Data Driven Approach to Turbomachinery Blade Design

Master of Science in Aeronautical Engineering

Author: **Antonio Pucciarelli**

Student ID: **974675**
Advisor: **Prof. Sergio Lavagnoli**
Promoter: **Prof. Paolo Gaetani**
Academic Year: **2022-2023**

Abstract

In the field of turbomachinery design, the current design models are often slow and repetitive. These models rely heavily on computers and frequently overlook any existing knowledge about the design space. This work introduces an innovative methodology in the domain of turbomachinery design, seamlessly integrating machine learning techniques into the blade design process.

The importance of this thesis lies in its ability to eliminate the requirement for CFD simulations in the design process. Moreover, it gives designers a clear understanding of the loading limit of a blade and a deep understanding of the correlations between the loading distribution and the blade geometry. It also sheds light on the physical limitations a designer might come across during the design process.

Highlighting the significant influence of this study, the thesis not only presents a fresh design approach but also offers in-depth understanding of the crucial steps that support its implementation. Covering the entire process from creating data to utilizing artificial intelligence in turbomachinery design, this research sets the groundwork for more effective design strategies.

To sum up, this thesis introduces a new approach to designing turbomachinery blades through the integration of machine learning methods. By creating a fast and independent design process and providing a clear explanation of how data is collected and analyzed, this research brings improved efficiency and accuracy to the world of turbomachinery design.

Keywords: *Turbomachinery, data analysis, artificial intelligence, modal reduction*

Abstract in lingua italiana

Nel campo della progettazione di turbomacchine, i modelli di progettazione attuali sono spesso lenti e ripetitivi. Questi modelli si basano pesantemente sui computer e spesso trascurano qualsiasi conoscenza esistente dello spazio di progettazione. Questo lavoro introduce una metodologia innovativa nel campo della progettazione di turbomacchine, integrando delle tecniche di intelligenza artificiale per la progettazione di palette di turbina.

L'importanza di questa tesi risiede nella sua capacità di eliminare la necessità di simulazioni CFD nel processo di progettazione di palette per turbomacchine. Inoltre, fornisce ai progettisti una chiara visione del limite di carico di una pala e una buona descirzione delle correlazioni tra la distribuzione del carico e la geometria della pala. Inoltre, getta luce sulle limitazioni fisiche che un progettista potrebbe incontrare durante il processo di progettazione.

Evidenziando l'influenza significativa di questo studio, la tesi non solo presenta un approccio di progettazione innovativo, ma offre anche una comprensione approfondita dei passaggi cruciali che ne sostengono l'attuazione. Coprendo l'intero processo dalla creazione dei dati all'utilizzo dell'intelligenza artificiale nella progettazione di turbomacchine, questa ricerca getta le basi per strategie di metodi di progettazione più efficaci.

In sintesi, questa tesi introduce un nuovo approccio alla progettazione delle pale per turbomacchine. Creando un processo di progettazione rapido e indipendente e fornendo una spiegazione chiara di come vengono raccolti e analizzati i dati, questa ricerca introduce l'integrazione dell'intelligenza artificiale nella progettazione di turbomacchine.

Parole chiave: *Turbomacchine, analisi di dati, intelligenza artificiale, riduzione modale*

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Definition
`SINL`	Inlet slope
`SOUT`	Outlet slope
`KUTTA`	Kutta condition
`CHINL`	Inlet channel
`CHOUT`	Outlet channel
`PITCH`	Blade pitch
`X(*)`	Blade point x coordinate
`Y(*)`	Blade point y coordinate
`INL SLOPE`	Inlet slope variable in `MISES`
`EXIT SLOPE`	Outlet slope variable in `MISES`
`LE STAGNATION`	Leading edge stagnatnio variable in `MISES`
`EXIT PRESSURE`	Outlet pressure variable in `MISES`
`INL MACH`	Inlet Mach number variable in `MISES`
`INL SLOPE DRIVE`	Solving strategy in `MISES` based on the inlet slope
`LE/TE KUTTA`	`MISES` constraint based on the Kutta condition at the leading edge and the trailing edge
`INL P0/P0r DRIVE`	Solving strategy in `MISES` based on the inlet pressure ratio
`OUT MACH DRIVE`	Solving strategy in `MISES` based on the outlet Mach number
`INL MACH`	Inlet Mach number value in `MISES`
`INL P1/P0a`	Inlet pressure fraction value in `MISES`
`INL SLOPE`	Inlet flow slope value in `MISES`
`INL POSITION`	Inlet boundary conditions position value in `MISES`

`OUT MACH`	Oulet Mach number value in `MISES`
`OUT P1/P0a`	Outlet pressure fraction value in `MISES`
`OUT SLOPE`	Outlet flow slope value in `MISES`
`OUT POSITION`	Outlet boundary conditions position value in `MISES`
`REYNOLDS`	Flow inlet Reynolds number in `MISES`
`NCRIT`	`MISES` turbulence model parameter
`XTRANS1`	Guess on the transition point at the prssure side of the blade in `MISES`
`XTRANS2`	Guess on the transition point at the suction side of the blade in `MISES`
`ISMOM`	`MISES` problem solving strategy
`MCRIT`	`MISES` critical Mach number for the shoch sensitivity setup
`MUCON`	`MISES` shock losses artificial dissipation
`MFRin | HWRAT`	`MISES` variables not used in the present flow setup
`BVR1in | BVR2in`	`MISES` variables not used in the present flow setup
CFD	Computational fluid dynamics
RBF	Radial basis functions

List of Symbols

Symbol	Definition	Unit
γ	Stagger angle	[∘]
χ_1	Metal inlet angle	[∘]
χ_2	Metal outlet angle	[∘]
n	Fist camberline parameter	[-]
a	Second camberline parameter	[-]
b	Third camberline parameter	[-]
x	Axial chord fraction	[-]
y	Camberline ordinate	[-]
y'	Camberline first derivative	[-]
$\boldsymbol{n}$	Camberline normal versor	[-]
n_x	Camberline normal versor x component	[-]
n_y	Camberline normal versor y component	[-]
$S_{(x,i,N)}$	Bernstein function	[-]
A_i	Bernstein weight	[-]
N	Bernstein number of modes	[-]
$C_{(x)}$	Shape function	[-]
$\zeta_{(x)}$	Thickness function	[-]
$\zeta_{TE_{(x)}}$	Trailing edge thickness function	[-]
R_{LE}	Leading edge radius adimensionalized by the chord	[-]
R_{TE}	Trailing edge radius adimensionalized by the chord	[-]
x_{PS}	Pressure side x component	[-]
y_{PS}	Pressure side y component	[-]
x_{SS}	Suction side x component	[-]
y_{SS}	Suction side y component	[-]
β	Blade wedge angle	[∘]

A_0	Kulfan parametrization $i = 0$ parameter: related to R_{LE}	[-]
A_N	Kulfan parametrization $i = N$ parameter: related to R_{TE}	[-]
$pitch$	Adimensionalized blade pitch over chord	[-]
N_{new}	New parameters counts for blade scaling	[-]
α_1	Inlet flow angle	[$\circ$]
α_2	Outlet flow angle	[$\circ$]
M_2	Oulet flow Mach number after mixed out	[-]
Re	Reynolds number computed at inlet	[-]
M_{LE}	Leading edge Mach number used for the aerodynamic style curve	[-]
M_{PEAK}	Peak Mach number used for the aerodynamic stule curve	[-]
M_{PRESS}	Pressure Mach number used for the aerodynamic style curve	[-]
S_{PEAK}	Surface position where M_{PEAK} is located over the aerodynamic style curve	[-]
M_{TE}	Trailing edge Mach number, it varies on the position: suction side or pressure side	[-]
S_{TOT}	Total surface length	[-]
M_1	Inlet Mach number	[-]
$M_{1,ax}$	Axial inlet Mach number	[-]
$\boldsymbol{x}_0$	Optimizer initial guess	[-]
$\boldsymbol{x}$	Optimizer blade geometry array at `iter`	[-]
`iter`	Iteration in the optimization algorithm	[-]
`iterTol`	Iteration tolerance in the optimization algorithm	[-]
`cost`	Convergence cost at `iter` in the optimization algorithm	[-]
`costTol`	Convergence tolerance in the optimization algorithm	[-]
σ	Standard deviation of the error at `iter` in the optimization algorithm	[-]
σ_{tol}	Standard deviation tolerance in the optimization algorithm	[-]
$RMSE$	Root mean square error over the blade	[-]

$\Delta\alpha_2$	Exit angle error	[$\circ$]
$cost$	Cost function that is used inside `TurbOpt`	[-]
$\left.\frac{M_{real}}{M_{TE,real}}\right\rvert_i$	Mach fraction, computed by `MISES`, at the i_{th} point over the blade surface length used in the $RMSE$ computation	[-]
$\left.\frac{M_{target}}{M_{TE,target}}\right\rvert_i$	Mach fraction, set up by the aerodynamic style, at the i_{th} point over the blade surface length used in the $RMSE$ computation	[-]
$\mathcal{X}$	Aerodynamic style and aerodynamic duty dataset	[-]
$\mathcal{Y}$	Blade geometry dataset	[-]
$\mathcal{X}_*$	Aerodynamic style and aerodynamic duty training dataset	[-]
$\mathcal{Y}_*$	Blade geometry training dataset	[-]
$\mathcal{X}_{**}$	Aerodynamic style and aerodynamic duty test dataset	[-]
$\mathcal{Y}_{**}$	Blade geometry test dataset	[-]
f	Analytical function for the interpolation of the blade geometry domain	[-]
$\hat{f}$	Numerical approximated function for the interpolation of the blade geometry domain	[-]
c	Radial basis functions length scale	[-]
d	Euclidean distance between to points inside the domain	[-]
α	Regularization parameter in the machine learning model	[-]
γ^i	i_{th} parameter for the idenfication of the stagger angle	[-]
$\boldsymbol{w}$	Machine learning weight	[-]
$\boldsymbol{\Phi}$	Matrix which containts the RBF	[-]
Φ	Radial basis function	[-]
m_*	Training set dimension	[-]
$J_{(\boldsymbol{x}_*)}$	Functional to minimize	[-]
$J_{(\boldsymbol{x}_{**})}$	Testing function	[-]
σ	Modal shape variance	[-]

$\frac{pitch}{chord}$	Blade solidity. The chord in the present work is set to 1	[-]

1 | Problem Framing

This chapter aims to *outline* the challenges that arise from designing a turbomachine using a **conventional approach**. Additionally, it introduces a new **potential design method** that can address some of these issues.

1.1. Design Process

Studying a turbomachinery system requires significant *effort*, both in terms of modeling the underlying physics that define the problem and in generating the optimal configuration that aligns with design constraints. An efficient approach to **tackling** this challenge involves **framing** it in terms of targets and constraints. By minimizing the complexity of the design process, one can *mitigate* the risks associated with poor design.

1.1.1. Problem Classification

An engineering problem can be described by **inputs** and **outputs**. An engineer should be able to get the best outputs giving the inputs in the less time possible. Inputs and outputs can be classified in order to better understand the problem and to solve it in the most effective way.

On the inputs side, they can be subdivided into:

- **Objectives**: these define the parameters that need optimization
- **Constraints**: these define the region of interest for the study

On the outputs side, they can be classified as follows:

- **Objectives**: the optimized parameters resulting from the design
- **Definitions**: the engineering representation of the outputs

A modeling framework serves as the link between the inputs and outputs of the problem. A *well-designed* framework allows reaching the best solution in reasonable time.

Figure 1.1 represents a possible design problem.

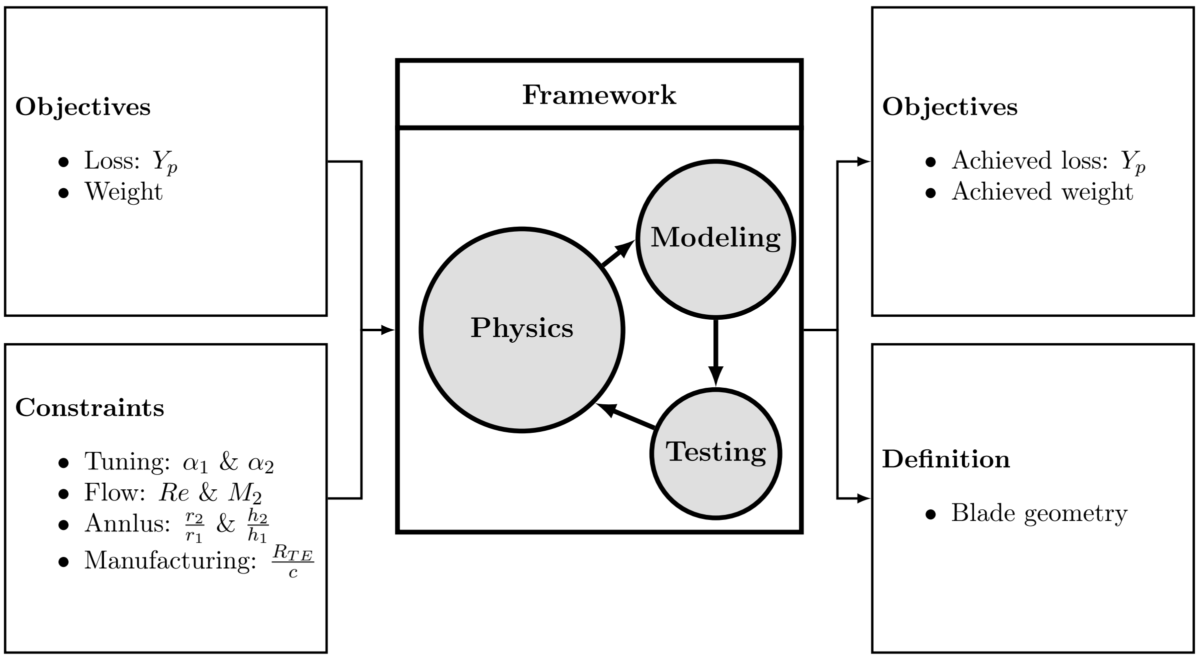


Figure 1.1: Turbomachinery design classification.

1.1.2. Turbomachinery Analysis in Complex Machines

A general modeling approach involves decomposing the design of a complex machine into various **sub-problems**, each of which plays a role in the overall system's performance. For instance, the study of an aeronautical engine can be subdivided into the examination of its primary components: the fan, compressor, combustion chamber, turbine, spool, gears, control systems, and other important assemblies.

The principal **macroscopic properties** of the system can be derived from an initial analysis of its primary components. To illustrate, when focusing on the **turbine**, it becomes feasible to estimate the number of stages and to make a preliminary approximation of the flow deflection angles in each stage. After this preliminary design is formulated, the next step is to design blades that satisfy these specified properties. Each blade, in turn, will be characterized by its **constituent sections**. The collective properties of the blade will then be *summarized* by its **mid-section**. Consequently, the design process for turbomachinery **invariably** leads back to the design of the blades' mid-section.

This outcome serves as the *focal point* of a new section design methodology.

Figure 1.2 depicts the central feature of problem decomposition.

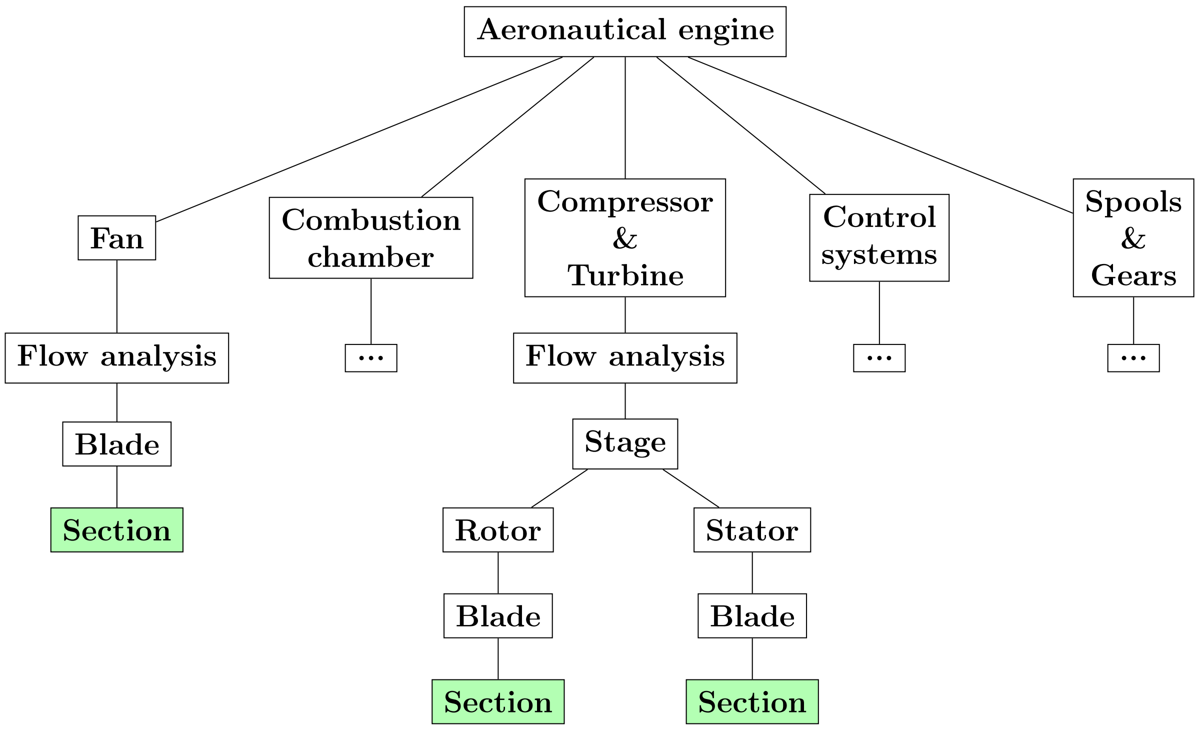


Figure 1.2: Problem decomposition.

1.2. Design Methods

Recognizing that the section generation is the most **repetitive task** in the turbomachinery design process, it is advantageous to also **comprehend** the current **state of the art** in the section generation. Various methods exist for designing a blade section, many of which can be classified as **iterative processes**. In these methods, a **numerical scheme** is employed to describe flow properties around a chosen geometry.

It is evident that *sidestepping* this optimization task, which **requires Computational Fluid Dynamics** (CFD), could substantially contribute to **reducing design time**.

1.2.1. Iterative Based Method

A further more classification can be made on the iterative based methods. These methods can be:

- **Automated**: done using **computers**, where the solver tends *blindly* to the optimal configuration.
- **Manaul**: done by **engineers**, where *prior knowledge* and *experience* allow reaching the optimum.

These methods all encounter the same challenge: **time is wasted on iterations involving CFD** [2]. On the other hand, each of these methods has its own **advantage**. In the case of **automated methods**, optimization is **expedited** through **computer usage** [12]. In contrast, **manual methods** leverage **prior knowledge** and **experience** to **crunch down the domain of study**, consequently reducing the time required for optimization.

1.2.2. Non-Iterative Based Method

The non-iterative-based method integrates **prior knowledge** and **experience**, utilizing data, along with the **speed** of computers through machine learning.

To describe this method, it is essential to introduce a *new way* of design a blade. Instead of generating a blade by *iterating through various geometries*, this method constructs a blade *using a dataset* [10] that represents a range of blade variations. Within this domain of blades, the optimal blade is determined by **maximizing the defined objectives** while adhering to the **imposed constraints**.

The **effectiveness** and **accuracy** of this method **rely** significantly on the **database** that must accurately represent the designated study domain. The sole optimization process employed by this method is associated with identifying appropriate blades that define the domain of study.

