such as Rao's method). Then a boundary-layer correction would be added to compensate for the viscous effects.

Recent advances in computational technology have allowed the integration of the full Navier-Stokes equations, as well as, enabling automatic design methods developed by combining the CFD and optimization

codes [43]. However, this has not always been the case, since optimization techniques have been utilized to design rocket nozzles since the 1950s [39]. For the most part, numerical methods had to be paired with an external optimization algorithm in order to ensure the functioning of the optimizer.

These numerical methods can, for the most part, be divided into low- and high-fidelity methods. Both are extensively used because of the computational cost of higher-fidelity models. When computationally expensive analyses are used it is often of benefit to utilize multiple levels of fidelity to reduce the size of the design space and initialize subsequent analyses from a lower-fidelity solution [38]. This approach is referred to as the Multi-Fidelity Approach.

Regarding the numerical methods themselves, Euler and RANS (Reynold-Averaged-Navier Stokes) are the most current methods used associated with CFD. An Euler inviscid CFD solution is considerably less expensive than a Navier-Stokes viscous solution due to both a reduced mesh size and simplified physics [39].

To reduce the computational time, sometimes simpler models are used for flow analysis. The aforementioned Rao's method though simple in formulation yields remarkable valuable results and it is still widely used in the preliminary design of rocket nozzles [38]. This method can be evaluated with essentially negligible computational cost, thus a large number of designs can be evaluated rapidly. However, its assumption of inviscid isentropic flow makes it deviate from the real flow. Over time many modifications and improvements of this method have been studied [40].

Finally, the Method of Characteristics is also used for nozzle design optimization, taking advantage of the concept of Prandtl-Meyer theory to compute a flowfield without the usage of CFD [44], similarly to the previous method.

As mentioned before, low-fidelity methods should be paired with higher-fidelity ones to ensure proper design optimization. However, it has to be taken into consideration that there is no general proof showing that a higher-fidelity optimization solution must exist within some quantifiable range of a lower-fidelity optimum [38].

Cai *et al.* [43] showed in his optimization study that CFD-based optimization processes achieve better nozzle performances than classical optimization method (e.g., Rao's Method), achieving a 1.5% improvement in performance.

Colonno *et al.* [38] concluded, on the other hand, that multidisciplinary optimization shows clear benefits compared to traditional optimization because a good compromise between disciplines is found to lead to an overall performance increase.

2.5.3 Optimization Algorithms

Nozzles are generally designed for maximum thrust or specific impulse. The traditional approach for solving any optimization problem is generally a mathematical approach, where the optimized value is expressed as a function in an explicit or implicit form with several design variables [45].

Depending on the behavior of the objective and constraint functions' response, a gradient-based optimization algorithm or genetic algorithms may perform better [38]. With technological advancement, machine learning is beginning to be used as a rocket nozzle optimization technique. DiDominic *et al.* [46] applied an artificial neural network with the ability to accurately relate input parameters to highly complex non-linear output

parameters. The trained neural network acted effectively as a rapid prediction tool when utilized as a surrogate model in optimization.

Matveev *et al.* [45] managed to reduce a nozzle's length by 15%, while only reducing its thrust by 0.18% by choosing a configuration part of a Pareto set computed by a multi-objective GA where the objective was to obtain maximum thrust with the minimum possible nozzle length.

2.5.4 Parameterization Methods

Imagine having to perform MDO for a complete aerospace configuration during the design phase. The choice of shape parameterization technique has an enormous impact on the formulation and implementation of the optimization problem. A parameterization method is considered favorable if it is i) automated; ii) provides consistent geometry changes; iii) fits into the contours description; and iv) produces a compact and effective set of design variables [47].

In general, shape parameterization techniques can be divided into eight categories: basis vector; domain element; discrete; analytical; free-form deformation (FFD); partial differential equation (PDE); polynomial and spline; and computer-aided design (CAD).

The parameterization that best fits all four previously mentioned conditions is the Free Form Deformation parameterization technique. Owing to the deformation mechanism of the FFD method, this method can easily realize local shape modification and synchronous deformation of the flow field mesh and improve the optimization efficiency [48]. The FFD formulation is independent of grid topology making it suitable for a variety of analysis codes, such as low- and high-fidelity analysis tools [47].

This is a versatile parameterization technique that was originally used with a solid modeling system [49]. More recently it has been proposed in a variety of contexts, for example, the parameterization of airfoils and wings in a shape optimization context [50].

Similarly to Rao's method, due to its popularity and simple implementation, many FFD variations were developed such as the FFD of solids with trivariate B-splines implemented by Greissmair and Purgathofer [51] and the method of extended free form deformation (EFFD) by Coquillart and Sabine [52].

Dang *et al.* [48] combined the FFD technique and CFD method to run an aerodynamic design optimization of the hypersonic rocket sled deflector and its ability to increase stability. The optimized deflector demonstrated a negative lift increase by 7.39%, thus confirming the effectiveness of the proposed method.

Koshakji *et al.* [50] demonstrated that FFD is a powerful, flexible and efficient parameterization method since it has the capability of also deforming the mesh defined inside the nozzle domain. In their study, they achieved a 27.6% drag reduction and a 17.1% of efficiency improvement for an aircraft's rudder.

Finally, Coquillard *et al.* [52] presented accurate three-dimensional sculpturing and molding simulated by their developed method of EFFD. EFFD shows to be an easy-to-use and efficient method for modeling cloth-like surfaces.

Chapter 3

Theoretical Background

This chapter contains all the information needed to implement all the methodologies used throughout this dissertation. A small recap of compressible nozzle flows is written along with an introduction to parameterization techniques and design optimization for the reader to get familiarized with the theoretical background of this thesis.

3.1 Compressible Flow

This section is structured in a way such that the equations get more complex along the way until the equations for inviscid compressible flows are reached. The deduction of the equations is rooted in the book "Fundamentals of Aerodynamics" by John D. Anderson Jr. [53].

3.1.1 Review on Thermodynamics

All lower caps flow properties in this section describe their specific properties. Specific properties are intensive properties that are expressed on a "per unit mass" basis.

Perfect Gas

A perfect gas is described as a collection of particles, where every particle of gas is sufficiently far apart from each other, resulting in negligible intermolecular forces. For a perfect gas it is possible to relate density ρ with pressure p and temperature T through the **Equation of State**:

$$p = \rho RT, \tag{3.1}$$

where R is the specific gas constant, which can be calculated as $R = R_{uni}/M$ considering the universal gas constant R_{uni} and the molecular mass M of gas, yielding a value of 287 J/(kg.K) for the atmospheric air [11].

Energy and Enthalpy

When dealing with a perfect gas, both enthalpy h and internal energy e are solely dependent on temperature meaning that

$$de = c_v dT, \tag{3.2}$$

$$dh = c_p dT, \tag{3.3}$$

where enthalpy is defined as

$$h = e + p\upsilon = e + \frac{p}{\rho}, \tag{3.4}$$

and c_v and c_p are the specific heats for constant volume and pressure, respectively. υ is referred to as specific volume.

Both specific heats are related through the specific gas constant as shown

$$c_p - c_v = R. \tag{3.5}$$

Defining a new variable as the ratio of specific heats $\gamma = c_p/c_v$ one obtains the following relations

$$c_p = \frac{\gamma R}{\gamma - 1}, \tag{3.6}$$

$$c_v = \frac{R}{\gamma - 1}. \tag{3.7}$$

Laws of Thermodynamics

The first law of thermodynamics dictates that the heat added δq and work done $\delta \varpi$ to a system causes an equal change in the internal energy de hence:

$$\delta q + \delta \varpi = de. \tag{3.8}$$

Before mentioning the second law of thermodynamics entropy ds is defined as:

$$ds = \frac{\delta q_{rev}}{T} = \frac{\delta q}{T} + ds_{\text{irrev}}, \tag{3.9}$$

where δq_{rev} is a reversible addition of heat and ds_{irrev} describes the generation of entropy due to an irreversible process.

The second law dictates that the spontaneous generation of entropy cannot be negative,

$$ds_{\text{irrev}} \geq 0. \tag{3.10}$$

Combining equations 3.9 and 3.10, one has

$$ds \geq \frac{\delta q}{T}, \tag{3.11}$$

which establishes the direction that the process will take place.

Reversible Process

In the special case that $ds_{\text{irrev}} = 0$, a reversible process takes place. In this particular scenario

$$\delta\varpi = -pd\upsilon, \tag{3.12}$$

where $d\upsilon$ describes a change in volume due to a displacement of the boundary of the system. Inserting equations 3.9 and 3.12 into the first law of thermodynamics (equation 3.8) assuming that heat is reversibly added we obtain

$$Tds = de + pd\upsilon, \tag{3.13}$$

which can easily be reformulated using equation 3.4 into

$$Tds = dh - \upsilon dp. \tag{3.14}$$

Equations 3.13 and 3.14 express the first law in terms of entropy. Both equations can be further developed using the previous expressed relations 3.2 and 3.3 obtaining

$$ds = c_v \frac{dT}{T} + \frac{pd\upsilon}{T}, \tag{3.15}$$

$$ds = c_p \frac{dT}{T} - \frac{\upsilon dp}{T}. \tag{3.16}$$

For a calorically perfect gas where c_p and c_v are both constant, equations 3.15 and 3.16 become, respectively,

$$s_2 - s_1 = c_v \, ln\frac{T_2}{T_1} + R \, ln\frac{\upsilon_2}{\upsilon_1}, \tag{3.17}$$

$$s_2 - s_1 = c_p \, ln\frac{T_2}{T_1} - R \, ln\frac{p_2}{p_1}. \tag{3.18}$$

Isentropic Relation

A process that is both reversible and adiabatic is defined as isentropic. According to equation 3.9 if $\delta q = 0$ (adiabatic) and $ds_{\text{irrev}} = 0$ (reversible) then $ds = 0$, concluding that entropy remains constant. Equaling equations 3.17 and 3.18 to zero and applying relations 3.6 and 3.7 we obtain the relations between pressure, density, and temperature for an isentropic process

$$\frac{p_2}{p_1} = \left(\frac{\rho_2}{\rho_1}\right)^{\gamma} = \left(\frac{T_2}{T_1}\right)^{\gamma/\gamma-1}. \tag{3.19}$$

These relations are very important because a large quantity of practical compressible flows can be assumed isentropic-like through a rocket nozzle. Assuming that no heat is being transferred due to good heat flux isolation and that the dissipative effects of the boundary layer can be neglected we experience an isentropic flow.

3.1.2 Governing Equations for Inviscid, Compressible Flow

Continuity Equation

The first governing equation dictates the physical principle that mass can neither be created nor be destroyed. The following equation is known as the **Continuity Equation**:

$$\frac{\partial}{\partial t}\iiint_{\mathcal{V}} \rho \, d\mathcal{V} + \oiint_{S} \rho \mathbf{V} \cdot \mathbf{dS} = 0, \tag{3.20}$$

where $\mathcal{V}$, $\mathbf{S}$ and $\mathbf{V}$ denote volume, surface, and velocity, respectively.

Equation 3.20 can also be written in the form of a partial differential equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{V}) = 0. \tag{3.21}$$

For a steady flow, $\partial / \partial t = 0$, equations 3.20 and 3.21 reduce, respectively, to

$$\oiint_{S} p\mathbf{V} \cdot \mathbf{dS} = 0, \tag{3.22}$$

$$\nabla \cdot (\rho \mathbf{V}) = 0. \tag{3.23}$$

Momentum Equation

The **Momentum Equation** is an equality between the rate of change of the momentum $m\mathbf{V}$ of a fluid and the forces F applied to it

$$F = \frac{d}{dt}(m\mathbf{V}) \, . \tag{3.24}$$

The force exerted in the fluid has two sources: (i) body forces $\mathbf{f}_b$, such as gravity, that act on the volume of the fluid, and (ii) surface forces, such as pressure and shear stress ($\tau = 0$ due to inviscid flow). These forces are mathematically represented below, respectively:

$$\text{Body force} = \iiint_{\mathcal{V}} \rho \mathbf{f}_b \, d\mathcal{V} \, , \tag{3.25}$$

$$\text{Pressure force} = -\oiint_{S} p \mathbf{dS} \, . \tag{3.26}$$

Regarding the time rate of change of momentum, it is comprised of the net flow of momentum out of the control volume and the time rate of change of momentum, resulting in the following momentum equation:

$$\frac{\partial}{\partial t}\iiint_{\mathcal{V}} \rho \mathbf{V} d\mathcal{V} + \oiint_{S} (p\mathbf{V} \cdot \mathbf{dS})\mathbf{V} = -\oiint_{S} p \mathbf{dS} + \iiint_{\mathcal{V}} \rho \mathbf{f}_b \, d\mathcal{V}, \tag{3.27}$$

that similarly to the continuity equation can be written in the form of a partial differential equation

$$\rho(\frac{\partial \mathbf{V}}{\partial t} + \mathbf{V}\nabla\mathbf{V}) = -\nabla p + \rho \mathbf{f}_b. \tag{3.28}$$

For a steady flow with no body forces ($\mathbf{f}_b = 0$) we obtain the **Euler Equations**:

$$\nabla \cdot (\rho u \mathbf{V}) = -\frac{\partial p}{\partial x}, \tag{3.29a}$$

$$\nabla \cdot (\rho v \mathbf{V}) = -\frac{\partial p}{\partial y}, \tag{3.29b}$$

$$\nabla \cdot (\rho w \mathbf{V}) = -\frac{\partial p}{\partial z}, \tag{3.29c}$$

where u, v, and w are the x, y, and z components of speed, respectively. This set of equations can be reduced to a single scalar equation:

$$dp = -\rho V dV, \tag{3.30}$$

which relates the change in velocity along a streamline dV to the change in pressure dp along the same streamline.

Energy Equation

It is known that internal energy e just like mass can neither be created nor destroyed. On that notice we have the following equation:

$$\frac{\partial}{\partial t}\iiint_{\mathscr{V}} \rho\left(e + \frac{V^2}{2}\right) d\mathscr{V} + \iint_S \rho\left(e + \frac{V^2}{2}\right)\mathbf{V}\cdot\mathbf{dS} = \iiint_{\mathscr{V}} \dot{q}\rho\, d\mathscr{V} - \iint_S p\mathbf{V}\cdot\mathbf{dS} + \iiint_V \rho(\mathbf{f}_b\cdot\mathbf{V}) d\mathscr{V}\ . \tag{3.31}$$

The terms represent respectively from the left to the right the time rate of change of total energy inside $\mathscr{V}$, the net rate of flow of total energy across the control surface, the total rate of heat addition, the rate of work done in the fluid due to pressure forces and finally the rate of work done on the fluid due to body forces. Modifying equation 3.31 into a partial differential equation we obtain

$$\frac{\partial}{\partial t}\left[\rho\left(e + \frac{V^2}{2}\right)\right] + \nabla\cdot\left[\rho\left(e + \frac{V^2}{2}\right)\mathbf{V}\right] = \rho\dot{q} - \nabla\cdot(p\mathbf{V}) + \rho(\mathbf{f}_b\cdot\mathbf{V})\ . \tag{3.32}$$

If the flow is steady, adiabatic ($\dot{q} = 0$) and without body forces ($\mathbf{f}_b = 0$), equation 3.31 is reduced to

$$\iint_S \rho\left(e + \frac{V^2}{2}\right)\mathbf{V}\cdot\mathbf{dS} = -\iint_S p\mathbf{V}\cdot\mathbf{dS} \tag{3.33}$$

or in differential form

$$\nabla\cdot\left[\rho\left(e + \frac{V^2}{2}\right)\mathbf{V}\right] = -\nabla\cdot(p\mathbf{V})\ . \tag{3.34}$$

When dealing with fluid dynamics, the definition of material or substantial derivative is of great importance. The material derivative computes the instantaneous time rate of change of any quantity for a portion of a

material moving with a velocity V [54]. The substantial derivative can be written as

$$\frac{D}{Dt} \equiv \frac{\partial}{\partial t} + (\mathbf{V} \cdot \nabla) \,. \tag{3.35}$$

Equation 3.35 is comprised of the *local derivative* and the *convective derivative*, which are the time rate of change at a fixed point and the time rate of change due to the movement of the fluid, respectively. All governing equations 3.21, 3.28, and 3.32 can be written in terms of the substantial derivative:

$$\frac{D\rho}{Dt} = 0 \,, \tag{3.36}$$

$$\rho \frac{Du}{Dt} = -\frac{\partial p}{\partial x} + \rho \mathrm{f}_{b_x} \,, \tag{3.37a}$$

$$\rho \frac{Dv}{Dt} = -\frac{\partial p}{\partial y} + \rho \mathrm{f}_{b_y} \,, \tag{3.37b}$$

$$\rho \frac{Dw}{Dt} = -\frac{\partial p}{\partial z} + \rho \mathrm{f}_{b_z} \,, \tag{3.37c}$$

$$\rho \frac{D(e + V^2/2)}{Dt} = \rho \dot{q} - \nabla \cdot (p\mathbf{V}) + \rho(\mathbf{f}_b \cdot \mathbf{V}) \,. \tag{3.38}$$

This set of five equations describes the fluid mechanics of inviscid compressible flows. More complex equations like the Navier-Stokes Equations, which take viscosity into account, could be deduced, however, they exceed the scope of this thesis.

3.1.3 Velocity Potential

Considering a two-dimensional, steady, irrotational ($\nabla \times \mathbf{V} = 0$), and isentropic flow, we can define a velocity potential $\phi = \phi(x, y)$ such that

$$\mathbf{V} = \nabla \phi \,. \tag{3.39}$$

Applying the velocity potential relation 3.39 to equations 3.23 and 3.30 one obtains

$$\begin{aligned} \left[1 - \frac{1}{a^2}\left(\frac{\partial \phi}{\partial x}\right)^2\right] \frac{\partial^2 \phi}{\partial x^2} + \left[1 - \frac{1}{a^2}\left(\frac{\partial \phi}{\partial y}\right)^2\right] \frac{\partial^2 \phi}{\partial y^2} \\ - \frac{2}{a^2}\left(\frac{\partial \phi}{\partial x}\right)\left(\frac{\partial \phi}{\partial y}\right) \frac{\partial^2 \phi}{\partial x \partial y} = 0, \end{aligned} \tag{3.40}$$

where a is the speed of sound.

For the sake of simplicity, only the final equation is presented which deductions are referred to in [53]. This equation is called the *velocity potential equation* and will be later used to elaborate on the *Method of Characteristics*.

3.2 Nozzle Flows

Nozzle flows describe all types of flows going through nozzles, which are devices that increase the velocity of a fluid at the expense of pressure. Their interior flows experience fascinating phenomena ranging from supersonic speeds to shock waves. Being the subject matter of this thesis, this section will describe the physics behind nozzle flows and present the main equations related to the flow mechanics of nozzles deduced in [53].

3.2.1 Supersonic Flow

A steady flow can be either subsonic, sonic, or supersonic. It depends if at a point the flow speed $V = \sqrt{u^2 + v^2 + w^2}$ is, respectively, less than, equal to, or greater than the speed of sound

$$a = \sqrt{\gamma RT} \tag{3.41}$$

at that particular point. Regarding the Mach number,

$$M = \frac{V}{a}, \tag{3.42}$$

one characterizes the flow as supersonic if $M>1$. In the specific case of the fluid being both subsonic and supersonic at different locations, it is described as a *transonic flow*. When dealing with a supersonic flow, one has to take into consideration the fact that the fluid is compressible. In fact, for Mach numbers over 0.3, the error due to compressibility cannot be neglected anymore, as observed in figure 3.1. This is the reason for

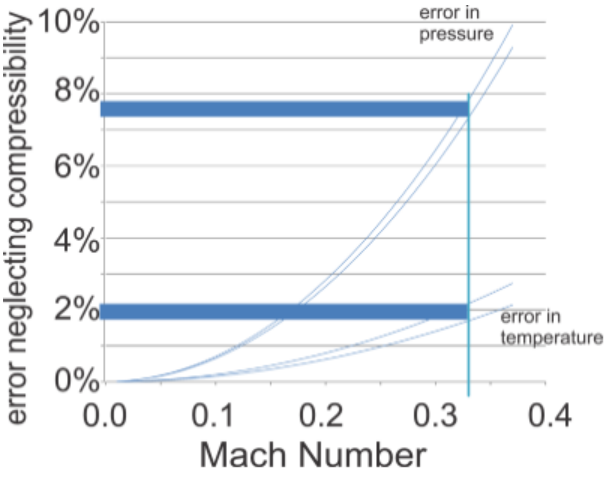


Figure 3.1: Compressibility error for increasing Mach number [55]

equation 3.29 deduction otherwise one could simply use *Bernoulli's Equation* [56].

3.2.2 Governing Equations for Quasi-One-Dimensional Flow

When dealing with a quasi-one-dimensional flow, one considers the flowfield variables and the area solely to depend on x, as illustrated in figure 3.2. Using this approximation the velocity $\mathbf{V}$ can be reduced to u, its x component of speed.

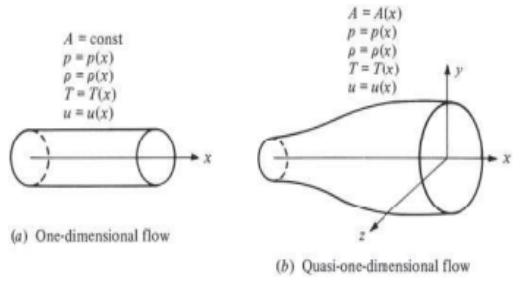


Figure 3.2: One-dimensional and quasi-one-dimensional flows [53].

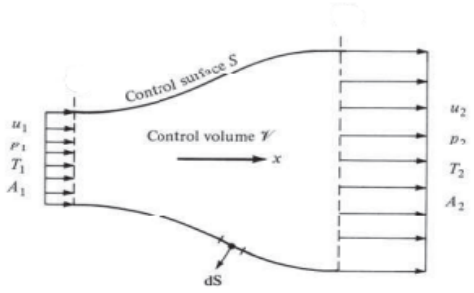


Figure 3.3: Finite control volume for quasi-one-dimensional flow [53].

Considering the nozzle flow control volume portrayed in figure 3.3, where the inflow and outflow are constrained to the nozzle's geometry, equation 3.22 applied to the control volume yields

$$\rho_1 A_1 u_1 = \rho_2 A_2 u_2 \, . \tag{3.43}$$

Regarding the momentum equation, one can simply make use of equation 3.30 in one dimension. Lastly, the energy equation for quasi-one-dimensional flows, like the continuity equation, can be deduced by applying equation 3.33 to the control volume

$$p_1 u_1 A_1 + \rho_1 u_1 A_1 \left(e_1 + \frac{u_1^2}{2} \right) = p_2 u_2 A_2 + \rho_2 u_2 A_2 \left(e_2 + \frac{u_2^2}{2} \right). \tag{3.44}$$

Dividing equation 3.44 by equation 3.43 and remembering that $h = e + p/\rho$ (equation 3.4) one obtains

$$h_1 + \frac{u_1^2}{2} = h_2 + \frac{u_2^2}{2} = \text{const} \, . \tag{3.45}$$

Differentiating equations 3.43, 3.30, and 3.45 one acquires the continuity, momentum, and energy equations for a steady, inviscid, adiabatic, and quasi-one-dimensional flow, respectively, as follows:

$$d(\rho \, u A) = 0 \, , \tag{3.46a}$$

$$dp = -\rho u \, du \, , \tag{3.46b}$$

$$dh + u \, du = 0 \, . \tag{3.46c}$$

3.2.3 Definition of Total Conditions

The energy equation, although many times disregarded, contains a lot of useful information regarding the flowfield properties of the nozzle flow. Equation 3.45 states that $h + V^2/2 = \text{const}$ along a streamline. A new

variable can be defined as the total enthalpy:

$$h_0 = h + \frac{V^2}{2} \,, \tag{3.47}$$

which is determined as that enthalpy that would exist at a point where the fluid element was brought to rest adiabatically. In the same way, total enthalpy is determined, one can define total temperature T_0, total pressure p_0 and total density ρ_0. In general, the index 0 indicates stagnation or total conditions. As an aggregate of equations 3.45 and 3.47 one obtains:

$$h_0 = \text{const.} \tag{3.48}$$

Assuming a calorically perfect gas, $h_0 = c_p T_0$, the total temperature is constant along a steady, inviscid, isentropic flow, that is

$$T_0 = \text{const}, \tag{3.49}$$

resulting in

$$c_p T_0 = c_p T + \frac{V^2}{2} \,. \tag{3.50}$$

From equation 3.50 and the isentropic relations 3.19 the total-to-static ratios of the flowfield variables are determined as a function of M and γ:

$$\frac{T_0}{T} = \left(1 + \frac{\gamma - 1}{2} M^2\right) \,, \tag{3.51a}$$

$$\frac{\rho_0}{\rho} = \left(1 + \frac{\gamma - 1}{2} M^2\right)^{1/(\gamma-1)} \,, \tag{3.51b}$$

$$\frac{p_0}{p} = \left(1 + \frac{\gamma - 1}{2} M^2\right)^{\gamma/(\gamma-1)} \,. \tag{3.51c}$$

3.2.4 Quasi-One-Dimensional Flow Properties

Using the set of equations 3.46, it is possible to obtain an equation that relates the change in velocity du to the change in area dA assuming isentropic flow. For simplicity's sake only the final equation is presented which deductions are referred to in [53]:

$$\frac{dA}{A} = (M^2 - 1)\,\frac{du}{u} \,. \tag{3.52}$$

Equation 3.52 is called the *area-velocity relation*. It highlights the fact that for a subsonic flow a decrease in area is associated with an increase in velocity since the term in parenthesis is negative. For a supersonic flow, however, an increase in the area results in a velocity increase, because the term changes the signal. This behavior is illustrated in figure 3.4.

Another important behavior of this function is that for a sonic flow, that is $M = 1$, equation 3.52 forces $dA = 0$, corresponding to a local maximum or minimum in the area distribution.

It is important to notice that to reach sonic flow, one must first accelerate the flow by decreasing the nozzle area. After achieving sonic conditions, the area can be increased in order to further increase the flow's velocity. Hence, a nozzle designed to achieve supersonic flow at its exit is a convergent-divergent duct. The minimum

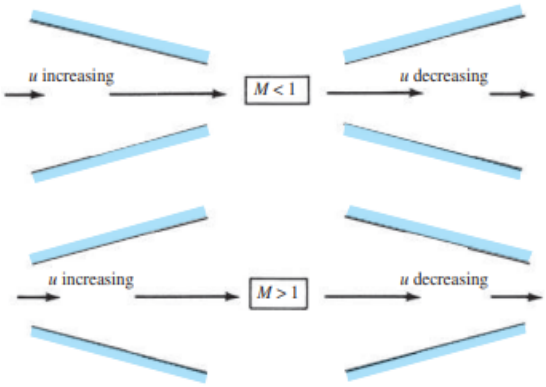


Figure 3.4: Compressible flow in converging and diverging ducts [53].

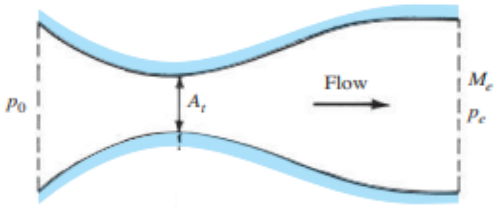


Figure 3.5: Illustration and comparison of a supersonic nozzle and a supersonic diffuser [53].

area, where sonic flow is achieved is called the *throat*. Another very important equation is the *area-Mach number relation*, which is deduced in [53]:

$$\left(\frac{A}{A_t}\right)^2 = \frac{1}{M^2}\left[\frac{2}{\gamma+1}\left(1+\frac{\gamma-1}{2}M^2\right)\right]^{(\gamma+1)/(\gamma-1)} . \tag{3.53}$$

This equation shows that the Mach at any point of the nozzle is a function of the ratio between the local and the throat area. Imagining a large tank, where the gas can be assumed stationary ($M \approx 0$), is located at the inlet of the nozzle, one can calculate all properties of the flow $p(x)$, $\rho(x)$, $T(x)$, and $M(x)$ knowing the area distribution $A(x)$ and applying equation 3.53 followed by the isentropic relations 3.51.

