POLITECNICO DI MILANO

Facoltà di Ingegneria Industriale

Corso di Laurea Specialistica in Ingegneria Meccanica

Advanced design of brake calipers

Adviser: Prof. Gianpiero Mastinu

Co-Adviser: Prof. Massimiliano Gobbi

Tesi di Laurea di:

Amir Reza Pishdad

Matricola n. 764787

Academic Year 2011–2012

To my parents, your unconditional love is the "optimal" support in my life

Acknowledgment

This dissertation would have not been possible without the guidance and deep insights of my adviser and mentor, Dr. Gianpiero Mastinu, who always steered the course of research in the right direction with his innovative ideas and effective words. If it was not for his truly inspiring course in "Advanced design" I would have never reached for such a fascinating thesis. My sincere thanks goes to Dr. Massimiliano Gobbi, my co-adviser, for his enthusiasm, and endless support during conducting the research project. It is impossible to imagine this project without his insightful comments and continuous encouragement. I can never forget unconditional and unfailing support of Ing. Federico Ballo. It has been a true honor to share your valuable viewpoints. I could never thank you enough for your constant help, back and forth emails, companionship in every step of the thesis and all you have done for this project. Celien, thanks for all the supporting Garfields, and helping me out with the proofing. Miguel, you always had a smart piece of advice for me, thanks pal. And Angie, it meant a lot to me that you never let me down and that you were there no matter what.

Summary

This research project is mainly concerned with assessment and optimization of brake caliper design. The caliper belongs to a racing car. It is desirable to discover new patterns in optimized caliper that would aid engineers in their future designs. This is done by creating a FE code written in MATLAB that can assess the displacement and stress of a simplified caliper design, resembling a frame, under a load configuration that is almost identical to the one of the actual caliper. The volume of the caliper and certain displacements have been selected as objective functions. These objective functions are optimized by altering the caliper shape to minimize the mass and maximize the stiffness of the caliper. Following this, the optimal solutions are compared to assist us realizing which design configuration would have the best behavior in terms of defined objective functions. In the end, an asymmetric design is gained that completely matches the results of topology optimization. This fresh design could be further developed and tested for industrial applications.

Contents

List of Figures

List of Tables

Nomenclature

ρ	Discrete Selection Field
ϵ_x	Longitudinal Normal Strain
ν	Poisson's ratio
Ω	Design Space
ϕ_y	Transverse Shear Factor
ϕ_z	Transverse Shear Factor
ρ	Density
σ'	*von Mises stress*
A	Cross Sectional Area
B.C.	Boundary Condition
E	Modulus of Elasticity
G	Shear Modulus
I_{yy}	Principal Moment of Inertia
I_{zz}	Principal Moment of Inertia
J	Polar Moment of Inertia
k	Element Stiffness
3D	Three Dimensional
$\mathbf{B}$	Strain-Displacement Matrix

D	Global Displacement Vector
K	Global Stiffness Matrix
N	Shape Function
R	Global Load Vector
CAE	Computer-aided engineering
d.o.f.	degree of freedom
FEM	Finite Element Method
m	Mass
SIMP	Solid Isotropic Material with Penalization
V	Volume

Chapter 1

Introduction

1.1 Introduction

In assessing the performance of any vehicle, two characteristics stand out: acceleration and deceleration. Meaning that, the car should reach a certain speed in minimum time and go back to zero in the shortest distance. For any car to halt in shortest distance, should possess a sophisticated braking system. This fact can be emphasized in high performance and racing cars.

The reason goes back to basic physics; the role of brake system in a vehicle is to transfer the kinetic energy to heat, meaning that the braking torque that is generated is converted to heat through friction, the kinetic energy in directly proportional to vehicle mass but as the square of vehicle speed [$\mathtt{k} = \frac{1}{2}\mathtt{m} \cdot \mathtt{v}^2$]. Therefore, for high performance cars that travel in high speeds this energy is much greater. Additionally, F1 cars they usually brake from speed around 300 $\mathtt{km/h}$ to 100 $\mathtt{km/h}$ in corners in less than 1.3 $\mathtt{s}$,with a deceleration over 4 $\mathtt{g}$. To do so, the brake system must generate a pressure gradient of over 4060 $\mathtt{psi/s}$, with the pressure in the system of usually over 725 $\mathtt{psi}$ [2]. That leads to fact that the high temperatures reached by the brake discs are enormous, sometimes exceeding 1000℃.