1.2.3. Concepts

After introducing this new method, its framework is *grounded* in four other key concepts. These concepts dictate the approach to **generating** and **processing data** within the method. These concepts are:

- The suitable machine learning model capable of *accommodating* the **physics inherent in midsection generation**.
- The *macroscopic* behavior of a blade.
- The *local distribution of the load* across the blade.
- The *dimensionality reduction* of the problem through a **parameterized representation** of both the blade and its loading.

1.2.3.1. Physics & Modeling

One of the primary aspects to consider is how data will be processed by the machine learning algorithm. Blade design is not susceptible to external random noise, implying that a blade will consistently exhibit the same behavior under identical operating conditions. This underscores the existence of a *one-to-one* correlation between the inputs and the resulting geometry.

This concept holds great significance for selecting the appropriate machine learning model. Due to this data correlation, a regression approach will be employed for interpolating within the domain.

1.2.3.2. Aerodynamic Duty

Because of the high dimensionality of the problem, it is necessary to classify the objectives of study in order to minimize at its best the number of inputs.

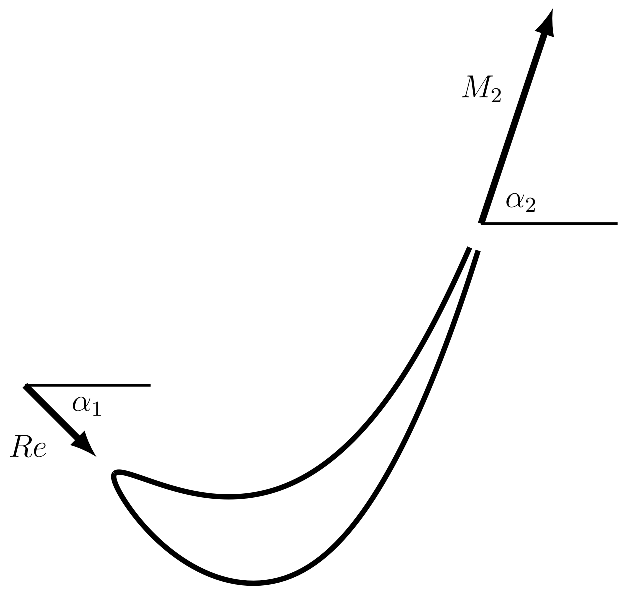


Figure 1.3: Aerodynamic duty parameters.

A way of describing the macroscopic working conditions of the flow is to define:

- α_1: inlet flow angle
- α_2: outlet flow angle
- M_2: outlet Mach number
- Re: Reynolds number of the flow

These four parameters, represented in Figure 1.3, are the minimum requirements for the definition of a working condition or **duty** of the blade. As result, these parameters will be indentified as **aerodynamic duty** of the blade.

1.2.3.3. Aerodynamic Style

In a similar way, a blade's behavior can be characterized by its **load distribution**. In contrast to the **aerodynamic duty**, the load can be defined as a localized property of the blade. Adhering to the same philosophy as the aerodynamic duty, it is possible to **parametrize** the load distribution around a blade, thereby enabling the examination of key load patterns relevant to the engineering field.

Within the context of the current study, only *pertinent load patterns* will be used for database generation. The usage of *useful load-patterns* avoid *unnecessary* diffusion and are the ones which generate the *highest load differences* between suction and pressure side.

1.2.3.4. Dimensionality

Having introduced two significant classifications for reducing constraint dimensionality, a dimensionality reduction technique will also be employed to define the blade.

Blade parametrization offers the advantage of **conserving memory** space and establishing a **uniform definition** for all potential blades within the database.

Furthermore, another noteworthy benefit emerges from the simplification of **data interpolation** due to parametrization: interpolating parameters is considerably simpler than interpolating the **intricate behavior of the blade's coordinates**.

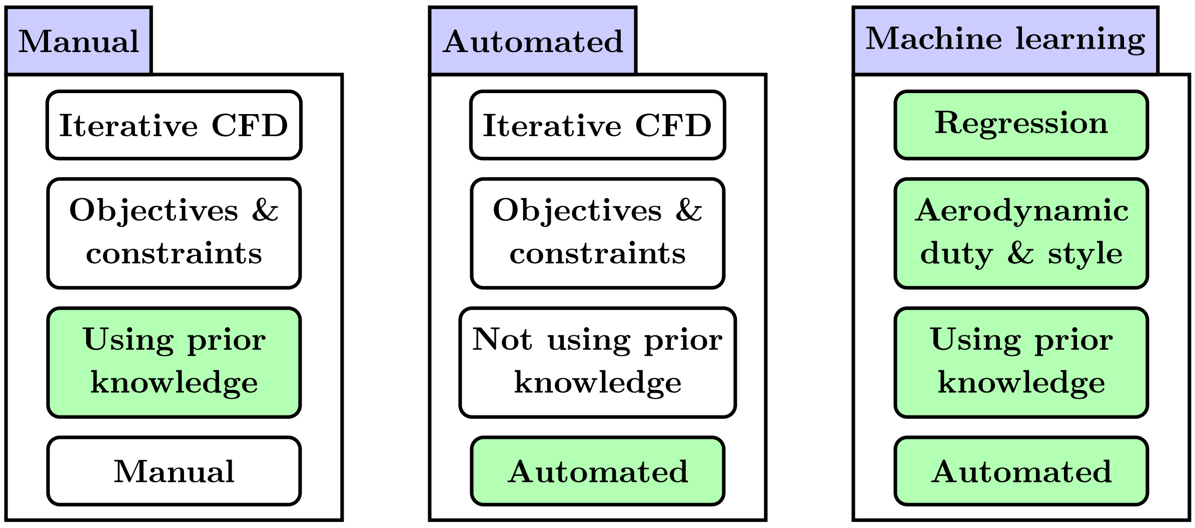


Figure 1.4: Representation of the main differences between the method.

2 | Framework

This chapter explains the main *building blocks* for **problem dimensionality reduction**, **database generation**, and **machine learning setup**. It will underscore the *key role* of **parametrization** in the **dimensionality reduction** of the database, in terms of its **dimensions** and the way of **representing** the data.

2.1. Blade Geometry Parametrization

The blade geometry is defined by two components: the **camberline** and the **profile line**. The following parametrization follows **Kulfan's parametrization** [8].

2.1.1. Camberline

The camberline serves as a *primary property layer* for defining the blade geometry. The suction side and pressure side geometry are added atop this *layer* to generate the complete blade. The **camberline** holds the utmost importance in blade generation. Even a *minor* alteration to the camberline shape can lead to a *significant* change in the flow behavior around the blade.

2.1.1.1. Formulation

The parameters used for camberline definition are:

- γ: stagger angle.
- χ_1: metal inlet angle.
- χ_2: metal outlet angle.

These three parameters (γ, χ_1, and χ_2) are then employed to compute *intermediate variables*, which will subsequently contribute to defining the camberline structure. The *intermediate variables* n, a, and b are computed as:

$$n = \frac{tan(\chi_2) + tan(\chi_1)}{tan(\gamma)} \tag{2.1}$$

$$a = \frac{tan(\chi_2)}{n} \tag{2.2}$$

$$b = -\frac{tan(\chi_1)}{n} \tag{2.3}$$

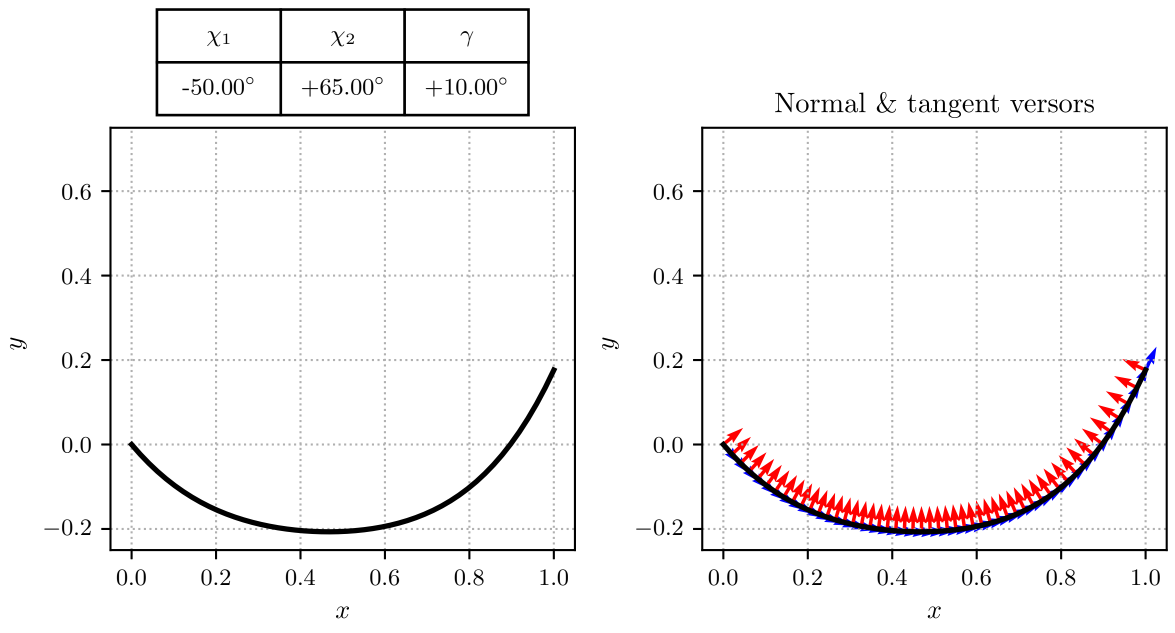
The camberline is defined as:

$$y = a \cdot x^n + b \cdot (1-x)^n \tag{2.4}$$

$$y' = a \cdot n \cdot x^{n-1} - b \cdot n \cdot (1-x)^{n-1} \tag{2.5}$$

$$\boldsymbol{n} = \begin{bmatrix} n_x \\ n_y \end{bmatrix} = \begin{bmatrix} -\frac{y'}{\sqrt{1+(y')^2}} \\ \frac{1}{\sqrt{1+(y')^2}} \end{bmatrix} \tag{2.6}$$

Figures 2.1 and 2.2 depict two potential camberlines.



χ_1	χ_2	γ
-50.00°	+65.00°	+10.00°

Figure 2.1: Camberline: $\gamma = 10°$, $\chi_1 = -50°$, and $\chi_2 = 65°$.

The normal and tangent vectors are employed for the thickness distribution along the camberline, as explained in Section 2.1.2.

χ_1	χ_2	γ
-20.00°	+72.00°	+30.00°

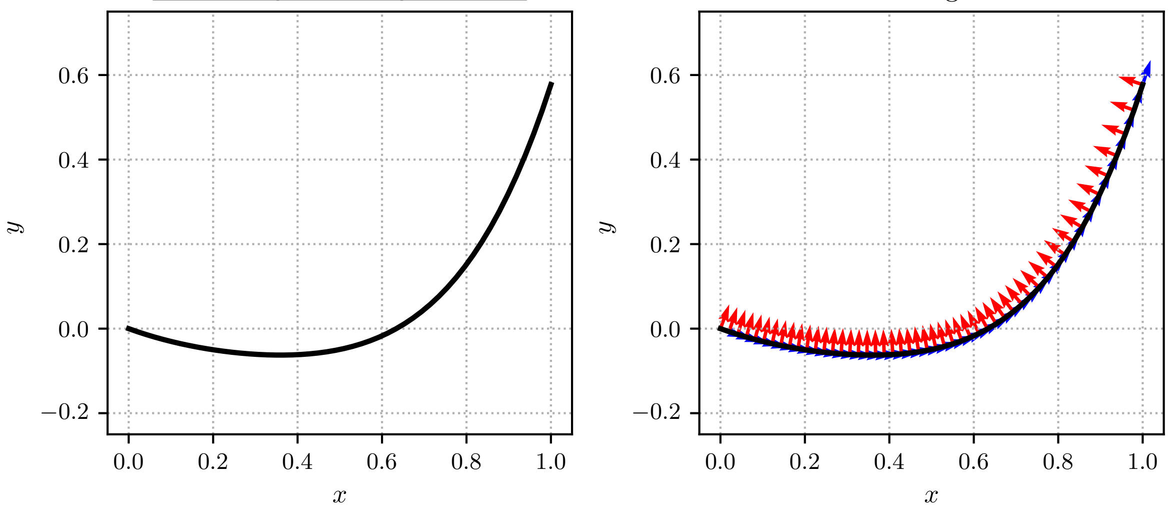


Figure 2.2: Camberline: $\gamma = 30°$, $\chi_1 = -20°$, and $\chi_2 = 70°$.

2.1.2. Profile Line

The profile line defines both the **suction side** and the **pressure side** of the blade. Utilizing **Kulfan's parametrization** [8], it is possible to generate a wide array of blades **using just a few parameters**. The preference for parametrization over coordinate-based representation arises from:

- optimization **speed**
- the fact that, considering the tool's ultimate purpose, parameters are **easier to correlate** than a pure coordinate-based representation

2.1.2.1. Formulation

The profile line is defined by $N + 1$ parameters: A_i for $i = 0 : N$. These parameters represent the **weights** of the **modes** that characterize the blade.

2.1.2.1.1 Bernstein Functions The shape modes of the blade are described by the Bernstein functions, denoted as $S_{(x,i,N)}$:

$$S_{(x,i,N)} = A_i \cdot \frac{N!}{(N-i)! \cdot i!} \cdot x^i \cdot (1-x)^{N-1}, \text{ for } i = 0 : N \tag{2.7}$$

The blade geometry modes are visualized in Figure 2.3.

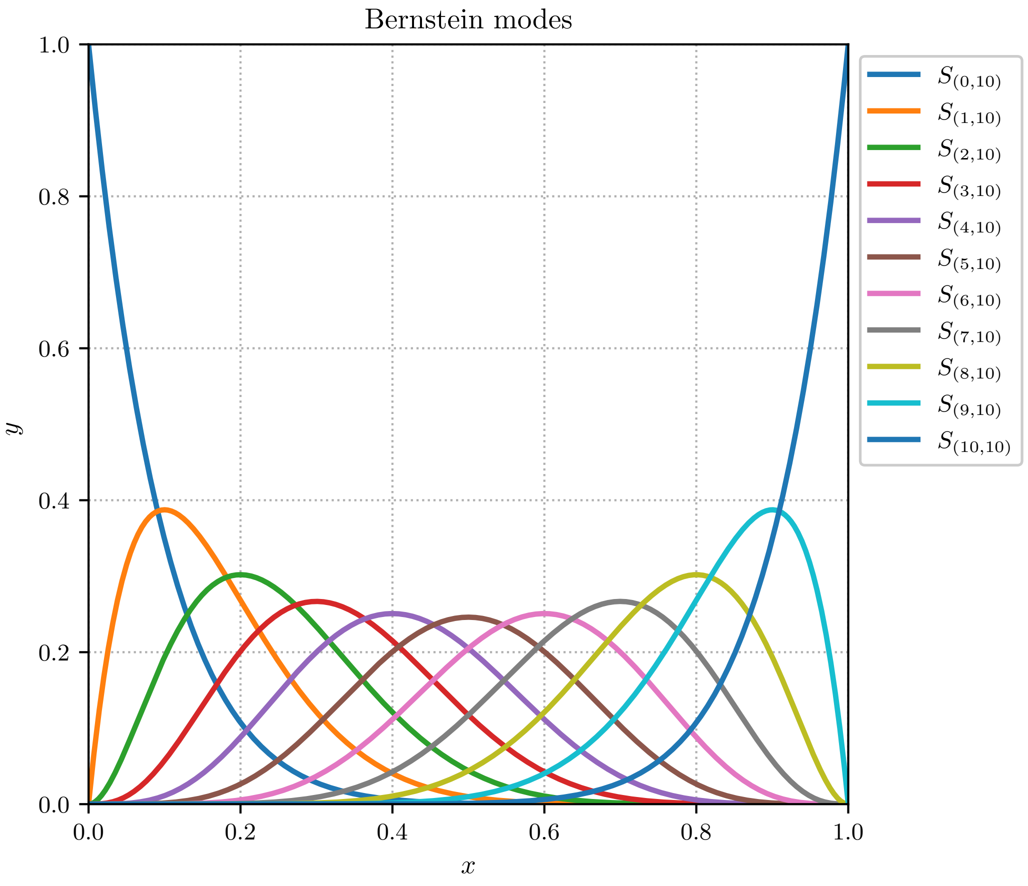


Figure 2.3: Bernstein function, $S_{(x,i,N)}$, with $A_i = 1$ and $N = 10$.

2.1.2.1.2 Shape Function In addition to the Bernstein functions as described in Equation (2.7), a **shape function** [9] is introduced for representing the *leading edge properties* and *trailing edge properties* of the blade. The shape function, denoted as $C_{(x)}$, takes the form:

$$C_{(x)} = x^{C_0} \cdot (1-x)^{C_1}, \text{ where: } C_0 = 0.5 \text{ and } C_1 = 1.0 \tag{2.8}$$

Equation (2.8) captures the *roundness* of the leading edge and the wedge angle properties at the trailing edge of the blade. The shape function is illustrated in Figure 2.4.

2.1.2.1.3 Thickness Distribution The blade thickness, denoted as ζ, is determined by combining Equation (2.7) and Equation (2.8). Additionally, the trailing edge radius, R_{TE}, is incorporated using a linear distribution, ζ_{TE}, over the camberline. The thick-

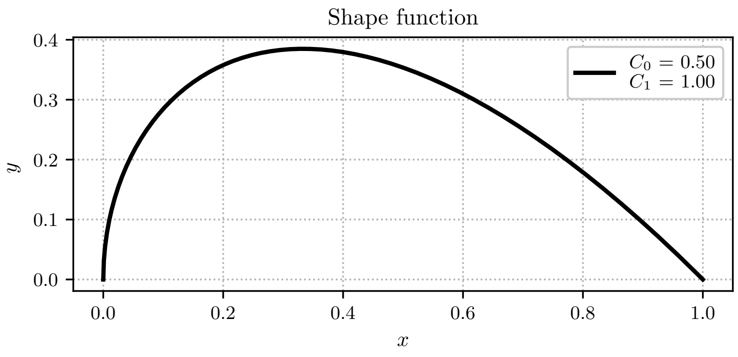


Figure 2.4: Shape function, $C_{(x)}$.

ness distribution $\zeta_{(x)}$ is related to camberline properties and **is distributed using a unit Bernstein function** - $S_{(x,0,2)}$. The distribution $\zeta_{TE_{(x)}}$ solely relies on camberline properties n_x and n_y.

$$\zeta_{(x)} = \sum_{i=0}^{N} S_{(x,i,N)} \cdot C_{(x)} \tag{2.9}$$

$$\zeta_{TE_{(x)}} = x \cdot R_{TE} \tag{2.10}$$

The pressure(suction) side (±) profile line coordinates are then computed as:

$$x_{PS/SS} = x_{camberline} \pm n_x \cdot \zeta_{(x)} \cdot S_{(x,0,2)} \pm n_x \cdot \zeta_{TE_{(x)}} \tag{2.11}$$

$$y_{PS/SS} = y_{camberline} \pm n_y \cdot \zeta_{(x)} \cdot S_{(x,0,2)} \pm n_y \cdot \zeta_{TE_{(x)}} + \zeta_{(x)} \cdot \left[1 - S_{(x,0,2)}\right] \tag{2.12}$$

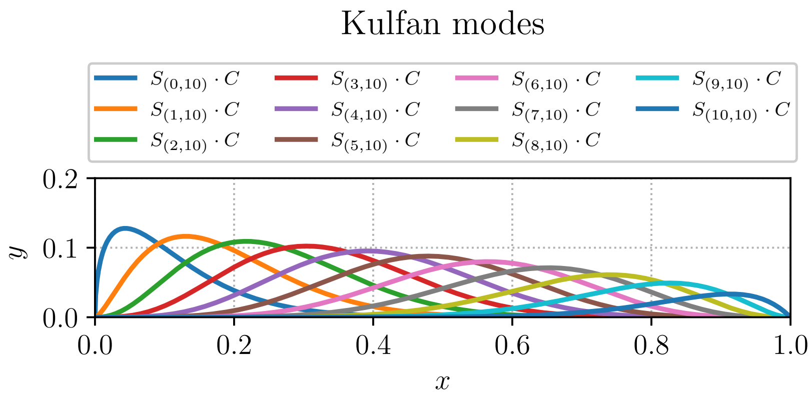


Figure 2.5: Kulfan modes, $S_{(x,i,N)} \cdot C_{(x)}$, with $A_i = 1$ and $N = 10$.