3.2.5 Convergent-Divergent Nozzle

A convergent-divergent nozzle, as spelt in the name, is made of a convergent duct followed by a divergent one as illustrated in figure 3.5. If $p_{amb} = p_0$, no pressure differential exists, hence no flow occurs inside the nozzle. Continuously increasing the pressure ratio $p_r = p_0/p_{amb}$ results in an increase in flow velocity. Upstream of the throat, the Mach number will increase until it reaches its maximum at the throat.

In case it does not reach a sonic state at the throat ($M < 1$), the flow will decelerate along the divergent section of the nozzle and the pressure increase until it reaches the ambient pressure at the exit section ($p_e = p_{amb}$) as shown in figure 3.6 in example a) and b).

For sufficiently low ambient pressure, the flow will accelerate at the converging duct until it reaches sonic behavior at the duct's throat. In this case, the flow will continue accelerating in the divergent sections, because it achieved a supersonic state. For supersonic behavior, however, a specific point in the flow is blind to everything downstream of it.

Three possible scenarios can occur from this instance: (i) the flow develops isentropic, without the occurrence of any shock waves (example f); (ii) the flow encounters a normal shock wave inside the nozzle (example c and d); or (iii) the flow develops similarly to the isentropic case inside the nozzle, however, it expands or compresses itself through oblique shocks or expansion fans after exiting the nozzle (example e and g).

The third possibility is the most common when working with rocket nozzles since it is very difficult to ensure that the ambient pressure equals the exit pressure when dealing with supersonic flows. In case the exit pressure is less than the ambient pressure, one considers the nozzle flow to be overexpanded, meaning oblique shock waves will be formed. On the contrary, if the exit pressure is larger than the ambient pressure, one observes an underexpanded flow characterized by the formation of expansion fans at the nozzles exit. These outcomes are shown in figure 3.7. The second scenario is discarded when mentioning rocket nozzles because the existence of a normal shock wave results in the flow becoming partially or fully subsonic.

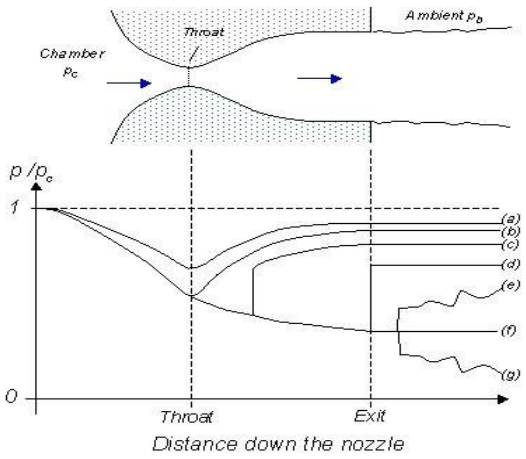


Figure 3.6: Pressure distribution along the nozzle for different pressure ratios p_0/p_{amb} [57].

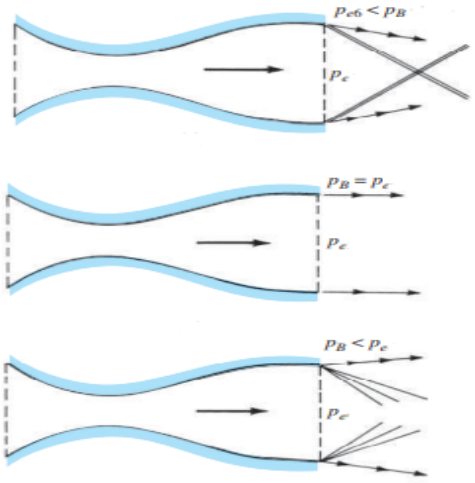


Figure 3.7: Supersonic nozzle flows with waves at nozzle exit for overexpanded, isentropic and underexpanded nozzle (up-to-down) [53].

3.2.6 The Optimum Nozzle

The thrust of a rocket nozzle is calculated as:

$$F = \dot{m}V_e + (p_e - p_{amb})A_e \ , \tag{3.54}$$

where p_e and p_{amb} are the exit and ambient pressures, respectively, and A_e the nozzle exit area. For a given mass flow $\dot{m}$ the following proportionality relation exists

$$V_e \propto \frac{A_e}{A_t} \ ,$$
$$\frac{1}{p_e} \propto \frac{A_e}{A_t} \ .$$

Knowing that both V_e and p_e contribute to the thrust and influence the area ratio oppositely, the main goal of this section is to find the optimal value of A_e/A_t that maximizes the thrust. In general, the subscripts e and t indicates exit and throat conditions, respectively. Proceeding with demonstration, the expression for the mass flow through a choked nozzle is presented in [53] as:

$$\dot{m} = \frac{p_0 A_t}{\sqrt{T_0}} \sqrt{\frac{\gamma}{R}\left(\frac{2}{\gamma+1}\right)^{(\gamma+1)/(\gamma-1)}} \,. \tag{3.55}$$

Inserting equation 3.55 into 3.54 and non-dimensionalizing it, one obtains the coefficient of thrust for a chocked throat:

$$C_T = \frac{F}{p_0 A_t} = \frac{V_e}{\sqrt{T_0}} \sqrt{\frac{\gamma}{R}\left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{(\gamma-1)}}} + \frac{A_e}{A_t}\frac{(p_e - p_{amb})}{p_0} \,. \tag{3.56}$$

For an isentropic flow reworking equation 3.50, one obtains:

$$V_e = \sqrt{2c_p\left[T_{0_e} - T_e\right]} = \sqrt{2c_p T_{0_e}}\left[1 - \frac{T_e}{T_{0_e}}\right]^{1/2} \,, \tag{3.57}$$

and applying equations 3.19 and 3.49 the coefficient of thrust reduced to:

$$C_T = \left[1 - \left(\frac{p_e}{p_0}\right)^{\frac{\gamma-1}{\gamma}}\right]^{1/2} \sqrt{\frac{2c_p\gamma}{R}\left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{(\gamma-1)}}} + \frac{A_e}{A_t}\frac{(p_e - p_{amb})}{p_0} \,. \tag{3.58}$$

This expression can be further developed. Finally, the necessary condition for maximum thrust is:

$$\frac{\partial C_T}{\partial p_e} = 0 \,, \tag{3.59}$$

resulting in the condition of optimality

$$p_e = p_{amb} \,. \tag{3.60}$$

The full demonstration is presented in [13].

3.3 Prandtl-Meyer Expansion Fans

Expansion fans occur when a supersonic flow is turned away from itself [53]. An expansion fan is a continuous expansion region that can be visualized as an infinite number of Mach waves, contrary to an oblique shock that is made of a single shock wave, a consequence of the flow turning into itself. Both phenomena can be visualized in figure 3.8.

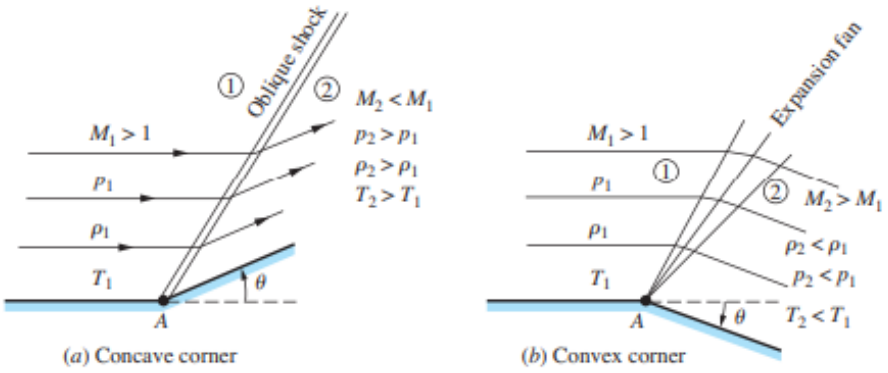


Figure 3.8: Supersonic flow over a sharp corner [53].

Since the expansion takes place across a continuous succession of Mach waves, defined as infinitely weak normal shock waves, the expansion is isentropic, contrary to oblique shocks, that experience entropy increase. An expansion wave appearing from a sharp convex corner is called a *centered expansion wave*. This kind of fan will be visualized during the implementation of the *Method of Characteristics* and is commonly addressed to as *Prandtl-Meyer expansion waves* in honor of Ludwig Prandtl and Theodor Meyer who first worked on a theory for centered expansion waves. From Prandtl-Meyer's theory comes an equation that relates the infinitesimal change in velocity dV to the infinitesimal deflection $d\theta$:

$$d\theta = \sqrt{M^2-1}\,\frac{dV}{V}\ . \tag{3.61}$$

Equation 3.61 can be integrated from region 1 (identified in figure 3.8 with 1), where the deflection angle is zero and Mach number M_1, to region 2 (identified in figure 3.8 with 2), where the deflection angle is θ_d and the Mach number is M_2:

$$\int_0^{\theta_d} d\theta = \theta_d = \int_{M_1}^{M_2} \sqrt{M^2-1}\frac{dV}{V}\ . \tag{3.62}$$

Solving the integral, one obtains

$$\theta_d = \nu(M_2) - \nu(M_1)\ , \tag{3.63}$$

where ν is called the *Prandtl-Meyer function*:

$$\nu(M) = \sqrt{\frac{\gamma+1}{\gamma-1}}\tan^{-1}\sqrt{\frac{\gamma-1}{\gamma+1}(M^2-1)} - \tan^{-1}\sqrt{M^2-1}\ . \tag{3.64}$$

3.3.1 Inverse Prandtl-Meyer Function

In several applications, it is necessary to obtain the value of M for a given value of ν. Hall [58] determined an expression expanding equation 3.64 as a series in powers of $\nu^{2/3}$ for a perfect gas with $\gamma = 1.4$ as follows:

$$M = \frac{1 + d_1 y + d_2 y^2 + d_3 y^3}{1 + e_1 y + e_2 y^2}\ , \tag{3.65a}$$

$$y = \left[\frac{\nu}{\nu_\infty}\right]^{2/3}\ , \tag{3.65b}$$

$$\nu_\infty = \frac{\pi}{2}\left[\frac{1}{\lambda} - 1\right], \tag{3.65c}$$

$$\lambda = \sqrt{\frac{\gamma-1}{\gamma+1}}\ , \tag{3.65d}$$

with $d_1 = 1.3604$, $d_2 = 0.0962$, $d_3 =$ -0.5127, $e_1 =$ -0.6722 and $e_2 =$ -0.3278.

3.4 Method of Characteristics in 2D

When dealing with supersonic flows the partial differential equations, i.e., the Euler Equations, that govern the flowfield properties ($u, v, p, T, \rho, etc.$) are hyperbolic. The main goal of this method is to simplify these

PDE (equation 3.40) into an ordinary differential equation (ODE). The deduction of this method based on [53] will be presented.

Before getting into the deduction of the Method of Characteristics itself, one has to define what a characteristic line is. Characteristic lines divide two adjacent regions described by expressions which are analytically different. In general, characteristic lines exist when the transition from one region to another involves discontinuities of some derivatives. This means that there are certain lines in the xy space along which the derivatives of the flow variables are indeterminate. These characteristic lines divide the region of the flow where the action is produced from the region of the flow that ignores the presence of the disturbances [59].

Since it is known that $\partial\phi/\partial x = u$ and $\partial\phi/\partial y = v$, equation 3.40 can be written as

$$\left(1-\frac{u^2}{a^2}\right)\frac{\partial^2\phi}{\partial x^2}+\left(1-\frac{v^2}{a^2}\right)\frac{\partial^2\phi}{\partial y^2}-\frac{2uv}{a^2}\,\frac{\partial^2\phi}{\partial x\partial y}=0\,. \tag{3.66}$$

Furthermore, from the relation for an exact differential

$$df=\frac{\partial f}{\partial x}\,dx+\frac{\partial f}{\partial y}\,dy,$$

one obtains

$$d\left(\frac{\partial\phi}{\partial x}\right)=du=\frac{\partial^2\phi}{\partial x^2}\,dx+\frac{\partial^2\phi}{\partial x\partial y}\,dy\,, \tag{3.67}$$

$$d\left(\frac{\partial\phi}{\partial y}\right)=dv=\frac{\partial^2\phi}{\partial x\partial y}\,dx+\frac{\partial^2\phi}{\partial y^2}\,dy\,. \tag{3.68}$$

Assuming these derivatives as unknowns and solving equations 3.66, 3.67, and 3.68 for $\partial\phi/\partial x\partial y$ using Cramer's rule [60], one obtains

$$\frac{\partial^2\phi}{\partial x\partial y}=\frac{\begin{vmatrix} 1-\frac{u^2}{a^2} & 0 & 1-\frac{v^2}{a^2} \\ dx & du & 0 \\ 0 & dv & dy \end{vmatrix}}{\begin{vmatrix} 1-\frac{u^2}{a^2} & -\frac{2uv}{a^2} & 1-\frac{v^2}{a^2} \\ dx & dy & 0 \\ 0 & dx & dy \end{vmatrix}}=\frac{N}{D}\,. \tag{3.69}$$

In equation 3.69, the differentials du and dv represent the changes in velocity along the increments dx and dy, respectively, as portrayed in figure 3.9. An important observation is that there exists some direction (dy/dx), along which D = 0. Since $\partial^2\phi/\partial x\partial y$ cannot be undefined, when the denominator equals zero so must the numerator. In conclusion, by choosing this specific direction one achieves indeterminacy proving the existence of lines where the flow fields derivatives are indeterminate. Equaling the denominator of equation 3.69 to zero, one has

$$\left(1-\frac{u^2}{a^2}\right)\left(\frac{dy}{dx}\right)^2+\frac{2uv}{a^2}\left(\frac{dy}{dx}\right)+\left(1-\frac{v^2}{a^2}\right)=0\,, \tag{3.70}$$

where dy/dx represents the slope of the characteristic lines. After applying the quadratic formula to equation

3.70 one obtains

$$\left(\frac{dy}{dx}\right) = \frac{-uv/a^2 \pm \sqrt{(u^2+v^2)/a^2 - 1}}{1 - u^2/a^2} \,. \tag{3.71}$$

With considerable algebraic and trigonometric manipulation equation 3.71 reduces to

$$\left(\frac{dy}{dx}\right)_{char} = tan(\theta \mp \mu) \,, \tag{3.72}$$

where

$$\mu = \sin^{-1}\left(\frac{1}{M}\right) \,, \tag{3.73}$$

and $u = Vcos(\theta)$ and $v = Vsin(\theta)$. When the flow is governed by a supersonic steady motion having constant entropy and constant total enthalpy, the characteristic surfaces are coincident with the envelope of Mach as shown by equation 3.72 and illustrated as dotted lines in figure 3.10 [59].

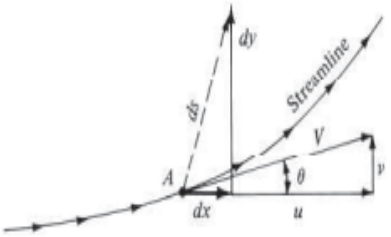


Figure 3.9: An arbitrary direction ds away from point A [53].

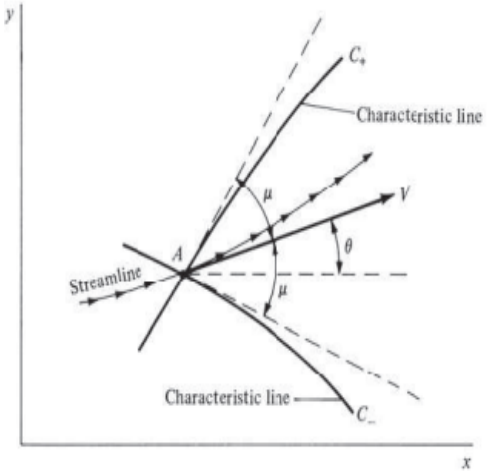


Figure 3.10: Left- and right-running characteristic lines through point A [53].

The characteristic lines are curved, however, because μ depends on the local Mach number and the local streamline direction θ varies along the flow. Regarding designation, the left- and right-running characteristics are denoted as C_+ and C_- respectively.

As previously said, N = 0 along the characteristic lines, resulting in the following equation:

$$\frac{dv}{du} = \frac{-(1 - u^2/a^2)}{1 - v^2/a^2}\frac{dy}{dx} \,. \tag{3.74}$$

Substituting equation 3.71 into 3.74 one obtains

$$\left(\frac{dv}{du}\right) = \frac{uv/a^2 \mp \sqrt{(u^2+v^2)/a^2 - 1}}{1 - v^2/a^2} \,, \tag{3.75}$$

which reduces to

$$d\theta = \mp\sqrt{M^2 - 1}\frac{dV}{V} \,. \tag{3.76}$$

As a consequence of making use of the characteristic lines, one can simplify the governing partial differential

equations (such as 3.40) into an ordinary differential equation (like 3.76), which was the ultimate goal of this method. However, the ordinary differential equation may only be applied along the characteristic lines. Equation 3.76 is identical to the expression used to describe Prandtl-Meyer expansion waves resulting in the following relations:

$$\theta + \nu(M) = const = K_- = KR \text{ (along the } C_- \text{ characteristic)} , \tag{3.77a}$$

$$\theta - \nu(M) = const = K_+ = KL \text{ (along the } C_+ \text{ characteristic)} , \tag{3.77b}$$

where $\nu(M)$ is called the *Prandtl-Meyer function* which is presented in equation 3.64.

3.4.1 Grid Generation

In order to simulate the supersonic flow inside a nozzle, it is necessary to generate grid points. There are two types of grid points: internal points and wall points.

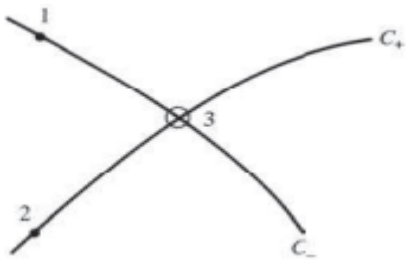


Figure 3.11: Internal point - MoC [53].

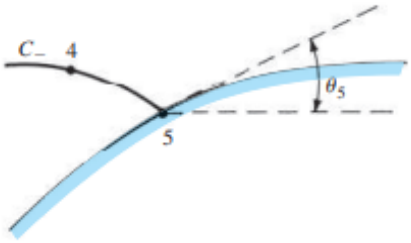


Figure 3.12: Wall point - MoC [53].

Internal Points

Consider the internal grid points 1, 2 and 3 as shown in figure 3.11. Assuming that the location, as well as, the flow properties at points 1 and 2 are known, one can obtain the intersection of both characteristics. Since the value of K_- and K_+ are constant along the characteristics (equation 3.77) the following equality is obtained:

$$\theta_3 + \nu_3 = (K_-)_3 = (K_-)_1 = \theta_1 + \nu_1 , \tag{3.78a}$$

$$\theta_3 - \nu_3 = (K_+)_3 = (K_+)_2 = \theta_2 - \nu_2 . \tag{3.78b}$$

Solving equation 3.78 one obtains

$$\theta_3 = \frac{1}{2}\left[(K_-)_1 + (K_+)_2\right] , \tag{3.79a}$$

$$\nu_3 = \frac{1}{2}\left[(K_-)_1 - (K_+)_2\right] . \tag{3.79b}$$

After obtaining ν_3, M_3 can be calculated using equation 3.65. From M_3 and the known total condition, it is possible to find p_3 and T_3 using the isentropic relations 3.19. Finally, V_3 is determined using equation 3.41 and

3.42. Regarding point 3 coordinates, assuming the characteristics as straight lines with slopes equal to

$$\text{slope}\{C_-\} = \frac{1}{2}(\theta_1 + \theta_3) - \frac{1}{2}(\mu_1 + \mu_3) \ , \tag{3.80a}$$

$$\text{slope}\{C_+\} = \frac{1}{2}(\theta_2 + \theta_3) + \frac{1}{2}(\mu_2 + \mu_3) \ , \tag{3.80b}$$

one obtains:

$$\frac{y_3 - y_1}{x_3 - x_1} = \tan[\text{slope}\{C_-\}] \ , \tag{3.81a}$$

$$\frac{y_3 - y_2}{x_3 - x_2} = \tan[\text{slope}\{C_+\}] \ . \tag{3.81b}$$

Adjusting equation 3.81 the coordinates of points 3 are calculated as [61]:

$$x_3 = -\frac{x_1 \cdot \tan[\text{ slope } \{C_-\}] - x_2 \cdot \tan[\text{ slope } \{C_+\}] + (y_2 - y_1)}{\tan[\text{ slope } \{C_-\}] - \tan[\text{ slope } \{C_+\}]} \ , \tag{3.82a}$$

$$y_3 = \frac{\tan[\text{slope}\{C_-\}] \cdot \tan[\text{ slope } \{C_+\}] \cdot (x_1 - x_2) + \tan[\text{slope}\{C_-\}] \cdot y_2 - \tan[\text{slope}\{C_+\}] \cdot y_1}{\tan[\text{ slope } \{C_-\}] - \tan[\text{slope}\{C_+\}]} \ . \tag{3.82b}$$

Wall Points

Considering points 4 and 5 presented in figure 3.12, the angle θ_5 is known. This way the value of ν_5 and $(K_+)_5$ can be directly obtained from equation 3.77

$$\nu_5 = \theta_4 + \nu_4 - \theta_5 \ , \tag{3.83}$$

$$(K_+)_5 = \theta_5 - \nu_5 \ . \tag{3.84}$$

The properties at the grid points are calculated from known properties at other grid points. Hence, in order to start a calculation using the Method of Characteristics, it is necessary to know the flow properties along some initial data line.

The MoC can also be applied in three dimensions, however, it presents a very complex implementation that is outside the scope of this thesis. A reference to get familiarized with the three-dimensional version of this method is [59].

3.5 Boundary Conditions

To properly model a flowfield around a given geometry, it is vital to define a domain and its boundaries, where the numerical solver will apply the equations of fluid dynamics. Figure 3.13 illustrates a domain $\Omega \subset \mathbb{R}^2$ and its boundary $\partial\Omega$.

Boundary conditions are a set of constraints applied to the boundaries of both the body and domain. Such restrictions impose one unique solution determined by the governing equations. Most boundary conditions can be divided into two types: Dirichlet and Neumann boundary conditions [62, 63]. Both types are commonly used during numerical computations.

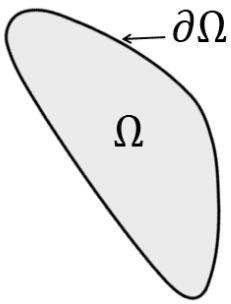


Figure 3.13: Domain and boundary [62].

Dirichlet Boundary Condition

Assuming a generic variable ϕ, Dirichlet boundary conditions, also referred to as first-type, specify the value of ϕ at the domain's boundary

$$\phi|_{\partial\Omega} = \phi_{\text{specified}} \ . \tag{3.85}$$

Since the solution is already known at the boundary, it is used as an input by the governing equations to compute the solution along the domain. Two widely used Dirichlet boundary conditions are the wall and the far-field condition.

Wall conditions themselves can be separated into multiple different variations. Since inviscid flow is assumed throughout the computed simulations presented in this thesis, the only wall condition taken into account is the Slip Boundary Condition, also known as Euler Condition. This condition erases the normal component of the velocity and keeps the tangential components untouched at the assigned surface [64]:

$$\mathbf{V}|_{\text{normal}} = 0 \ . \tag{3.86}$$

As for the far-field condition, the variables at the boundary acquire their values from the fixed free-stream condition. This condition is however formulated for external flows. When working with internal flows, like nozzle flows, the inlet ($\phi|_{\text{inlet}}$) and outlet ($\phi|_{\text{outlet}}$) variables are assumed by inflow (ϕ_{inflow}) and outflow (ϕ_{outflow}) condition, respectively, similar to the far-field condition:

$$\phi|_{\text{inlet}} = \phi_{\text{inflow}} \ \wedge \ \phi|_{\text{outlet}} = \phi_{\text{outflow}} \ . \tag{3.87}$$

Neumann Boundary Conditions

Contrary to the previous boundary conditions, where variables are directly stated, Neumann boundary conditions state that the derivative of ϕ must have a given value on the boundary [65]:

$$\left.\frac{\partial\phi}{\partial x}\right|_{\partial\Omega} = \left.\frac{\partial\phi}{\partial x}\right|_{specified} \ . \tag{3.88}$$

Two widely used Neumann boundary conditions are the Adiabatic Wall Condition and Symmetry Condition. Adiabatic, meaning not allowing heat to be transferred, is a straightforward condition that sets the heat flux to

zero

$$\dot{q}|_{\partial\Omega} = 0 \, . \tag{3.89}$$

The symmetry condition, very useful to reduce the computational price of simulations, defines a mirror surface. One of its conditions states that the flux of every variable across this surface is zero

$$\frac{d\phi}{dn} = 0 \, . \tag{3.90}$$

This condition is highly used in rocket nozzle simulations since most nozzles have an axis of symmetry or a symmetry surface.

3.6 The Finite Volume Method

Similarly to other numerical methods, the finite volume method (FVM) transforms a set of partial differential equations into a system of linear algebraic equations. Instead of discretizing the PDE like the finite difference methods (FDM), the finite volume method uses the integral form of the governing equations. This requires changing surface and volume integrals into discrete algebraic relations over elements and their surfaces resulting in a set of semi-discretized equations. Thereafter, interpolation profiles are chosen to approximate the variation of the variables within the element and relate the surface values of the variables to their cell values and thus obtaining algebraic equations. The advantage of this approach is that it manages well irregularly shaped domains and complex geometry, and the integral form of the equations handles discontinuities like normal shock waves much better [63, 66].

3.7 Parameterization Techniques

Parameterization is a mathematical process consisting of expressing a geometry as a function of some independent quantities called parameters. It has shown to be a very useful tool in design optimization since its main goal consists in exploring as many design solutions as possible while keeping the number of design variables to a minimum [67]. The effective operation of an optimization model is considerably influenced by the parameterization technique chosen to represent an object geometry influencing its computational time cost as well as the quality of the results. Some popular parameterization techniques are described next.

3.7.1 Polynomial and Spline Approach

The use of polynomial and spline representation for shape optimization is a useful tool to reduce the total number of design variables compared with a discrete model. The polynomial form is a compact and powerful representation for shape optimization [47]. Nevertheless, it is only suited for simple curves, since complex structures require high-degree polynomials, which greatly increases the number of parameters required.

Bézier Curve

Bézier curves are widely used in computer graphics, often to produce smooth curves, and yet they are a very simple tool. This parameterization technique takes advantage of the Bernstein polynomial and De Castaljeu algorithm in order to write a Bézier curve by defining a set of control points [68]. Defined by a set of control points $\mathbf{P} = [P_0, P_1, ..., P_n]$, the curve order is equal to the number of control points minus one. In case of wanting to obtain a quadratic curve, given the control points P_0, P_1, and P_2, a segment of a parabola is formed when varying t from 0 to 1:

$$\mathbf{B}(t) = (1-t)^2\mathbf{P}_0 + 2t(1-t)\mathbf{P}_1 + t^2\mathbf{P}_2, \quad 0 \leq t \leq 1\ . \tag{3.91}$$

In general, a Bézier curve of degree n is defined by as n+1 control points as:

$$\mathbf{B}(t) = \sum_{\mathrm{i}=0}^{n} B_{\mathrm{i}}^{n}(t) \cdot \mathbf{P}_{\mathrm{i}}, \quad \mathrm{i} = 0, 1, \ldots, n\ ; \tag{3.92}$$

where $B_i^n(t)$ describe the Bernstein polynomials of degree n

$$B_i^n(t) = \begin{pmatrix} n \\ \mathrm{i} \end{pmatrix} t^{\mathrm{i}}(1-t)^{n-1}. \tag{3.93}$$

Binomial coefficients are defined as

$$\begin{pmatrix} n \\ i \end{pmatrix} = \frac{n!}{i!(n-i)!} \tag{3.94}$$

To summarise, Bézier curves are easy to set up and allow rotation and translation by moving the control points. One major disadvantage is that no base geometry can be used for this type of parameterization because although any possible shape can be achieved using Bézier curves it is not possible to know the number and the location of each control point *a priori*. Reference [69] tries to tackle this problem.

Another downside to Bézier curves is their capacity to only accurately represent simple curves without drastically needing to increase their dimensions. In order to represent a complex curve, it is more efficient to separate the curve into segments and use a set of low-order Bézier curves instead [70].