Figures 2.5 presents the **profile blade modes**, which are a combination of Equation (2.7) and Equation (2.8).

The suction side and pressure side coordinates are computed using a *custom set of points* over the camberline. The camberline is discretized with **Chebyshev nodes** all along the chord. This approximation allows better accuracy at the leading edge and at the traling edge of the blade.

Figure 2.6 and Figure 2.7 show clearly the blade coordinates starting from the chord discretization passing through the camberline discretization to the suction side and pressure side points. Using the **Chebyshev nodes** allows having a **faster generation** of the blade while keeping **good accuracy** over the blade coordinates representation.

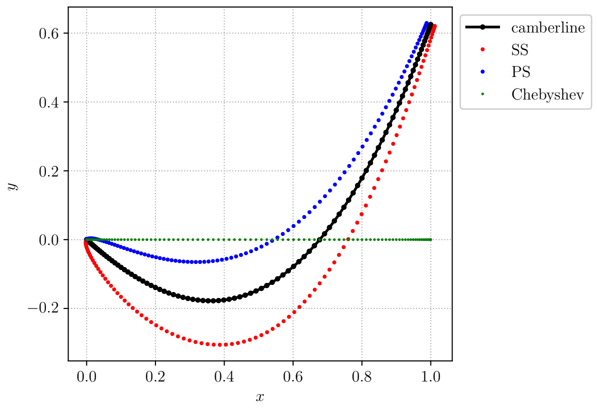


Figure 2.6: Coordinate based representation of the blade.

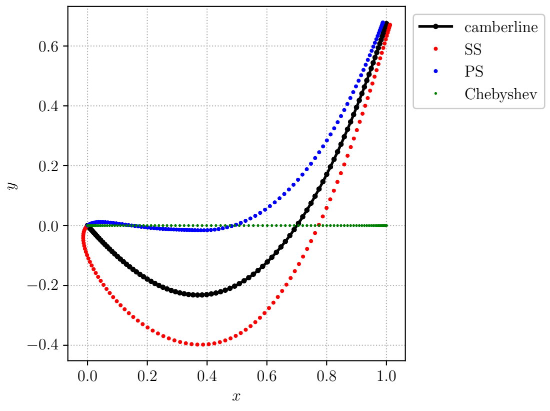


Figure 2.7: Coordinate based representation of the blade.

2.1.2.2. Main Features

Kulfan's parametrization retains several crucial geometric properties:

- The leading edge radius, R_{LE}, is determined solely by the A_0 parameter [8, App. B]. $A_0 = \sqrt{2 \cdot R_{LE}}$.
- The trailing edge angle, β, is directly related to the A_N parameter [8, App. A]. $A_N = tan(\beta) + R_{TE}$.

These features hold great **significance** in studying the **blade characteristics** in relation to flow properties.

2.1.2.3. Scaling

The blade optimization takes place through **incremental steps**, utilizing a strategy that ensures *faster convergence*. One key aspect of this strategy is to perform a **scaling** of the blade once convergence is achieved. This step is crucial for optimizing a blade with many degrees of freedom. The primary aim of scaling is to address potential *pits* within the domain during the blade optimization.

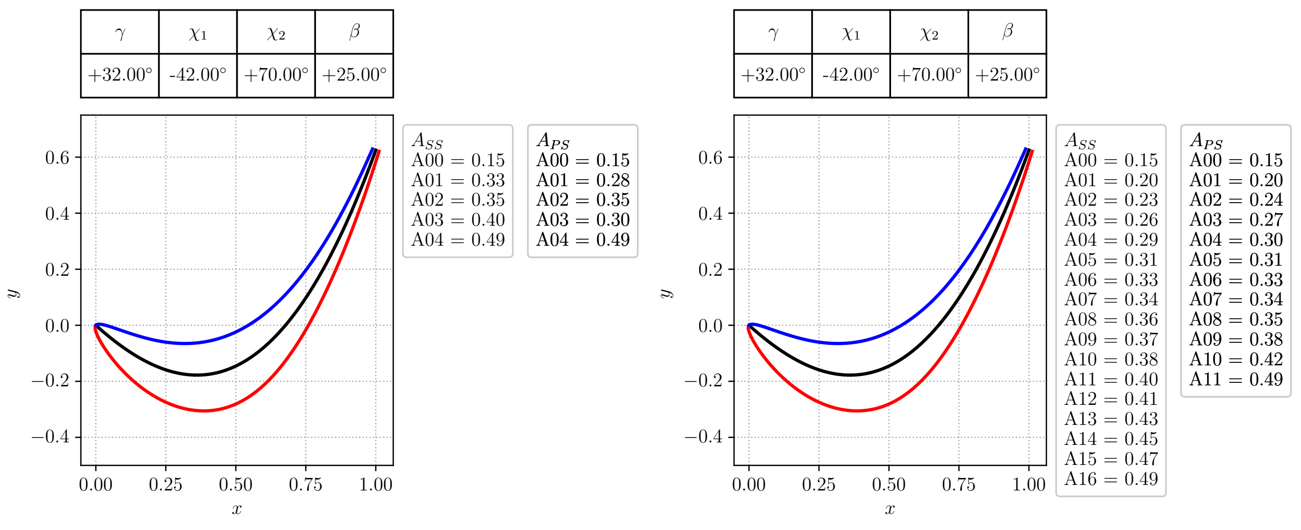
This scaling is generated solving a linear system. The linear system uses a matrix of $N_{new} \times N_{new}$ dimension. To build up this matrix, the blade, parametrized with lower number of parameters N, is evaluated on an equispaced number of points, N_{new} times, along the axial chord. The resulting matrix is then used to compute the new parametrization, made by N_{new} parameters.

It's important to note that the only unaltered parameters are:

- For the suction side and pressure side: R_{LE} and β
- For the camberline: χ_1, χ_2 and γ

These parameters remain unchanged during the scaling process due to their direct correlation with the blade's geometrical properties

Figures 2.8 and 2.9 illustrate different blades and the outcomes resulting from their scaling.



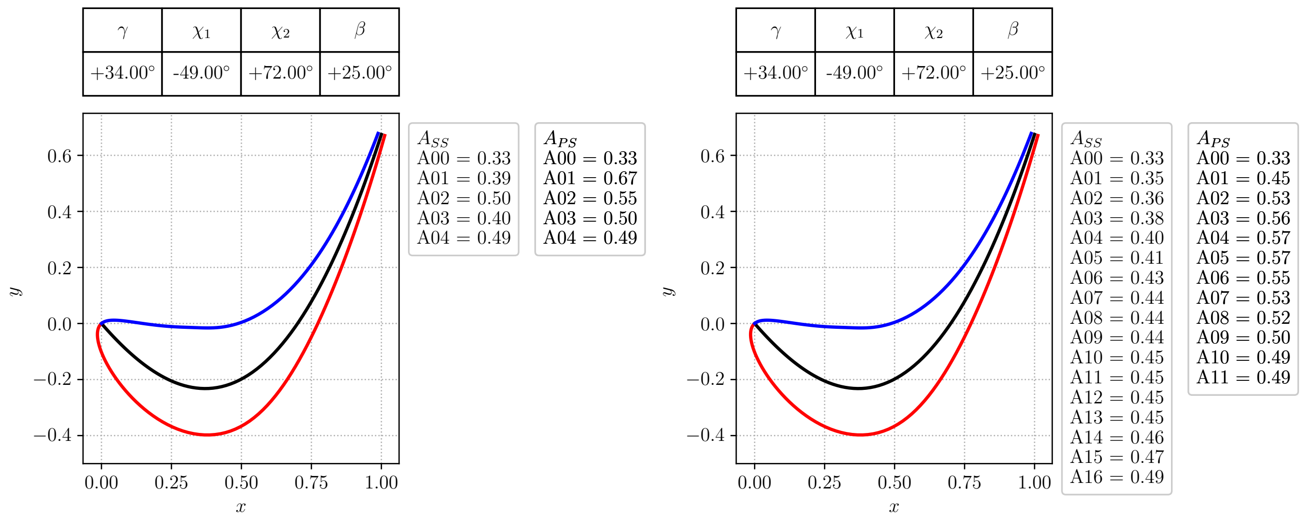
γ	χ_1	χ_2	β
+32.00°	-42.00°	+70.00°	+25.00°

A_{SS}	A_{PS}
A00 = 0.15	A00 = 0.15
A01 = 0.33	A01 = 0.28
A02 = 0.35	A02 = 0.35
A03 = 0.40	A03 = 0.30
A04 = 0.49	A04 = 0.49

γ	χ_1	χ_2	β
+32.00°	-42.00°	+70.00°	+25.00°

A_{SS}	A_{PS}
A00 = 0.15	A00 = 0.15
A01 = 0.20	A01 = 0.20
A02 = 0.23	A02 = 0.24
A03 = 0.26	A03 = 0.27
A04 = 0.29	A04 = 0.30
A05 = 0.31	A05 = 0.31
A06 = 0.33	A06 = 0.33
A07 = 0.34	A07 = 0.34
A08 = 0.36	A08 = 0.35
A09 = 0.37	A09 = 0.38
A10 = 0.38	A10 = 0.42
A11 = 0.40	A11 = 0.49
A12 = 0.41	
A13 = 0.43	
A14 = 0.45	
A15 = 0.47	
A16 = 0.49	

Figure 2.8: Example No. 1 of blade scaling following Kulfan's parametrization.



γ	χ_1	χ_2	β
+34.00°	-49.00°	+72.00°	+25.00°

A_{SS}	A_{PS}
A00 = 0.33	A00 = 0.33
A01 = 0.39	A01 = 0.67
A02 = 0.50	A02 = 0.55
A03 = 0.40	A03 = 0.50
A04 = 0.49	A04 = 0.49

γ	χ_1	χ_2	β
+34.00°	-49.00°	+72.00°	+25.00°

A_{SS}	A_{PS}
A00 = 0.33	A00 = 0.33
A01 = 0.35	A01 = 0.45
A02 = 0.36	A02 = 0.53
A03 = 0.38	A03 = 0.56
A04 = 0.40	A04 = 0.57
A05 = 0.41	A05 = 0.57
A06 = 0.43	A06 = 0.55
A07 = 0.44	A07 = 0.53
A08 = 0.44	A08 = 0.52
A09 = 0.44	A09 = 0.50
A10 = 0.45	A10 = 0.49
A11 = 0.45	A11 = 0.49
A12 = 0.45	
A13 = 0.45	
A14 = 0.46	
A15 = 0.47	
A16 = 0.49	

Figure 2.9: Example No. 2 of blade scaling following Kulfan's parametrization.

2.2. Aerodynamic Style Parametrization

In contrast to the **aerodynamic duty**, the **aerodynamic style** allows to define locally the load distribution along the blade.

The **aerodynamic style** is defined by the Mach fraction, $\frac{M}{M_{TE}}$, over the surface length fraction, $\frac{S}{S_{TOT}}$.

There are multiple *reasons* about the use of these parameters. The key ones are:

- The leading edge is a regione where, due to the high curvature of the geometry, there is a **high change in surface fraction over the axial chord**. Since this region is a very important for the correlation study, the $\frac{M}{M_{TE}}$ *vs* $\frac{S}{S_{TOT}}$ formulation results appropriate.
- The **boundary layer** properties, such as transition and detachament, is mostly dependent on the **total flow path length** done by the flow - described by S - and not over its projection over the axis - described by x.

The **aerodynamic style** [1] is defined by these variables:

- $\frac{M_{LE}}{M_{TE}}\frac{M_2}{M_1}$: leading edge Mach fraction. This parameter defines the load at the leading edge on the suction side.
- $\frac{M_{PEAK}}{M_{TE}}$: peak Mach fraction. Defines the peak Mach fraction on the suction side, representing the highest Mach value over the blade.
- $\frac{M_{PRESS}}{M_{TE}}\frac{M_2}{M_{1,ax}}$: pressure Mach number. It is a **double descriptor** of the leading edge load on the suction side and the Mach fraction before the Mach fraction raises to reach the trailing edge on the pressure side.
- $\frac{S_{PEAK}}{S_{TOT}}$: surface fraction position where the peak Mach fraction, $\frac{M_{PEAK}}{M_{TE}}$, is positioned over the load distribution.

It is important to understand that **the load varies with respect to $\frac{M_1}{M_2}$ and $\frac{M_{1,ax}}{M_2}$**. This is due to the fact that the leading edge load distribution is highly sensitive to the inlet flow properties[1].

2.2.1. Trial & Error

Having introduced the variables which define the loading distribution, it is important to notice that these variables have to be tuned in order to meet the manufacturing requirements - such as the leading edge radius ($\frac{R_{LE}}{c} \geq 1.25 \cdot 10^{-2}$). The trailing edge radius ($\frac{R_{TE}}{c}$) is set at $1.25 \cdot 10^{-2}$ throughout the whole study. These values are dictated mainly by the manufacturing capabilities in the turbomachinery industry.

This concept is very important because there are blades which might not satisfy both the aerodynamic style and the duty imposed. The trial and error tuning of the loading behavior at the leading edge can be seen as a design limit but it can be also seen as a first *filtering operation* over the aerodynamic style and the aerodynamic duty for appreciable results.

[1]Defined by M_1 and α_1.

The ***trial and error*** approach has been used mainly over the loading behavioiur at the leading edge - both at the suction side and the pressure side of the blade. These corrections are mainly made on the behavior of the conjunction point between the ellipse shaped loading at the leading edge with the straight line after it.

This behavior is of paramount importance as it enables the convergence of a specific blade set in terms of loading distribution and the exit flow angle.

Figures 2.10 provide visualizations of the load properties obtained using a **spline parametrization**:

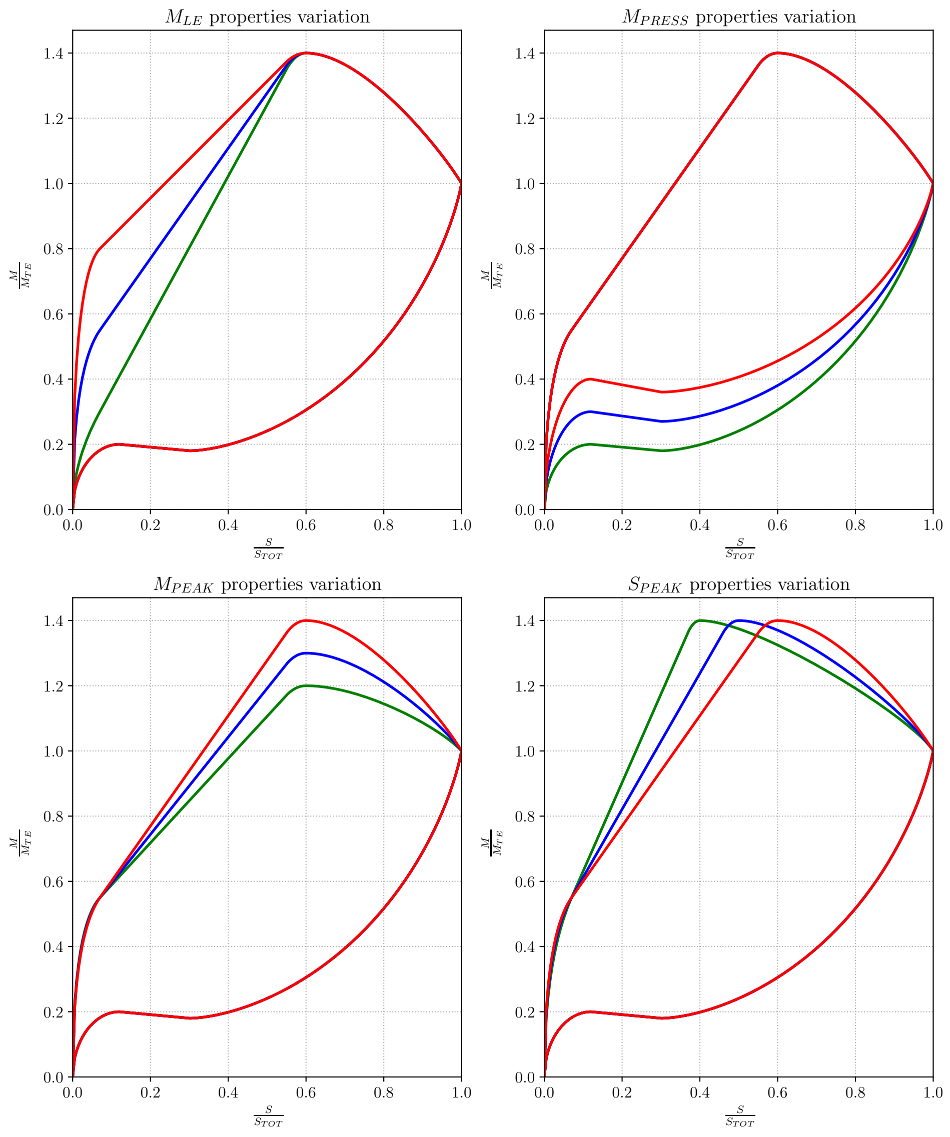


Figure 2.10: Aerodynamic style properties.

3 | MISES

The following sections aim to explain a key software component for the computation of the database.

The optimizer, which generates the blades that will be part of the database, incorporates the `MISES` program[1]. This program utilizes a **streamtube solver** to solve the flow. The flow computation is rapid, and its results are employed for extrapolating the **load distribution along the blade** and determining **main flow properties**, such as the flow exit angle.

`MISES` is a widely recognized software in the turbomachinery industry, renowned for its robustness, speed, and reliable results. Extensive testing has been conducted to analyze the reliability of its results. Grid independence has also been evaluated across various test cases.

The program comprises distinct blocks, each serving a specific purpose. The primary blocks include:

- `ISES`: A library responsible for generating the grid based on the blade geometry.
- `ISET`: A library used to compute the flow properties within the blade channel.

The subsequent sections will explain the configuration files used within the program and the main results obtained.

3.1. ISET

The initial step involves **loading** the geometry into the software and performing **grid generation**.

The grid generation process is closely tied to streamtubes that define flow properties. The blade geometry is uploaded using the `blade.datablade` file. This file contains the **blade coordinates** in the **blade-to-blade plane**. The grid computation is initiated

[1]A closed source program developed by `MIT`.

through an initial incompressible flow simulation. This rapid grid generation exhibits high correlation, particularly for high-pressure turbine blades, with the **final flow** path.

Listing 3.1: `blade.datablade` setup.

```
blade
-1.191754 999 2.000000  2.0  0.861 ! SINL | SOUT/KUTTA | CHINL |
   CHOUT | PITCH
0.988528      1.172492            ! X(0)    Y(0)
0.988486      1.172401            ! X(1)    Y(1)
0.988148      1.171680            ! X(2)    Y(2)
0.987473      1.170240            ! X(3)    Y(3)
...           ...                 ! ...     ...
1.010386      1.159661            ! X(240)  Y(240)
1.011081      1.161517            ! X(241)  Y(241)
1.011428      1.162447            ! X(242)  Y(242)
```

The first row of Listing 3.1 establishes the **primary flow properties** and **geometrical properties** of the grid, which include:

- `SINL`: Inlet flow slope
- `SOUT/KUTTA`: Outlet flow slope or Kutta condition (set to 999)
- `CHINL`: Grid inlet position
- `CHOUT`: Grid outlet position
- `PITCH`: Blade pitch

Following the definition of the main grid and flow properties, the **coordinates are imported** into `blade.datablade`.

It is important to note that the analyzed blades have an open trailing edge. This is due to the fact that wake model used in `MISES` is built on an open trailing edge, especially for simulations involving HPT blades.

`ISET` specifically focuses on loading the blade geometry and generating the grid. `CHINL` and `CHOUT` define the grid boundaries, while `SINL` and `SOUT/KUTTA` provide an initial approximation of the potential flow around the blade, which contributes to the grid generation.

3.2. ISES

The second step involves configuring the flow solver. The `ises.datablade` configuration file is used to set up the flow solver, including flow solution and turbulence modeling.

Listing 3.2 outlines the general problem setup.

Listing 3.2: `ises.datablade` setup.

```
1 2 5 6 15 ! INL SLOPE | EXIT SLOPE | LE STAGNATION | EXIT PRESSURE | INL MACH
1 3 4 6 17 ! INL SLOPE DRIVE | LE/TE KUTTA | INL P0/P0r DRIVE | OUT MACH DRIVE
0.25 0. 1.3 -0.5 ! INL MACH  | INL  P1/P0a | INL SLOPE | INL POSITION
1.10 0.68 0. 1.3 ! OUT MACH  | OUT P2/P0a  | OUT SLOPE | OUT POSITION
0.   0.          ! MFRin     | HWRAT       |
6E+5 4.0         ! REYNOLDS  | NCRIT       |
0.15 0.15        ! XTRANS1   | XTRANS2     |
4    0.9 1.0     ! ISMOM     | MCRIT       | MUCON
0.   0.          ! BVR1in    | BVR2in      |
```

3.2.1. Variables and Constraints

The first row of Listing 3.2 sets up the **flow variables** which in this case are:

- `INL SLOPE`: Inlet flow angle α_1, expressed as a tangent value
- `EXIT SLOPE`: Outlet flow angle α_2, expressed as a tangent value
- `LE STAGNATION`: Stagnation point at the leading edge, adjustable by the solver from the computed value computed by `ISET`
- `EXIT PRESSURE`: Outlet static pressure
- `INL MACH`: Inlet Mach number

The second row specifies the **constraints** applied during the simulation and the **strategy** for solving the flow. Constraints include:

- `LE KUTTA`: Kutta condition at the leading edge
- `TE KUTTA`: Kutta condition at the trailing edge

Solver strategies are:

- `INL SLOPE DRIVE`: Driving α_1 to the user-defined value
- `INL P0/P0r DRIVE`: Driving the pressure ratio $\frac{p_0}{p_{0,r}}$ to the user-defined value
- `OUT MACH DRIVE`: Driving M_2 to the user-defined value

The initial flow properties and target variables for achieving solution convergence are set in the third and fourth rows of `ises.datablade`.

3.2.2. Boundary Conditions

Similar to `ISET` for the grid properties, `ISES` uses variables to define inlet and outlet boundary condition positions. Additionally, boundary condition properties are established to numerically define the problem. Imposed boundary conditions include:

- Inlet flow angle α_1
- Exit Mach number M_2
- Kutta condition at the trailing edge of the blade
- the Kutta condition at the leading edge of the blade - it can be seen as a reverse Kutta condition at the trailing edge adapted to the leading edge: this in order to gurantee physical values and directionality of the flow at the leading edge.

3.2.3. Flow Solver Strategy

The solver adopts a *driven-based approach*.

The flow properties are initialized using `ises.datablade`. Through multiple iterations, the solver refines the flow approximation while driving the solution towards the specified boundary condition values, such as α_1 (`INL SLOPE`), M_2 (`OUT MACH`), and $\frac{p_0}{p_{0,r}}$ (`INL P1/P0a`).

The exit flow angle is not part of the *driven-variables* as it is an optimization parameter.

The solver within `ISES` offers multiple strategies to solve the problem, [4, Ch. 4]. The solver utilized in `datablade` is a *partially isentropic solver*. It solves the problem treating the flow as **isentropic** by default. However, if shocks are present in the blade channel, the solver considers these entropy variations, adjusting the flow properties [14]. The presence of shocks is analyzed based on the density field ρ.

3.2.4. Turbulence Modeling

Turbulence is modeled inside the program [4, Ch. 4] following the e^n model [3] and the **Abu-Ghannam-Shaw** model [3]. The *swapping* between these two models is related to the NCRIT parameter which is a **weight used for the turbulence models setup**. Once one of the two models is *triggered*, that model **predicts** the flow **transition** and the **turbulence properties**.

In order to increase the simulation speed, XTRANS1 and XTRANS2 are set to 0.15; these two variables predict the transition point over the blade. The transition over the blade is then *modified* by the **turbulence models** inside ISES. The program starts simulating the flow using the transition point set by the designer. If the transition point does not suit physics, the program starts computing the boundary layer separation point using one of the aforementioned transition models. Once the separation point is computed, the physical quantities, related to the laminar and turbulent regime of the flow, are computed by the flow solver. To sum up, the key parameters for the turbulence modeling are the Reynold number, REYNOLDS, the NCRIT parameter which is a triggering parameter between the two transition models and the XTRANS1 and XTRANS2 values which are the speeding up parameters for the transition point approximation.

Row 6 and 7 in Listing 3.1 show the turbulence modeling setup.

3.3. MISES Steps

Figure 3.1 displays a blade pre-processed by ISET for a spline interpolation of the geometry. This can be seen as the preprocessing for the grid generation of the problem

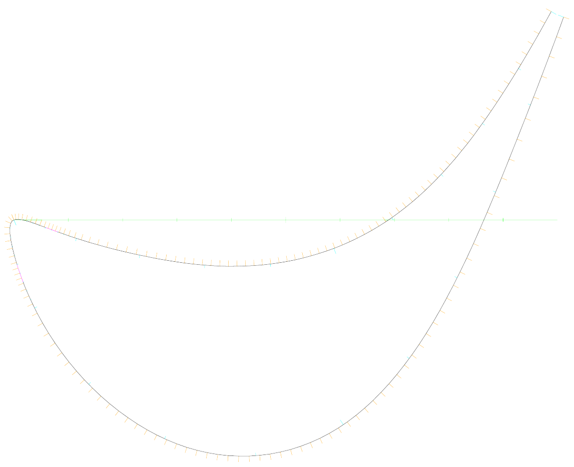


Figure 3.1: Spline-interpolated blade.

Following the spline approximation, there is an intermediate step for allocating the grid properties and defining the position of the boundary conditions, as shown in Figure 3.2.

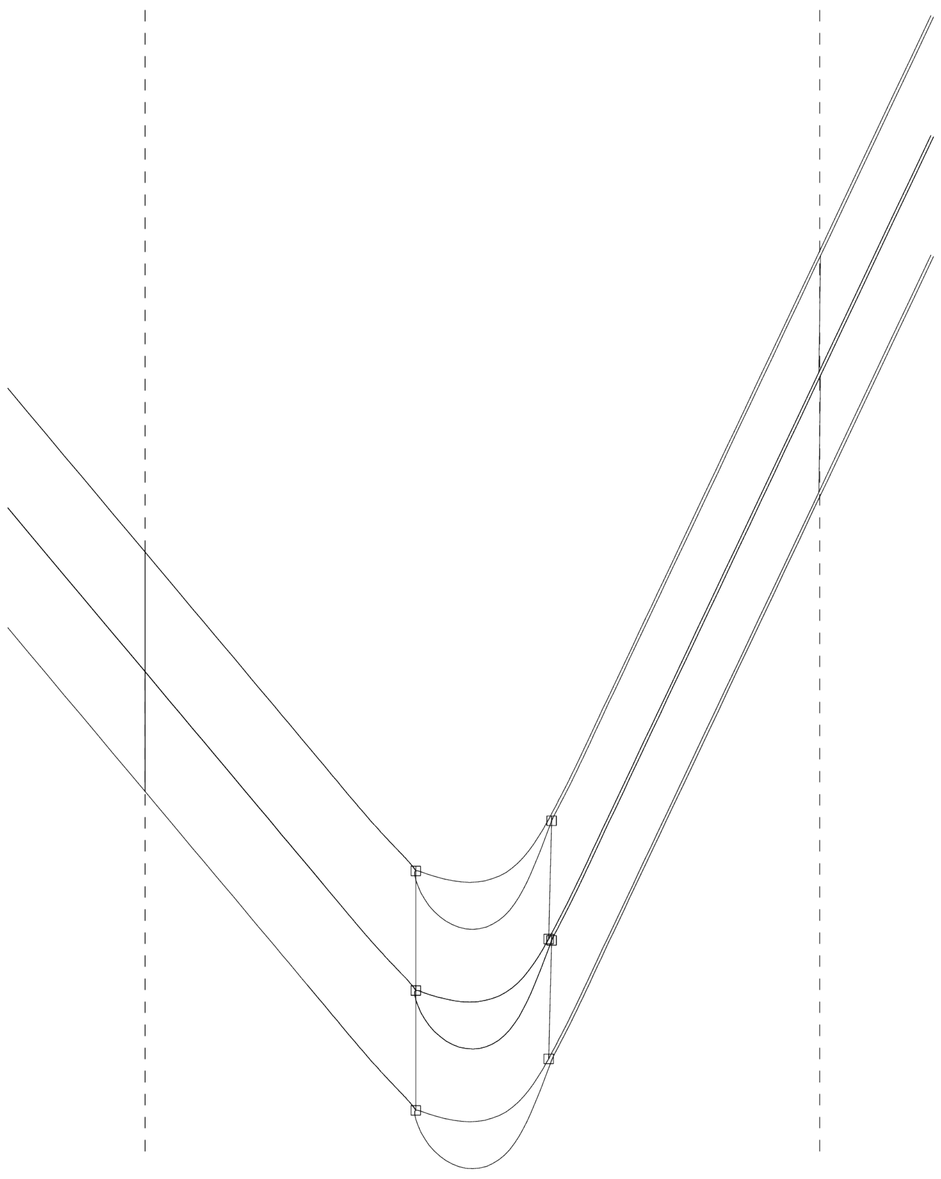


Figure 3.2: Grid properties allocation and boundary conditions setup.

The grid is consequently computed using a potential solver which generates an approximation of the flow that allows the generation of the study cells inside the domain.

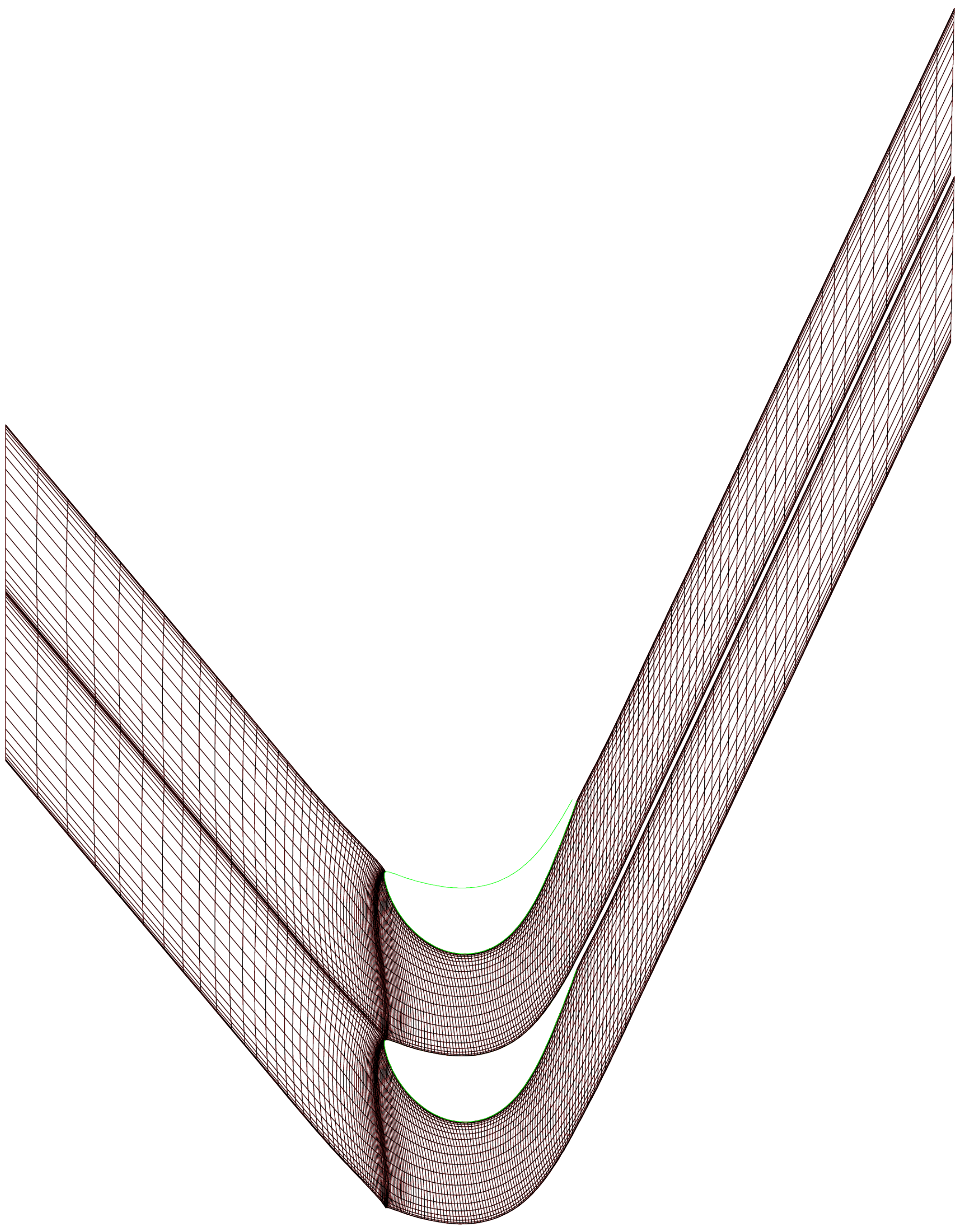


Figure 3.3: Grid used by `ISES` for flow property computation.

Flow properties are computed in `ISES` based on `ises.datablade` settings, as visualized in Figure 3.4.

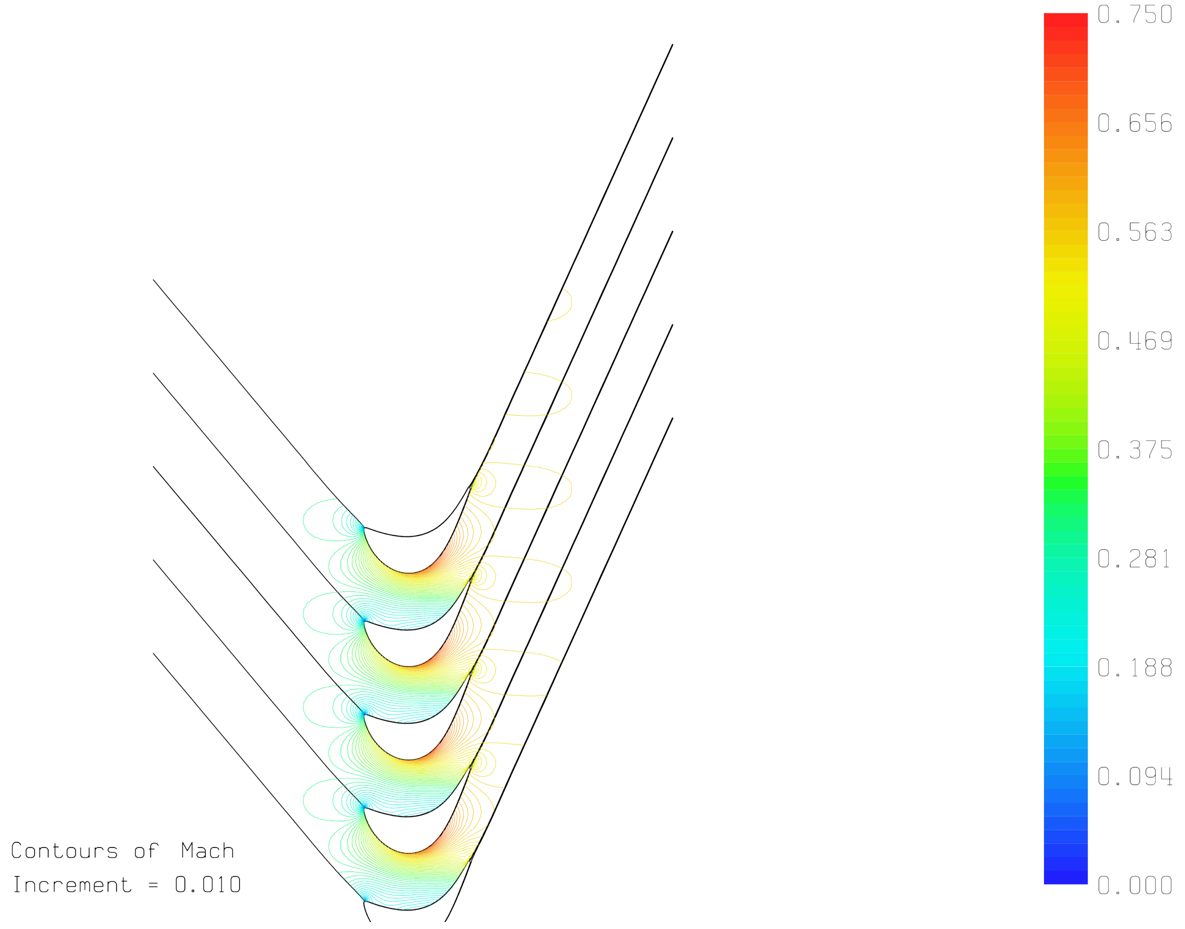


Figure 3.4: Contour plot of flow properties computed by `ISES` module.

Once flow is computed, the Mach fraction distribution along the blade is extracted using the `EDP` module in `MISES`. This module reads selected flow property (Mach number in this case) and saves it in a `.dat` file. The Mach fraction distribution on both suction and pressure sides of the blade is calculated using the two Mach numbers at the trailing edge of the blade which result different because the presence of the wake.

4 | datablade

This chapter introduces the **code** developed for the **database generation** and **analysis**. It will also explain the main aspects of the **code** in terms of **blade generation** and **optimization strategy**. The code has been named `datablade`.

4.1. Structure

Figure 4.1 represents the directory subdivision of `datablade`. The program structure is based on blocks that can be combined for different purposes.