B-Splines

The last statement refers to a spline. B-splines to be precise, are constructed based on a set of basic functions and a set of control points. Multi-segmented B-spline curves with n control points and a node vector $\mathbf{v}_k$ such that $x_1 \leq x_2 \leq \ldots \leq x_{n+k}$ can be defined as [71]:

$$C_{k,n}(t) = \sum_{i=1}^{n} \mathbf{P}_i \cdot \mathcal{N}_i^k(t), \tag{3.95}$$

where $\mathbf{P}_i$ are the control points and $\mathcal{N}_i^k(t)$ is the ith basis function of degree k. Setting $\mathcal{N}_i^1(t) = 1$ if $x_i \leq t < x_{i+1}$ and zero otherwise and assuming $k = p + 1$ one gets

$$\mathcal{N}_i^k = \mathcal{N}_i^{p+1}(t) = \frac{t - x_i}{x_{i+p} - x_i}\mathcal{N}_i^p(t) + \frac{x_{i+p+1} - t}{x_{i+p+1} - x_{i+1}}\mathcal{N}_{i+1}^p(t)\,. \tag{3.96}$$

More complex parameterization methods can be developed from the method like rational B-Splines or non-uniform rational B-splines (NURBS) where the control points are weighted and the nodes are not evenly spaced apart.

Ferguson's Spline

First introduced in 1964 Ferguson's Parametric Splines' basic logic is very similar to the previous techniques. A Ferguson's spline is a curve connecting two control points (**A**,**B**) where a tangent has been imposed to the curve's extremities ($\mathbf{T}_A$, $\mathbf{T}_B$), as displayed in figure 3.14. The curve $S_{Fer}(t)$ is defined as a cubic polynomial

$$S_{Fer}(t) = \sum_{i=0}^{3} \mathbf{a}_i t^i, t \in [0,1]\,, \tag{3.97}$$

where the coefficient is calculated as

$$\begin{aligned} \mathbf{a}_0 &= \mathbf{A}\,, \\ \mathbf{a}_1 &= \mathbf{T}_A\,, \\ \mathbf{a}_2 &= 3[\mathbf{B} - \mathbf{A}] - 2\mathbf{T}_A - \mathbf{T}_A\,, \\ \mathbf{a}_3 &= 2[\mathbf{A} - \mathbf{B}] + \mathbf{T}_A + \mathbf{T}_B\,. \end{aligned} \tag{3.98}$$

By substituting the coefficients in equation 3.97, the equation is rewritten as

$$S_{Fer}(t) = \mathbf{A}\left(1 - 3t^2 + 2t^3\right) + \mathbf{B}\left(3t^2 - 2t^3\right) + \mathbf{T}_A\left(t - 2t^2 + t^3\right) + \mathbf{T}_B\left(-t^2 + t^3\right)\,. \tag{3.99}$$

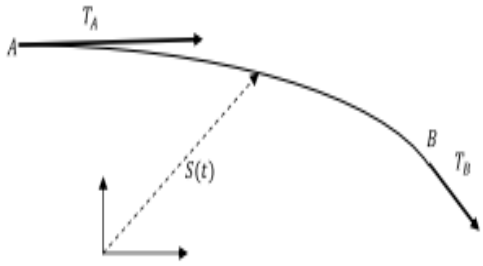


Figure 3.14: Ferguson spline and its boundary conditions [72].

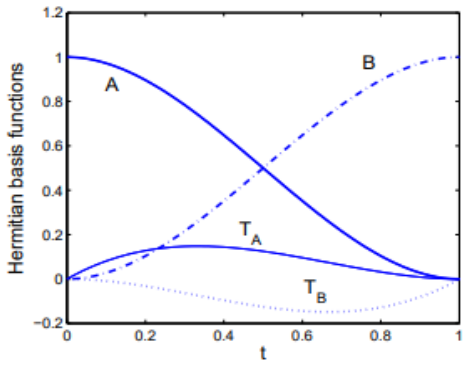


Figure 3.15: The four basis functions of equation 3.99 are shown with their respective multipliers [72].

$S_{Fer}(t)$ can be considered a Hermitian interpolant and the bracketed factors in equation 3.99 are its basis functions [72]. Figure 3.15 gives an intuitive understanding of the basis function's effect on the shape of the interpolant. The principal difference between Bézier curves and Ferguson's Splines parameterization techniques

is that one is controlled by changing the control points' position, while the other is controlled by changing the tangents of the endpoints directly.

This parameterization method, although first introduced for airfoil shape control, demonstrates good utility for rocket nozzle design because of its capability to impose derivatives. On the other hand, having the ability to regulate only the position and tangent of two control points results in low controllability.

3.7.2 Hicks-Henne "bump" Functions

Hicks and Henne introduced in 1978 a compact formulation of the parameterization of airfoil sections [73]. This parameterization technique is based on applying a perturbation to a base contour. This perturbation is characterized by a linear superposition of "bumps" or analytical shape functions that are defined as

$$f_{Hicks}(t) = \left[\sin\left(\pi t^{\frac{\ln(0.5)}{\ln(h_i)}}\right)\right]^{w_i} , \tag{3.100}$$

where h_i is the position of the maximum point of the function and w_i controls the width of the bump.

The parameters contribute by determining the value of the coefficients α_i associated with the shape function [70]. This way, the parameterized contour y is calculated as:

$$y = y_{\text{base}} + \sum_{i=1}^{m} a_h f_{Hicks}(t), \quad 0 \leq t \leq 1 . \tag{3.101}$$

If all coefficients a_h are set to zero the result will be the base geometry. This method has been demonstrated to be both useful and versatile in dealing with numerical optimization in wing design.

Unfortunately, this is not the case for rocket nozzle. One of the most influential characteristics to improve thrust in a rocket nozzle is its exit area and this method does not allow the extremities of the base geometry to be altered since $f_{Hicks}(t)|_{t=0} = f_{Hicks}(t)|_{t=1} = 0$.

3.7.3 Free Form Deformation

The idea behind Free Form Deformation is to enclose a geometry into a parallelepiped of control points that define the parametric domain. A physical analogy for FFD would be to imagine a parallelepiped of clear, flexible plastic undergoing deformation [49]. FFD can be based on different principles like Sederber's scheme based on Bernstein's polynomials or NURBS FFD that is more complex and outside the scope of this thesis [74]. To apply Sederber's technique every point of the base geometry has to be converted into the coordinate system (x, y, z) established by the parallelepiped region

$$\mathbf{X} = \mathbf{X}_0 + x\mathbf{S} + y\mathbf{T} + z\mathbf{U}, \quad 0 \leq x, y, z \leq 1 . \tag{3.102}$$

Let $P_{ijk}, i = 0, ..., l, j = 0, ..., m, k = 0, ..., n$, represent the movement of the control points from their latticed position, one computes the new points $\mathbf{X}_{new}$ by

$$\mathbf{X}_{new} = \mathbf{X} + \mathbf{X}_{def} , \tag{3.103}$$

where

$$\mathbf{X}_{def} = \sum_{i=0}^{l} \binom{l}{i} (1-x)^{l-i} x^i \left[\sum_{j=0}^{m} \binom{m}{j} (1-y)^{m-j} y^j \left(\sum_{k=0}^{n} \binom{n}{k} (1-z)^{n-k} z^k \mathbf{P}_{ijk} \right) \right] . \tag{3.104}$$

The algorithm for Sederberg's scheme is easy to program and the pseudocode can be built from [74] as shown by algorithm 1.

Algorithm 1 Free Form Deformation Algorithm

Input: Parallelepiped points (P_{ijk}), parameters (x,y,z)
Output New coordinates of the point **X** for parameters (x,y,z)

```
for i = 0 to l do
  for j = 0 to m do
    for k = 0 to n do
      X = X + binom(l, i) · pow(1 − x, l − i) · pow(x, i)·
      binom(m − 1, j) · pow(1 − y, m − j) · pow(y, j)·
      binom(n, k) · pow(1 − z, n − k) · pow(z, k) · P_ijk
    end for
  end for
end for
```

The FFD parameterization method is a powerful tool because it can deform almost any type of geometrical model. It provides a simple and intuitive method that allows deforming arbitrary shapes with a minimal number of control variables [67]. The method can be applied on the totality of the domain or locally, guarantying derivative continuity (C^1) with the adjacent, undeformed parts of the model. The FFD technique provides also control over surface continuity as well as volume change [49].

3.8 Design Optimization

Design optimization deals with the improvement of a structure's properties by improving its shape. In the specific case of a rocket nozzle properties such as the coefficient of thrust and the specific impulse can be analyzed to improve its performance. Optimization is often confused with the term improvement. Mathematically speaking, it means finding the best possible solution by changing control variables, often subjected to constraints [75].

3.8.1 Basic Concepts

Being methodical in the formulation of an optimization problem is crucial since the optimizer is blind to any contradiction and tends to exploit any weaknesses in the problem's formulation. In order to properly set up an optimization problem, one has to be familiar with some basic concepts.

Objective Function

To find the optimal design, it is necessary to define an objective function whose goal is to differentiate better from worse designs. Properties like the coefficient of thrust and the specific impulse can be quantified using an objective function, where different design variables result in different outcomes. Objective functions

can be implemented in various ways. In this work, both the Method of Characteristics and CFD are used to compute the objective functions of interest.

Design variables

The variables that describe the optimizer system are called *design variables* and are represented by a column vector:

$$\mathbf{x} = [x_1, x_2, ..., x_{n_d}] \,, \tag{3.105}$$

where n_d determined the problem's dimensionality. The design variables are independent of each other and the optimizer can choose any element of $\mathbf{x}$ freely. It is through these variables that the solver evaluates the objective function.

Constraints

Most engineering applications are constrained problems. Constraints are used to restrict possible solutions to a feasible region. The constraints can limits the design values directly, denominated bounds, or indirectly limit the results through equality and/or inequality constraints [76]. A typical constrained optimization problem can be formulated as:

$$\min_x f(\mathbf{x}) \text{ such that } \begin{cases} \mathrm{A} \cdot \mathbf{x} \leq \mathrm{b} \\ \mathrm{A}_{eq} \cdot \mathbf{x} = \mathrm{b}_{eq} \\ \mathrm{lb} \leq \mathbf{x} \leq \mathrm{ub}, \end{cases} \tag{3.106}$$

where $\mathbf{x}$ is the vector of design variables, f is the objective function and lb, ub are the lower and upper boundaries of the design variables, respectively. A,b and $\mathrm{A}_{eq}, \mathrm{b}_{eq}$ are linear inequality and equality constraints, respectively.

The main goal of an optimization algorithm is to minimize or maximize its objective function. It is conventional for an optimizer to minimize its objective function, however, maximization can be achieved in two separate ways: Finding the minimum of $f(x)^{-1}$ and consequently inverting it

$$\max[f(\mathbf{x})] = (\min[f(\mathbf{x})^{-1}])^{-1} \,, \tag{3.107}$$

or finding the minimum of the negative of f and then changing the signal

$$\max[f(\mathbf{x})] = -\min[-f(\mathbf{x})] \,. \tag{3.108}$$

A possible optimization setup for a rocket nozzle design could be to maximize thrust and minimize drag while constraining any kind of shock formation in the inflow.

3.9 Optimization Algorithms

The process of optimization consists in iteratively guessing the optimal value of the design variables until a solution is obtained. No single optimization algorithm is effective for all optimization problems, since each different algorithm has a different strategy to move between consecutive iterations [75]. A common classification to characterize optimization algorithms is by dividing them into gradient-free and gradient-based algorithms.

3.9.1 Gradient-free

The gradient-free optimization approach is solely based on moving from one point to another if the value of the objective function decreases. A gradient-free algorithm is easier to set up because no additional coding is needed other than the objective function and its constraints. On the other hand, gradient-free methods require a lot more objective function evaluations than gradient-based when the number of design variables increases as depicted in figure 3.16.

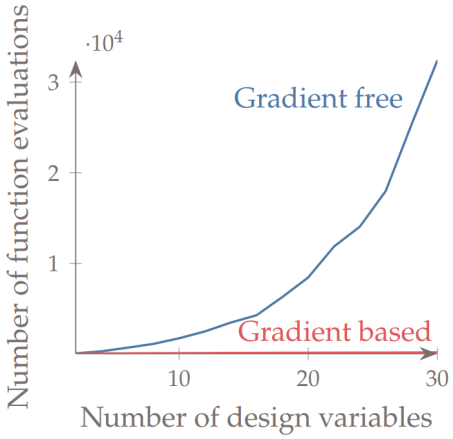


Figure 3.16: Computational cost of gradient-based vs. gradient-free optimization algorithms [75]

3.9.2 Gradient-based

Gradient-based algorithms take advantage of both the objective function and its derivative with respect to the design variables to converge to the optimum more efficiently. Gradients are used to ensure that the optimizer converges to the mathematical optimality condition. Though they usually need fewer iterations, gradient-based algorithms require the objective function to be sufficiently smooth. In practice, however, they can tolerate discontinuities as long as they are not near the optimum [75].

3.9.3 Local vs. Global Search

There are many ways to search the design space as shown before. These can be labeled as being local or global. A local search is characterized by a single starting point that forms a sequence that converges to a local optimum. A global search, however, tries to scan the whole design space to find the global optimum. It is not possible however to guarantee that the global optimum is the one that is found. Although local and

global search is usually matched up with gradient-based and gradient-free methods, respectively, these are independent attributes since they can be mix-matched [75].

3.9.4 Mathematical vs. Heuristic Algorithms

Another attribute of optimization algorithms is whether an algorithm is based on probable mathematical principles or heuristics. Gradient-based algorithms are usually based on mathematical principles, whereas gradient-free are more evenly split [75].

Heuristic-based optimality is flawed in the sense that it is only expected to find a close enough solution to a local optimum because it tries to mimic some natural behavior and uses a certain amount of randomness. Nonetheless, in the practical engineering world, it is considered a desirable outcome to find a better solution than the previous one, even if it is not optimal, meaning that the heuristic method should not be discarded. Heuristic methods are generally used in problems where the gradients cannot be calculated or with different local optima. Algorithms based on mathematical principles, on the other hand, converge to the optimum within the limits of the working precision [75].

3.10 Multiobjective Optimization

Multiobjective optimization helps explore the trade-offs between different metrics. Compared to the typical optimization problem 3.106, the only change occurs in the objective description that now has more than one objective:

$$\underset{x}{\text{minimize}} \quad f(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_{nf}(x) \end{bmatrix}, \text{ where } \quad n_f \geq 2 \,. \tag{3.109}$$

3.10.1 Pareto Optimality

When dealing with multiple objectives, one has to redefine the meaning of optimal. To this end, the concept of Pareto optimality is introduced. One design point is said to dominate another in case it is superior in all objectives. In the same way, a point is called *nondominated* if no other evaluated point dominates it. In figure 3.17, the region in the shaded rectangle highlights the points that are dominated by design A. In other words, B is dominated by A and C is non-dominated.

A point is considered Pareto optimal if it is nondominated by any point in the entire domain [75]. The collection of all Pareto optimal points forms the Pareto set. When observed for two different objectives, the Pareto set resembles a hyperbola, as visualized in figure 3.18.

3.11 Surrogate-Based Optimization

A surrogate model consists of an approximation of a functional output that represents a curve fit to some data. The main purpose of a surrogate-based optimization is to perform optimization using a model that is

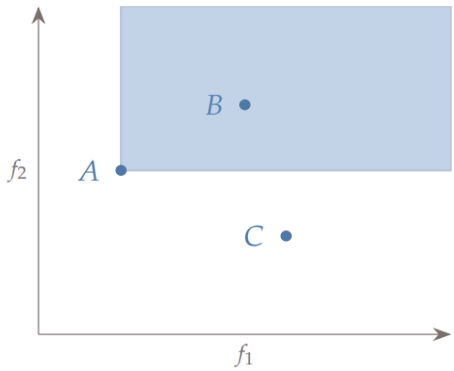


Figure 3.17: Three designs, A, B, and C, plotted against two objectives, f_1 and f_2 [75].

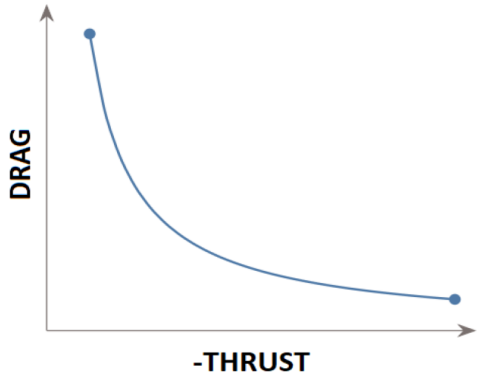


Figure 3.18: A notional Pareto front representing thrust and drag trade-offs for a rocket nozzle design optimization problem. [75].

much faster to compute than the original function, without losing substantial accuracy [75]. Surrogate models are a computationally cheap and easy alternative for models which are computationally expensive, such as high-fidelity CFD simulations.

In addition, they are also helpful to visualize how the objective function varies with respect to the design variables. On the contrary, surrogate models display poor scalability, or in other words, the larger the number of inputs, the more model evaluations are needed to construct a surrogate model that is accurate enough.

3.11.1 Sampling

To construct a surrogate model, one first needs to evaluate several selected points. This selection follows a *sampling plan*. Some sampling methods are presented next.

Full Factorial

One of the more straightforward approaches is to discretize each dimension and evaluate all possible combinations. This method is not efficient since it scales exponentially with the number of input variables.

Random Sampling

As the name indicates, random sampling randomly generates points across all dimensions, where each sample is independent of past samples. This sampling technique displays several clustering of points, resulting in some parts of the design space not being properly represented. This behavior can be observed in figure 3.19.

Latin Hypercube Sampling

This popular sampling method, also built on a random process, gets rid of the clustering of points by maximizing the distance between the samples. Latin Square is a concept based on one and only one point existing in any given row or column. Latin Hypercube is a generalization of the Latin Square for higher dimensions. LHS can be formulated as an optimization problem where the goal is to maximize the distance

between samples constrained to the projection on each axis following a chosen probability distribution, often uniform. More on the Latin hypercube sampling technique can be read about in [77].

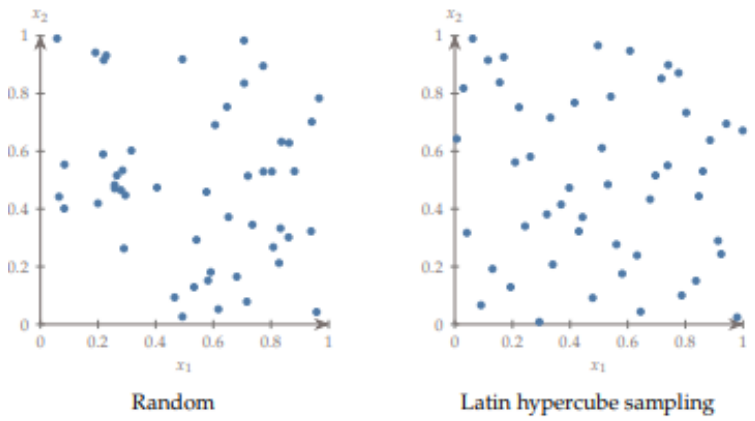


Figure 3.19: Contrast between random and Latin hypercube sampling [75]

3.11.2 Surrogate Construction

Once sampling is completed, one obtains the so-called *training data*

$$\left(x^{(i)}, f^{(i)}\right), \tag{3.110}$$

where $x^{(i)}$ is the input variable vector from the sampling and $f^{(i)}$ contains the outputs directly from evaluating the model. The main objective is to construct the surrogate model out of this data set. This is achieved through interpolation or regression. While interpolation generates a function that exactly matches the sampled data set regardless of its smoothness, regression models minimize the error between a smooth trend and the training data, as shown in figure 3.20. Regression is advantageous when dealing with noisy data, since interpolatory models may produce undesired oscillations when fitting the noise. Nonetheless, interpolation is a better choice for multimodal data, not exposed to noise, because regression may smooth over variations that are actually physical [75].

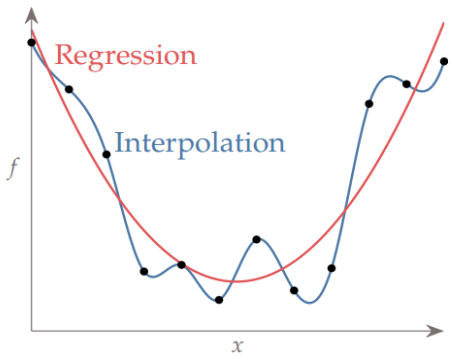


Figure 3.20: Interpolation versus Regression Models [75].

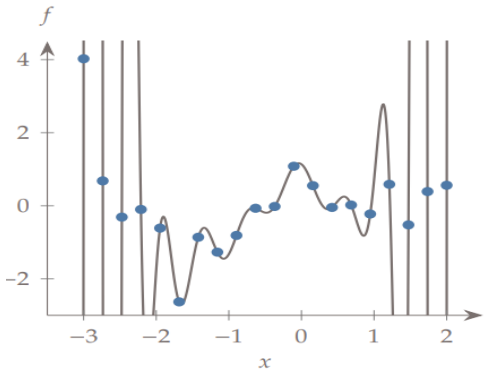


Figure 3.21: A 19th-order polynomial fit to the data with low error but poor predictability. [75].

3.11.3 Validation

Validation is a crucial step to ensure the proper functioning of the surrogate model. After performing the sampling, some points are not included in the training data. This set of points forms the testing data, which is used during the validation to show if there are big discrepancies between the surrogate and the original model. This type of validation is referred to as *simple cross-validation*.

One of the main reasons to perform validation is to prevent overfitting, which occurs when there are too many degrees of freedom and closely fitting a given set of data results in the model having a poor predictive ability. This can occur when using high-order polynomial fitting to reduce fitting error discarding the fact that the model may perform poorly in its predictions. Overfitting for a one-dimensional surrogate model can be visualized in figure 3.21.

Underfitting, as well as overfitting, has a poor predictive ability. It consists in not having enough degrees of freedom to create a useful model, like using linear regression for a multimodal dataset.

Chapter 4

Implementation

The following chapter will introduce two different methods to simulate a flowfield inside a nozzle (Method of Characteristics in MATLAB® and Computational Fluid Dynamics) and the respective optimization techniques applied to each method. The chapter is structured in the chronological order in which the work was developed.

4.1 Implementation of the Method of Characteristics in MATLAB®

In order to apply the Method of Characteristics to a rocket nozzle flow, the program MATLAB® was used to implement the equations deduced in section 3.4.

4.1.1 Minimum-Length Nozzle Design

Supersonic nozzles can be divided into two different types: gradual-expansion nozzles, and minimum-length nozzles. Gradual expansion nozzles are typically used when maintaining a high-quality flow at the exit is desired, like for supersonic wind tunnels. However, due to being lengthy and heavy, a minimum-length nozzle, which utilizes a sharp corner at the throat (as shown in figure 4.1), is a better option for rocket nozzle [78].

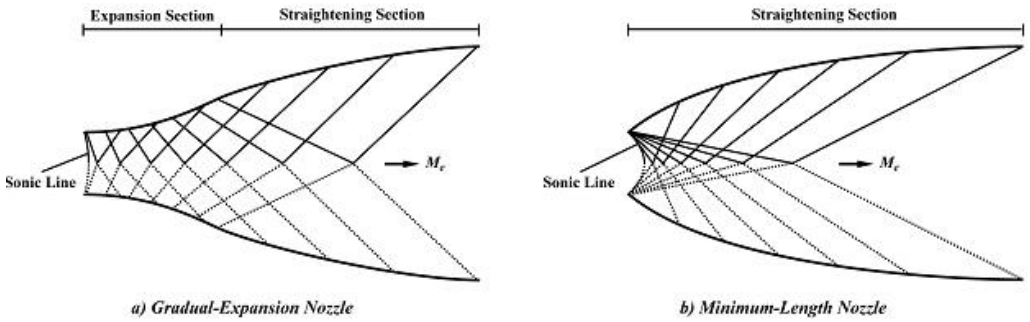


Figure 4.1: Gradual-expansion nozzle versus minimum-length nozzle [78].

If the nozzle contour is not properly shaped, shock waves can occur inside the duct. The Method of Characteristics provides a technique for properly designing the contour of a supersonic nozzle for shock-free, isentropic flow, taking into account the multidimensional flow inside the duct.

An important piece of information to know when visualizing characteristic grid is that if the characteristics are veering away from each other the flow is expanding, whereas if the characteristics are coming together the flow is compressing. In the case that the characteristics collapse into each other a shock wave occurs. In order to guarantee a shock-free flow, a constraint is implemented to the algorithm prohibiting the collapse of the characteristic lines.

As aforementioned, for a minimal nozzle length design, the expansion section is shrunk to a point (see section 3.3), and the expansion takes place through a centered Prandtl-Meyer wave emanating from a sharp-corner throat with an angle $\theta_{w_{max}}$ as demonstrated in figure 4.2.

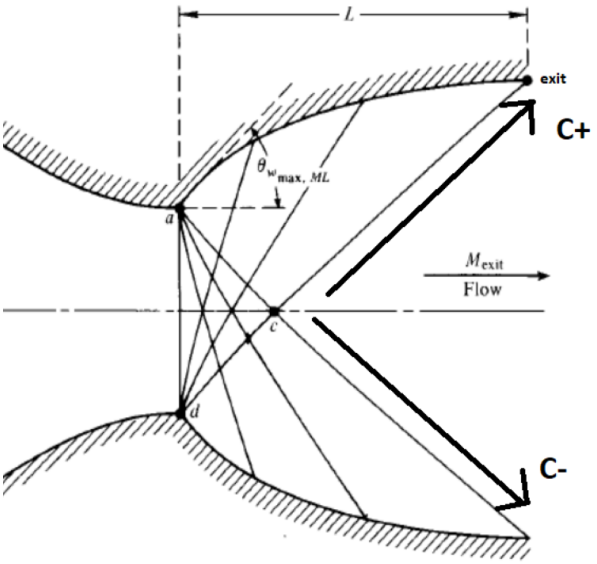


Figure 4.2: Minimum length Supersonic Nozzle Design [79].

A relation between the exit Mach number M_e and the maximum wall angle $\theta_{w_{max}}$ can be calculated [79]. It is known from equation 3.78 that along C_+

$$\theta_c - \nu_c = (K_+)_c = \theta_e - \nu_e \ .$$

Since point c is located at the centerline which is an axis of symmetry $\theta_c = 0$, resulting in

$$\nu_c = \nu_e - \theta_e \ .$$

Similarly along C_-

$$\nu_c = \nu_e + \theta_e \ .$$

Adding both expressions one realizes that

$$\nu_c = \nu_e \ .$$

Finally, repeating the process along C_- at point a

$$\theta_{w_{max}} + \nu_a = \nu_c = \nu_e \ .$$

Applying equation 3.63 at point a it is shown that

$$\theta_{w_{max}} = \nu_a - \nu|_{M=1} = \nu_a \ .$$

Adding the previous two equalities, the criterion for minimum nozzle length is determined:

$$\theta_{w_{max}} = \frac{\nu(M_e)}{2} \ . \tag{4.1}$$

Another important aspect regarding the minimum nozzle length is that its contour is made to absorb expansion waves instead of reflecting them. This means that the flow near the wall will neither expand nor compress meaning its direction angle θ stays equal according to equation 3.76. The conditions for no compressibility is

$$\theta_{\text{wall}} = \theta_{\text{adj}} \qquad (\text{along } K_{-}) \ . \tag{4.2}$$

As mentioned before, the Method of Characteristics needs some initial data to calculate the flow properties. From the desired exit Mach number M_e using equation 4.1 the angle at the starting point is obtained. Since all characteristics come up from the same point and have to be initiated with a different initiation angle θ_i^{ini} in order to display different paths, one computes the initiation angle vector θ^{ini} as a distribution with the intervals $[0,\ \theta_{w_{max}}]$. This distribution can present multiple behaviors, from uniform to quadratic to sinusoidal. During the implementation of the MoC, the sinusoidal distribution method is used. Algorithm 2 is responsible for this task.