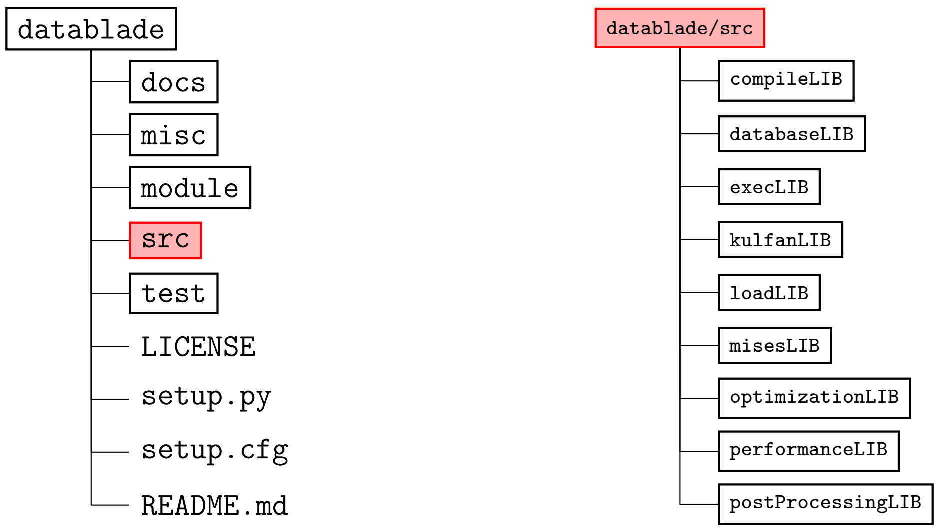


Figure 4.1: `datablade` structure

4.2. Configuration File

`datablade` optimization is configured using a **configuration file**. The configuration file is in `.json` format for allowing better **reading** and **handability**.

The following listings are parts of the configuration file - `config.json` - used in `datablade`. All the dictionary entries are called *as closer as possible* to the physical variables.

Listing 4.1 initilizes the initial guess for the blade optimization. Starting from its geometry - camberline, suction side and pressure side properties - and allocating the pitch and the trailing edge radius (which is kept constant at $R_{TE} = 1.25 \cdot 10^{-2}$ for the whole database).

Listing 4.1: `config.json` structure: initial guess ($\boldsymbol{x}_0$).

```
"testNumber": 0, // test case number: optimization ID
"geometry": {  // geometry definition
  "x0": [       // initial guess setup
    2.842E+01, // stagger angle (in degrees)
    -5.300E+01, // metal inlet angle (in degrees)
    7.550E+01, // metal outlet angle (in degrees)
    3.663E-01, // Kulfan parametrization A0 (equal for the suction
        side and pressure side)
    5.994E-01, // Kulfan parametrization A1suct -> for the suction
        side (lower part)
    6.504E-01, // Kulfan parametrization A2suct
    6.268E-01, // Kulfan parametrization A3suct
    5.173E-01, // Kulfan parametrization A4suct
    4.958E-01, // Kulfan parametrization A5suct
    5.081E-01, // Kulfan parametrization A6suct
    6.151E-01, // Kulfan parametrization A7suct
    8.558E-01, // Kulfan parametrization A8suct
    7.865E-01, // Kulfan parametrization A1press ->  for the
       pressure side (upper part)
    5.516E-01, // Kulfan parametrization A2press
    6.878E-01, // Kulfan parametrization A3press
    8.276E-01, // Kulfan parametrization A4press
    1.005E+00, // Kulfan parametrization A5press
    1.197E+00, // Kulfan parametrization A6press
    1.594E+00, // Kulfan parametrization A7press
    2.127E+00, // Kulfan parametrization A8press
    1.094E+00  // blade pitch
],
  "Nsuct":  8, // number of Kulfan parameters used at the suction
     side
  "Npress": 8, // number of Kulfan parameters used at the pressure
      side
  "TEradius": 1.25E-2 // trailing edge radius
},
```

Listing 4.2 defines the aerodynamic style and aerodynamic duty of the blade.

Listing 4.2: `config.json` structure: aerodynamic style and aerodynamic duty setup.

```
"aerodynamicDuty": { // aerodynamic duty properties -> macroscopic
    properties of the flow
  "alpha1": -50, // flow inlet angle (in degrees)
  "alpha2": 72.5, // flow outlet angle (in degrees)
  "MOUT": 0.7, // outlet Mach number (used at the boundary
    conditions for the MISES simulation)
  "Reynolds": 6E+05 // Reynolds number (used for the turbulence
    model setup in MISES)
},
"aerodynamicStyle": { // aerodynamic style properties -> local
   load distribution of the load (expressed in Mach number
   fraction [M / Mte]) over the blade (surface length fraction [S
   / Stot])
  "Mleading": 1.8, // leading edge Mach number fraction at the
    suction side
  "Mpeak": 1.4, // peak Mach number fraction at the suction side
  "Mpress": 1.0, // peak Mach number fraction at the pressure side
     before raising
  "Lpeak": 0.6, // position of the peak Mach number fraction on
    the suction side (expressed in surface fraction)
  "LEfrac": 0.1, // position of the leading edge Mach number
    fraction at the suction side (expressed in surface fraction)
  "PRESSfrac": 0.4 // position of Mach fraction raising at the
    pressure side (expressed in surface fraction)
},
```

Listing 4.3 sets the `MISES` simulation's entries and defines the parameters used in the cost function.

Listing 4.3: `config.json` structure: `MISES` configuration and cost function parameters.

```
"mises": { // MISES simulation properties
  "MINL":   0.2, // inlet Mach number guess (it will be corrected
    by MISES during simulation)
  "PINL":   0.6, // inlet static pressure over total pressure
    guess (it will be corrected by MISES during simulation)
  "POUT":   0.2, // outlet static pressure over total pressure
    guess (it will be corrected by MISES during simulation)
  "BCINL": -0.5, // inlet boundary conditions station
  "BCOUT":  1.5, // outlet boundary conditions station
  "nIter": -50,  // number of MISES iterations (check MISES user-
     guide for understanding the '-' sign)
  "nCrit":  2.0, // transition critical number between e^n
     transition model and Abu-Ghannam-Shaw transition model
  "Xtr1":   0.2, // transition position in chord percentage for
     the upper side of the blade (this value will be used as guess
      by MISES)
  "Xtr2":   0.2  // transition position in chord percentage for
     the lower side of the blade (this value will be used as guess
      by MISES)
},
"cost": { // cost function properties -- cost = RMSE * [1.0 +
   factor * (max(0, |angleError - threshold|)) * 2.0]**exponent
  "exponent":  1.5, // exponent of the cost function
  "factor":    0.1, // factor of the cost function
  "threshold": 0.4  // angle threshold of the cost function
},
```

At the end of `config.json`, in Listing 4.4, there is the definition of the optimization strategy used in `datablade`.

Listing 4.4: `config.json` structure: optimization strategy setup.

```
"optimization": { // optimization properties
"method": {
    "method": "Nelder-Mead", // enables Nelder-Mead simplex method
    "tol":    0.025, // cost function tolerance
    "PITCH":  null, // enables pitch as DOF treated by the
       optimization algorithm
    "iterationTol": 1000, // max number of iteration
    "stdDeviationTol": 1E-1, // max value for the standard
       distribution of the error
    "metalInBounds": 4, // bounds for the inlet metal angles
    "metalOutBounds": 4, // bounds for the outlet metal angles
    "Nsuct":  8, // suction side number of DOF
    "Npress": 8  // pressure side number of
  }
}
```

4.3. Optimization

The optimization algorithm used in `datablade` is a classic **gradient-free** method. The method used is the **Simplex** method [11]. Even though the method finds the **local minima**, satisfactory results can be guaranteed using:

- an appropriate choice of the initial guess, $\boldsymbol{x}_0$
- a **dimensionality adaptation strategy** for the problem
- an appropriate cost function which guarantees reliable results

4.3.1. Dimensionality Adaptation

The optimization of blades poses a challenging multidimensional optimization problem, demanding significant time and resources to attain the optimal solution. Given this complexity, an *intelligent approach* is necessary to streamline blade optimization. The strategy employed in this study demonstrates the ability to yield *favorable outcomes* for constructing the database. This strategy involves optimizing blades by **manipulating the dimensionality of the problem** [1].

The optimization process initiates by optimizing a blade with a **limited set of parameters**. While a blade with restricted degrees of freedom **may not produce an ideal outcome**, it does contribute to **flattening the design space**, thereby **mitigating unfavorable design intervals** where the optimizer might struggle to converge. Once optimization with fewer parameters achieves convergence, it's highly probable that increasing the degree of freedom will lead to **improved solutions**.

Empirical tests have revealed that the most favorable outcomes are achieved by **doubling** the degree of freedom in blade parametrization after each optimization cycle. Should a blade successfully converge with a low degree of freedom parametrization, the computed blade is subsequently scaled to populate the database with blades parametrized with the same number of parameters.

4.3.2. Cost Function

The blade optimization is related to two main errors:

- the **load error** over the blade, which is computed as the root mean squared error over the loading
- the **flow exit angle error**

The root mean squared error is computed using Equation (4.1). The exit angle flow error is then computed using Equation (4.2).

$$RMSE = \sqrt{\frac{1}{N_{points}} \cdot \sum_{i=1}^{N_{points}} \left(\frac{M_{real}}{M_{TE,real}}\bigg|_i - \frac{M_{target}}{M_{TE,target}}\bigg|_i \right)^2} \tag{4.1}$$

$$\Delta\alpha_2 = \left|\alpha_{2,real} - \alpha_{2,target}\right| \tag{4.2}$$

Equation (4.1) gets the Mach fraction distribution along the blade using the `.dat` file computed by `EDP` module in `MISES` and the Mach fraction distribution using the aerodynamic style curve computed in Chapter 2.

Once $RMSE$ and $\Delta\alpha_2$ have been computed, the cost function, Equation (4.3), is a blend of those properties. It is important that the cost function relates the $RMSE$ and $\Delta\alpha_2$ with a **product**; this in order to reduce the *competition* between the two variables during the optimization.

$$cost = RMSE \cdot \left[1 + 0.04 \cdot \Big(max\big(0,\ \Delta\alpha_2 - 1.0\big)\Big)^{2.0}\right] \tag{4.3}$$

Equation 4.3 features the product between the $RMSE$ with another term which is a blend of different quantities. The 0.04 factor represents a scale which adapts the angle error to the overall cost function, it dictates how important the angle error, $\Delta\alpha_2$, should be compared to the $RMSE$. The scaling value acts on the terms in round parenteses. The value in the round parenteses is a threshold switch for the angle error. The threshold is set up by the $max(0,\ \Delta\alpha_2 - 1.0)$ term. Once $\Delta\alpha_2 - 1.0 > 0$, the switch activates and $\Delta\alpha_2$ contributes to the cost function. On top of that, the *switch* is **squared**; this in order to allow a **smooth transition of the error** from a region where *just* $RMSE$ contributes to the cost function - for $\Delta\alpha_2 - 1.0 \leq 0$ - to the region where both $RMSE$ and $\Delta\alpha_2$ contribute to the overall cost function - for $\Delta\alpha_2 - 1.0 > 0$.

The cost function serves a dual purpose. Firstly, it establishes a *guideline* governing the convergence behavior of the optimization algorithm. Secondly, it provides a *threshold of acceptability* for the optimized blades. If the cost function value for an optimized blade is high, that particular blade might be excluded to ensure a more precise database.

4.4. Optimizer

On top of the optimization strategy there is the optimizer. The `datablade` optimizer combines the `MISES` software with the `scipy` module [13]. These two components allow good performance and robustness.

Figure 4.2 illustrates how the blade is optimized by `datablade`. It comprehends the optimization algorithm keeping the blade dimensionality fixed alongside the scaling process to increase convergence. Figure 4.2 starts with the loading of the configuration file, `config.json`, which provides the initial guess - where the optimization starts -, the aerodynamic style and the aerodynamic duty of the blade. Once these properties are loaded, the program starts to optimize the blade using an optimization based on the dimensionality adaptation of the blade to the problem.

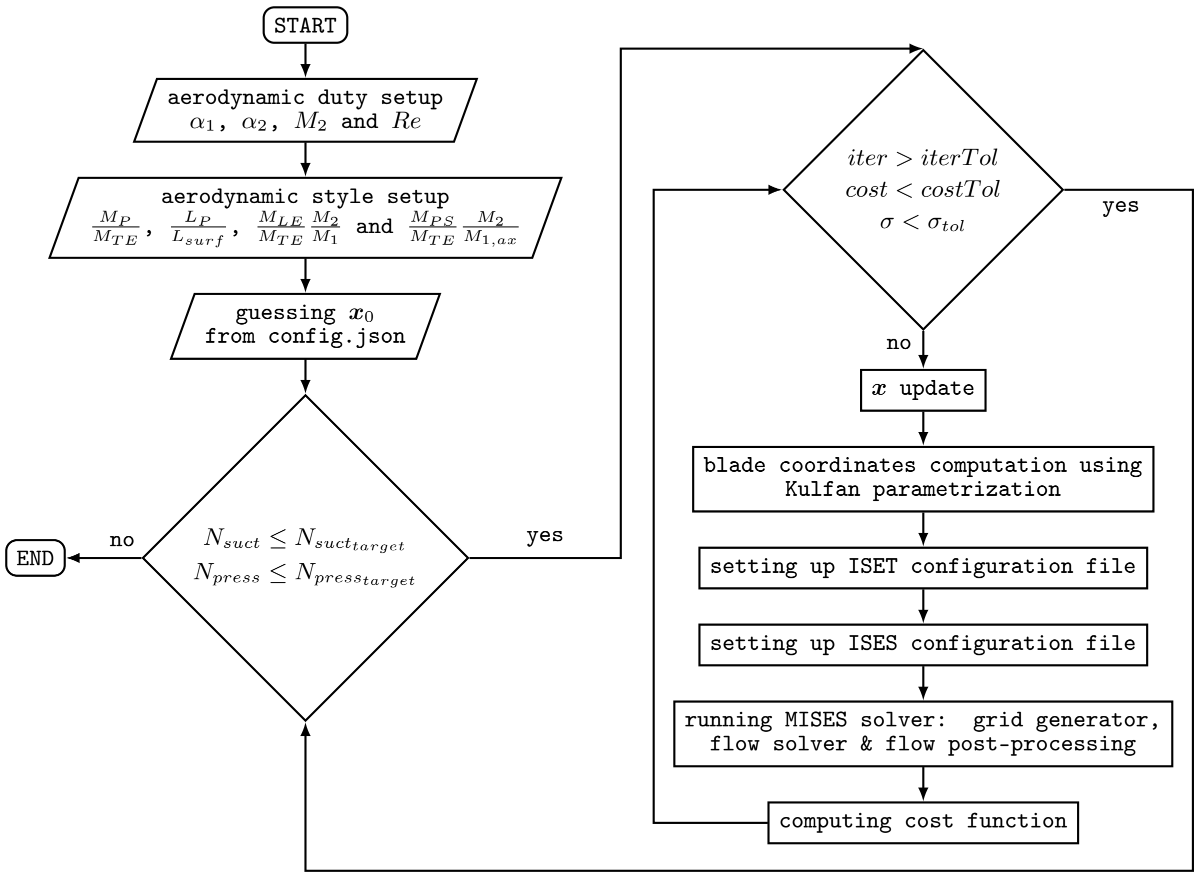

START
aerodynamic duty setup α_1, α_2, M_2 and Re
aerodynamic style setup $\frac{M_P}{M_{TE}}$, $\frac{L_P}{L_{surf}}$, $\frac{M_{LE}}{M_{TE}}\frac{M_2}{M_1}$ and $\frac{M_{PS}}{M_{TE}}\frac{M_2}{M_{1,ax}}$
guessing x_0 from config.json
$N_{suct} \leq N_{suct_{target}}$ $N_{press} \leq N_{press_{target}}$
no
END
yes
$iter > iterTol$ $cost < costTol$ $\sigma < \sigma_{tol}$
yes
no
x update
blade coordinates computation using Kulfan parametrization
setting up ISET configuration file
setting up ISES configuration file
running MISES solver: grid generator, flow solver & flow post-processing
computing cost function

Figure 4.2: `datablade` optimizer structure.

5 | Database

The purpose of this chapter is to explain the generation of the database that will be used by the interpolation algorithm in Chapter 6. It starts explaining the domain of study and then it will define an appropriate strategy for increasing the overall convergence speed of the whole database.

This strategy can be seen as an additional *acceleration* layer for the computation of an *appropriate* initial guess, $\boldsymbol{x}_0$, used in the optimizer.

Table 5.1: Domain boundaries and discretization.

Variable	**Min**	**Max**	**Points**
α_1	$-50°$	$-20°$	3
α_2	$65°$	$72.5°$	4
M_2	0.4	0.7	3
Re	$6 \cdot 10^5$	$6 \cdot 10^5$	1
$\frac{M_P}{M_{TE}}$	1.2	1.4	3
$\frac{L_P}{L_{surf}}$	0.5	0.6	3
$\frac{M_{LE}}{M_{TE}}\frac{M_2}{M_1}$	1.2	1.8	3
$\frac{M_{PS}}{M_{TE}}\frac{M_2}{M_{1,ax}}$	0.8	1.2	3

5.1. Domain

This study pertains to high-pressure turbine sections. The domain of study is presented in Table 5.1.

Due to the **Reynolds indipendence** of the studied flow,the investigation has been carried out using a fixed Reynolds number value - $Re = 6 \cdot 10^5$. α_1, α_2, M_2, and Re are physical quantities readily visualized within the problem setup.

The other four parameters, $\frac{M_{LE}}{M_{TE}}\frac{M_2}{M_1}$, $\frac{M_{PRESS}}{M_{TE}}\frac{M_2}{M_{1,ax}}$, $\frac{M_{PEAK}}{M_{TE}}$ and $\frac{L_{PEAK}}{L_{TOT}}$, constitutes engineering parameters that define the aerodynamic style properties. These parameters come out from numerours tests, aiming to identify values that both *align with physics* and *fit within blade capabilities.* These parameters are completelly arbitrary and are the most suitable for the intent of the present work.

The additional optimization strategy consists in linearizing the inner domain using outer domain points. This will result in optimizing first the outer points of the domain, Table 5.1, which will then provide a *suitable* initial guess, $\boldsymbol{x}_0$, for the inner domain points.

Figure 5.1 shows the linear interpolation of the stagger angle, γ, with respect to the inlet flow angle, α_1, and outlet flow angle, α_2.

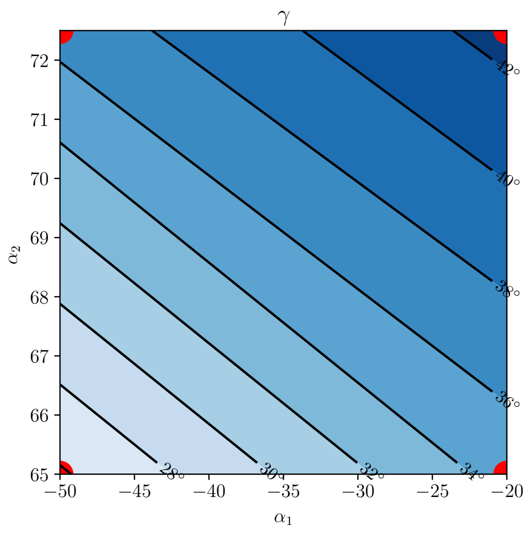


Figure 5.1: Linear interpolation of the stagger angle, γ, with respect to α_1 and α_2 from the corner points in red.

This interpolation procedure is applied to all the blade parameters: starting from the camberline parameters and ending to the blade thickness parameters.

5.1.1. Outer Points

The computation of the corner points of the domain is the first step for the generation of the database. These points are the most time-consuming to converge, as the optimizer might start from an initial guess far from the optimal configuration.

Computing these corner points is critical, as it accelerates the computation of the inner points of the domain and increases the probability of optimizer convergence for inner points.

With the domain boundaries listed in Table 5.1, `datablade` optimizes 128 blades to generate the corner points of the domain.

5.1.2. Inner Points

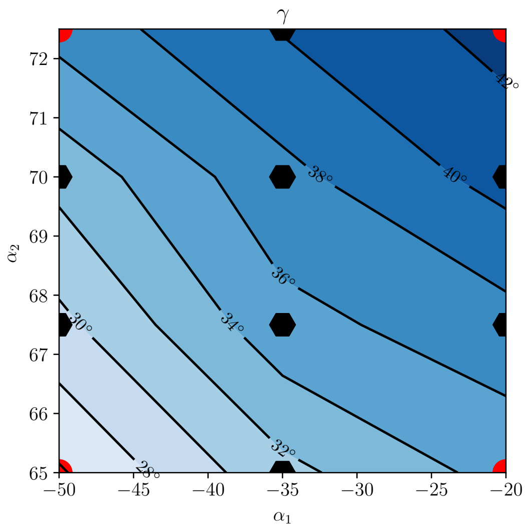


Figure 5.2: Complete γ variation with respect to α_1 & α_2. Corner points in red and inner points in black.

Once the corner points are generated, inner points are computed using linear interpolation. An example of linear interpolation is shown in Figure 5.1, which provides a preliminary estimation of the stagger angle, γ, variation based on the inlet flow angle, α_1, and the outlet flow angle, α_2.

According to Table 5.1, 2916 blades will be optimized for the inner points study.

Figure 5.2 shows the interpolated γ with respect to α_1 and α_2 after all database points have been computed.

5.2. Optimized Data

The database contains Kulfan's parameters for the definitions of the blade geometry. Below, some key blades of interest for initial database analysis are defined. The database's blades are categorized as follows:

- Blade configuration with **low error** on the loading distribution *and* **low error** on the exit flow angle
- Blade configuration with **low error** on the loading distribution *but* **high error** on the exit flow angle[1].
- Blade configuration with **high error** on the loading distribution *and* **high error** on the exit flow angle

Table 5.2 defines the complete database after optimization.

Table 5.2: Data properties inside the optimized database.

	count	$\boldsymbol{\mu}$	$\boldsymbol{\sigma}$	***min***	**25%**	**75%**	***max***
cost	2916	0.018399	0.006787	0.005445	0.013499	0.021949	0.049249
$\boldsymbol{\Delta\alpha_2}$	2916	0.965338°	0.542963°	0.000366°	0.472465°	1.382774°	2.377860°

Table 5.2 reveals an acceptable mean value for both *cost* and $\Delta\alpha_2$. It is noteworthy that more than 75% of the database has a cost below 2.75%, indicating that the majority of the blades converge toward the desired aerodynamic style. This initial analysis is crucial as it indirectly suggests that only a few blades will be excluded from the machine learning analysis.

Another important observation from Table 5.2 is that the machine learning algorithm must also address the correction of the exit flow angle, α_2. This is necessary because over 25% of the database exhibits an exit flow angle error, $\Delta\alpha_2$, exceeding 1°.

Figure 5.3 showcases the best results achievable by the optimizer. The computed load adheres to the target load, while maintaining an exit angle error, $\Delta\alpha_2$, below an acceptable threshold of 1°.

[1]In the present work it is considered a high exit flow angle error: $\Delta\alpha_1 > 1°$.

$\frac{M_{LE}}{M_{TE}} \frac{M_2}{M_1}$	$\frac{M_{PRESS}}{M_{TE}} \frac{M_2}{M_{1,ax}}$	$\frac{M_{PEAK}}{M_{TE}}$	$\frac{S}{S_{TOT}}$
1.80	1.00	1.40	0.50

cost	$\Delta\alpha_2$
1.91E-02	0.20°

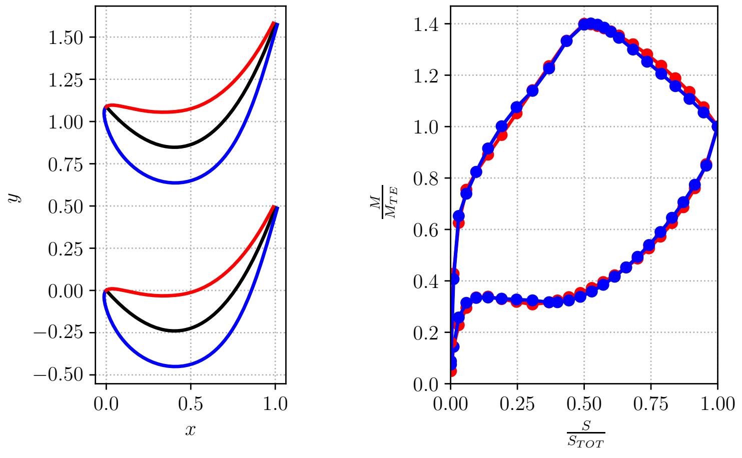


Figure 5.3: Blade with good performances on the load distribution and on the exit angle error.

Due to physical constraints and the optimization algorithm, not every optimized blade simultaneously achieves the desired loading distribution and a low exit angle error.

Figure 5.4 represents a *poor-performing* blade. Such blades are excluded from the machine learning analysis. The exclusion is due to the high exit angle error and the unacceptable discrepancy between the computed load and the target load.

$\frac{M_{LE}}{M_{TE}} \frac{M_2}{M_1}$	$\frac{M_{PRESS}}{M_{TE}} \frac{M_2}{M_{1,ax}}$	$\frac{M_{PEAK}}{M_{TE}}$	$\frac{S}{S_{TOT}}$
1.20	0.80	1.20	0.60

cost	$\Delta\alpha_2$
3.92E-02	1.42°

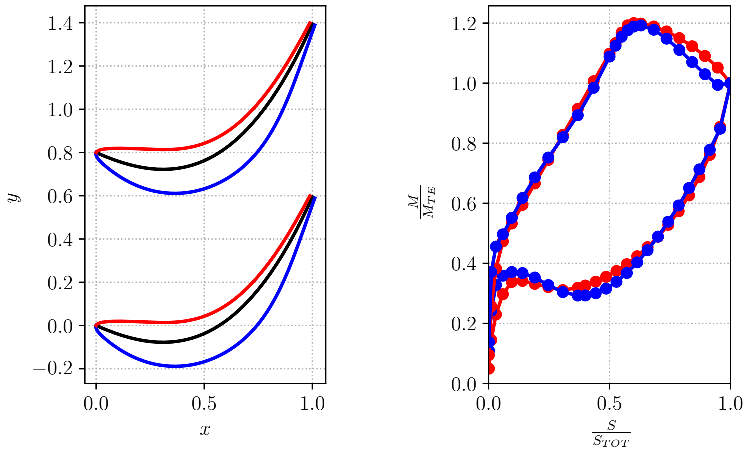


Figure 5.4: Blade with poor performances on the load distribution and on the exit angle error.

These blades do not contribute effectively to the database because they do not appropriately match the loading distribution. The primary issue with these blades is the loading distribution error. If a blade exhibits a substantial error in loading distribution, the information it provides holds no relevance within the studied design space. This is because the blade configuration lacks correlation with the desired aerodynamic style.

Figure 5.5 depicts an acceptable blade. Despite its elevated exit angle error, the machine learning algorithm can rectify this error by shifting the interpolated field to achieve a better fit for the exit angle. As Chapter 6 will demonstrate, the exit angle error can be corrected using a suitable machine learning network. This error can be interpreted as an information about exit angle behavior within specific regions of the design space.

$\frac{M_{LE}}{M_{TE}}$ $\frac{M_2}{M_1}$	$\frac{M_{PRESS}}{M_{TE}}$ $\frac{M_2}{M_{1,ax}}$	$\frac{M_{PEAK}}{M_{TE}}$	$\frac{S}{S_{TOT}}$
1.20	0.80	1.40	0.60

cost	$\Delta\alpha_2$
1.63E-02	1.64°

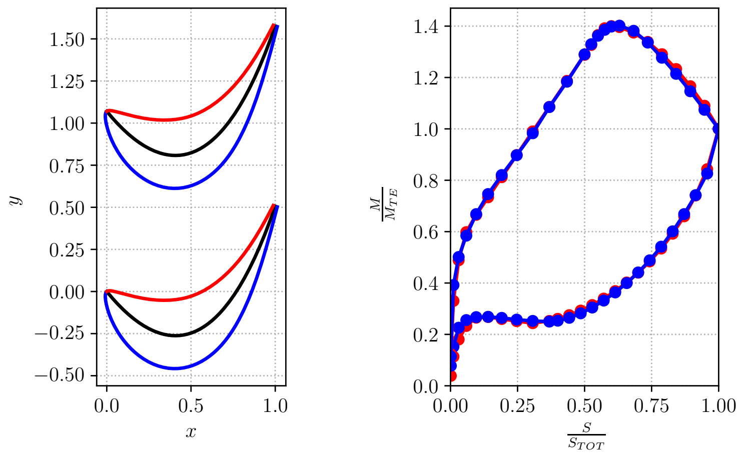


Figure 5.5: Blade with good performances on the load distribution but low performance on the exit angle error.

5.2.1. Error Distribution

As discussed in the previous sections, there are blades which present a significant error on the loading distribution, denoted by $RMSE$. This error cannot be corrected by the machine learning algorithm under any circumstances; this is related by the fact that **bad input data** will inevitably lead to **bad output data**. Blades falling into this category of *unacceptable designs* are referred to as **unfeasible designs**.

These particular configurations will be omitted to provide the machine learning algorithm only with useful data for interpolation.

Having introduced the concept of **unfeasible design** it is necessary to define the regions of the database where the error on the aerodynamic style can be considered *not-acceptable*. In this study, the error in the aerodynamic style is considered *not-acceptable* if it exceeds a cost threshold of 2.75%.

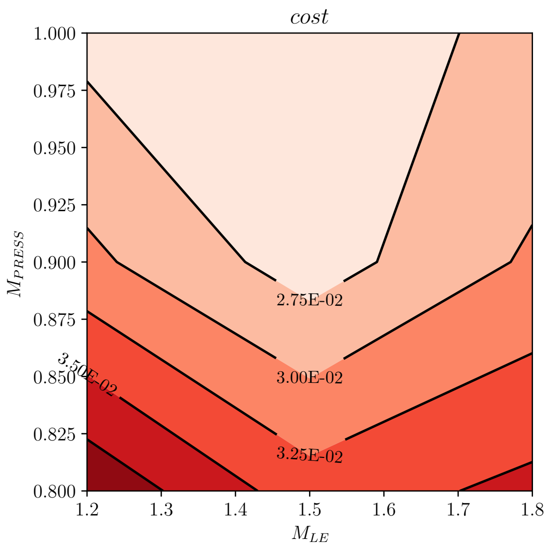


Figure 5.6: *cost* distribution for $\alpha_1 = -50°$, $\alpha_2 = 65°$, $\frac{M_{PEAK}}{M_{TE}} = 1.4$, $\frac{S_{PEAK}}{S_{TOT}} = 0.5$ and $M_2 = 0.4$.

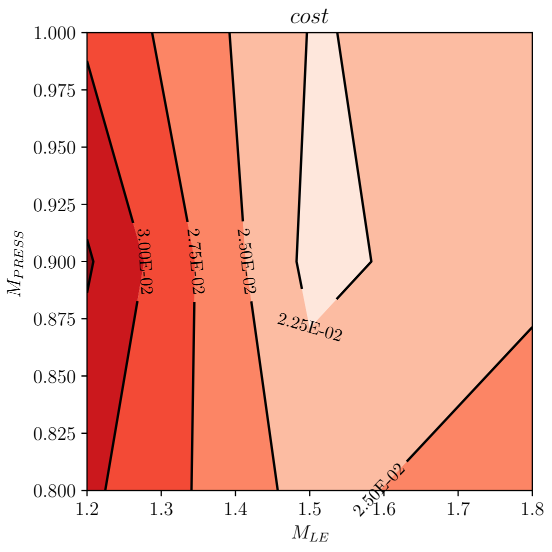


Figure 5.7: *cost* distribution for $\alpha_1 = -50°$, $\alpha_2 = 65°$, $\frac{M_{PEAK}}{M_{TE}} = 1.2$, $\frac{S_{PEAK}}{S_{TOT}} = 0.5$ and $M_2 = 0.4$.

Figure 5.6 and Figure 5.7 represent the two main regions where the error on the aerodynamic style is *not-acceptable*.

5.2.2. Filtering

Although the data within the database can be processed by the machine learning algorithm, it is necessary to *filter* the data in order to *feed* only *useful* entries to the interpolation algorithm - data which have significative relevance for the engineering purposes.

Table 5.3 presents the filtered database, representing a refined version after undergoing this data filtering process.

Table 5.3: Filtered database.

	count	$\boldsymbol{\mu}$	$\boldsymbol{\sigma}$	***min***	**25%**	**75%**	***max***
cost	2787	0.015233	0.004874	0.005554	0.011249	0.018777	0.027483
$\boldsymbol{\Delta\alpha_2}$	2787	0.846587°	0.421034°	0.001229°	0.513713°	1.143698°	2.373522°

In this revised dataset, the number of blades is slightly reduced compared to the original database. This reduction is attributed to the exclusion of certain blades due to their inadmissible errors in blade loading.

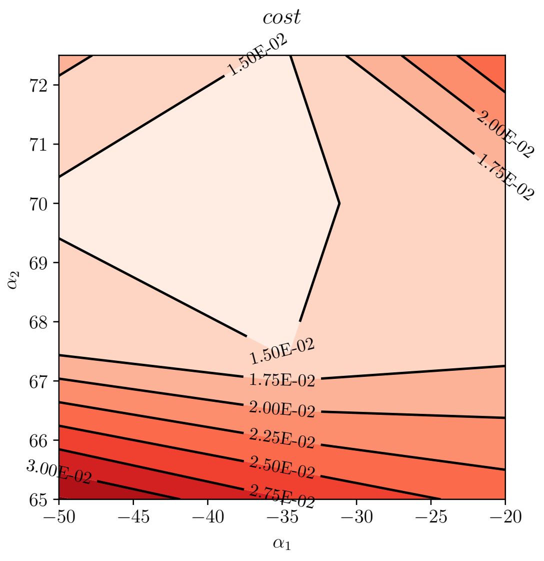


Figure 5.8: *cost* function behavior in the domain.

The **unfeasible designs** are concentrated at the *boundaries* of the domain. Figure 5.8 shows the cost function of a *slice* of the domain. The data with a cost above 2.75% have been dropped off.

The filtering operation has been conducted *only* over the ***cost*** properties of the blade. Even though the ***cost*** relates the $RMSE$ and the $\Delta\alpha_2$, its value is dictated mainly by the $RMSE$.

Figure 5.9 shows the error over the exit angle. The filtering operation are not made on the $\Delta\alpha_2$ domain.

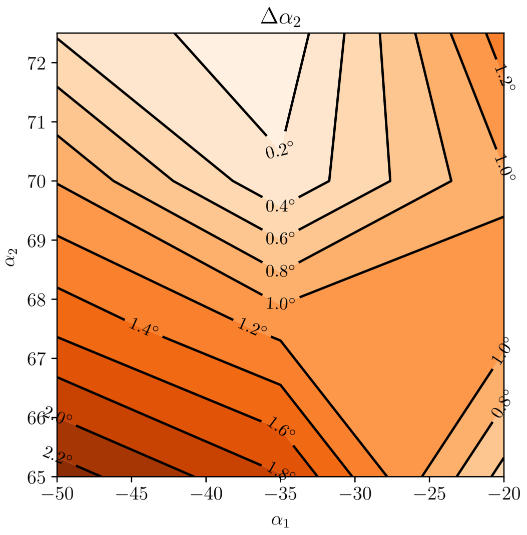


Figure 5.9: $\Delta\alpha_2$ function behavior in the domain.

6 | Artificial Intellingence

The present chapter has the purpose of *explaining* the database **interpolation** using a **machine learning model**.

6.1. Problem Framing

The machine learning algorithm is based on a regression algorithm. The regression algorithm is based on a domain interpolation which does not include any aleatory variable.

This algorithm *predicts* a **blade geometry** giving as input the **aerodynamic duty** and the **aerodynamic style**. The blades are parametrized the same way the database's blade are.

Figure 6.1 is a representation of the machine learning structure.

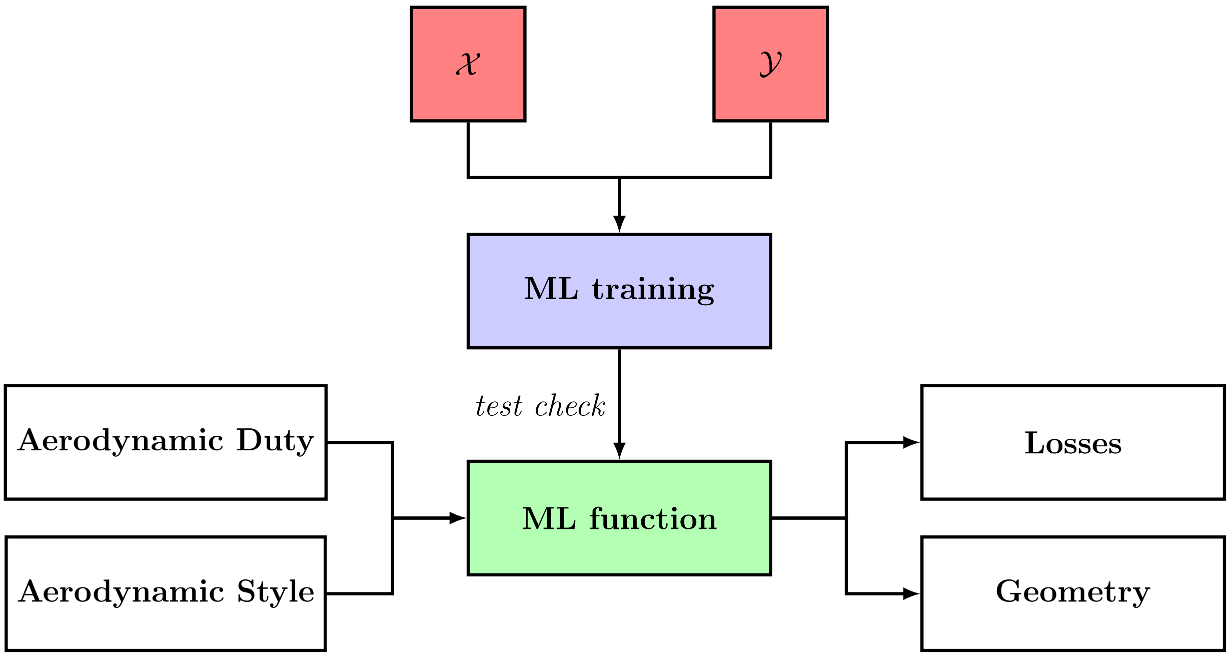


Figure 6.1: Machine learning model.

6.2. Interpolation

The regression starts with the definition of the domains of study:

- $\mathcal{X}$ contains the aerodynamic style and aerodynamic duty properties. $\mathcal{X} \in \mathbb{R}^{2787\times 8}$: where 2787 is the number of filtered blades and 8 are the number of parameters for defining the aerodynamic style and the aerodynamic duty of the blade. The $\boldsymbol{x}$ vector is the general element of $\mathcal{X}$
- $\mathcal{Y}$ stores the camberline and Kulfan parameters for the description of the blade. $\mathcal{Y} \in \mathbb{R}^{2787\times 42}$: where 2787 is the number of blades and 42 is the number of parameters used for representing each blade - Kulfan parameters and pitch. The $\boldsymbol{y}$ vector is the general element of $\mathcal{Y}$

The machine learning algorithm has the goal to find an approximation of a mapping function, f, which relates the two domains, $\mathcal{X}$ and $\mathcal{Y}$.