Algorithm 2 Initialization Method of Characteristics

Input: Mach at Exit (M_e), Number of characteristics (n)
Output: Initiation Angle Vector θ^{ini}

1: $\theta_{w_{max}} = \nu(M_e)/2$
2: $x \leftarrow 0$
3: **for** $i = 1$ to n **do**
4: $\quad \theta_i^{ini} = \theta_{w_{max}} \cdot (1 - \cos(\text{x}))$
$\quad \text{x} \leftarrow \text{x} + 90/(\text{n} - 1)$

After obtaining the initial data, the Minimum-length Nozzle Algorithm (see algorithm 3) can be applied, where the first left-running characteristic is computed from the initial data. The MATLAB® code can be visualized in Appendix A.

After obtaining the output, it is possible to calculate the nozzle contour using equations 3.65, 3.73, and 3.82 enumerated in chapter 3.

4.1.2 Arbitrary Nozzle Design

In the last section, the Method of Characteristics was used to define a contour. In this section, the main goal is to apply the MoC on preexisting contours (conical and bell nozzles) and calculate its flowfield properties in order to calculate its coefficient of thrust, which will later be used during the optimization process.

Algorithm 3 Minimum-Length Nozzle using Method of Characteristics

Input: Initiation Angle Vector (θ^{ini}) , Number of characteristics (n)
Output: $\theta, \nu, \mathrm{KR}, \mathrm{KL}$

1: $i = 1$
2: **for** l=1 to n **do**
3: $\theta_i = 0$ {Symmetry Condition}
4: **if** $l = 1$ **then**
5: $\nu_i = \theta_i^{ini}$
6: **else**
7: $\nu_i = \mathrm{KR}_{i-n}$
8: **end if**
9: $\mathrm{KL}_i = \theta_i - \nu_i$
10: $\mathrm{KR}_i = \theta_i + \nu_i$
11: $i \leftarrow i + 1$
12: **for** $j = 1$ to $n-1$ **do**
13: $\mathrm{KL}_i = \mathrm{KL}_{i-1}$ {See Equation 3.77a}
14: **if** l=1 **then**
15: $\mathrm{KR}_i = 2 \cdot \theta_i^{ini}$
16: **else**
17: $\mathrm{KR}_i = \mathrm{KR}_{i-n}$ {See Equation 3.77b}
18: **end if**
19: $\theta_i = 0.5 \cdot (\mathrm{KR}_i + \mathrm{KL}_i)$ {See Equation 3.78}
20: $\nu_i = 0.5 \cdot (\mathrm{KR}_i - \mathrm{KL}_i)$
21: $i \leftarrow i + 1$
22: **end for**
23: $\theta_i = \theta_{i-1}$ {See Equation 4.2}
24: $\nu_i = \nu_{i-1}$
25: $\mathrm{KL}_i = \mathrm{KL}_{i-1}$
26: $\mathrm{KR}_i = \mathrm{KR}_{i-1}$
27: $i \leftarrow i + 1$
28: **end for**

Conical Nozzle Design

The simplest possible geometry of a rocket nozzle is a conical nozzle. In order to use the Method of Characteristics for a conical nozzle it is necessary to make some small changes to the previous algorithm. First of all, the desired Mach number at the exit is no longer taken into consideration since the geometry of the nozzle rules over the flowfield variables. Secondly, the computation of the flowfield variables at the wall change, because in order to comply with the Euler boundary condition (see section 3.5):

$$\theta_{wall} = \theta_{contour} = \alpha_{half} \ . \tag{4.3}$$

The algorithm for a conical nozzle is similar to algorithm 3 but changes slightly between lines 23 to 27, as shown in algorithm 4.

Bell Nozzle Design

The Bell nozzle is the most common type of nozzle. Similarly to conical nozzles, the flow angle at the wall has to obey the Euler condition. However, in the case of a bell nozzle, the slope of the contour is constantly

Algorithm 4 Conical Nozzle using Method of Characteristics

Input: Divergent Conical Nozzle Half-Angle (α_{half})
Output: $\theta, \nu, \text{KR}, \text{KL}$
Swap Lines 23-27 from Algorithm 3

1: $\theta_i = \alpha_{half}$ {See Equation 4.3}
2: $\nu_i = \theta_i - \text{KL}_{i-1}$
3: $\text{KL}_i = \text{KL}_{i-1}$
4: $\text{KR}_i = \theta_i + \nu_i$
5: $i \leftarrow i + 1$

changing. This increases the difficulty of the problem since the intersection of the characteristics with the contour becomes an iterative problem.

While in a simple conical nozzle the direction of the flow near the wall θ_{wall} is already known and equal to α_{half} the slope of the characteristic can be obtained from equation 3.80 and the intersection is easily found out. In a bell nozzle, however, the location of the intersection is needed to obtain the flow direction near the wall θ_{wall}, which in turn can only be found out knowing the characteristic's slope, which is calculated with θ_{wall}.

To solve this problem, equation 4.2 is assumed to determine the intersection of the characteristic line with the contour and obtain θ_{wall}, which is used to update the characteristic slope leading to a new intersection. This process is repeated until the difference in θ_{wall} between two iterations is negligible. Algorithm 5 presents such an iterative solver.

Algorithm 5 Intersection Location Iterative Solver

Input: Flow angle adjacent to the wall θ_{i-1}
Output: Updated flow angle at the wall θ_{new}

1: $\theta_i = \theta_{i-1}$
2: $\theta_{new} \leftarrow \text{Interp}(\theta_i)$
3: **while** $|\theta_i - \theta_{new}| \geq 0.001$ **do**
4: $\quad \theta_{new} \leftarrow \text{Interp}(\theta_i)$
5: **end while**

Implementing the Method of Characteristics on a contour is not as straightforward as creating a contour, because the grid generated by the MoC is not identical to the domain of the nozzle. This happens because the contour is continuous in space, while the grid is discrete. In other words, the algorithm in charge of computing the characteristics generates each left-running characteristic line one by one and will only stop when the last characteristic line goes beyond the nozzle contour. This leads to the last left-running characteristic being partially located outside the nozzle. as shown in figure 4.3.

In order to calculate the flowfield variables at the exit, a "virtual" characteristic is imagined between the last two left-running characteristics, where the flowfield properties can be calculated through linear interpolation.

4.1.3 Coefficient of Thrust - Method of Characteristics

The thrust of a quasi-one dimensional nozzle flow is known and calculated as $F = \dot{m}V_e + (p_e - p_{amb})A_e$ (equation 3.54 in chapter 3). As mentioned before, this equation is used for quasi-one-dimensional flow, meaning the flowfield variables only vary in x.

In order to obtain the thrust of the nozzle using the MoC it is necessary to apply a Riemann sum to the

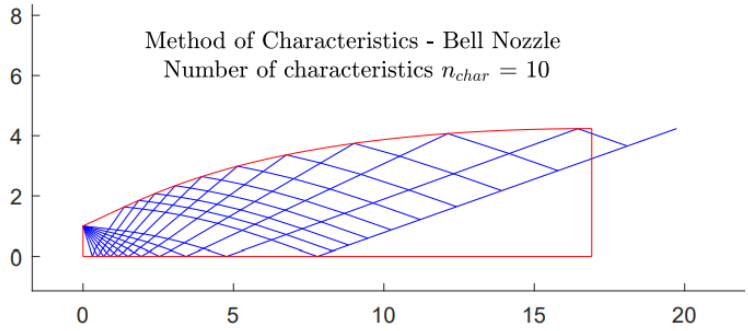


Figure 4.3: Characteristic "overshooting" using the MoC with $n_{char} = 10$ on a bell nozzle geometry.

normal component of the exit velocity

$$V_n = V_x = V_e \cdot cos(\theta_e) \,, \tag{4.4}$$

and pressure p_e along the last two characteristic lines and interpolate the values obtaining the average exit pressure and velocity along the "virtual" characteristic. A Riemann sum of a variable f with respect to the tagged partition $x_0, ..., x_n$ together with $t_0, ..., t_{n-1}$ is [80]

$$\sum_{i=0}^{n-1} f\,(t_i)(x_{i+1} - x_i) \,. \tag{4.5}$$

The Riemann sum is normally separated between lower and upper Riemann sums:

$$\begin{aligned} L(f,P) &= \sum_{i=0}^{n-1} \inf_{t\in[x_i,x_{i+1}]} f(t)(x_{i+1} - x_i) \,, \\ U(f,P) &= \sum_{i=0}^{n-1} \sup_{t\in[x_i,x_{i+1}]} f(t)(x_{i+1} - x_i) \,. \end{aligned} \tag{4.6}$$

However, for this problem a "centered" Riemann sum is applied as:

$$R_{sum} = \sum_{i=0}^{n-1} (f\,(x_{i+1}) + f\,(x_i)) \cdot (x_{i+1} - x_i)\,/2 \,, \tag{4.7}$$

and the average value

$$f_{avg} = \frac{R_{sum}}{n} \,. \tag{4.8}$$

Proceeding to the interpolation, one obtains:

$$\phi_e = f_{\text{before}} + \frac{x_e - x_{\text{wall}_1}}{x_{\text{wall}_2} - x_{\text{wall}_1}} \cdot (f_{\text{after}} - f_{\text{before}}) \,, \tag{4.9}$$

where x_{wall_1} and x_{wall_2} are the x-location of the intersection of the characteristics with the contour before and after the exit point, respectively. One may ask how the last characteristic intersects the wall if it's already outside the nozzle. To tackle this problem the algorithm assumes a fictitious contour that continues indeterminably as a straight line with the slope of the exit point.

After the determination of both exit velocity and pressure, the thrust can be obtained using equation 3.54

as well as the thrust coefficient

$$C_T = \frac{F}{p_0 A_t} \, . \tag{4.10}$$

4.2 Design optimization in MATLAB®

Before implementing any kind of optimization method, it is necessary to define an objective function. As mentioned before, the main goal of the optimizer is to increase the coefficient of thrust of a rocket nozzle. A possible objective function algorithm is presented in algorithm 6.

Algorithm 6 Objective Function

Input: Design variables vector $\mathbf{x}$, Number of characteristics (n), Exit Mach (M_e)
Output: Objective Value ($f(\mathbf{x})$)

1: $\text{GEO}_{\text{base}} \leftarrow \text{MIN}_{\text{nozzle}}(n, M_e)$ {See Algorithm 3}
2: $\text{GEO}_{\text{def}} \leftarrow \text{FFD}(\mathbf{x}, \text{GEO}_{\text{base}})$ {See Algorithm 1}
3: $\text{C}_T \leftarrow \text{MoC}(\text{GEO}_{\text{def}})$
4: $f = 1/C_T$ {See Equation 3.107}

In a nutshell, the objective function receives as input design variables $\mathbf{x}$ used to deform a base geometry GEO_{base} using the Free Form Deformation technique (see subsection 3.7.3). The base contour is obtained through the minimum nozzle method specifying the number of characteristics and the exit Mach. Finally, the Method of Characteristics is applied to the deformed geometry and the coefficient of thrust is calculated. Since this value is supposed to be maximized and most optimizers consider the optimum as the minimum, the objective function output is the inverse of the coefficient of thrust (see section 3.8).

Regarding the optimizer, two different optimization methods were applied to the Method of Characteristics which are presented next. When using an optimizer for the MoC, one has to take into consideration that the restriction to the optimizer is intrinsic to the method itself instead of being expressed separately. Equation 3.106 reduced to:

$$\min_x f(\mathbf{x}) \text{ such that } \Big\{ \text{lb} \leq \mathbf{x} \leq \text{ub} \, . \tag{4.11}$$

Fmincon

Fmincon is a gradient-based method used to find the minimum of a constrained nonlinear multivariable function identical to equation 3.106. It is implemented in MATLAB® as follows:

$$x = \text{fmincon}(f, \mathbf{x}_0, \text{A}, \text{b}, \text{A}_{eq}, \text{b}_{eq}, \text{lb}, \text{ub}, \text{nlcon}) \, , \tag{4.12}$$

where nlcon are the nonlinear constraints. Like most gradient-based methods it is designed to operate on problems where the objective function and its derivative are continuous.

This method performs a local search since its characterized by a single starting point and presents a deterministic behavior because it always evaluates the same points given the same initial condition.

There are multiple different algorithms to choose from when using this method. Sequential Quadratic Programming (SQP) was opted to be used as the Fmincon algorithm. SQP methods are a class of optimization methods that solve quadratic programming (QP) subproblems at each iteration. Each QP subproblem minimizes

a quadratic model of a certain modified Lagrangian function subject to linearized constraints [81]. To be better informed about the SQP algorithm applied to this specific example, where the only constraints are the boundaries, one may read [82].

NSGA-II

The genetic algorithm NSGA-II stands for nondominated sorting genetic algorithm II. Contrarily to the previous algorithm, NSGA-II is a gradient-free algorithm that is population-based like most genetic algorithms. The optimization starts with a set of design points rather than a single starting point, and each optimization iteration updates this set in some way. This algorithm performs a global search since it does not begin at a specific point and evolves throughout the problem domain. Furthermore, it involves randomness in the population generation. Another important characteristic of this algorithm is associated with the fact that it is a multiobjective optimizer.

As the second objective to be analyzed, the width, or rather, the height of the nozzle is chosen since it shows a direct proportionality to the drag forces subjected to the rocket nozzle. During take-off, minimizing forces opposed to the thrust is critical, being drag force one of the most dominant due to the high air density near sea level.

The NSGA-II algorithm is inspired by biological reproduction and evolution using three main steps: selection, crossover, and mutation [75]. As a result of this algorithm, a Pareto front will be obtained. The details behind each step of the NSGA-II algorithm can be read about in [83]. The implementation from Seshadri [84] is used. Table 4.1 enumerates each of the main characteristics of both algorithms.

	Algorithm Characteristics				
	Gradient	Search	Behavior	Modeling	Optimization
Fmincon	Based	Local	Mathematical	Deterministic	Single
NSGA-II	Free	Global	Heuristic	Stochastic	Multi

Table 4.1: Fmincon versus NSGA-II

4.3 Computational Fluid Dynamics

Computational fluid dynamics (CFD) is a very powerful tool for the analysis of systems involving fluid flow, heat transfer, and other associated phenomena. From the 1960s onwards CFD techniques have been used in the aerospace industry. CFD leads to substantial time and cost reduction of new designs as well as enables the study of systems where controlled experiments are difficult, such as simulating rocket nozzle supersonic flows [66].

On the other hand, one has to take into consideration its computational cost. When working with a computational budget, it is necessary to be able to find a trade-off between the accuracy and the computational cost of the solver.

4.3.1 Mesh Generation

The discretization of the physical domain, also referred to as mesh generation, produces a computational mesh on which the governing equations are subsequently solved using the Finite Volume Method (see section 3.6). This so-called discretization consists in splitting the continuous domain into non-overlapping sub-domains called cells [63]. Typically a mesh geometry is discretized using either a structured or an unstructured grid. Grids with both structured and unstructured regions are known as hybrid grids. Some grid properties that affect solution accuracy are [66]:

Outer boundary size - The location of the outer boundary of the grid can have a large impact on the results. However, for internal flows, this is not the case since the domain is finite and known.

Aspect ratio - The aspect ratio is essentially the ratio of the length and width of a cell. To ensure good grid quality the aspect ratio should be in the order of one. For cells within the boundary layer near solid surfaces unfortunately this is rarely the case.

Cell stretching - Grid stretching has to do with the relationship of adjacent cells. When applying the finite volume method the derivative approximations assume an evenly spaced grid system, which is only true if there is no stretching, therefore large discontinuities in grid size should be avoided. A rule of thumb for cell stretching is that any two adjacent cells should never have lengths that differ by more than 20%.

Skewness - An important issue of unstructured meshes is for the location of the face centroid connecting two cells to be centrally located with respect to the neighboring cell centroids. This discrepancy is called skewness and must be averted as much as possible since it affects the correct application of the finite volume method.

Alignment - Near solid walls, the gradients in the streamwise direction are orders of magnitude lower than those in the transverse direction due to the no-slip viscous boundary condition. The geometry most favorable for an accurate solution dealing with these large gradients is when the element faces are aligned parallel to the wall boundary. Hybrid meshes are useful in this manner.

Near-wall treatment - As stated before, the no-slip condition at the wall generates high gradients. A grid refinement is needed in order to properly capture the viscous phenomena, such as the three sublayers of the inner region.

In order to produce a good mesh, one has to take all these characteristics in mind and apply them according to the complexity of the domain to be discretized. While the mesh generation of airfoils design has to tackle most of the aforementioned grid properties, the grid generation of a rocket nozzle inflow proves to be very simple. A structured mesh can easily be implemented without compromising the solver's accuracy or demanding too much computational power.

GMSH

GMSH is an open-source meshing tool that was used to generate the mesh to be used in the CFD solver. The steps to create a mesh consist in importing/creating a nozzle geometry and marking its boundaries. Afterward, it is possible to choose from multiple types of mesh geometries, being the structured one selected. One of the advantages of this program is its capacity to receive as input a *.geo* file generated in MATLAB® with all the

necessary information and exporting a *.su2* to be later used.

4.3.2 SU^2 Framework

SU^2 is an open-source collection of software tools written in C++ and Python for the analysis of partial differential equations (PDEs) and PDE-constrained optimization problems [85]. In the present work, SU^2 was chosen over other open-source frameworks because it had shown accurate behavior on past works (e.g.[67]). One of the main disadvantages of being an open-source framework is its limited documentation on code. Even though providing some tutorials, the process of setting up the configuration file showed to be very exhaustive

To properly run the simulation applied to the nozzle mesh a configuration *.cfg* file had to be developed. Some of the decisions regarding the choices used in the configuration are presented next.

Configuration File

A configuration file is a file containing the user's options for a particular problem to be solved. For the simulation of a supersonic nozzle flow, an Euler solver was applied, in order to compare this high-fidelity method with the Method of Characteristics since both are based on the Euler equation. As a result, no turbulence or viscosity models needed to be implemented.

The fluid was assumed as an ideal gas with the ratio of specific heats γ=1.4 and a specific gas constant of R = 287 J/(kg·K), the default value for standard air. Regarding the boundary conditions (see section 3.5), the Euler condition was applied to the nozzle wall, while the symmetry condition was enforced on the centerline. As for the inlet (throat) of the nozzle, a supersonic inflow Dirichlet condition was applied and the temperature, static pressure and velocity direction were assumed. Concerning the output of the nozzle one simply enforced the static outlet pressure.

To compute the gradients of the flow variables the weighted least-squares numerical method for spatial gradients was implemented. Methods based on least squares produce more accurate approximations of the solution gradients on distorted meshes. Since SU^2 has a weighted least-squares method already implemented, it will be used in this work to calculate the gradients of the flow variables at all grid nodes, which are then averaged to obtain the gradients at the cell faces [67].

Regarding the Courant-Friedrichs-Lewy (CFL) condition an adaptive CFL number was implemented which could fluctuate between 0.5 and 100. In a CFD simulation, this number broadly indicates how much the information travels (U) across a computational grid cell (Δh) in a unit of time (Δt) [86]. The stability requirements of time-integration schemes can be defined using the CFL condition:

$$C = \frac{U\Delta t}{\Delta h} \leq C_{max} \, , \tag{4.13}$$

where C_{max} varies depending on the type of time integration scheme, but it is generally equal to 1 or less [86]. Implicit schemes alleviate such a strict CFL condition allowing the use of time steps where CFL is higher than 1. This is the case for this setup where an Euler implicit time discretization method is applied.

A multigrid with two levels is used to carry out the simulations faster and help convergence. The convergence criteria used to observe if the solver converges is the Root-Mean-Square (RMS).

If a sequence X_N converges to X, then the mean squared difference should become smaller and smaller by increasing n. The root-mean-square deviation (RMSD) of an estimator $\hat{\Theta}$ with respect to an estimated parameter Θ is defined as the square root of the mean square error:

$$\mathrm{RMSD}(\hat{\Theta}) = \sqrt{\mathrm{MSE}(\hat{\Theta})} = \sqrt{\mathrm{E}\left((\hat{\Theta}-\Theta)^2\right)} = \sqrt{\frac{\sum_{i=1}^{n}(X_N - X_i)^2}{n}} \,. \tag{4.14}$$

If the problem converges the deviation also referred to as residual will gradually decrease until it achieves a convergence criterion value which was assumed as $\log_{10}$(-5).

Finally and most importantly a ROE scheme was used as the convective numerical method. ROE is a modern high resolution, shock-capturing scheme. The main idea behind this scheme is to determine the approximate solution by solving a constant coefficient linear system instead of the original nonlinear system [87]. This scheme has shown to be particularly effective when dealing with the Euler equation and was implemented for this reason.

Other input variables of the configuration file lay outside of this thesis scope and will not be described. However, their importance must be underlined because these variables have a big influence on the convergence of the simulation. Regarding the outputs, to calculate the coefficient of thrust for the nozzle flows some custom outputs had to be implemented. After finalizing the setup of the CFD problem the simulation revealed a converging response. The setup of the configuration file can be visualized in Appendix B.

4.3.3 Grid Convergence Study

Before commencing with the optimization of the nozzle design a grid convergence study was conducted in order to define the optimal mesh discretization. As mentioned before, a trade-off between accuracy and computational cost has to be done. The assessment of the spatial convergence of a simulation involves performing the simulation on two or more successively finer grids. As the grid is refined and the time step is reduced the spatial and temporal discretization errors, respectively, should asymptotically approach zero, excluding computer round-off error. A usual method to examine spatial and temporal convergence is known as Richardson's Extrapolation.

Richardson's Extrapolation

Taking the values relative to three differently refined meshes with a constant grid refinement ratio r, the Richardson's Extrapolation value, which estimates the continuum value (value at zero grid spacing) from a series of lower-order discrete values can be expressed as [88]:

$$f_{h=0} \cong f_1 + \frac{f_1 - f_2}{r^p - 1} \,, \tag{4.15}$$

where

$$p = \ln\left(\frac{f_3 - f_2}{f_2 - f_1}\right) / \ln(2) \,, \tag{4.16}$$

and f_1, f_2 and f_3 represent the finest to the coarsest grid, respectively.

The grid refinement ratio r is defined as:

$$r = \frac{h_{s_{coarse}}}{h_{s_{fine}}}\,, \tag{4.17}$$

where h_s is the grid element spacing. For a structure grid

$$h_s \propto \frac{1}{\sqrt{N_s}}, \tag{4.18}$$

where N_s is the number of cells on the mesh. Merging equations 4.17 and 4.18 it results in:

$$r = \sqrt{\frac{N_{s_{fine}}}{N_{s_{coarse}}}}\,. \tag{4.19}$$

4.3.4 Surrogate Model

After completing the grid convergence study the surrogate model may be initiated. First and foremost, a sufficiently big nozzle geometry dataset has to be generated. Secondly, the mesh generation, with the optimal refinement deduced in the grid study is applied using GMSH.

Having obtained a database of meshes, running the simulation is the next step to gather the training data to build the surrogate model (see section 3.11). Implementing polynomial fitting (in MATLAB®/EXCEL®) the surrogate model is computed and later cross-validated with the test data, which was generated simultaneously with the training data. If the surrogate model presents negligible errors regarding the test data, one can assume that the model simulates the SU^2 framework to a certain extent. Finally, an optimizer (Fmincon and/or NSGA-II) can be applied to obtain the optimal design and compare it to the design obtained using the Method of Characteristics.

Although SU^2 provides a feature to perform shape design optimization the computational budget available for this thesis would not sustain such complex simulations. Furthermore, to use this design capability in SU^2 one has to adapt the source code such that nozzle design with the aforementioned features is enabled (e.g. nozzle outputs and flow post-processing), which is outside the scope of this thesis.

Seeing that the surrogate model implementation showed good results the SU^2 framework ended up only being used to compute the training and testing data of the model. The optimization in turn became a lot less complex and computationally cheaper.

Figure 4.4 shows a flowchart of the assignments that were implemented in order to achieve a surrogate-based optimization featured by the SU^2 framework.

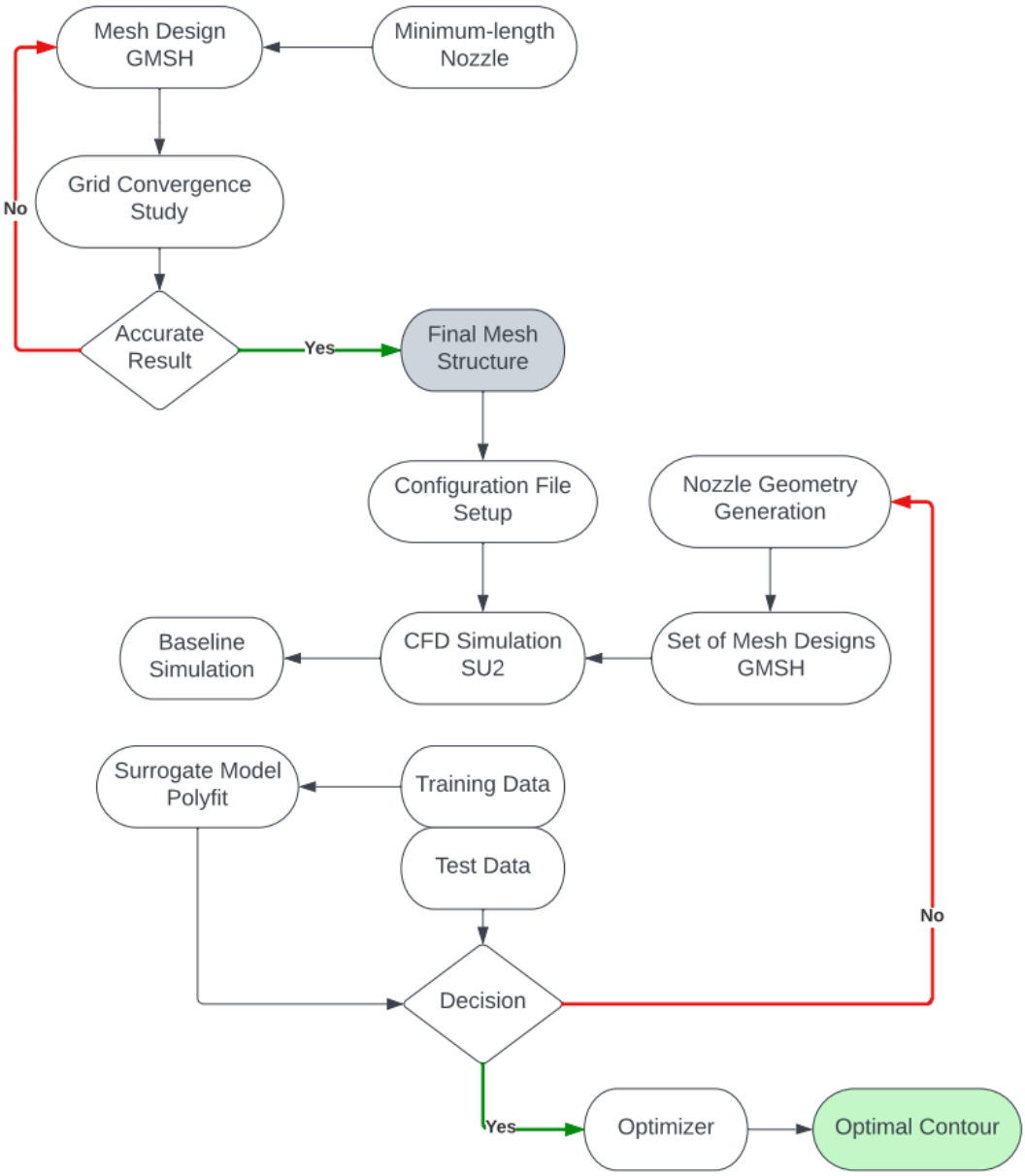


Figure 4.4: Flowchart describing the critical steps to conduct a surrogate-based optimization.

Chapter 5

Results

Similarly to chapter 4, the results of this thesis will be presented in the chronological order in which the thesis was developed. At the end of the chapter, a comparison between both simulation methods (MoC and CFD) will be conducted in order to rank the fidelity of the Method of Characteristics. Before starting with the discussion of the results, it is important to underline that a two-dimensional MoC produces a wedge-shaped nozzle with a rectangular outlet (figure 5.1) [89]. Additionally, the nozzles produced with the two-dimensional algorithm are calculated for a unit breadth and unit throat height.