Ideally, the mapping function, f, defines Equation (6.1):

$$f_{(\boldsymbol{x})} = \boldsymbol{y} \tag{6.1}$$

The machine learning algorithm has the role of computing $\hat{f}$ which is the numerical approximation of f. The new function, $\hat{f}$, will define the following problem:

$$\hat{f}_{(\boldsymbol{x})} \approx \boldsymbol{y} \tag{6.2}$$

For the computation of $\hat{f}$, the $\mathcal{X}$ and $\mathcal{Y}$ domains are splitted into two sub-domains each. For the present work, a splitting factor of 30% has been used after extensive tests over the many possible factors. The splitting will then generate four sub-domains:

- $\mathcal{X}$ domain:
  - training set: $\mathcal{X}_*$, made by $\boldsymbol{x}_*$
  - test set: $\mathcal{X}_{**}$, made by $\boldsymbol{x}_{**}$
- $\mathcal{Y}$ domain:
  - training set: $\mathcal{Y}_*$, made by $\boldsymbol{y}_*$
  - test set: $\mathcal{Y}_{**}$, made by $\boldsymbol{y}_{**}$

6.2.1. Radial Basis Function

Once introduced the main purpose of the machine learning algorithm, it is necessary to find a **suitable regression algorithm** to interpolate the domain. In the present work, the radial basis functions [7] (RBF) are used as the kernel of the machine learning model. These functions are a very powerful tool for the interpolation of data. The overfitting problem is manageable and the model allows perfect interpolation of the training points, $(\boldsymbol{x}_*, \boldsymbol{y}_*)$.

Equation (6.4) formulates the radial basis functions kernel:

$$d_j(\boldsymbol{x}) = |\boldsymbol{x} - \boldsymbol{x}_*^j| \tag{6.3}$$

$$\Phi_{j_{(\boldsymbol{x},\ \boldsymbol{x}_*^j,\ c_j)}} = exp\big(- c_j^2\ d_j(\boldsymbol{x})^2\big) \tag{6.4}$$

The kernel on its side relies on its length scale, c_j, and on the euclidean norm between two points in the domain, $\boldsymbol{x}$ and $\boldsymbol{x}_*^j$, as presented in Equation (6.3).

From Equation (6.4) it is possible to understand that the higher the euclidean distance the lower the influence of a point $\boldsymbol{x}$ in the system. This is a very **powerful feature** because it decorrelates blades which define aerodynamic duties and aerodynamic styles that are completely different.

At the same time, the usage of gaussian functions for the approximation of the domain allows to **smooth out** $\hat{f}$. This feature is very important because it guarantees a smooth variation of the geometrical properties of the blade allowing a smooth change of the camberline and Kulfan parameters [5]. Figure 6.2 shows the radial basis functions with a fixed length scale.

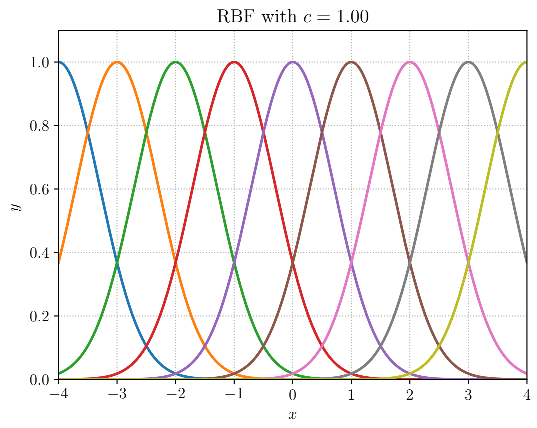


Figure 6.2: Radial Basis Functions.

6.2.2. Training & Testing

Having set the fundamentals of the machine learning algorithm, it is necessary to construct the system for the computation of $\hat{f}$. The problem is defined by a linear system of equations. These equations are a linear combinations of radial basis functions, Equation (6.4), which are weighted by parameters which take the name of *weights*. These weights have the role to *properly* fit the radial basis functions into the database. The following set of equations represent the problem:

$$f \approx \hat{f} = \boldsymbol{\Phi}_{(\boldsymbol{x})} \cdot \boldsymbol{w} \tag{6.5}$$

$$\boldsymbol{y} \approx \hat{f}_{(\boldsymbol{x},\boldsymbol{w})} = \boldsymbol{\Phi}_{(\boldsymbol{x})} \cdot \boldsymbol{w} \tag{6.6}$$

$$\boldsymbol{\Phi}_{(\boldsymbol{x})} = \begin{bmatrix} | & & | \\ \Phi_{0_{(\boldsymbol{x},\ c_0)}} & \dots & \Phi_{n_{(\boldsymbol{x},\ c_{m*})}} \\ | & & | \end{bmatrix} \tag{6.7}$$

Once the linear system is set, it is necessary to define a way for computing $\boldsymbol{w}$. The problem will reduce to a minimization problem. The problem is characterized by the search of the weight $\boldsymbol{w}$ such that a cost function is reduced to its minimum. The problem is represented by Equation (6.8):

$$min\left(J_{(\boldsymbol{x}*)} = \frac{1}{m_*} \sum_{j=0}^{m_*-1} \left(\hat{f}_{(\boldsymbol{x}_*^j,\boldsymbol{w})} - \boldsymbol{y}_*^j \right)^2 \right) \tag{6.8}$$

For example, the matricial form for the γ degree of freedom will be written as:

$$\begin{bmatrix} \gamma^0 \\ \vdots \\ \gamma^{m_*-1} \\ \gamma^{m_*} \end{bmatrix} = \begin{bmatrix} \Phi_{0_{(\boldsymbol{x}_*^0,c_0)}} & \dots & \Phi_{m_*-1_{(\boldsymbol{x}_*^0,c_{m*-1})}} & \Phi_{m_*{(\boldsymbol{x}_*^0,c_{m*})}} \\ \vdots & \ddots & \vdots & \vdots \\ \Phi_{0_{(\boldsymbol{x}_*^{m*-1},c_0)}} & \dots & \Phi_{m_*-1_{(\boldsymbol{x}_*^{m*-1},c_{m*-1})}} & \Phi_{m_*{(\boldsymbol{x}_*^{m*-1},c_{m*})}} \\ \Phi_{0_{(\boldsymbol{x}_*^{m*},c_0)}} & \dots & \Phi_{m_*-1_{(\boldsymbol{x}_*^{m*},c_{m*-1})}} & \Phi_{m_*{(\boldsymbol{x}_*^{m*},c_{m*})}} \end{bmatrix}_\gamma \cdot \begin{bmatrix} w^0 \\ \vdots \\ w^{m_*-1} \\ w^{m_*} \end{bmatrix}_\gamma \tag{6.9}$$

Equation (6.9) shows a piece of the minimization problem in Equation (6.8). In this case the left hand side of the equation shows a part of the $\boldsymbol{y}_*$ vector which correspond to the γ optimization problem. The whole system is a concatenation of problems spanning all the Kulfan's parameters and the pitch. The matrix in Equation (6.9) is a more detailed form of Equation (6.7). The right hand side of Equation (6.9) takes the γ subscript: this to indicate that the elements are referred only to the γ degree of freedom.

The optimization problem is solved by an iterative algorithm which finds the optimum solution, $\boldsymbol{w}$. The optimization is conducted over the **training set**, $(\mathcal{X}_*, \mathcal{Y}_*)$.

Once $\boldsymbol{w}$ is computed, it is necessary to understand the *quality* of the weights. This test is done using the **test set**, $(\mathcal{X}_{**}, \mathcal{Y}_{**})$ following Equation (6.10):

$$J_{(\boldsymbol{x}_{**})} = \frac{1}{m_{**}} \sum_{j=0}^{m_{**}-1} \left(\hat{f}_{(\boldsymbol{x}^j_{**}, \boldsymbol{w})} - \boldsymbol{y}^j_{**}\right)^2 \tag{6.10}$$

6.2.3. Sckit-Learn

The machine learning analysis of the system has been done using a robust and consolidated library: **Sckit-Learn**.

Listing 6.1 represents the commands for the normalization of the whole database. In terms of operativity and interpolation errors, it is common use to normalize data in $[0, 1]$ interval.

Listing 6.1: Data preprocessing

```python
from sklearn import preprocessing

# scaling database
# input
xScaler = preprocessing.MinMaxScaler()
xModel = xScaler.fit(x)
xNorm = xModel.transform(x)

# output
yScaler = preprocessing.MinMaxScaler()
yModel = yScaler.fit(y)
yNorm = yModel.transform(y)
```

Having preprocessed the data, it is necessary to split the normalized data into the training and test sets. Once the entries are randomly splitted, the RBF kernel is initialized for the usage in the interpolation algorithm. The final step for the interpolation of the database consists into the minimization of the J functional which computes the $\boldsymbol{w}$ vector.

Listing 6.2 represents all the above steps.

Listing 6.2: RBF interpolation

```python
from sklearn import model_selection
from sklearn.gaussian_process import GaussianProcessRegressor
from sklearn.gaussian_process.kernels import RBF

# splitting data into test and train data
xTrainNorm, xTestNorm, yTrainNorm, yTestNorm = model_selection.
    train_test_split(xNorm, yNorm, test_size=0.30, random_state
    =42)

# kernel setup
kernel = RBF(length_scale=0.1)

# geometry function object optimization
geometryFunc = GaussianProcessRegressor(kernel=kernel,
    random_state=0).fit(xTrainNorm, yTrainNorm)
```

Having optimized the radial basis function network, it is necessary to understand its properties; a test over the network is done using the test set.

Listing 6.3 computes the score of the interpolated network.

Listing 6.3: RBF scoring

```python
# computing SCORE
SCORE = geometryFunc.score(xTestNorm, yTestNorm)
print('SCORE = {0:.2E}'.format(SCORE)) # SCORE = 9.75E-01
```

The test scores the computed function, $\hat{f}$. The score gives an idea of how well the model fits the data in terms of explained variance.

The test scores a 3% error. For the present work, the $\hat{f}$ computed satisfies the projects requirements. As result, $\hat{f}$ approximates the blade geometry given as inputs the aerodynamic style and the aerodynamic duty.

6.2.4. Tuning

Tuning the **hyperparameters** for a RBF kernel is an important step to achieve *optimal performance*. The key hyperparameters to focus for the RBF network are:

- Length Scale (l): This hyperparameter controls the *smoothness* of the RBF kernel. A **smaller** length scale leads to a more **wiggly** and **flexible** kernel, while a **larger** length scale makes it **smoother**.
- α: This parameter determines the **regularization strength** of the model. It helps to **prevent overfitting** and stabilize the computations. A **larger** α places stronger

emphasis on **noise reduction**.

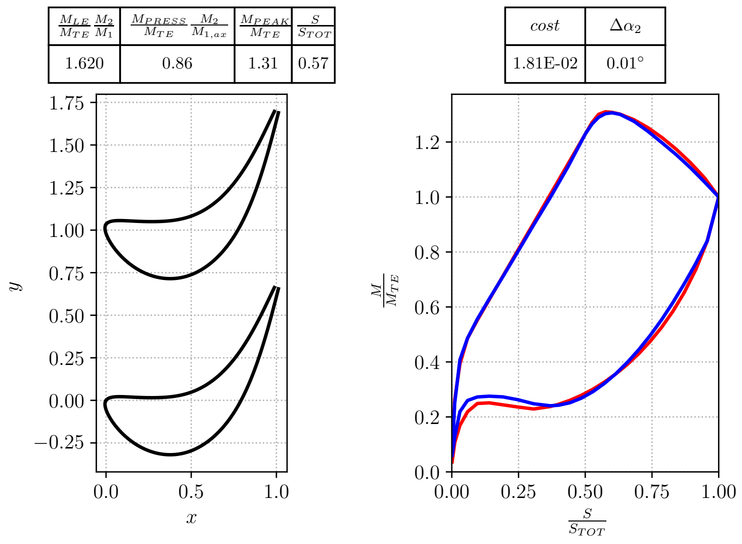
The tuning has been done using a trial and error approach over the RBF lenght scale and the regularization parameter used by the regressor. The tuning has given:

- $l = 0.1$
- $\alpha = 1 \cdot 10^{-3}$

Different parameters have been tested but they generated unacceptable errors at the boundaries of the domain of study due to the RBF properties. The gaussian functions do not show good accuracy at the boundaries because of the poor extrapolation properties of the method in that region of the domain.

6.3. Results and Accuracy

This section presents the extrapolation of the blade geometry and the load testing using the computed $\hat{f}$.



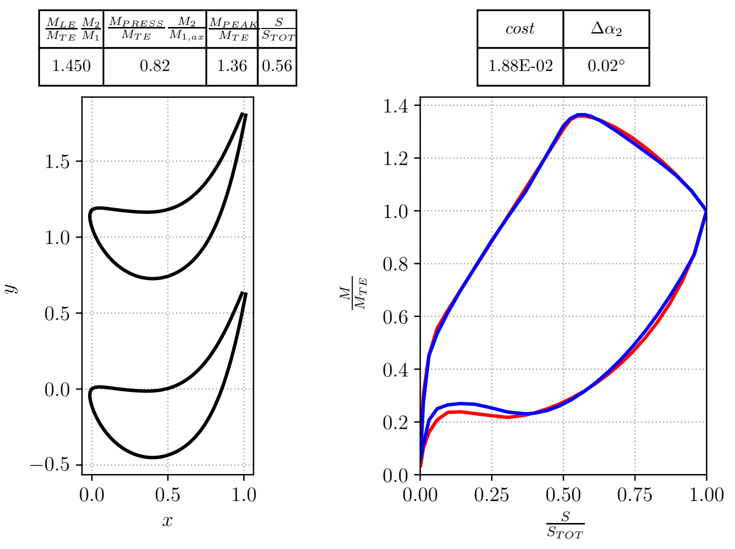
$\frac{M_{LE}}{M_{TE}}\ \frac{M_2}{M_1}$	$\frac{M_{PRESS}}{M_{TE}}\ \frac{M_2}{M_{1,ax}}$	$\frac{M_{PEAK}}{M_{TE}}$	$\frac{S}{S_{TOT}}$
1.620	0.86	1.31	0.57

cost	$\Delta\alpha_2$
1.81E-02	0.01°

Figure 6.3: Blade generated by $\hat{f}$ with $\alpha_1 = -34.04°$, $\alpha_2 = 70.15°$ and $M_2 = 0.57$.

In Figure 6.3, a blade computed from $\hat{f}$ is shown. This blade exhibits a 1.8% error in the loading distribution and an error of 0.01° in the exit angle. The aerodynamic duty and style of the blade differ from the entries in the $\mathcal{X}$ dataset.

Figure 6.4 shows a blade which is considered acceptable for the design study of a section.

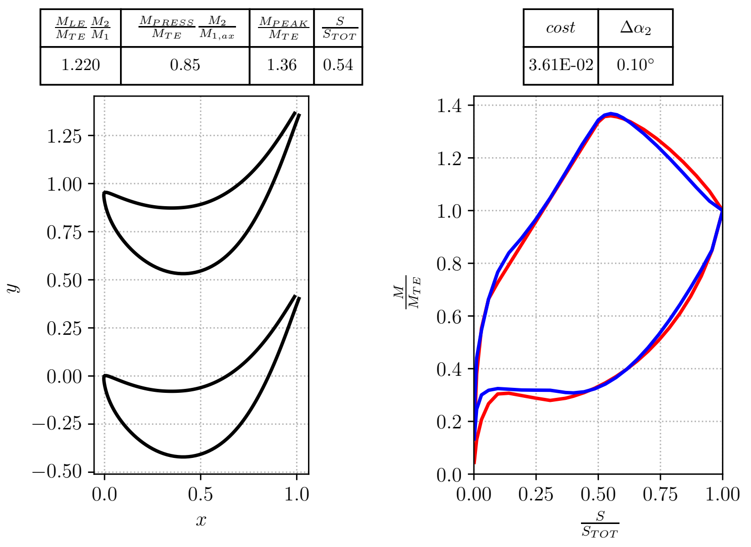


$\frac{M_{LE}}{M_{TE}}$ $\frac{M_2}{M_1}$	$\frac{M_{PRESS}}{M_{TE}}$ $\frac{M_2}{M_{1,ax}}$	$\frac{M_{PEAK}}{M_{TE}}$	$\frac{S}{S_{TOT}}$
1.450	0.82	1.36	0.56

cost	$\Delta\alpha_2$
1.88E-02	0.02°

Figure 6.4: Blade generated by $\hat{f}$ with $\alpha_1 = -44.67°$, $\alpha_2 = 72.11°$ and $M_2 = 0.41$.

Figure 6.5 illustrates the discrepancy between the generated blade and the target load due to the limitations of the machine learning model based on the study datasets $(\mathcal{X}, \mathcal{Y})$.



$\frac{M_{LE}}{M_{TE}}$ $\frac{M_2}{M_1}$	$\frac{M_{PRESS}}{M_{TE}}$ $\frac{M_2}{M_{1,ax}}$	$\frac{M_{PEAK}}{M_{TE}}$	$\frac{S}{S_{TOT}}$
1.220	0.85	1.36	0.54

cost	$\Delta\alpha_2$
3.61E-02	0.10°

Figure 6.5: Blade generated by $\hat{f}$ with $\alpha_1 = -49.16°$, $\alpha_2 = 65.16°$ and $M_2 = 0.67$.

Figure 6.6 demonstrates the same error pattern as seen in Figure 6.5. After an extensive analysis of the computed $\hat{f}$ function, the error throughout the entire interpolated domain of study remains under 4%, while the exit angle error $(\Delta\alpha_2)$ remains below 0.5°. These thresholds are considered acceptable for an initial analysis of a blade using a rapid design tool.

$\frac{M_{LE}}{M_{TE}}\frac{M_2}{M_1}$	$\frac{M_{PRESS}}{M_{TE}}\frac{M_2}{M_{1,ax}}$	$\frac{M_{PEAK}}{M_{TE}}$	$\frac{S}{S_{TOT}}$
1.310	0.89	1.25	0.59

cost	$\Delta\alpha_2$
3.47E-02	0.34°

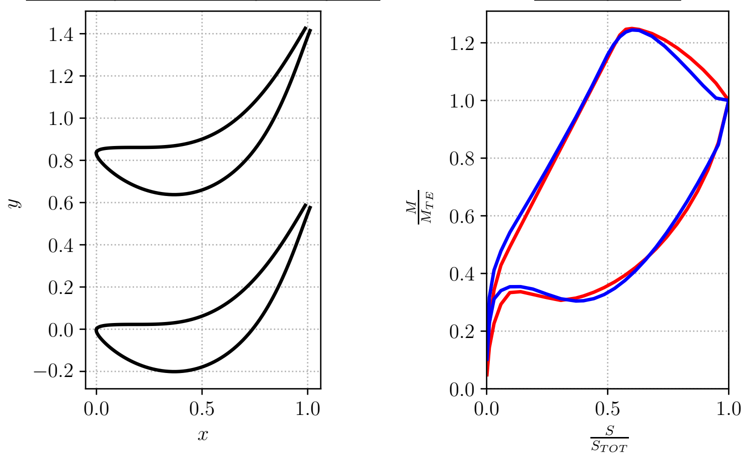


Figure 6.6: Blade generated by $\hat{f}$ with $\alpha_1 = -20.07°$, $\alpha_2 = 65.16°$ and $M_2 = 0.60$.

7 | Modal Reduction

The following sections aim to explain the dimensionality reduction of the problem comparing the **Kulfan** based representation of the domain to a more **physical** representation using a **modal decomposition** of data.

7.1. Problem Framing

The principal component analysis is one of the main methods for the dimensionality reduction of a problem [6]. Unlike the machine learning part, the study is conducted only over the $\mathcal{Y}$ dataset.

7.2. Principal Component Analysis

The method starts with a global analysis of the correlation between the different blade parameters:

- γ, χ_1 and χ_2 for the camberline
- A_{suct} for the suction side parametrization
- A_{press} for the pressure side parametrization
- *pitch*

From this correlation analysis, the method extracts the main correlation directions. These directions will then define a vector which can be seen as one of the eigenvectors which defines the $\mathcal{Y}$ dataset. After having computed the first direction, it is possible to compute the other *modal directions*.