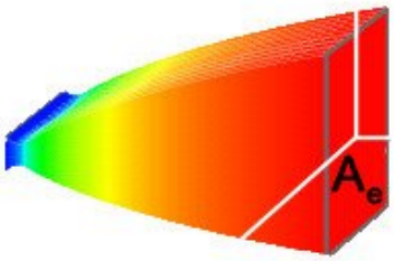


Figure 5.1: Example of a typical wedge nozzle [89].

5.1 Minimum-length Nozzle Results

Being the easiest nozzle design to implement, the minimum-length nozzle is a good candidate to test the implementation of the Method of Characteristics. Since the algorithm applied to arbitrary nozzles is a variation of the one developed for minimum-length nozzle it is of utmost importance the accurate functioning of algorithm 3. The implementation of the MoC for a minimum nozzle with an exit Mach $M_e = 3$, with varying number of characteristics n_{char} is shown in figure 5.2

The resulting nozzle length L and exit height h_{nozzle} are presented in table 5.1.

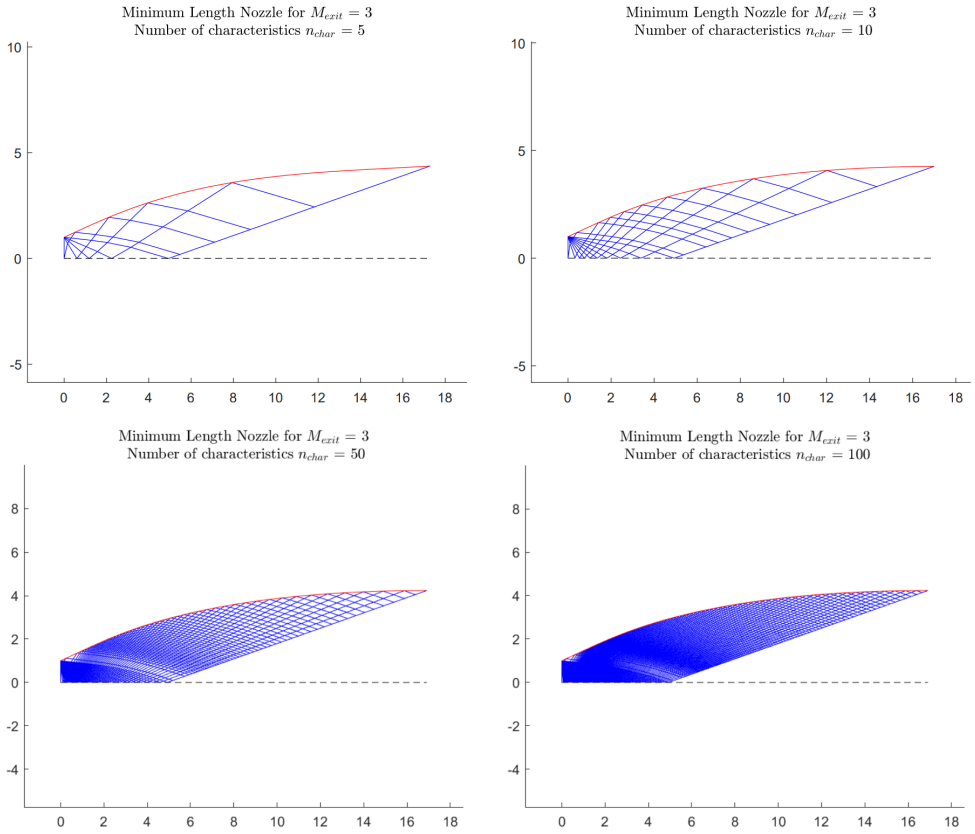


Figure 5.2: Minimum-length nozzle design with $M_e = 3$, for various number of characteristic lines ($n_{char} = [5, 10, 50, 100]$).

Minimum Nozzle Geometry for $M_e = 3$		
Number of Characteristics	L	h_{nozzle}
$n_{\text{char}} = 5$	17.285	4.368
$n_{\text{char}} = 10$	16.983	4.261
$n_{\text{char}} = 50$	16.912	4.236
$n_{\text{char}} = 100$	16.910	4.235

Table 5.1: Influence of the number of characteristics on the nozzle design

One of the main attributes of this nozzle type is the uniform exit flow that occurs using this nozzle contour. This flow distribution is the most alike to a quasi-one-dimensional nozzle flow (see subsection 3.2.2). In order to verify the accuracy of this method one will compare it to the area-Mach number relation (equation 3.53 in chapter 3) for a different number of characteristics.

It is important to underline that the previous equation is based on three-dimensional flow. Since the Method of Characteristics is computed in 2D, considering the throat radius equal to one, the relation can be rewritten as:

$$h_{\text{nozzle}} = \sqrt{\frac{1}{M_e^2}\left[\frac{2}{\gamma+1}\left(1+\frac{\gamma-1}{2}M_e^2\right)\right]^{(\gamma+1)/(\gamma-1)}}. \tag{5.1}$$

Comparing equation 5.1 to the numerical solution of the MoC for various numbers of characteristics one

obtains the following evolution of the relative error.

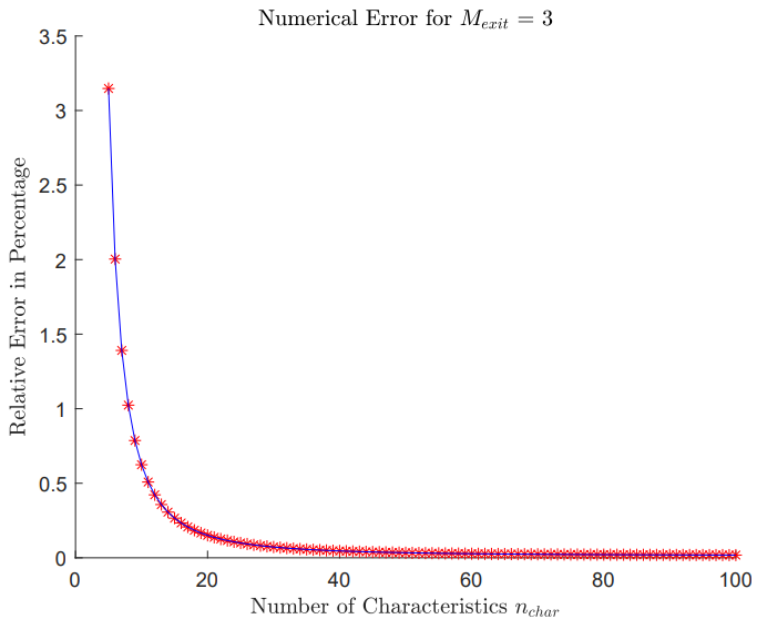


Figure 5.3: Analytical versus Numerical Solution - Relative Error for varying number of characteristics ranging from 5 to 100 for a minimum-length nozzle with $M_e = 3$.

Figure 5.3 shows how quickly the error descends by increasing n_{char}. For further calculation of the MoC, fifty characteristic lines will be used, since it shows a negligible error compared to the analytical solution and too many characteristics are not desired because they raise the computational cost.

Another important aspect of the minimum-length nozzle is noticing that the characteristics maintain an identical distance between each other outside of the kernel zone. This reaches back to equation 4.2, where it was assumed that the flow neither experiences expansion nor compression, as proven by figure 5.2. The kernel zone is defined as the zone of flow expansion and it is outlined by the characteristics that are initialized at the throat.

For $n_{\text{char}} = 50$ the minimum nozzle contour for varying M_e is presented below. Observing figure 5.4 one

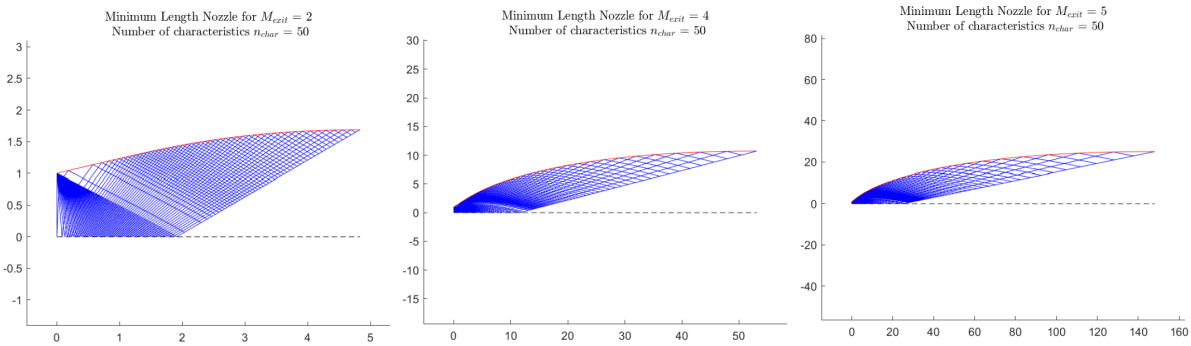


Figure 5.4: Minimum-length nozzle design with $n_{char} = 50$, for various exit Mach numbers ($M_e = [2, 4, 5]$).

can notice that the characteristic grid near the nozzle exit is coarser for higher exit Mach numbers. This results

in larger numerical errors. As a consequence, the number of characteristics has to be adjusted to the size of the nozzle.

5.2 Arbitrary Nozzle Contours

The Method of Characteristics proved to be an accurate tool to design the contour of a nozzle. However, how will it work on a preexisting contour?

5.2.1 Conical Nozzle - Results

As mentioned in chapter 4 (subsection 4.1.2), the algorithm in charge of computing the characteristics generates each left-running characteristic line one by one and will only stop when the last characteristic line goes beyond the nozzle contour. This means that for a sufficiently long nozzle the right-running characteristics reflected from the nozzle wall will intersect the centerline and reflect off it generating new left-running characteristics. This phenomenon can be observed in figure 5.5.

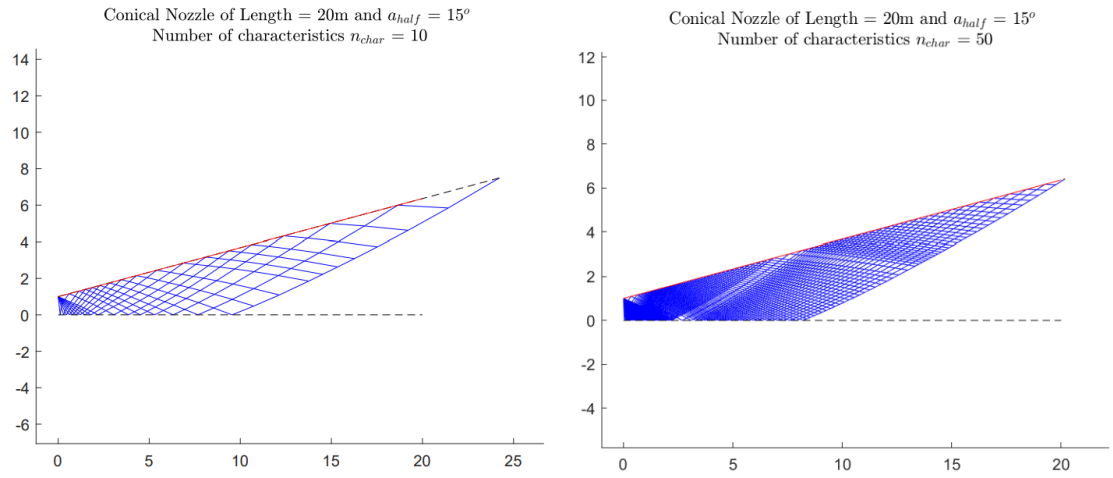


Figure 5.5: Conical nozzle design with $\alpha_{half} = 15^o$, for various number of characteristic lines ($n_{char} = [10, 50]$).

This occurrence generates a discontinuity in the grid spacing which leads to the development of small errors as shown in the Mach distribution along the nozzle wall compared to the analytical equation 5.1 portrayed in figure 5.6.

Another consequence of imposing the nozzle geometry rather than creating it is the fact that the exit flow no longer shows uniform behaviour. This means that to calculate the average exit Mach, a Riemann sum has to be applied. Since the Mach number near the centerline is smaller and grows toward the wall (in a conical nozzle), it is expected that the average M_e value is smaller compared to the Mach number along the wall. This comparison is shown in figure 5.7. This discrepancy between both results dictates that the relation 5.1 for a quasi-one-dimensional flow may only be applied for uniform conditions. Additionally, figure 5.5 also shows that the "overshooting" (see section 3.4) can be minimized using a higher number of characteristic lines.

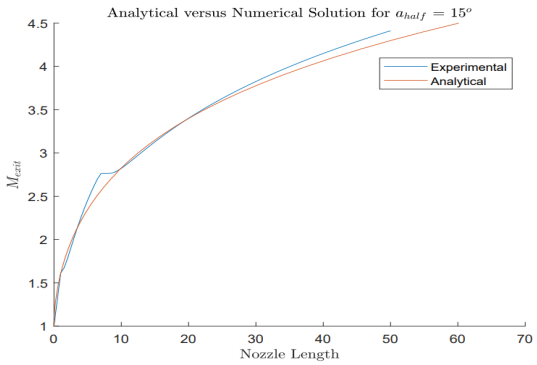


Figure 5.6: Analytical versus Numerical Solution - Relative Error for varying nozzle length for a conical nozzle with $\alpha_{half} = 15^o$.

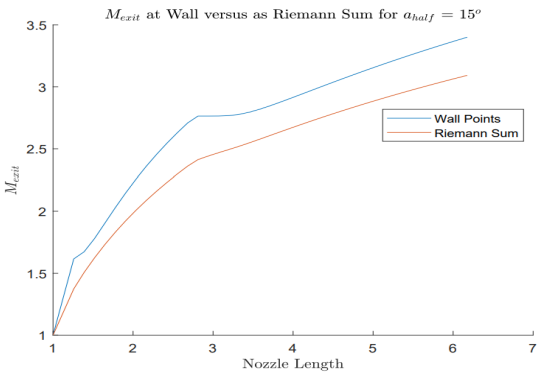


Figure 5.7: Exit Mach versus Nozzle Length for two different methods - Wall Point and Riemann Sum

5.2.2 Bell Nozzle - Results

To design a bell nozzle contour the parameterization method FFD (see section 3.7) is applied to a base geometry, which in turn is generated from a minimum-length nozzle. For all bell nozzle simulations the contour of a minimum-length nozzle with $M_e = 3$ and $n_{\text{char}} = 50$ was chosen as the base geometry.

The decision behind this choice stems from the fact that for relative higher Mach numbers (M>5) the flow can no longer be perceived as perfect due to the thermal dissociation of the air. Thermal dissociation is defined as the breakdown of chemical bonds in molecules resulting in the production of smaller molecules or atoms under the influence of temperature [90]. More on hypersonic flows in [91].

Before deforming any kind of geometry a free-form deformation box has to be built around the base geometry. The starting point as well as the endpoints of the nozzle's geometry delimit the size of the box. A FFD box can have as many control points as wanted, however, the influence of each control point reduces when increasing the refinement of the grid, as shown by equation 3.103 and observed in figure 5.8.

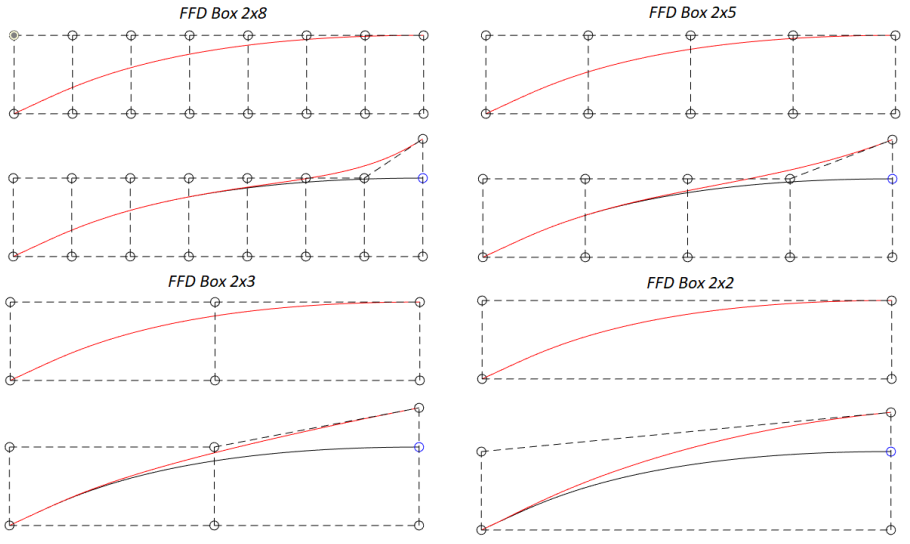


Figure 5.8: Free Form Deformation Boxes with various grid sizes: [2x2], [3x2], [5x2] and [9x2].

In figure 5.8 a deformation in the vertical direction is applied to the most up-right node with the value $\mathbf{x}_{\text{FFD}} = 0.5$. This value is a parameter that describes how far away the deformed node is from the original position. If this value is equal to 1 the total deflection equals the height of the base nozzle geometry.

In order to test the bell nozzle algorithm a simulation is conducted for an undeformed nozzle ($\mathbf{x}_{\text{FFD}} = 0$).

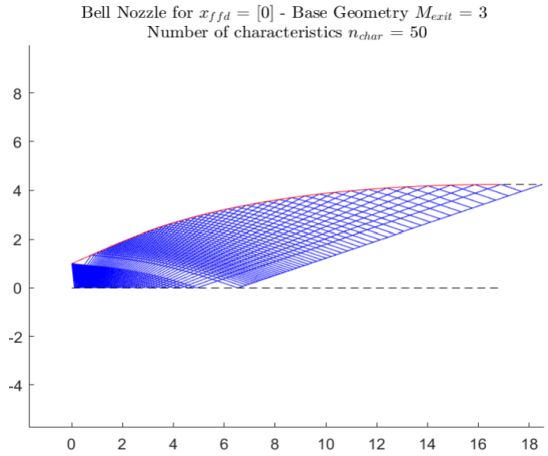


Figure 5.9: MoC applied to an undeformed bell nozzle $x_{FFD} = 0$ with $M_e = 3$ and $n_{char} = 50$.

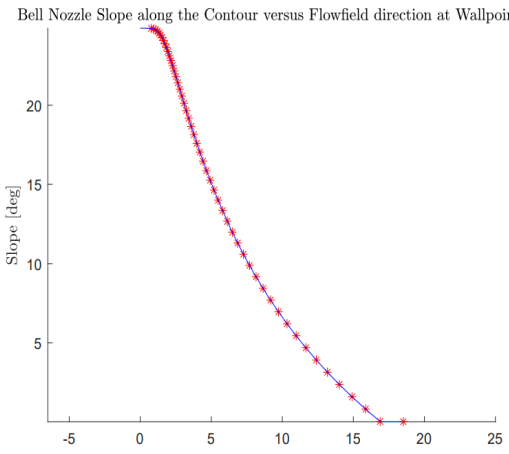


Figure 5.10: Slope intersection by MoC algorithm along undeformed bell nozzle

Figure 5.9 clearly shows an "overshoot" regarding the last characteristic as seen by the black dotted line. However, due to the interpolation applied to calculate the coefficient of thrust (see subsection 4.1.3) the influence of the overshooting is minimized.

Nevertheless, the proper implementation of the MoC applied to the bell nozzle is determined by the correct functioning of algorithm 5. The iterative intersection of the characteristic lines with the nozzle wall shows accurate results as demonstrated by figure 5.10, where each asterisk represents the slope of the flow at different positions of intersection.

The Method of Characteristics can now be used for different bell nozzles by changing the deformation parameters $\mathbf{x}_{\text{FFD}}$, which are directly linked to $\mathbf{P}_{ijk}$. The node most up-right of the FFD box is the one that will be dislocated in the vertical direction to obtain new geometries. This decision is taken, because that control node is the one that has the most influence on the coefficient of thrust since the exit velocity and pressure are mainly dependent on the exit area, which is directly controlled by the node as seen in figure 5.8.

5.3 Optimization in MATLAB® - Results

After verifying that the Method of Characteristics is capable of determining the flowfield variables along the nozzle domain and concluding the coefficient of thrust for various geometries, it is possible to set up an objective function to run an optimization.

Before carrying out any optimization some properties of the rocket nozzle have to be clarified. The base nozzle is engineered in a way that its exit pressure equals 1 atmosphere, where 1 atm = 101325 Pa. To achieve this feat the chamber pressure p_0 has to be exactly 3723300 Pa. The nozzle is structured this way in order

for it to be optimal for an ambient pressure near the sea level, since the condition of optimality states that $p_e = p_{amb}$ (equation 3.60 in chapter 3). Additionally, a chamber temperature T_0 of 3000 K was chosen.

5.3.1 Fmincon - Results

Various simulations were run using the Fmincon method. As discussed before the most up-right node of the FFD box has the biggest influence on the coefficient of thrust of a rocket nozzle. For that reason, this will be the only node to be deformed, with a lower and upper boundary of -0.5 and 2, respectively. In order to have some variety throughout the results 4 different FFD boxes will be used similarly to those illustrated in figure 5.8.

After conducting all simulations, one obtains the results listed in table 5.2.

Fmincon - Results								
Grid Size	Ambient Pressure p_{amb}							
	0 Atm		0.25 Atm		0.5 Atm		0.75 Atm	
	$\mathbf{x}_{\text{FFD}}$	C_T	$\mathbf{x}_{\text{FFD}}$	C_T	$\mathbf{x}_{\text{FFD}}$	C_T	$\mathbf{x}_{\text{FFD}}$	C_T
2x2	1.1167	1.61217	0.7721	1.56271	0.4773	1.52023	0.2225	1.48377
3x2	0.9020	1.60647	0.6582	1.56037	0.4302	1.51956	0.2123	1.48371
5x2	0.6046	1.59609	0.4693	1.55548	0.3325	1.51788	0.1756	1.48345
9x2	0.3441	1.58533	0.2826	1.54993	0.2158	1.51558	0.1323	1.48302

Table 5.2: Fmincon Results for various FFD grids and ambient pressures.

The deformation done to the nozzle to achieve its optimal design is demonstrated in figure 5.11 for each FFD grid.

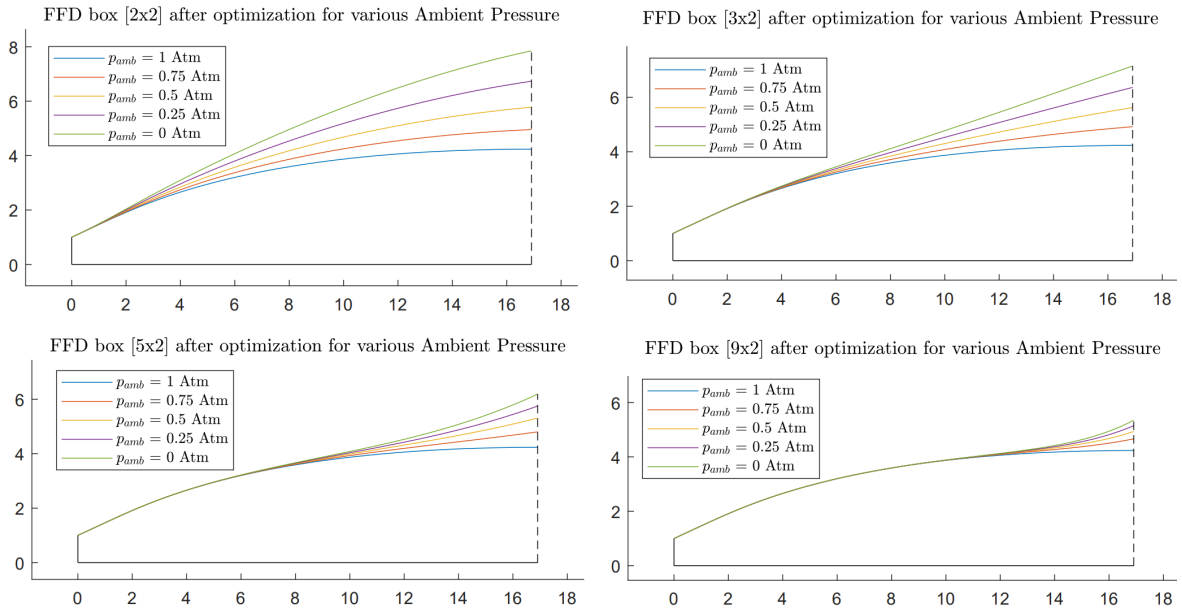


Figure 5.11: Optimal Nozzle Geometry for various FFD grids and ambient pressures - Fmincon.

Figure 5.11 demonstrates that the optimal nozzle geometry for sea level pressure is the base geometry, as mentioned before. For this particular condition the nozzle outputs a coefficient of thrust equal to:

$C_{T_{SSL}} = 1.45258$ and $C_{T_{vac}} = 1.56784$

Regarding lower atmospheric pressures, the outcomes are expected since the nozzle widens for lower pressure to minimize the difference between the exterior and the exit pressures (optimality condition). However, a total optimal condition ($p_e = p_{amb}$) cannot be achieved because the flow may not be expanded too extensively.

As mentioned before, the influence of deformed nodes is smaller for finer grids, meaning the deformed domain will be smaller, resulting in more abrupt flow expansions. This is the reason why the deformation parameter $\mathbf{x}_{\text{FFD}}$ is smaller for finer FFD grids and the geometry difference between atmospheric pressures is smaller compared to courser grids.

To achieve better outcomes for finer grids, a possibility would be to give other control nodes the possibility to be deformed, in order to achieve a smoother widening of the rocket nozzle. By applying this change the coefficient of thrust can be increased even further.

Fmincon - Results										
Grid Size	Ambient Pressure $p_{\text{amb}} = 0$ atm									
	$\mathbf{x}_{\text{FFD}_1}$	$\mathbf{x}_{\text{FFD}_2}$	$\mathbf{x}_{\text{FFD}_3}$	$\mathbf{x}_{\text{FFD}_4}$	$\mathbf{x}_{\text{FFD}_5}$	$\mathbf{x}_{\text{FFD}_6}$	$\mathbf{x}_{\text{FFD}_7}$	$\mathbf{x}_{\text{FFD}_8}$	$\mathbf{x}_{\text{FFD}_9}$	C_T
5x2	0.7837	-0.0423	1.0452	0.7899	1.1946	-	-	-	-	1.61568
9x2	0.7351	0.0385	0.5621	0.4219	0.5059	0.8118	0.9041	0.9822	1.1569	1.61612

Table 5.3: Fmincon Results for various FFD grids using multiple optimization variables

Table 5.3 shows the deformations applied to all the upper line control points for 2 different grids. An important observation is that the coefficient of thrust now increases when the grid is refined. This is due to the optimal deformation of the base geometry no longer being detained by non-deformed control points as before. The resulting geometries can be visualized in figure 5.12.

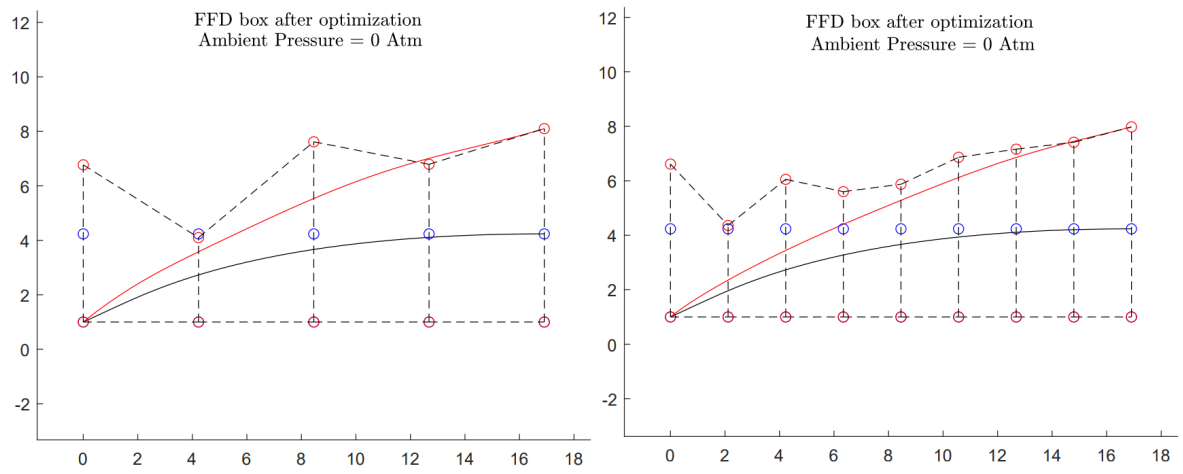


Figure 5.12: Optimal Nozzle Geometry for various FFD grids using multiple optimization variables in vacuum.

As mentioned in chapter 2 a TOC produces an internal shock wave emanating from the throat. A similar occurrence for the [9x2] optimized contour can be visualized in figure 5.13, with the clustering of characteristics near the throat, meaning a shock wave is about to form.

Additionally, the TOC can be accurately modelled by a parabola, becoming a TOP described by the following

equation: $y = -0.0122x^2 + 0.6158x + 1$ and displayed in figure 5.14.

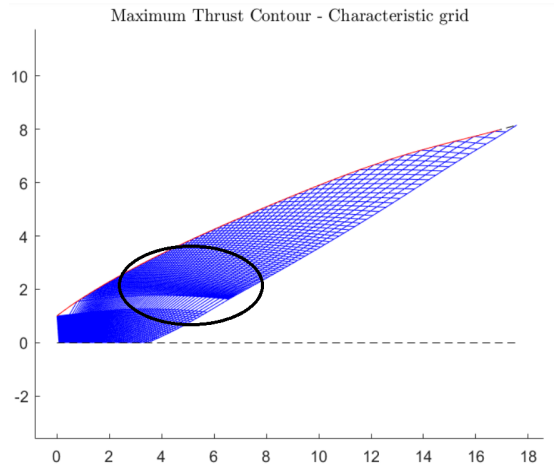


Figure 5.13: Characteristic Grid of Thrust Optimized Contour - Characteristic Collapse highlighted

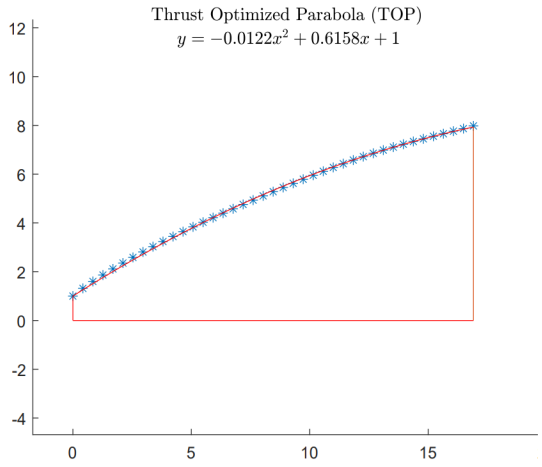


Figure 5.14: Regression of Thrust Optimized Parabola with a $R^2 = 0.9996$.

However, the method using multiple deformation variables has its downside. For grids that are too refined the optimal geometries show some curling, similarly to "overfitting" a polynomial curve (see section 3.11), as portrayed in figure 5.15, where the slope derivative is constantly changing signal.

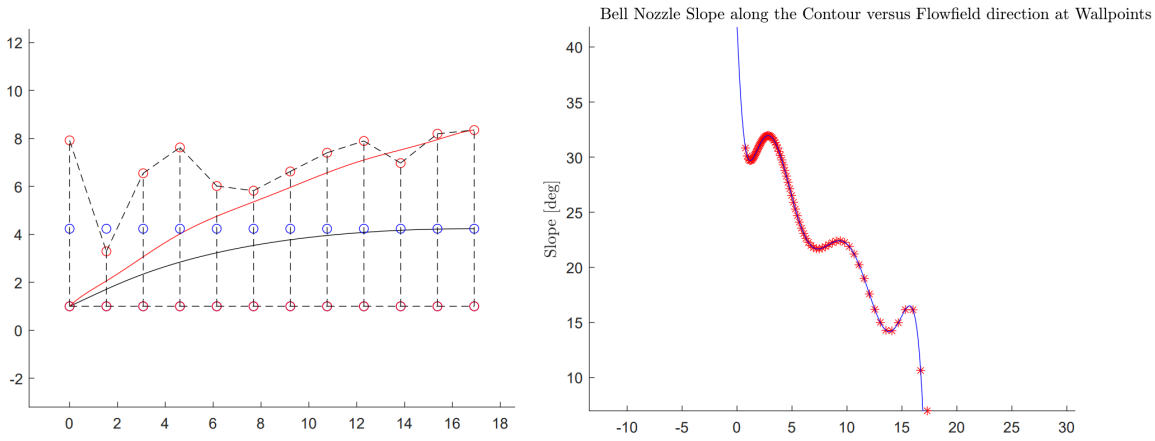


Figure 5.15: Thrust Optimal Contour Overfitting - Contour and Slope

Overall, Fmincon is shown to be a useful optimization method when using a small number of design variables that achieved an overall increase of the coefficient of thrust in vacuum $C_{T_{vac}}$ from 1.56784 to 1.61612.

5.3.2 NSGA-II - Results

In order to compare the results between both optimization algorithms the simulation conducted with NSGA-II used the same parameters. Since this method is multiobjective the results will be presented as Pareto fronts, as demonstrated in figure 5.16.

All simulations were computed using a single design variable: the deformation of the up-right control point

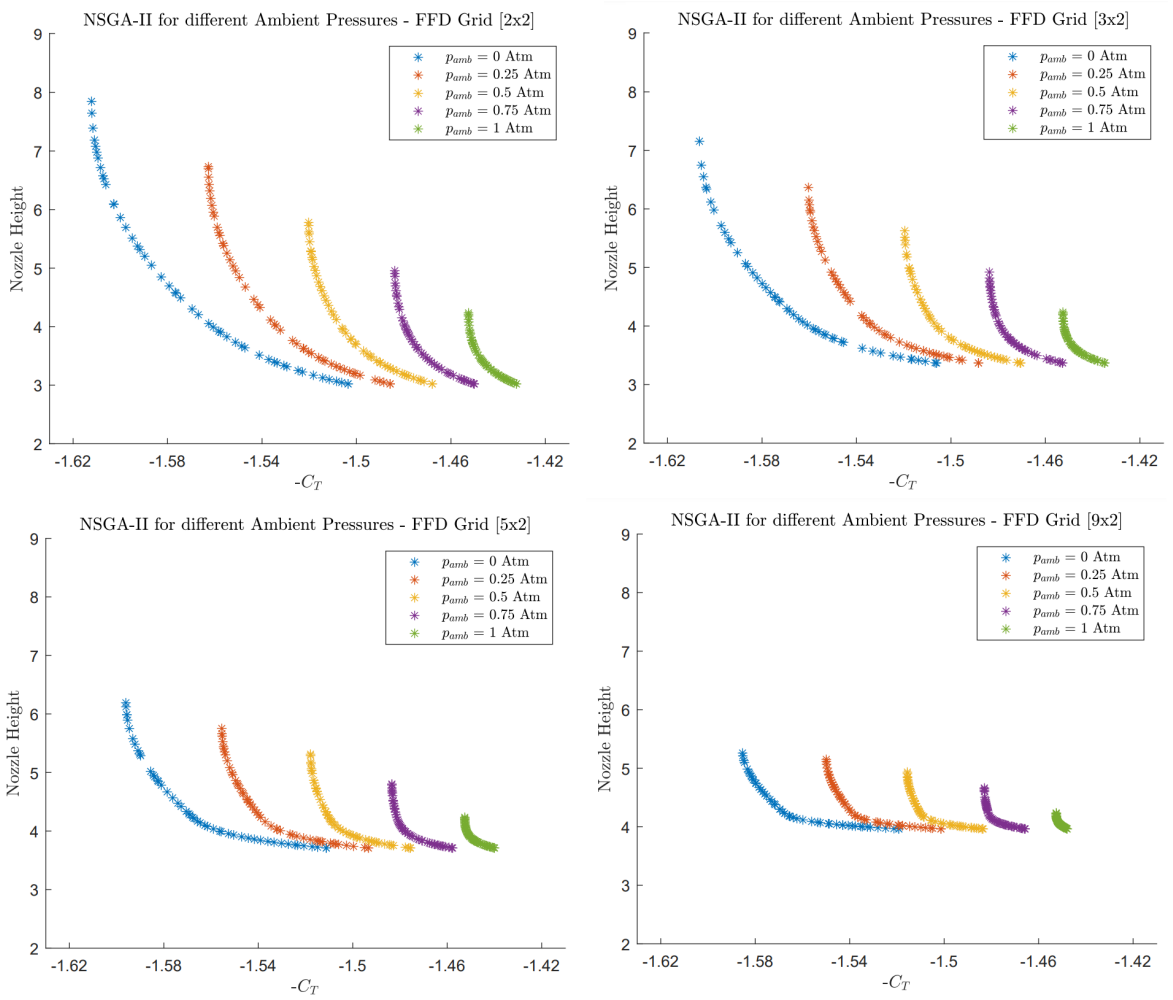


Figure 5.16: Pareto front for different ambient pressure [0, 0.25, 0.5, 0.75, 1] atm and FFD grids [2x2], [3x2], [5x2], [9x2] - NSGA-II.

of the FFD grid, similarly to the Fmincon method. A lower and upper boundary of -0.5 and 2, respectively, were again implemented.

It is possible to observe the trade-off between the coefficient of thrust and the nozzle's height or rather its width, which correlates directly to its drag. Since the nozzle height is directly related to the control point's deformation, the lower boundary bars the height from becoming even lower. This is the reason why the height value of the lower extremity of the Pareto front is equal within the same FFD grid and generated by $\mathbf{x}_{\text{FFD}}$=-0.5.

Regarding the higher extremity, it is attributed to the best coefficient of thrust and should resemble the results of the Fmincon method as shown in table 5.4.

NSGA-II - Results of best Thrust								
Grid Size	Ambient Pressure p_{amb}							
	0 Atm		0.25 Atm		0.5 Atm		0.75 Atm	
	$\mathbf{x}_{\text{FFD}}$	C_T	$\mathbf{x}_{\text{FFD}}$	C_T	$\mathbf{x}_{\text{FFD}}$	C_T	$\mathbf{x}_{\text{FFD}}$	C_T
2x2	1.1167	1.61217	0.7720	1.56271	0.4774	1.52023	0.2223	1.48377
3x2	0.9028	1.60647	0.6589	1.56037	0.4302	1.51956	0.2123	1.48371
5x2	0.6044	1.59610	0.4690	1.55548	0.3332	1.51788	0.1756	1.48345
9x2	0.3170	1.58539	0.2828	1.54993	0.2157	1.51558	0.1321	1.48302

Table 5.4: NSGA-II Results for various FFD grids and ambient pressures - Optimal thrust coefficient

Almost all coefficients of thrust are identical between both methods. Small variations drive from the opposite nature of both optimizers.

Running NSGA-II with two objectives, being one almost linearly proportional to the design variable and using a single design variable seems a little bit futile since the same results could be achieved by calculating the objective function for a range of $\mathbf{x}_{\text{FFD}}$ and determining the Pareto front visually as demonstrated in figure 5.17.

However, this algorithm is extremely effective when using multiple objectives and different design variables that affect the objectives in various alternative ways. However, it has to be taken into consideration that this algorithm is gradient-free (see section 3.9), meaning that too many design variables can become computationally expensive. When running NSGA-II, similarly to Fmincon, for multiple design variables, only the 5x2 FFD grid was implemented, since for a 9x2 grid the algorithm would need to complete more generations to obtain an accurate result meaning taking a longer computational time. Table 5.5 and figure 5.18 present the results of the simulation using multiple design variables for a 5x2 FFD grid.

NSGA-II - Results						
Grid Size	Ambient Pressure $p_{\text{amb}} = 0$ atm					
	$\mathbf{x}_{\text{FFD}_1}$	$\mathbf{x}_{\text{FFD}_2}$	$\mathbf{x}_{\text{FFD}_3}$	$\mathbf{x}_{\text{FFD}_4}$	$\mathbf{x}_{\text{FFD}_5}$	C_T
5x2	0.2248	0.5825	0.6661	0.8760	1.204	1.61390

Table 5.5: NSGA-II Results for various FFD grids using multiple optimization variables in vacuum

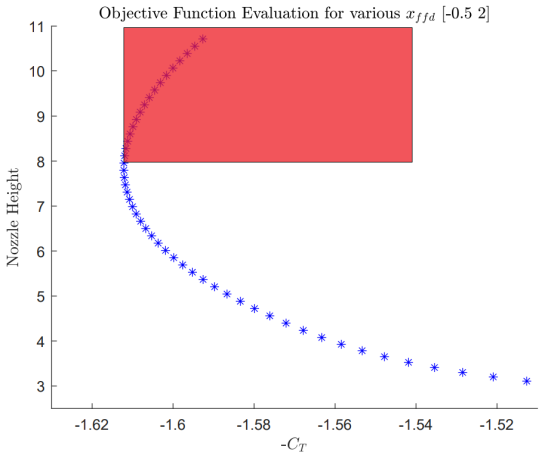


Figure 5.17: Generation of the Pareto front - Red square shows dominated conditions

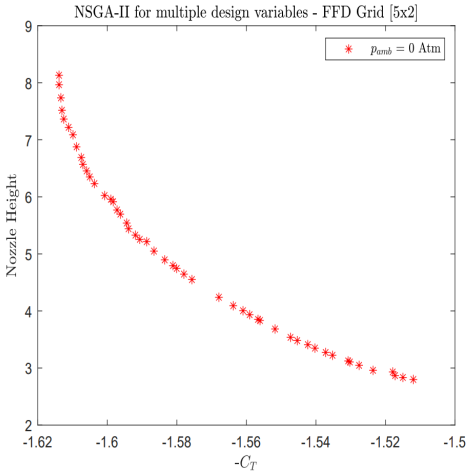


Figure 5.18: Pareto front for multiple design variables in vacuum for [5x2] grid - NSGA-II

Similarly to the result obtained using Fmincon, the optimal coefficient of thrust was enhanced using multiple design variables.

5.4 Computational Fluid Dynamics

In order to validate the optimization completed using the Method of Characteristics it is necessary to use a high-fidelity method to compare its results to the previous ones.

Firstly, one might implement computational fluid dynamics to a minimum-length nozzle (the simplest geometry) to confirm that the fundamentals of the Method of Characteristics are correct. Figure 5.19 displays the flowfield along a minimum-length nozzle for a $M_e = 3$ generated by the MoC. The outcomes are expected since the exit Mach number is approximately 3 and no disturbances (shock and expansion waves) are visible. The contour lines painted in black indicate an increment of 0.1 Mach ranging from 1 to 3 Mach.

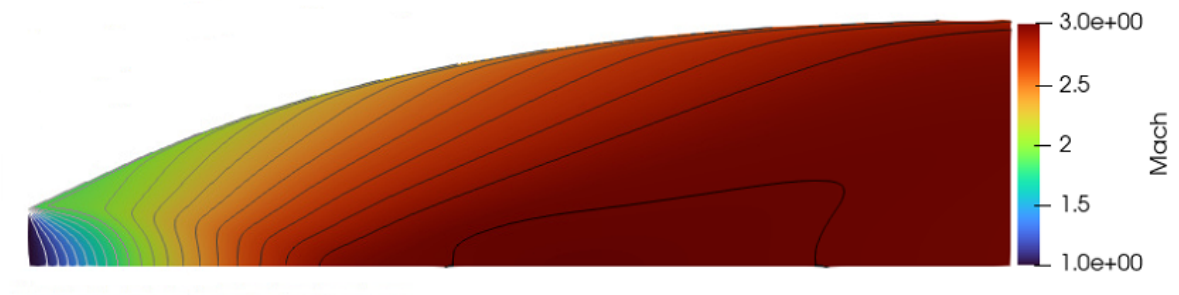


Figure 5.19: Minimum-length nozzle flow field - CFD

After verifying the MoC a grid convergence study is executed for 5 different structured meshes, presented in figure 5.20, in order to choose the most adequate grid sizing for the implementation of the surrogate model. The finest mesh is not displayed because the resolution is not enough to visualise the cell elements.

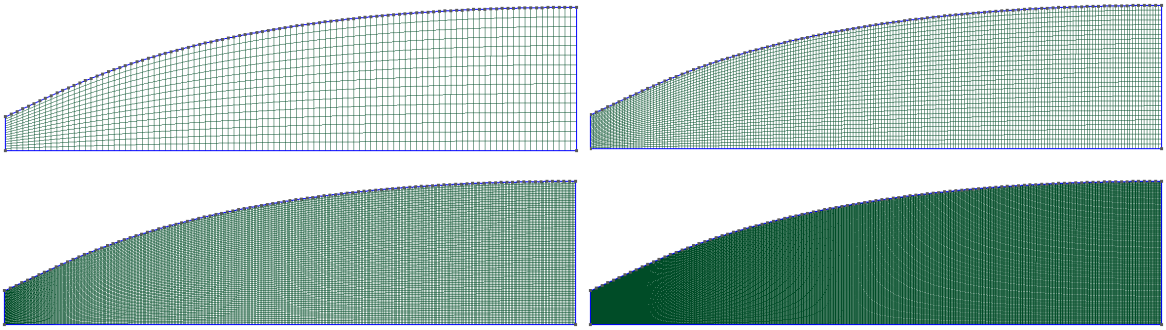


Figure 5.20: Convergence study meshes - [100x15], [200x30], [400x60] and [800x120].

Grid Convergence Study							
Grid Size	Grid Elements	C_T	$C_{T_{1/2}}$	$C_{T_{vac}}$	ΔC_T	Relative Error C_T	Time (s)[1]
100x15	1500	1.4277	1.4853	1.5429	-	1.714%	1.6
200x30	6000	1.4358	1.4934	1.5511	0.567%	1.157%	3.4
400x60	24000	1.4409	1.4985	1.5561	0.355%	0.805%	16.8
800x120	96000	1.4440	1.5016	1.5592	0.215%	0.592%	109.6
1600x240	384000	1.4458	1.5035	1.5611	0.125%	0.468%	770.6
RE	∞	1.4483	1.5065	1.5641	-	0.297%	-
MoC	-	1.4526	1.5102	1.5678	-	-	-

Table 5.6: Grid convergence study for a minimum-length nozzle with $M_e = 3$

[1] Simulation Time for Computer with the following configuration: 64 GB de RAM; e Intel Xeon(R) CPU e5-2620 v2 @ 2.10GHz (x24)

Table 5.6 displays how the coefficient of thrust converges when the number of grid elements is increased as well as the Richardson Extrapolation estimation from the three finest grids (see subsection 4.3.3).

One can say that the grid is converging, because ΔC_T, which calculates the relative error of the coefficient of thrust between two consecutive grids, decreases along the way, indicating that the value of C_T will eventually converge asymptotically to a value. As shown in figure 5.21 this value is expected to be the one calculated through the Richardson Extrapolation (dashed blue line). Compared to the results obtained through the MoC even the coarsest grid presented a relative error lower than 2%, meaning that the Method of Characteristics presents good results, as well as, a coarser grid may be used without the loss of too much accuracy.

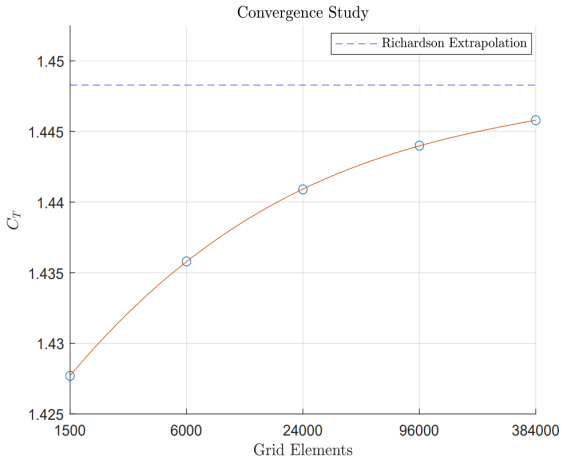


Figure 5.21: Evolution of Thrust Coefficient - Grid Convergence Study

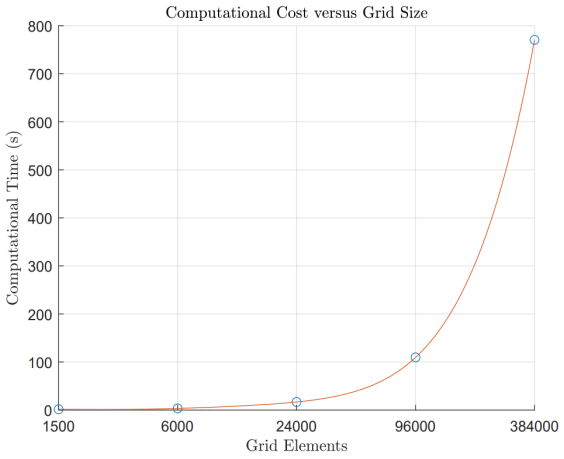


Figure 5.22: Evolution of Computational Cost - Grid Convergence Study

For the implementation of the surrogate model, the data set was computed using the [200x30] grid to keep the computational cost low. As displayed in figure 5.22, the computational time increases exponentially with the increase of grid elements and since a sufficiently large dataset has to be simulated to build the surrogate model it is wiser to keep the computational cost on the lower end.

5.4.1 Surrogate Model - Results

After running the SU^2 simulation for various geometries obtained by the Free Form Deformation of a base geometry for various FFD grid sizing (likewise to the Fmincon and NSGA-II methods) one obtains a training dataset. All points of the training data can then be fitted to a polynomial as demonstrated in figure 5.23.

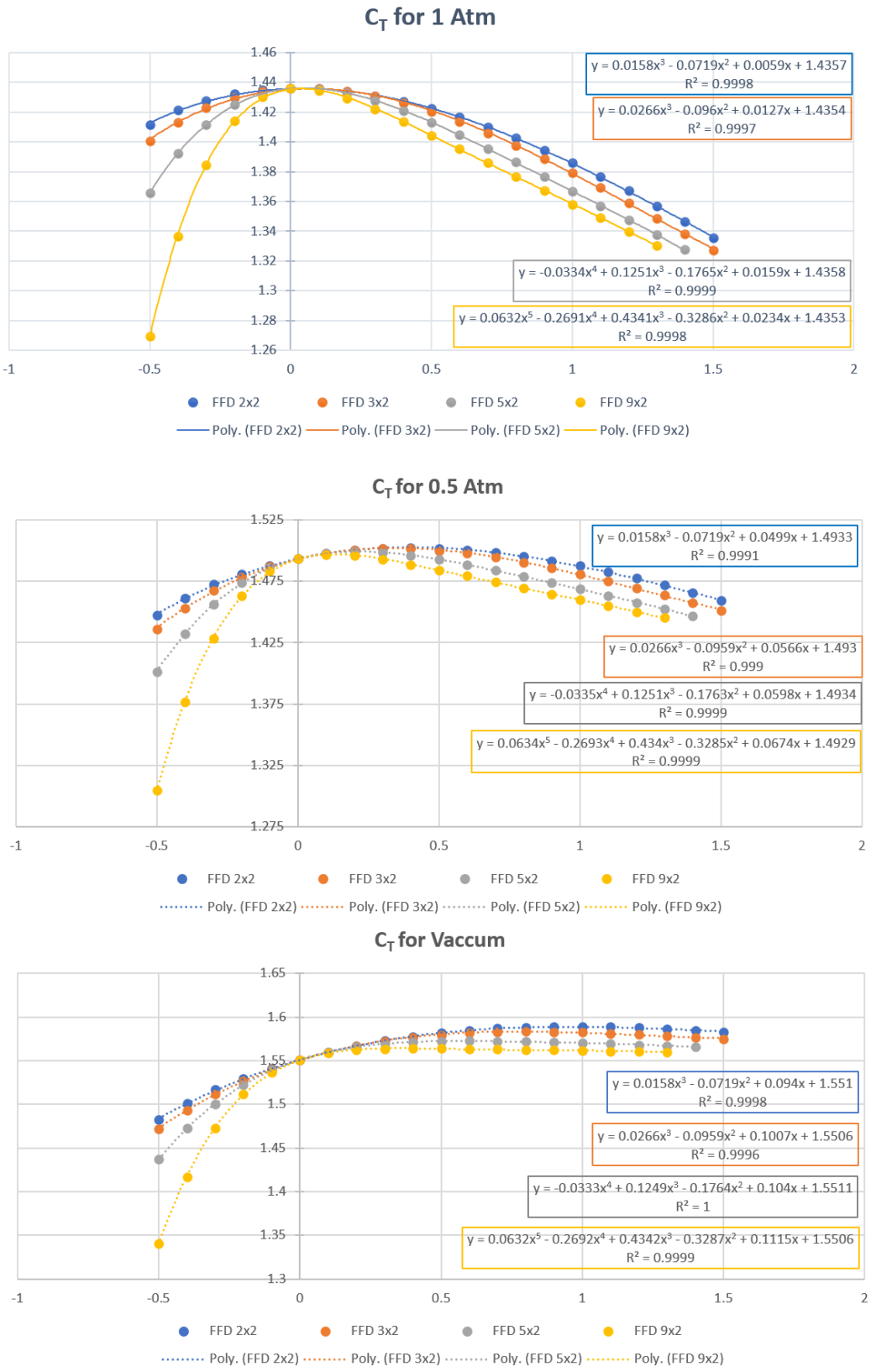


Figure 5.23: Curve fitting of the training data in three different ambient conditions and various FFD grids - Surrogate Model.

Being able to attribute a polynomial to each distribution simplifies the implementation of an optimization method drastically.

To obtain the optimal design for maximum thrust, one only has to apply the derivative to the polynomial expression and equal them to 0, limiting $\mathbf{x}_{\text{FFD}}$ between [-0.5,1.5] resulting in the evidence exhibited in table 5.7.

SU^2 - Results of best Thrust						
Grid Size	Ambient pressure p_{amb}					
	0 Atm		0.5 Atm		1 Atm	
	$\mathbf{x}_{\text{FFD}}$	C_T	$\mathbf{x}_{\text{FFD}}$	C_T	$\mathbf{x}_{\text{FFD}}$	C_T
2x2	0.9531	1.58895	0.400	1.50276	0.0416	1.43582
3x2	0.7748	1.58342	0.3445	1.50220	0.0681	1.43582
5x2	0.5696	1.57268	0.2150	1.49927	0.0474	1.43617
9x2	0.3450	1.56426	0.1347	1.49699	0.0384	1.43573

Table 5.7: SU^2 Results for various FFD grids and ambient pressures - Optimal thrust coefficient

Comparing the results of the SBO, the MoC seems to overestimate the coefficient of thrust. Regarding the optimal deformation, in vacuum the MoC significantly overshoots the value of $\mathbf{x}_{\text{FFD}}$ for coarse FFD grids ($\Delta\mathbf{x}_{\text{FFD}} \approx 0.16$), while for an ambient pressure of 0.5 amb finer FFD grids are the ones dealing with overshooting issues ($\Delta\mathbf{x}_{\text{FFD}} \approx 0.10$). For sea level conditions, the surrogate model outcomes dictate a small deformation, which can be neglected and assumed zero, meaning the nozzle geometry is developed for an optimal nozzle at take-off. The discrepancies can be caused by various factors. These can be divided between limitations of the Method of Characteristics and limitations of the surrogate model.

Limitation of the Surrogate Model

The surrogate model was implemented using a wide array of data points, meaning that the polynomial fitting can lead to errors due to "overfitting". Another limitation is due to the value of the data points themselves being poorly precise because the SU^2 framework only outputs the coefficient of thrust with four decimal places. In some cases, the value of C_T is equal for adjacent design variables, resulting in difficulties in accurately fitting the data distribution to a polynomial without having to increase its order inducing the occurrence of "overfitting".

As mentioned before, during the grid convergence study, the sizing of the mesh applied to a CFD solver has an impact on the underestimation of the coefficient of thrust. Finally, the main limitation of the surrogate model approach is its inability to work with more than one design variable at a time without the curve fitting process becoming too complex.

Limitation of The Method of Characteristics

Being a low-fidelity method, the Method of Characteristics has considerably more limitations that any high-fidelity model.