Listing 7.1 shows the code used for the the principal component analysis of the $\mathcal{Y}$ dataset.

Listing 7.1: PCA decomposition

```python
from sklearn.decomposition import PCA

pca = PCA()
```

```
pcaVal = pca.fit_transform(X=X)
varianceDistr = np.cumsum(pca.explained_variance_ratio_)
```

Once computed all the *modal directions*, it is possible to understand their importance for the definition of the whole dataset. For each *modal direction* a variance index can be extracted. This index represents the importance of the mode in the whole database. The higher the variance the higher the dataset coverage of its respective *mode*.

A good dataset coverage is presented for a variance coverage above of 95%.

Figure 7.1 shows the variance distribution related to the dataset modes. With *just* three modes it is possible to cover more than 95% of the entire database.

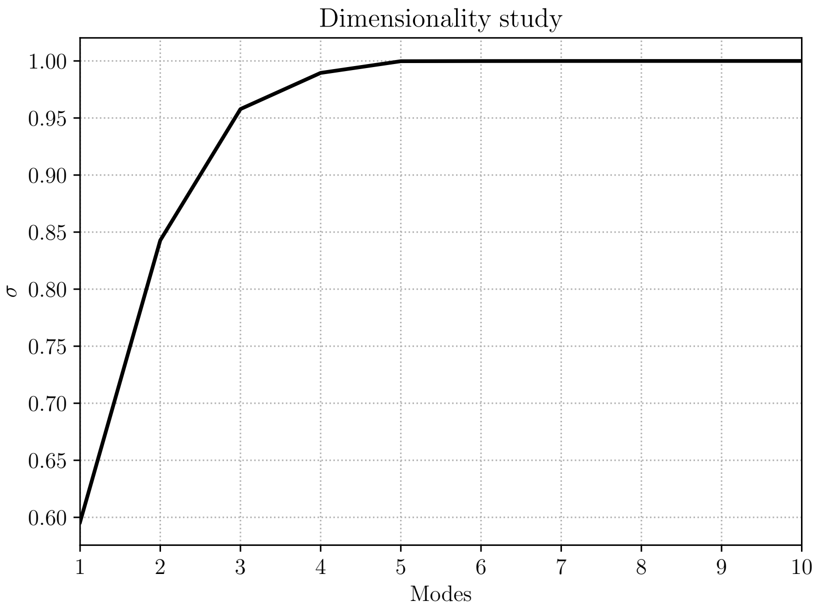


Figure 7.1: Principal Components of $\mathcal{Y}$ dataset.

7.2.1. Modal Analysis

After analyzing the primary modes within $\mathcal{Y}$, it becomes feasible to establish a connection between the load variations attributed to these modes. This procedure facilitates a further refinement of the study domain by solely considering a single dataset, namely $\mathcal{Y}$. This filtering process is not directly linked to the $\mathcal{X}$ dataset; instead, it can be better understood as a direct manifestation of the key geometric characteristics inherent in the $\mathcal{Y}$ dataset.

Figure 7.2 shows the first mode. This mode is represented letting varying its amplitude over a reference blade. Figure 7.2 represents also the load distribution varying the mode amplitude. From the plot it is possible understand that the first mode comprehends a lot of variation of the camberline. This variation will result in a change of the suction side load, especially on the peak Mach number and on the leading edge loading.

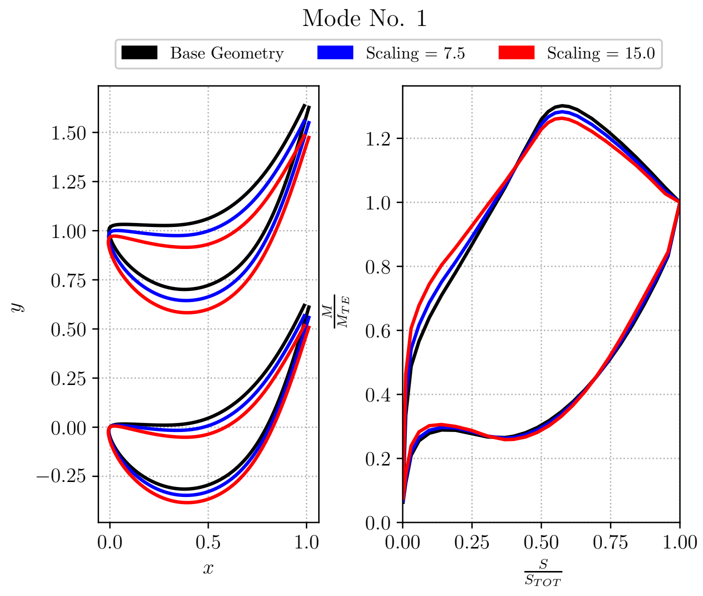


Figure 7.2: Mode No. 1 with the respective modal loading distribution.

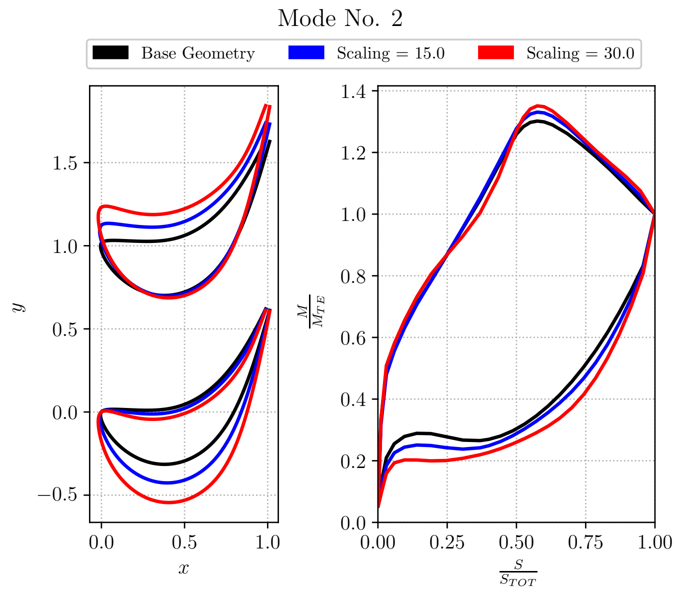


Figure 7.3: Mode No. 2 with the respective modal loading distribution.

The second mode, represented in Figure 7.3, involves heavily the blade thickness variation. The blade thickness variations do not change the loading at the leading edge on the suction side but it has an evident effect on the loading distribution at the leading edge on the pressure side. Even though the suction side leading edge is not affected by this mode, the peak Mach number on the suction side of the blade shows dependence on this mode; it can be seen as a mode which changes locally the load distribution over the suction side.

The first and second mode do not affect the position of the peak Mach number over the suction side. The third mode, represented in Figure 7.4, features the change in the position of the peak Mach number on the suction side. At the same time, appreciable changes are made on the pressure side Mach distribution.

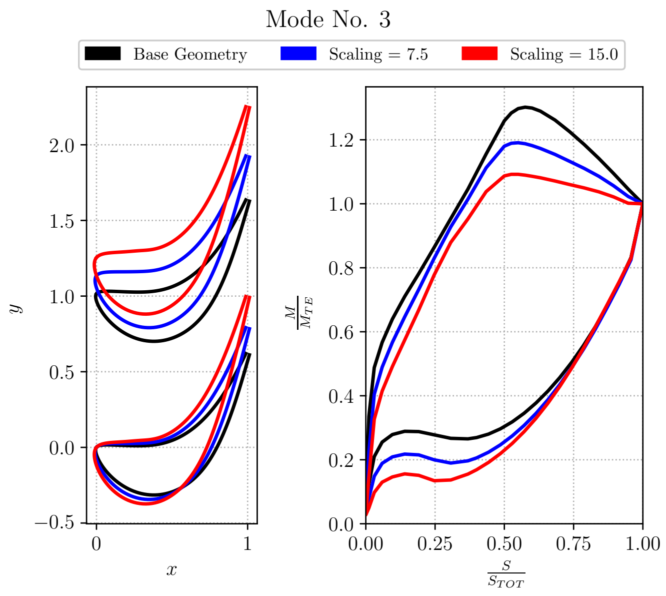


Figure 7.4: Mode No. 3 with the respective modal loading distribution.

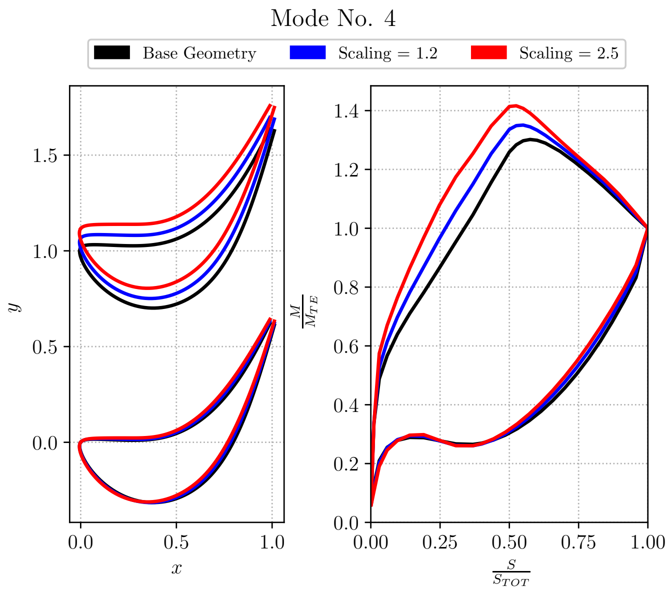


Figure 7.5: Mode No. 4 with the respective modal loading distribution.

Figure 7.5 represents the fourth dataset mode. Even though this mode can be seen as important, the dataset it spans is relatively small. This because the variance, σ, attributed to the fourth mode is smaller than the variance of its previous modal forms. This mode features best the changes over the suction side of the blade. It is clear that the leading edge loading and the peak load on the suction side of the blade are heavily dependent on this mode. Even though this behavior, the influence of this mode on the dataset is of lower importance compared to the first mode.

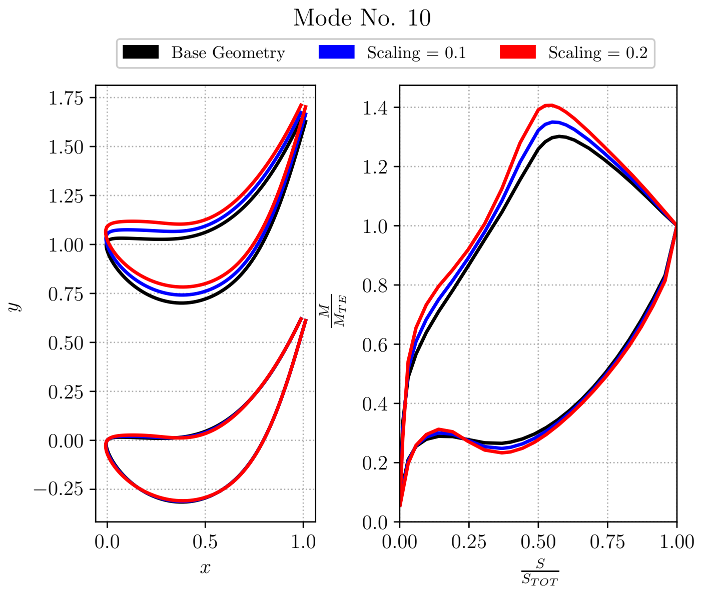


Figure 7.6: Mode No. 10 with the respective modal loading distribution.

The present work has shown the principal dataset modes - from the first mode to the third - but it does not show the whole modal spectrum of the $\mathcal{Y}$ dataset. The following modes are more related to important modal forms from which important results can be taken.

One of the most particular modal shapes are the 10th mode, the 30th mode and 42nd mode, the last mode. These modes underline important physical properties:

- the fact that the modal spectrum does not follow the same engineering strategy for the design of a section but it relies **only** on the dataset: mode 10th.
- there are modes which act on a very specific part of the loading distribution: mode 30th.
- there are modes which can be seen as noise and have low importance in the section design process: mode 42nd.

Figure 7.6 shows the 10th mode. This mode affects mainly the pitch degree of freedom. This mode changes the load distribution over the blade because a variation of the channel size. At the same time, the exit flow angle is heavily dependent on the solidity ($solidity = \frac{pitch}{chord}$) which is related to the pitch.

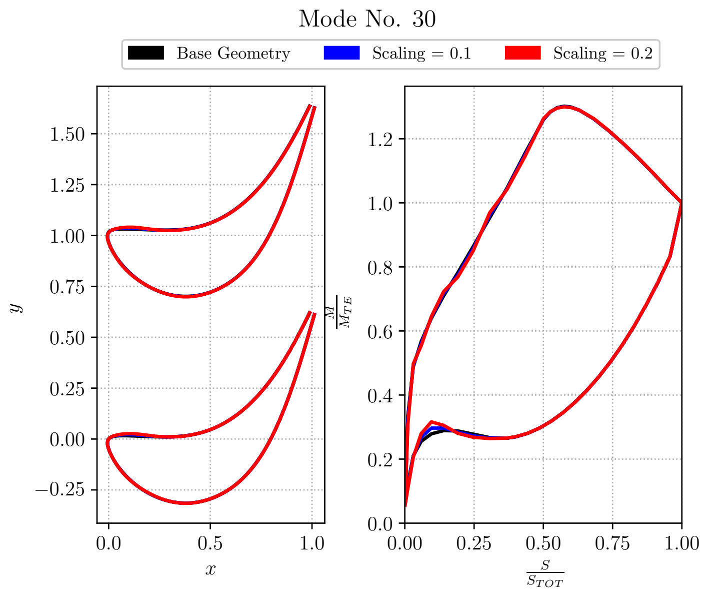


Figure 7.7: Mode No. 30 with the respective modal loading distribution.

Even though the 10th mode can be seen as an important mode because of the turbomachinery theory, it is of secondary importance inside the modal behavior of the system. This because the previous modes **span much more space** than the 10th.

Using Figure 7.7, it is possible to understand the influence, both over geometry and loading, of a low-variance mode. In this figure, the main geometrical variation is made over the pressure side at the leading edge of the blade. This change affects mostly the local Mach fraction distribution at the leading edge on the pressure side. With this mode, it is possible to highlight the fact that avoiding modes with low variance does not generate huge errors in the geometry representation.

Figure 7.8 shows the last modal shape. This shape can be seen as a wobbling noise around the blade and it has the lowest importance in the representation of the blade domain. As the 30th modal shape, the 42nd mode can be dropped off for a modal representation of a blade without loosing accuracy both in geometry and loading.

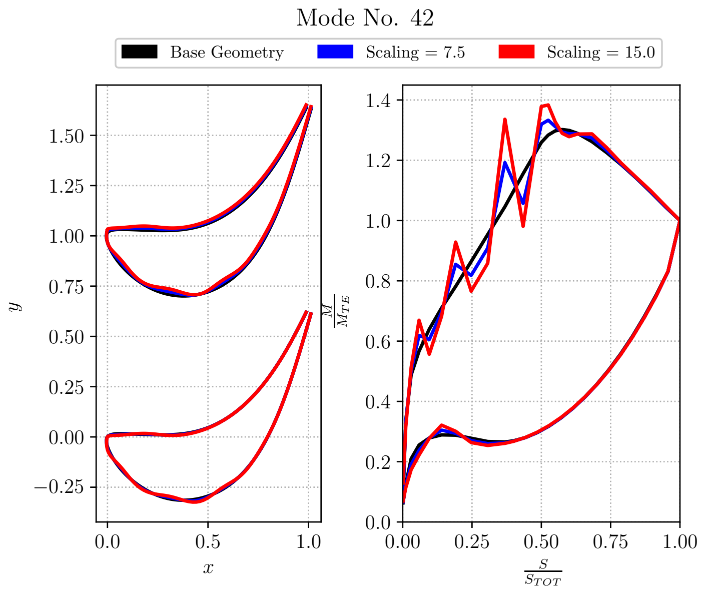


Figure 7.8: Mode No. 42 with the respective modal loading distribution.

8 | Conclusions

This chapter concludes the present work and highlights the main **results**, **limitations**, and possible **achievable improvements**.

8.1. Introduction and Reiteration

In this study, a groundbreaking design methodology has been developed for the field of turbomachinery blade design. This novel approach eliminates the need for conventional CFD simulations, significantly reducing design time while establishing insightful correlations between data and blade geometries.

8.2. Summary of Findings

The most crucial outcome of this research is the successful validation of the proposed methodology. It demonstrates applicability in blade design, offering speed and the ability to break down the design space into principal geometrical modes.

8.3. Implication and Significance

These identified modes present the opportunity to design blades more efficiently, employing a limited number of parameters compared to traditional blade parametrization methods. The establishment of a correlation between blade geometry and parametrized loading distribution further empowers the model. Simultaneously, the model defines the design's limitations in terms of loading, granting designers a comprehensive understanding of the design space and constraints in turbomachinery design.

8.4. Limitations

It's important to acknowledge that the model's boundaries are defined by the laws of physics. The sensitivity of database generation to loading parametrization highlights the close connection to physics. Moreover, the quality of the input database significantly impacts the quality of the resulting blades. Ensuring a high-quality database is essential for optimal results.

8.5. Future Directions

Future research directions could explore how blade geometry behaves with variations in the Reynolds number. Investigating blade geometry changes under diverse loading conditions also holds promise for advancing the methodology.

8.6. Closing Statements

The model's practicality in the industry is evident, serving as an initial design layer for blade generation. By seamlessly integrating physics and machine learning through data, it opens up a new avenue for designing turbomachinery systems. This efficient tool accelerates the design process, making it accessible to designers across the field. The profound significance lies in its ability to reshape the design landscape, creating a faster and more intuitive approach for designing turbomachinery systems.

Bibliography

[1] C. J. Clark. A step towards an intelligent aerodynamic design process. In *Turbo Expo: Power for Land, Sea, and Air*, volume 58578, page V02CT41A033. American Society of Mechanical Engineers, 2019.

[2] J. D. Denton. Some limitations of turbomachinery cfd. In *Turbo Expo: Power for Land, Sea, and Air*, volume 44021, pages 735–745, 2010.

[3] M. Drela. Mises implementation of modified abu-ghannam/shaw transition criterion. *Mises User's Guide, MIT*, 1995.

[4] M. Drela and H. Youngren. A user's guide to mises 2.53. *Massachusetts Institute of Technology, Cambridge, MA*, 1998.

[5] D. Duvenaud. The kernel cookbook: Advice on covariance functions. *URL https://www. cs. toronto. edu/duvenaud/cookbook*, 2014.

[6] A. Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media, 2022.

[7] J. Hensman, N. Fusi, and N. D. Lawrence. Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*, 2013.

[8] B. M. Kulfan. Universal parametric geometry representation method. *Journal of aircraft*, 45(1):142–158, 2008.

[9] B. M. Kulfan. Recent extensions and applications of the 'cst'universal parametric geometry representation method. *The Aeronautical Journal*, 114(1153):157–176, 2010.

[10] T. M. Mitchell. Artificial neural networks. *Machine learning*, 45(81):127, 1997.

[11] J. A. Nelder and R. Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.

[12] S. Shahpar. A review of automatic optimization applications in aerodynamic design

of turbomachinery components. In *CD Proceedings of the ERCOFTAC Conference in Design Optimization: Methods and Applications, Athens, Greece*, 2004.

[13] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 2020.

[14] H. Youngren and M. Drela. Viscous/inviscid method for preliminary design of transonic cascades. In *27th Joint Propulsion Conference*, page 2364, 1991.