The main downside of the MoC is its exit conditions not being coincident with the real nozzle exit. This generated error when the flowfield is not uniform leaving the last characteristic line, since part of the flow is not simulated. This is particularly damaging when the deformation applied to the base geometry only occurs

towards the exit (finer FFD grids) since the expansion of the flow near the centre of the nozzle is not computed. Another problem leading to a loss in accuracy is the fact that the characteristic grid becomes coarser along the nozzle, reinforcing the statement that variations in the flow due to geometry changes near the exit are less precisely computed. Additionally, the "overshooting" and intersection of the last characteristic line with a made-up contour, as well as, the interpolation and Riemann sum, induce errors, since the distance between the "virtual" exit characteristic and its adjacent ones is not constant.

One might ask if all these inaccuracies could be avoided by increasing the number of characteristic lines used during the implementation of the MoC. The problem with that statement is that for too fine a grid the characteristic line might collapse when experiencing slight compression. This outcome is strictly undesired since the whole characteristic grid might break down introducing far worse errors than the ones trying to be avoided.

The outcomes presented in table 5.7 can also be visualized in figure 5.24, where the overestimation of both the nozzle width (proportional to the deformation) and the coefficient of thrust by the Method of Characteristics can be identified.

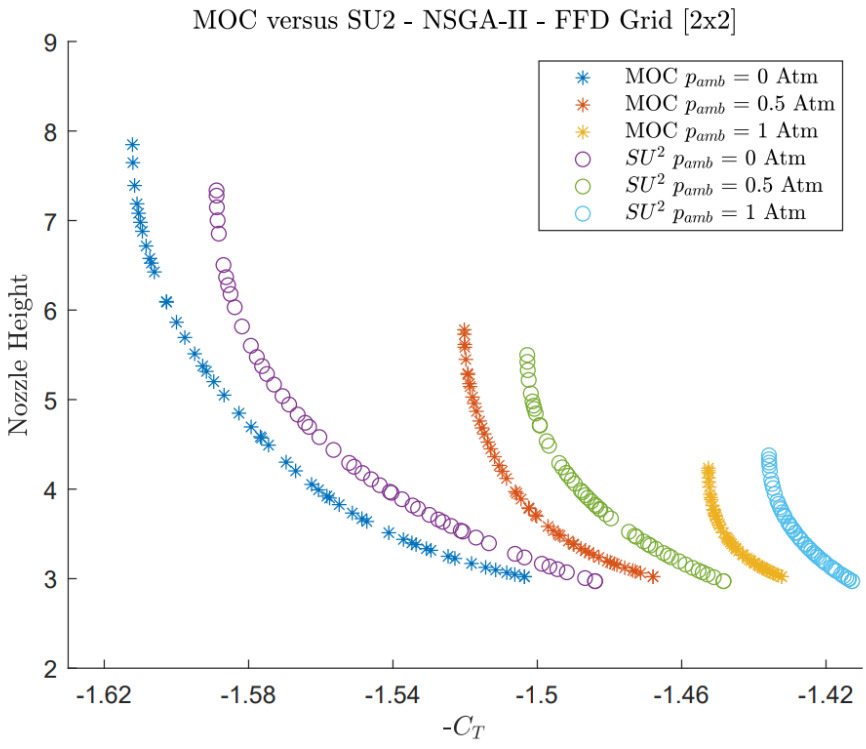


Figure 5.24: Pareto fronts for [2x2] FFD grid - SU^2 versus MoC - NSGA-II.

The deformation done to the nozzle in order to achieve its optimal design resulting from the surrogate-based optimization applied to a high-fidelity simulation tool is demonstrated in figure 5.25.

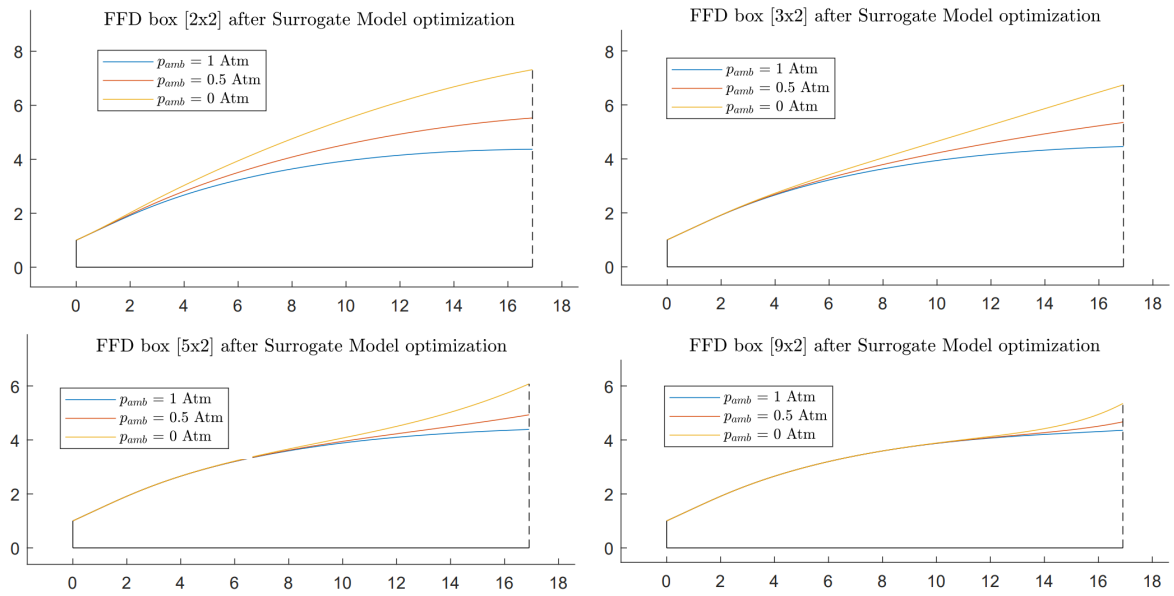


Figure 5.25: Optimal Nozzle Geometry for various FFD grids and ambient pressures - Surrogate Model.

Chapter 6

Conclusions

This chapter will summarise the achievements of this dissertation, followed by suggestions to improve the approach of optimization via the Method of Characteristics and conclude with a few ideas for future work.

6.1 Achievements

The present work consisted in developing a fast and reliable tool for preliminary nozzle design optimization. Throughout the development of this dissertation, many objectives were achieved and obstacles were conquered towards the ultimate goal of testing out the reliability of the proposed low-fidelity optimization method.

Firstly, a simulation tool based on the Method of Characteristics in 2D was developed. It demonstrates the capability of contouring ideal nozzles with high accuracy. For a total number of 100 characteristic lines, the nozzle exit height h_{nozzle} differs from the analytical value by a margin of 0.18%.

After updating the previous algorithm it became also able to compute the evolution of the flowfield variables along arbitrary smooth nozzle geometries, as well as, calculate important values regarding the performance of the nozzle (e.g. its coefficient of thrust). The word smooth is critically mentioned, since for unusual contours the MoC may demonstrate some difficulties in computing the flow field.

A free-form deformation-based parameterization technique was implemented to automatize the generation of nozzle geometries to be evaluated and optimized. Other parameterization techniques based on polynomial and spline approaches along with the Hicks-Henne technique were applied, however, the FFD technique demonstrated clear superiority compared to the others. The first major achievement was attaining an automated procedure to deform a base contour and simulate its flowfield. An ideal nozzle with an exit Mach number of 3 was chosen as the base contour, calibrated for optimum thrust at sea level conditions.

Having established an optimization process, based on the MoC, capable of resembling an objective function two contrasting optimization algorithms were employed to maximize the thrust (and minimizing drag) produced by the nozzle for various ambient pressures. The resulting thrust-optimized contour (TOC) demonstrated being able to be contoured accurately by a parabola, meaning the optimization process deformed an ideal nozzle into a TOP.

This type of contour has been shown to induce the formation of an internal shock initiated from the throat.

This occurrence could be observed by applying the MoC to the thrust-optimized parabola contour. However, as listed in the MoC limitation, the algorithm is not able to simulate the existence of shock waves, due to the collapse of the characteristic lines inducing major inaccuracies. This leads one to believe that an even better contour could be produced. Nevertheless, an improvement of 3.08% is achieved for the coefficient of thrust in vacuum $C_{T_{vac}}$ by the developed optimization process using multiple design variables and a 9x2 FFD grid.

In order to validate the results obtained by the implementation of the Method of Characteristics a CFD simulation using the Euler solver of the SU^2 framework was configured. After getting the solver to properly converge using an ROE numerical scheme and an Euler solver a mesh converge study was conducted and the proper functioning of the CFD solver was confirmed. Applying a Richardson Extrapolation to the various results obtained for $C_{T_{SSL}}$ the study showed a relative error of 0.297% concerning the MoC for an ideal nozzle with $M_e = 3$. For the following simulations, a mesh of 200x30 elements was appointed due to its low computational time of 3.4 seconds and its still negligible relative difference of 1.157%.

Subsequently, a data set of contours was produced using the FFD technique and numerous simulations were run using the high-fidelity solver. An accurate surrogate model based on the CFD training data was built as a way to simplify the optimization process, since implementing a surrogate model-based optimization is considerably computationally cheaper, than running a CFD-based optimization.

Having implemented an optimization based on a high-fidelity method, even though in some sort simplified by the implementation of a surrogate model, the fidelity of the MoC-based optimization was verified. A good agreement regarding the results of both routes was achieved, however with a small offset concerning the design variable $\mathbf{x}_{\text{FFD}}$ and coefficient of thrust C_T for the optimal design. The maximal deviation of the design variable $\Delta\mathbf{x}_{\text{FFD}}$ between both methods showed not to exceed a 20% absolute variation. A list enumerating both methods was established as a way to indicate where the loss in accuracy might derive from.

In conclusion, the present work proved that the MoC is a strong and reliable tool for preliminary design optimization since it reduces the computational cost regarding CFD-based optimization for a small loss in accuracy. The method developed throughout this work is ideal for a multi-fidelity approach.

6.2 Future Work

An Euler solver was implemented when conducting the simulation in CFD to obtain similar results to the Method of Characteristics, because of its inviscid nature. However, to truly judge the MoC merits towards a real flow, a CFD-based optimization should be applied using a RANS solver and a suitable turbulence model. The reason behind this decision is that separation can occur in a real flow. As discussed in chapter 2 the occurrence of separation in a nozzle due to an overly expanded flow can lead to major performance losses.

Also, combining multiple disciplines such as propulsion, aerodynamics, structures, and thermodynamics among others, to obtain an overall optimal performance is an important step to take in future works.

The implementation of the MoC itself is far from optimal and has room for improvement. In the future, a crucial task would be to further develop the algorithm presented in this thesis (e.g. extending the characteristic grid over the whole domain as a way to properly simulate the exit flow). A major advancement would be to extend the algorithm to three dimensions since most nozzles are axisymmetric as opposed to rectangular.

Bibliography

[1] Every Satellite Orbiting Earth and Who Owns Them. https://dewesoft.com/daq/every-satellite-orbiting-earth-and-who-owns-them, January 2022. Last accessed on: October 2022.

[2] How to See Starlink Satellite Train 2022. https://starwalk.space/en/news/spacex-starlink-satellites-night-sky-visibility-guide, October 2022.

[3] Specific Impulse. https://actiflow.com/cfd-the-truth-and-the-tales-2/. Last accessed on: October 2022.

[4] Ideal Rocket Equation. https://www.grc.nasa.gov/www/k-12/rocket/rktpow.html. Last accessed on: October 2022.

[5] G. V. R. Rao. Exhaust Nozzle Contour for Optimum Thrust. *Journal of Jet Propulsion*, 28(6):377–382, 1958. doi:10.2514/8.7324.

[6] E. Terry. CFD: the truth and the tales. https://actiflow.com/cfd-the-truth-and-the-tales-2/, November 2018. Last accessed on: October 2022.

[7] M. N. Thombre, H. A. Preisig, and M. B. Addis. Developing surrogate models via computer based experiments. In K. V. Gernaey, J. K. Huusom, and R. Gani, editors, *12th International Symposium on Process Systems Engineering and 25th European Symposium on Computer Aided Process Engineering*, volume 37 of *Computer Aided Chemical Engineering*, pages 641–646. 2015. doi:10.1016/B978-0-444-63578-5.50102-X.

[8] S. Khare and U. Saha. Rocket nozzles: 75 years of research and development. *Sādhanā*, 46:76, 2021. doi:10.1007/s12046-021-01584-6.

[9] F. R. Harris. The Parsons Centenary—a Hundred Years of Steam Turbines. *Proceedings of the Institution of Mechanical Engineers, Part A: Power and Process Engineering*, 198(3):183–224, 1984. doi:10.1243/PIME_PROC_1984_198_024_02.

[10] R. H. Goddard. *A method of reaching extreme altitudes*. 1919. doi:10.5479/sil.918318.39088014683783.

[11] G. P. Sutton and O. Biblarz. *Rocket Propulsion Elements*. John Wiley & Sons, Inc., seventh edition, 2001. ISBN 0-471-32642-9.

[12] A. H. Shapiro. *The dynamics and thermodynamics of compressible fluid flow. 1*. Wiley, New York, 23. print edition, 1976. ISBN 9780471066910.

[13] S. A. Whitmore. The optimum rocket nozzle. `http://mae-nas.eng.usu.edu/MAE_5420_Web/section5/section.5.3.pdf`, . Last accessed on October 2022.

[14] R. A. O'Leary and J. E. Bech. Nozzle Design. `http://www.rocket-propulsion.info/resources/articles/NozzleDesign.pdf`, 1992.

[15] M. Frey, K. Makowka, and T. Aichner. The tictop nozzle: a new nozzle contouring concept. *CEAS Space Journal*, 9(2):175–181, 2017. doi:10.1007/s12567-016-0139-z.

[16] F. J. Malina. Characteristics of the rocket motor unit based on the theory of perfect gases. *Journal of the Franklin Institute*, 230(4):433–454, 1940. doi:10.1016/S0016-0032(40)91348-5.

[17] G. V. R. Rao. Recent Developments in Rocket Nozzle Configurations. *ARS Journal*, 31(11):1488–1494, 1961. doi:10.2514/8.5837.

[18] H. M. Darwell and H. Badham. Shock formation in conical nozzles. *AIAA Journal*, 1(8):1932–1934, 1963. doi:10.2514/3.1965.

[19] J. G. Allman and J. D. Hoffman. Design of maximum thrust nozzle contours by direct optimization methods. *AIAA Journal*, 19(6):750–751, 1981. doi:10.2514/3.50999.

[20] J. H. Ahlberg, S. Hamilton, D. Migdal, and E. N. Nilson. Truncated Perfect Nozzles in Optimum Nozzle Design. *ARS Journal*, 31(5):614–620, 1961. doi:10.2514/8.5577.

[21] G. V. R. Rao. Approximation of Optimum Thrust Nozzle Contour. *ARS Jounal*, 30(6), June 1960. URL `https://cir.nii.ac.jp/crid/1572261550453372544`.

[22] I. Mohamed and R. Gopalapillai. *Shock Interactions in Thrust Optimised Parabolic (TOP) Nozzles during Start-Up and Shutdown*, pages 531–541. 04 2019. doi:10.1007/978-3-319-91017-8_68.

[23] A. Hadjadj and M. Onofri. Nozzle flow separation. *Shock Waves*, 19(3):163–169, 2009. doi:10.1007/s00193-009-0209-7.

[24] A. Shams, S. Girard, and P. Comte. Numerical simulation of shock-induced separated flows in overexpanded rocket nozzles. *Progress in Flight Physics*, 3:169–190, 2012. doi:10.1051/eucass/201203169.

[25] E. Martelli, F. Nasuti, and M. Onofri. Numerical calculation of fss/rss transition in highly overexpanded rocket nozzle flows. *Shock Waves*, 20:139–146, 2010.

[26] A. Conte, A. Ferrero, F. Larocca, and D. Pastrone. Numerical Tool Optimization for Advanced Rocket Nozzle Performance Prediction. In *AIAA Propulsion and Energy 2019 Forum*, Indianapolis, IN, USA, August 2019. doi:10.2514/6.2019-4115.

[27] M. Frey and G. Hagemann. Restricted shock separation in rocket nozzles. *Journal of Propulsion and Power*, 16(3):478–484, 2000. doi:10.2514/2.5593.

[28] G. Hagemann, H. Immich, and M. Terhardt. Flow phenomena in advanced rocket nozzles - the plug nozzle. In *34th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, July 1998. doi:10.2514/6.1998-3522.

[29] J. L. Tapee. Experimental aerodynamic analysis of a plug nozzle for supersonic business jet application. Master's thesis, Purdue University, West Lafayette, Indiana, August 2009.

[30] K. HERMAN and F. W. CRIMP Jr. Performance of plug-type rocket exhaust nozzles. *ARS Journal*, 31(1): 18–23, 1961. doi:10.2514/8.5373.

[31] G. Hagemann, H. Immich, T. Van Nguyen, and G. E. Dumnov. Advanced rocket nozzles. *Journal of Propulsion and Power*, 14(5):620–634, 1998. doi:10.2514/2.5354.

[32] K. A. Schomberg, G. Doig, J. Olsen, and A. J. Neely. Geometric analysis of the linear expansion-deflection nozzle at highly overexpanded flow conditions. In *50th AIAA/ASME/SAE/ASEE Joint Propulsion Conference*. doi:10.2514/6.2014-4001. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2014-4001`.

[33] G. V. R. Rao. *Analysis Of A New Concept Rocket Nozzle*, pages 669–682. doi:10.2514/5.9781600864759.0669.0682.

[34] R. Stark, C. Génin, D. Schneider, and C. Fromm. Ariane 5 Performance Optimization Using Dual-Bell Nozzle Extension. *Journal of Spacecraft and Rockets*, 53:743–750, 06 2016. doi:10.2514/1.A33363.

[35] M. Horn and S. Fisher. Dual-bell altitude compensating nozzles. `https://ntrs.nasa.gov/citations/19940018584`, November 1993. Pennsylvania State Univ., NASA Propulsion Engineering Research Center.

[36] D. Chasman, M. Birch, S. Haight, and R. Graffam. A Multi-Disciplinary Optimization Method for Multi Nozzle Grid (MNG) Design - Final Report. In *43rd AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, USA, January 2005. doi:10.2514/6.2005-706.

[37] D. Chasman, S. Haight, and R. Loehr. Viscous losses of MNG in Hybrid Motor Tests. In *48th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit*, Atlanta, Georgia, USA, August . doi:10.2514/6.2012-4266.

[38] M. Colonno, E. Van der Weide, and J. J. Alonso. The Optimum Vacuum Nozzle: an MDO Approach. In *46th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, USA, January 2008. doi:10.2514/6.2008-911.

[39] M. Yumusak and S. Eyi. Design optimization of rocket nozzles in chemically reacting flows. *Computers Fluids*, 65:25–34, 2012. doi:10.1016/j.compfluid.2012.05.002.

[40] D. Sun, T. Luo, and Q. Feng. New Contour Design Method for Rocket Nozzle of Large Area Ratio. *International Journal of Aerospace Engineering*, 2019:4926413, 2019. doi:10.1155/2019/4926413.

[41] J. D. Hoffman. Approximate analysis of nonisentropic flow in conical nozzles. *Journal of Spacecraft and Rockets*, 6(11):1329–1334, 1969. doi:10.2514/3.29824.

[42] K. Kumar, G. Murugesan, D. Antony, K. Ravichandran, and C. Viswanadh. Design and Optimization of Aerospike nozzle using CFD. *IOP Conference Series: Materials Science and Engineering*, 247:012008, 2017. doi:10.1088/1757-899X/247/1/012008.

[43] G. Cai, J. Fang, X. Xu, and M. Liu. Performance prediction and optimization for liquid rocket engine nozzle. *Aerospace Science and Technology*, 11(2):155–162, 2007. doi:10.1016/j.ast.2006.07.002.

[44] M. GOEING. *Nozzle design optimization by method-of-characteristics*. doi:10.2514/6.1990-2024. URL `https://arc.aiaa.org/doi/abs/10.2514/6.1990-2024`.

[45] V. Matveev, V. Zubanov, L. Shabliy, and A. Korneeva. Optimization of nozzle shape of hydrogen-oxygen rocket engine. pages 365–370, 01 2018. doi:10.5220/0006890003650370.

[46] D. DiDominic, E. Gist, J. Fitzgerald, and M. N. Glauser. *Complex Nozzle Optimization Techniques using Machine Learning*. doi:10.2514/6.2020-1866. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2020-1866`.

[47] J. A. Samareh. Survey of Shape Parameterization Techniques for High-Fidelity Multidisciplinary Shape Optimization. *AIAA Journal*, 39(5):877–884, 2001. doi:10.2514/2.1391.

[48] T. Dang, B. Li, D. Hu, Y. Sun, and Z. Liu. Aerodynamic design optimization of a hypersonic rocket sled deflector using the free-form deformation technique. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 235(15):2240–2248, 2021. doi:10.1177/0954410021994984.

[49] T. W. Sederberg and S. R. Parry. Free-Form Deformation of Solid Geometric Models. *ACM SIGGRAPH Computer Graphics*, 20(4):151–160, 1986. doi:10.1145/15886.15903.

[50] A. Koshakji, A. Quarteroni, and G. Rozza. Free Form Deformation Techniques Applied to 3D Shape Optimization Problems. *Communications in Applied and Industrial Mathematics*, 4, 2013. doi:10.1685/journal.caim.452.

[51] J. Greissmair and W. Purgathofer. Deformation of Solids with Trivariate B-splines. In *10th Eurographics '89*, Hamburg, Germany, September 1989.

[52] S. Coquillart. Extended Free-Form Deformation: A Sculpturing Tool for 3D Geometric Modeling. *ACM SIGGRAPH Computer Graphics*, 24(4):187–196, 1990. doi:10.1145/97880.97900.

[53] J. D. Anderson. *Fundamentals of Aerodynamics*. McGraw Hill Education, sixth edition, 2017. ISBN 9781259129919.

[54] Material Derivative. `https://www.continuummechanics.org/materialderivative.html`. Last accessed on: October 2022.

[55] J. Stevens. Is Incompressible Good Enough? `http://heat-transfer-thermodynamics.blogspot.com/2015/05/is-incompressible-good-enough.html`, May 2015. Last accessed o:n October 2022.

[56] R. Qin and C. Duan. The principle and applications of bernoulli equation. *Journal of Physics: Conference Series*, Oct 2017. doi:10.1088/1742-6596/916/1/012038. URL https://doi.org/10.1088/1742-6596/916/1/012038.

[57] W. J. Devenport. An educational Java Applet for those studying converging-diverging nozzle flows, Version 1.0. https://www.engapplets.vt.edu/fluids/CDnozzle/cdinfo.html. Last accessed on October 2022.

[58] I. M. Hall. Inversion of the Prandtl-Meyer relation. *The Aeronautical Journal (1968)*, 79(777):417–418, 1975. doi:10.1017/S0001924000035892.

[59] A. Ferri. *G. The Method of Characteristics*, pages 583–669. Princeton University Press, Princeton, 2015. ISBN 9781400877553. doi:doi:10.1515/9781400877553-009. URL https://doi.org/10.1515/9781400877553-009.

[60] J. Abramson. Solving Systems with Cramer's Rule. https://openstax.org/books/college-algebra/pages/7-8-solving-systems-with-cramers-rule. Last accessed on October 2022.

[61] S. A. Whitmore. Introduction to the method of characteristics and the minimum length nozzle. http://mae-nas.eng.usu.edu/MAE_5540_Web/propulsion_systems/section8/section.8.1.pdf, . Last accessed on October 2022.

[62] What Are Boundary Conditions? https://www.simscale.com/docs/simwiki/numerics-background/what-are-boundary-conditions/. Last accessed on: October 2022.

[63] F. Moukalled, L. Mangani, and M. Darwish. *The Finite Volume Method in Computational Fluid Dynamics: An Advanced Introduction with OpenFOAM® and Matlab*. Springer Cham, 2016. doi:10.1007/978-3-319-16874-6.

[64] Wall Boundary Condition. https://www.simscale.com/docs/simulation-setup/boundary-conditions/wall//. Last accessed on: October 2022.

[65] B. E. Rapp. Chapter 3 - engineering mathematics. In B. E. Rapp, editor, *Microfluidics: Modelling, Mechanics and Mathematics*, Micro and Nano Technologies, pages 21–50. Elsevier, Oxford, 2017. ISBN 978-1-4557-3141-1. doi:10.1016/B978-1-4557-3141-1.50003-4.

[66] R. M. Cummings, W. H. Mason, S. A. Morton, and D. R. McDaniel. *Applied Computational Aerodynamics: A Modern Engineering Approach*. Cambridge Aerospace Series. Cambridge University Press, 2015. doi:10.1017/CBO9781107284166.

[67] J. P. B. Lourenço. Supersonic and Transonic Adjoint-based Optimization of Airfoils. Master's thesis, Instituto Superior Técnico, November 2018.

[68] T. Van To and H. Ngoc Phien. Development of Bézier-based curves. *Computers in Industry*, 20(1):109–115, 1992. doi:https://doi.org/10.1016/0166-3615(92)90132-7.

[69] S. Baydas and B. Karakas. Defining a curve as a Bezier curve. *Journal of Taibah University for Science*, 13 (1):522–528, 2019. doi:10.1080/16583655.2019.1601913.

[70] V. Sripawadkul, M. Padulo, and M. Guenov. A Comparison of Airfoil Shape Parameterization Techniques for Early Design Optimization. In *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Fort Worth, Texas, USA, September 2010. doi:10.2514/6.2010-9050.

[71] B. Bertka. An Introduction to Bezier Curves, B-Splines, and Tensor Product Surfaces with History and Applications. `https://pzs.dstu.dp.ua/DataMining/spline/bibl/IntroSplines.pdf`, May 2008. University of California Santa Cruz. Last accessed on: October 2022.

[72] A. Sobester and T. Barretty. Quest for a Truly Parsimonious Airfoil Parameterization Scheme. *The 26th Congress of ICAS and 8th AIAA ATIO*, September 2008. doi:10.2514/6.2008-8879.

[73] R. M. Hicks and P. A. Henne. Wing Design by Numerical Optimization. *Journal of Aircraft*, 15(7):407–412, 1978. doi:10.2514/3.58379.

[74] J. Procházková. Free form deformation methods the theory and practice. In *16th Conference on Applied Mathematics, APLIMAT*, Bratislava, Slovak Republic, January-February 2017.

[75] J. R. R. A. Martins and A. Ning. *Engineering Design Optimization*. Cambridge University Press, 2021. doi:10.1017/9781108980647.

[76] C. J. F. Ribeiro. Surrogate based Multidisciplinary Design Optimization of a VTOL aircraft. Master's thesis, Instituto Superior Técnico, May 2021.

[77] W.-L. Loh. On Latin hypercube sampling. *The Annals of Statistics*, 24(5):2058 – 2080, 1996. doi:10.1214/aos/1069362310.

[78] V. D. Kulkarni. Lecture 34 in Gas Dynamics. `https://nptel.ac.in/courses/112103021`. Last accessed on: October 2022.

[79] M. Khan, S. Kumar, M. Sharath, and H. Chowdary. Design of a Supersonic Nozzle using Method of Characteristics. *International Journal of Engineering Research & Technology*, 2(11), 2013. doi:10.17577/IJERTV2IS110026.

[80] S. Axler. *Riemann Integration*, volume 282, pages 1–12. 2020. ISBN 978-3-030-33142-9. doi:10.1007/978-3-030-33143-6_1.

[81] A. Barclay, P. E. Gill, and J. Ben Rosen. Sqp methods and their application to numerical optimal control. In W. H. Schmidt, K. Heier, L. Bittner, and R. Bulirsch, editors, *Variational Calculus, Optimal Control and Applications*, pages 207–222, Basel, 1998. Birkhäuser Basel. ISBN 978-3-0348-8802-8.

[82] R. Cambini and C. Sodini. A sequential method for a class of box constrained quadratic programming problems. *Mathematical Methods of Operations Research*, 67(2):223–243, 2008. doi:10.1007/s00186-007-0173-x.

[83] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002. doi:10.1109/4235.996017.

[84] A. Seshadri. NSGA - II: A multi-objective optimization algorithm. `https://www.mathworks.com/matlabcentral/fileexchange/10429-nsga-ii-a-multi-objective-optimization-algorithm`, 2022. Last accessed on: 2022-10-23.

[85] T. D. Economon, F. Palacios, S. R. Copeland, T. W. Lukaczyk, and J. J. Alonso. SU2: An Open-Source Suite for Multiphysics Simulation and Design. *AIAA Journal*, 54(3):828–846, 2016. doi:10.2514/1.J053813.

[86] Courant number in CFD simulations. `https://www.idealsimulations.com/resources/courant-number-cfd/`. Last accessed on: October 2022.

[87] P. Wesseling. *Principles of Computational Fluid Dynamics*, volume 29 of *Springer Series in Computational Mathematics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. ISBN 9783642051456 9783642051463. doi:10.1007/978-3-642-05146-3.

[88] Examining Spatial (Grid) Convergence. `https://www.grc.nasa.gov/www/wind/valid/tutorial/spatconv.html`, February 2021. Last accessed on: October 2022.

[89] M. Murnaghan. Study of minimum length, supersonic nozzle design using the method of characteristics. Master's degree in space and aeronautical engineering, Universitat Politècnica de Catalunya, June 2019.

[90] A. Mianowski and W. Urbańczyk. Thermal dissociation in terms of the second law of chemical thermodynamics. *Journal of Thermal Analysis and Calorimetry*, 126:863–870, 2016. doi:10.1007/s10973-016-5569-5.

[91] J. Urzay. The physical characteristics of hypersonic flows. `https://web.stanford.edu/~jurzay/hypersonicsCh2_Urzay.pdf`, 2020. Center for Turbulence Research, Stanford University, Stanford. Last accessed on: October 2022.

Appendix A

MATLAB® Code

Functions in MATLAB®

Fmincon Optimization

```
lb = [-0.5,-0.5,-0.5,-0.5,0.5];
ub = [2,2,2,2,2];
x0 = [0,0,0,0,0];
ob = @(x) obj_fun(x);
nl = @(x) constraint(x);
opts = optimset('Algorithm','sqp','Display','iter','tolcon',1e-3);
[x_opt,fval,output] = fmincon(ob,x0,[],[],[],[],lb,ub,nl,opts);
```

Objective Function

```
function f = obj_fun(x)
%Free Form Deformation
[x_curve,y_curve]=ffd_y(x);
slope = zeros();
x0 = x_curve(1);
D = y_curve(1);
L = x_curve(end);
for i=1:1:length(x_curve)
    if i==1
        slope(i) = (y_curve(i+1) - y_curve(i))/(x_curve(i+1) - x_curve(i));
    elseif i==length(x_curve)
        slope(i) = (y_curve(i) - y_curve(i-1))/(x_curve(i) - x_curve(i-1));
    else
        slope(i) = (y_curve(i+1) - y_curve(i-1))/(x_curve(i+1) - x_curve(i-1));
    end
end
n =100;
```

```
theta0 = atand(slope(1));
x_curve(end+1) = x_curve(end)+100;
y_curve(end+1) = y_curve(end)+100*slope(end);
slope(end+1) = slope(end);
%% Prandtl-Expansion %%
tt0=zeros(1,n);
v0=zeros(1,n);
mu0=zeros(1,n);
M0=zeros(1,n);
tt = zeros();
v = zeros();
KL = zeros();
KR = zeros();
M = zeros();
mu = zeros();
x = zeros();
y = zeros();
for i=1:n
    x=x+90/n;
    tt0(i)=theta0*(1-cosd(x));
end
for i=1:n
    v0(i)=tt0(i);
    M0(i)=InvPrandtlMeyer(v0(i));
    mu0(i)=Mu(M0(i));
end
i=1;
tt(i)=0;
v(i)=2*tt0(i);
KL(i)=tt(i)-v(i);
KR(i)=tt(i)+v(i);
M(i)=InvPrandtlMeyer(v(i));
mu(i)=Mu(M(i));
x(i)= -D/tand((tt0(i)-(mu0(i)+mu(i)))/2)+x0;
y(i)=0;
i=i+1;
for c=1:n-1
    KL(i)=KL(i-1);
    KR(i)=2*tt0(i);
    tt(i)=0.5*(KL(i)+KR(i));
    v(i)=0.5*(KR(i)-KL(i));
    M(i)=InvPrandtlMeyer(v(i));
    mu(i)=Mu(M(i));
    slopeL = (tt(i)+tt(i-1)+(mu(i)+mu(i-1)))/2;
    slopeR = (tt(i)+tt0(i)-(mu(i)+mu0(i)))/2;
    x(i) = (x0*tand(slopeR)-x(i-1)*tand(slopeL)
    +y(i-1)-D)/(tand(slopeR)-tand(slopeL));
    y(i) = (tand(slopeR)*tand(slopeL)*(x0-x(i-1))
    +tand(slopeR)*y(i-1)-tand(slopeL)*D)/(tand(slopeR)-tand(slopeL));
```

```
    i=i+1;
end
index = find(x(i-1)-x_curve < 0, 1,'first');
start = index-1;
tt(i) = tt(i-1);
v(i)= v(i-1);
KL(i)=KL(i-1);
KR(i)=KR(i-1);
M(i)=InvPrandtlMeyer(v(i));
mu(i)=Mu(M(i));
SL = tand(((tt(i-1)+tt(i))+(mu(i-1)+mu(i)))/2);
char_line =[SL y(i-1)-SL*x(i-1)];
x_st = x_curve(start:end);
y_st = y_curve(start:end);
y_line = polyval(char_line, x_st);
y_dif = y_st - y_line;
index = find(y_dif < 0, 1, 'first');
if isempty(index)==1 || index < 2 || isreal(y_dif)==0
    f=100;
    return
end
x_int = interp1(y_dif(index-1:index),x_st(index-1:index),0);
SL_new = interp1(x_curve,slope,x_int);
tt_new = atand(SL_new);
while (abs(tt(i)-tt_new) ≥ 0.001)
    tt(i) = tt_new;
    v(i)=tt(i)-KL(i-1);
    KL(i)=KL(i-1);
    KR(i)=tt(i)+v(i);
    M(i)=InvPrandtlMeyer(v(i));
    mu(i)=Mu(M(i));
    SL = tand(((tt(i-1)+tt(i))+(mu(i-1)+mu(i)))/2);
    char_line =[SL y(i-1)-SL*x(i-1)];
    x_st = x_curve(start:end);
    y_st = y_curve(start:end);
    y_line = polyval(char_line, x_st);
    y_dif = y_st - y_line;
    index = find(y_dif < 0, 1, 'first');
    if isempty(index)==1 || index < 2 || isreal(y_dif)==0
        f=100;
        return
    end
    x_int = interp1(y_dif(index-1:index),x_st(index-1:index),0);
    SL_new = interp1(x_curve,slope,x_int);
    tt_new = atand(SL_new);
end
theta_0 = tt_new;
%% Initiation Point %%
tt0=zeros(1,n);
```

```
v0=zeros(1,n);
mu0=zeros(1,n);
M0=zeros(1,n);
x=0;
for i=1:n
    x=x+90/n;
    tt0(i)=theta_0*(1-cosd(x));
end
for i=1:n
    v0(i)=tt0(i);
    M0(i)=InvPrandtlMeyer(v0(i));
    mu0(i)=Mu(M0(i));
end
%% Array Preallocation %%
tt = zeros();
v = zeros();
KL = zeros();
KR = zeros();
M = zeros();
mu = zeros();
x = zeros();
y = zeros();
x_w = zeros();
y_w = zeros();
tt_w = zeros();
M_w = zeros();
%% 2D MOC %%
l=1;
i=1;
x_end = 0;
while (x_end < L)
    if l==1
        tt(i)=0;
        v(i)=2*tt0(i);
        KL(i)=tt(i)-v(i);
        KR(i)=tt(i)+v(i);
        M(i)=InvPrandtlMeyer(v(i));
        mu(i)=Mu(M(i));
        x(i)= -D/tand((tt0(i)-(mu0(i)+mu(i)))/2)+x0;
        y(i)=0;
        i=i+1;
        for c=1:n-1
            KL(i)=KL(i-1);
            KR(i)=2*tt0(i);
            tt(i)=0.5*(KL(i)+KR(i));
            v(i)=0.5*(KR(i)-KL(i));
            M(i)=InvPrandtlMeyer(v(i));
            mu(i)=Mu(M(i));
            slopeL = (tt(i)+tt(i-1)+(mu(i)+mu(i-1)))/2;
```

```
            slopeR = (tt(i)+tt0(i)-(mu(i)+mu0(i)))/2;
            x(i) = (x0*tand(slopeR)-x(i-1)*tand(slopeL)
            +y(i-1)-D)/(tand(slopeR)-tand(slopeL));
            y(i) = (tand(slopeR)*tand(slopeL)*(x0-x(i-1))
            +tand(slopeR)*y(i-1)-tand(slopeL)*D)/(tand(slopeR)-tand(slopeL));
            i=i+1;
        end
    else
        tt(i)=0;
        v(i)=KR(i-n);
        KL(i)=tt(i)-v(i);
        KR(i)=tt(i)+v(i);
        M(i)=InvPrandtlMeyer(v(i));
        mu(i)=Mu(M(i));
        x(i)= -y(i-n)/tand((tt(i-n)-(mu(i-n)+mu(i)))/2)+x(i-n);
        y(i)=0;
        i=i+1;
        for c=1:n-1
            KL(i)=KL(i-1);
            KR(i)=KR(i-n);
            tt(i)=0.5*(KL(i)+KR(i));
            v(i)=0.5*(KR(i)-KL(i));
            M(i)=InvPrandtlMeyer(v(i));
            mu(i)=Mu(M(i));
            slopeL = (tt(i)+tt(i-1)+(mu(i)+mu(i-1)))/2;
            slopeR = (tt(i)+tt(i-n)-(mu(i)+mu(i-n)))/2;
            x(i) = (x(i-n)*tand(slopeR)-x(i-1)*tand(slopeL)
            +y(i-1)-y(i-n))/(tand(slopeR)-tand(slopeL));
            y(i) = (tand(slopeR)*tand(slopeL)*(x(i-n)-x(i-1))
            +tand(slopeR)*y(i-1)-tand(slopeL)*y(i-n))/(tand(slopeR)-tand(slopeL));
            if x(i)<x(i-1) || y(i)<y(i-1)
                f = 100;
                return;
            end
            i=i+1;
        end
    end
    index = find(x(i-1)-x_curve < 0, 1,'first');
    start = index-1;
    iter = 1;
    tt(i) = tt(i-1);
    v(i)= v(i-1);
    KL(i)=KL(i-1);
    KR(i)=KR(i-1);
    M(i)=InvPrandtlMeyer(v(i));
    mu(i)=Mu(M(i));
    SL = tand(((tt(i-1)+tt(i))+(mu(i-1)+mu(i)))/2);
    char_line =[SL y(i-1)-SL*x(i-1)];
    x_st = x_curve(start:end);
```

```
    y_st = y_curve(start:end);
    y_line = polyval(char_line, x_st);
    y_dif = y_st - y_line;
    index = find(y_dif < 0, 1, 'first');
    if isempty(index)==1 || index < 2 || isreal(y_dif)==0
        f=100;
        return
    end
    x_int = interp1(y_dif(index-1:index),x_st(index-1:index),0);
    y_int = polyval(char_line,x_int);
    SL_new = interp1(x_curve,slope,x_int);
    tt_new = atand(SL_new);
    while (abs(tt(i)-tt_new) ≥ 0.001) && (iter≤50)
        tt(i) = tt_new;
        v(i)=tt(i)-KL(i-1);
        KL(i)=KL(i-1);
        KR(i)=tt(i)+v(i);
        M(i)=InvPrandtlMeyer(v(i));
        mu(i)=Mu(M(i));
        SL = tand(((tt(i-1)+tt(i))+(mu(i-1)+mu(i)))/2);
        char_line =[SL y(i-1)-SL*x(i-1)];
        x_st = x_curve(start:end);
        y_st = y_curve(start:end);
        y_line = polyval(char_line, x_st);
        y_dif = y_st - y_line;
        index = find(y_dif < 0, 1, 'first');
        if isempty(index)==1 || index < 2 || isreal(y_dif)==0
            f=100;
            return
        end
        x_int = interp1(y_dif(index-1:index),x_st(index-1:index),0);
        y_int = polyval(char_line,x_int);
        SL_new = interp1(x_curve,slope,x_int);
        tt_new = atand(SL_new);
        iter = iter + 1;
    end
    x(i) = x_int;
    y(i) = y_int;
    x_w(l) = x(i);
    y_w(l) = y(i);
    tt_w(l) = tt(i);
    x_end = x(i);
    M_w(l) = M(i);
    i=i+1;
    l=l+1;
end
x_curve(end) = [];
y_curve(end) = [];
%Flowfield variables
```

```
%Chamber Temperature & Pressure
T0 = 3000;
p0 = 3723300;
points = length(M);
%Universal Gas Constant
R = 8314.46261815324; %J/Kmmol
MM = 28.97; %g/mol
R_spec = R/MM;
gamma = 1.4;
T = zeros();
p = zeros();
a = zeros();
vel = zeros();
for i=1:points
    p(i)=p0*(1+(gamma-1)/2 * M(i)^2)^(-gamma/(gamma-1));
    T(i)=T0*(1+(gamma-1)/2 * M(i)^2)^(-1);
    a(i)=sqrt(gamma*R_spec*T(i));
    vel(i)=M(i)*a(i);
end
%Average Exit
u_e1 = vel(end-(2*n+1):end-(n+1));
p_e1 = p(end-(2*n+1):end-(n+1));
y_e1 = y(end-(2*n+1):end-(n+1));
tt_e1 = tt(end-(2*n+1):end-(n+1));
u_x1 = u_e1.*cosd(tt_e1);
intg_u1 = 0;
intg_p1 = 0;
for i=1:n
    intg_u1 = intg_u1 + (u_x1(i+1)+u_x1(i))*(y_e1(i+1)-y_e1(i))/2;
    intg_p1 = intg_p1 + (p_e1(i+1)+p_e1(i))*(y_e1(i+1)-y_e1(i))/2;
end
u_exit1 = intg_u1/y_e1(end);
p_exit1 = intg_p1/y_e1(end);
u_e2 = vel(end-n:end);
p_e2 = p(end-n:end);
y_e2 = y(end-n:end);
tt_e2 = tt(end-n:end);
u_x2 = u_e2.*cosd(tt_e2);
intg_u2 = 0;
intg_p2 = 0;
for i=1:n
    intg_u2 = intg_u2 + (u_x2(i+1)+u_x2(i))*(y_e2(i+1)-y_e2(i))/2;
    intg_p2 = intg_p2 + (p_e2(i+1)+p_e2(i))*(y_e2(i+1)-y_e2(i))/2;
end
u_exit2 = intg_u2/y_e2(end);
p_exit2 = intg_p2/y_e2(end);
x_exit = x_curve(end);
y_exit = y_curve(end);
%Interpolation
```

```
u_exit = u_exit1 + (x_exit-x_w(end-1))/(x_w(end)-x_w(end-1)) * (u_exit2 - u_exit1);
p_exit = p_exit1 + (x_exit-x_w(end-1))/(x_w(end)-x_w(end-1)) * (p_exit2 - p_exit1);
%Thrust
C_f = 0;
pa = 0 * 101325; %Ambient Pressure
m_dot = (p0*D)/sqrt(T0)*sqrt(gamma/R_spec * (2/(gamma+1))^((gamma+1)/(gamma-1)));
F = m_dot*u_exit + (p_exit-pa)*y_exit;
C_f = C_f + F/(p0*D);
f = 1/C_f;
end
```

Free Form Deformation

```
function [x_contour, y_contour] = ffd_y(x)
%FFD Grid Size
l = 5;
m = 2;
%Base Geometry
[x_base,y_base] = base_contour(100,3);
y_base = y_base - 1;
x_max = max(x_base);
y_max = max(y_base);
s = x_base/x_max;
t = y_base/y_max;
X = t;
%Deformation of Lattice Points
P = zeros(m,l);
k=1;
numPts = length(x);
if numPts > l
    msg = 'Error occurred.';
    error(msg)
end
for i=l-(numPts-1):l
    P(m,i) = x(k);
    k = k + 1;
end
l = l - 1;
m = m - 1;
%Free Form Deformation
for p=1:length(X)
    for i=0:l
        for j=0:m
            X(p) = X(p) + nchoosek(l,i)*(1-s(p))^(l-i)*s(p)^i*nchoosek(m,j)
            *(1-t(p))^(m-j)*t(p)^j*P(j+1,i+1);
        end
    end
```

```
end
%New Contour
x_contour = x_base;
y_contour = (X(:)*y_max+1)';
```

Base Geometry

```
function [xx,yy] = base_contour(n,Me)
    gamma=1.4;
    D=1;
    theta_max = PrandtlMeyer(Me,gamma)/2;
    [v,KL,KR,tt,tt0]=moc_2d_reflect(theta_max,n);
    node = n*(n+1);
    M = zeros(1,node);
    mu = zeros(1,node);
    x=zeros(1,node);
    y=zeros(1,node);
    x_w=zeros(1,n+1);
    y_w=zeros(1,n+1);
    v0=zeros(1,n);
    mu0=zeros(1,n);
    M0=zeros(1,n);
    for i=1:node
        M(i)=InvPrandtlMeyer(v(i));
        mu(i)=Mu(M(i));
    end
    for i=1:n
        v0(i)=tt0(i);
        M0(i)=InvPrandtlMeyer(v0(i));
        mu0(i)=Mu(M0(i));
    end
    %COORDINATE CALCULATION
    j=1;
    w=2;
    x_w(1) = 0;
    y_w(1) = D;
    for l=1:n
        if l==1
            for i=1:n+1
                if i==1
                    x(i)= -D/tand((tt0(i)-(mu0(i)+mu(i)))/2);
                    y(i)=0;
                elseif 1<i && i<n+1
                    slopeL = (tt(i)+tt(i-1)+(mu(i)+mu(i-1)))/2;
                    slopeR = (tt(i)+tt0(i)-(mu(i)+mu0(i)))/2;
                    x(i) = ...
                        (-x(i-1)*tand(slopeL)+y(i-1)-D)/(tand(slopeR)-tand(slopeL));
```

```
                    y(i) = (tand(slopeR)*tand(slopeL)*(-x(i-1))+
                    tand(slopeR)*y(i-1)-tand(slopeL)*D)/(tand(slopeR)-tand(slopeL));
                else
                    slopeL = tt(i-1)+mu(i-1);
                    slopeR = (tt(i-1)+tt0(i-1))/2;
                    x(i) = ...
                        (-x(i-1)*tand(slopeL)+y(i-1)-D)/(tand(slopeR)-tand(slopeL));
                    y(i) = (tand(slopeR)*tand(slopeL)*(-x(i-1))
                    +tand(slopeR)*y(i-1)-tand(slopeL)*D)/(tand(slopeR)-tand(slopeL));
                    x_w(w) = x(i);
                    y_w(w) = y(i);
                end
            end
        else
            for i=j:j+n
                if i==j
                    x(i)= -y(i-n)/tand((tt(i-n)-(mu(i-n)+mu(i)))/2)+x(i-n);
                    y(i)=0;
                elseif j<i && i<j+n
                    slopeL = (tt(i)+tt(i-1)+(mu(i)+mu(i-1)))/2;
                    slopeR = (tt(i)+tt(i-n)-(mu(i)+mu(i-n)))/2;
                    x(i) = (x(i-n)*tand(slopeR)-x(i-1)*tand(slopeL)
                    +y(i-1)-y(i-n))/(tand(slopeR)-tand(slopeL));
                    y(i) = (tand(slopeR)*tand(slopeL)*(x(i-n)-x(i-1))+tand(slopeR)
                    *y(i-1)-tand(slopeL)*y(i-n))/(tand(slopeR)-tand(slopeL));
                else
                    slopeL = tt(i-1)+mu(i-1);
                    slopeR = (tt(i-1)+tt(i-(n+1)))/2;
                    x(i) = (x(i-(n+1))*tand(slopeR)-x(i-1)*tand(slopeL)
                    +y(i-1)-y(i-(n+1)))/(tand(slopeR)-tand(slopeL));
                    y(i) = ...
                        (tand(slopeR)*tand(slopeL)*(x(i-(n+1))-x(i-1))+tand(slopeR)
                    *y(i-1)-tand(slopeL)*y(i-(n+1)))/(tand(slopeR)-tand(slopeL));
                    x_w(w) = x(i);
                    y_w(w) = y(i);
                end
            end
        end
    j=j+n+1;
    w=w+1;
    end
    %Splines
    ll = length(x_w);
    mesh = 1001;
    xx = linspace(0,x_w(ll),mesh);
    pp = spline(x_w,[tand(theta_max) y_w 0]);
    yy = ppval(pp,xx);
end
```

```
function [v,KL,KR,theta,theta_ini]=moc_2d_reflect(theta_max,n)
theta_ini=zeros(1,n);
x=0;
for i=1:n
    theta_ini(i)=theta_max*(1-cosd(x));
    x=x+90/(n-1);
end
node=n*(n+1);
theta=zeros(1,node);
v=zeros(1,node);
KL=zeros(1,node);
KR=zeros(1,node);
i=1;
for l=1:n
    if l==1
        theta(i)=0;
        v(i)=theta_ini(i);
        KL(i)=theta(i)-v(i);
        KR(i)=theta(i)+v(i);
        i=i+1;
        for c=1:n-1
            KL(i)=KL(i-1);
            KR(i)=2*theta_ini(i);
            theta(i)=0.5*(KL(i)+KR(i));
            v(i)=0.5*(KR(i)-KL(i));
            i=i+1;
        end
    else
        theta(i)=0;
        v(i)=KR(i-n);
        KL(i)=theta(i)-v(i);
        KR(i)=theta(i)+v(i);
        i=i+1;
        for c=1:n-1
            KL(i)=KL(i-1);
            KR(i)=KR(i-n);
            theta(i)=0.5*(KL(i)+KR(i));
            v(i)=0.5*(KR(i)-KL(i));
            i=i+1;
        end
    end
    theta(i)=theta(i-1);
    v(i)=v(i-1);
    KL(i)=KL(i-1);
    KR(i)=KR(i-1);
    i=i+1;
```

```
47 end
```

Prandtl-Meyer

```
1 function [v] = PrandtlMeyer(M,gamma)
2 v=180/pi*(sqrt((gamma+1)/(gamma-1))*
3 atan(sqrt((gamma-1)/(gamma+1)*(M^2-1)))-atan(sqrt(M^2-1)));
4 end
```

Inverse Prandtl-Meyer

```
1 function [Mach] = InvPrandtlMeyer(v)
2 %Based on Hall, I. M. "Inversion of the prandtl-meyer relation."
3 v=v*pi/180;
4 A=1.3604;
5 B=0.0962;
6 C=-0.5127;
7 D=-0.6722;
8 E=-0.3278;
9 v_0=0.5*pi*(sqrt(6)-1);
10 y=(v/v_0)^(2/3);
11 Mach=(1 + A*y + B*y^2 + C*y^3)/(1 + D*y + E*y^2);
12 end
```

Mach Angle

```
1 function mu=Mu(M)
2 mu=asind(1/M);
3 end
```

Appendix B

SU^2 Configuration File

```
SOLVER= EULER
MATH_PROBLEM= DIRECT
KIND_TURB_MODEL = NONE
% ---------------------- REFERENCE VALUE DEFINITION --------------------------%
REF_LENGTH= 1.0
REF_AREA= 0
FLUID_MODEL= IDEAL_GAS
GAMMA_VALUE= 1.4
GAS_CONSTANT= 287.0
% --------------------- BOUNDARY CONDITION DEFINITION -------------------------%
MARKER_EULER= ( Nozzle )
MARKER_SUPERSONIC_INLET= ( Inlet, 2500, 1.966951580207355e+06, 1000, 0.0, 0.0 )
MARKER_SYM = ( Symmetry )
MARKER_OUTLET = ( Outlet, 101325 )  %posso usar p = 1E-12
MARKER_PLOTTING= ( Outlet )
MARKER_MONITORING= ( Outlet, Inlet )
% ------------- COMMON PARAMETERS DEFINING THE NUMERICAL METHOD ---------------%
NUM_METHOD_GRAD= WEIGHTED_LEAST_SQUARES
CFL_NUMBER= 1.0
CFL_ADAPT= YES
CFL_ADAPT_PARAM= ( 0.5, 1.5, 0.5, 100.0 )
RK_ALPHA_COEFF= ( 0.66667, 0.66667, 1.000000 )
ITER= 1000
LINEAR_SOLVER= FGMRES
LINEAR_SOLVER_ERROR= 1E-4
LINEAR_SOLVER_ITER= 10
% -------------------------- MULTIGRID PARAMETERS -----------------------------%
MGLEVEL= 2
MGCYCLE= V_CYCLE
MG_PRE_SMOOTH= ( 1, 2, 3, 3 )
% --------------------- FLOW NUMERICAL METHOD DEFINITION ----------------------%
CONV_NUM_METHOD_FLOW= ROE
MUSCL_FLOW= NO
```

```
34 SLOPE_LIMITER_FLOW= VENKATAKRISHNAN
35 VENKAT_LIMITER_COEFF= 100
36 JST_SENSOR_COEFF= ( 0.5, 0.02 )
37 TIME_DISCRE_FLOW= EULER_IMPLICIT
38 % ---------------------------- CONVERGENCE PARAMETERS ----------------------------%
39 CONV_FIELD= RMS_DENSITY
40 CONV_RESIDUAL_MINVAL= -5
41 CONV_STARTITER= 10
42 CONV_CAUCHY_ELEMS= 100
43 CONV_CAUCHY_EPS= 1E-8
44 % ------------------------ INPUT/OUTPUT INFORMATION ------------------------%
45 MESH_FORMAT= SU2
46 MESH_OUT_FILENAME= mesh_out.su2
47 SOLUTION_FILENAME= solution_flow.dat
48 SOLUTION_ADJ_FILENAME= solution_adj.dat
49 TABULAR_FORMAT= CSV
50 HISTORY_OUTPUT = (ITER, RMS_RES, CFL_NUMBER)
51 CONV_FILENAME= history
52 VOLUME_OUTPUT = (COORDINATES, SOLUTION, PRIMITIVE)
53 RESTART_FILENAME= restart_flow.dat
54 RESTART_ADJ_FILENAME= restart_adj.dat
55 MESH_FILENAME= M3nozzle.su2
56 VOLUME_FILENAME= flow
57 SURFACE_FILENAME= surface_flow
58 VOLUME_ADJ_FILENAME= adjoint
59 GRAD_OBJFUNC_FILENAME= of_grad.dat
60 SURFACE_ADJ_FILENAME= surface_adjoint
61 CUSTOM_OUTPUTS = 'exit_pressure : AreaInt{PRESSURE}[Outlet];\
62           exit_pressure_avg : AreaAvg{PRESSURE}[Outlet];\
63           velocity_x : AreaInt{VELOCITY_X}[Outlet];\
64           velocity_y : AreaInt{VELOCITY_Y}[Outlet];\
65           velocity_x_avg : AreaAvg{VELOCITY_X}[Outlet];\
66           velocity_y_avg : AreaAvg{VELOCITY_Y}[Outlet];\
67           kin_energy : Macro{pow(VELOCITY_X, 2) * DENSITY};\
68           area : AreaInt{1}[Outlet];\
69           mom_thrust : AreaInt{$kin_energy}[Outlet];\
70           amb_pressure : AreaInt{101325.0}[Outlet];\
71           amb_pressure_half : AreaInt{50662.5}[Outlet];\
72           pre_thrust : Function{exit_pressure - amb_pressure};\
73           pre_thrust_half : Function{exit_pressure - amb_pressure_half};\
74           thrust : Function{mom_thrust + pre_thrust};\
75           thrust_half : Function{mom_thrust + pre_thrust_half};\
76           thrust_vac : Function{mom_thrust + exit_pressure};\
77           quot : AreaInt{3723300}[Inlet];\
78           C_T : Function{thrust/quot};\
79           C_Tvac : Function{thrust_vac/quot};\
80           C_T1/2 : Function{thrust_half/quot};'
81 SCREEN_OUTPUT=(INNER_ITER,RMS_DENSITY, C_T, C_T1/2 C_Tvac)
```