Simultaneously, it is clear that the mass should be as low as possible. The caliper, being part of wheel system, is no exception to this rule and might be even thought of more significance. Therefore, after selecting the material, reducing the mass means that the volume must be decreased. Nevertheless, the mass cannot be reduced unconditionally. Of course, in the case that entire design variables being equal reducing the mass will result in lower stiffness and toughness. Specifically the application of a braking caliper is

to insert pressure on the baking pads. Therefore, we can not afford to lose stiffness in the component.

The calculation of the braking torque on a wheel and the circumferential force acting on a single brake disc describes the forces involved during this type of braking. As an example, 1650 Nm are reached, corresponding to 28200 N on the disc friction surfaces.

As a result, the brake system must be quick, reactive, and easy to modulate, and offer a high level of repeatability. This allows the driver to anticipate the next stage and therefore to start braking always the same point.

Two main brake system now available are Drum Brakes (Figure 1.2) and Disk Brakes (Figure 1.1). Disk Brakes are generally prominent due to their fade resistance[1] and reduced tendency for "pull"[2]. Consequently, the only configuration that allows acceptable performance at the moment is a brake disc/caliper combination of carbon fiber.

1.2 Disk Brakes

Disc brakes are a combination of a brake caliper mounted on the stub and a brake disc mounted on the axle, designed to generate brake torque. Their high thermal resilience and their smooth, reproductive response means that they are used in all high performance vehicles. Disc brakes are axial brakes. The clamping forces of the brake caliper are applied to the brake pads in an axial direction by means of hydraulic cylinders. The brake pads then act on the flat friction surface of the disc (also known as the "rotor"). The pistons and the pads are located and retained in a housing that fits like a saddle over the outer circumference of the disc. In Figure 1.1 all parts of disc brake system is pointed out.

Some of the chief characteristics of disc brakes are as following:

- Acceptable release behavior
- Simple/self-adjustment

[1] The kinetic energy is converted to heat. This heat is transfer to disk/drum mainly. If the action of braking is repeated excessively, the accumulated thermal load would be more than cooling rate. This will result in a "fading" of brake system. Here the coefficient of friction of lining would reduce severely. In an extreme case a whole chunk of lining material could vaporize and create a gas shield between the lining material and friction surface.

[2] A vital characteristic of brake system is the ratio of tangential force at drum or disk to clamping force that they are able to generate. This ratio is a function of brake lining and geometry. Drum brakes have higher ratios. However, undesirable variation lining fiction coefficient can impact drum brake far more than disk brakes

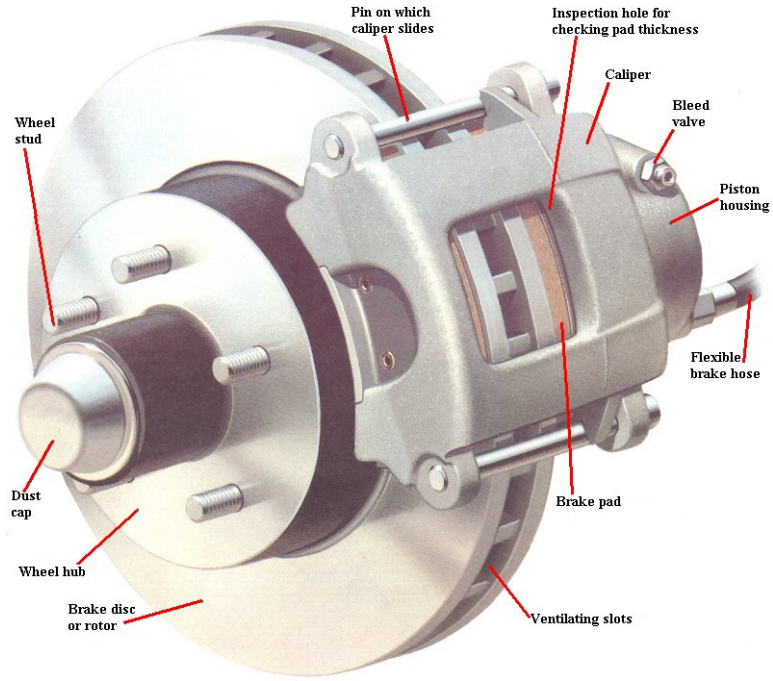


Figure 1.1: Different Parts of a Disc Brake

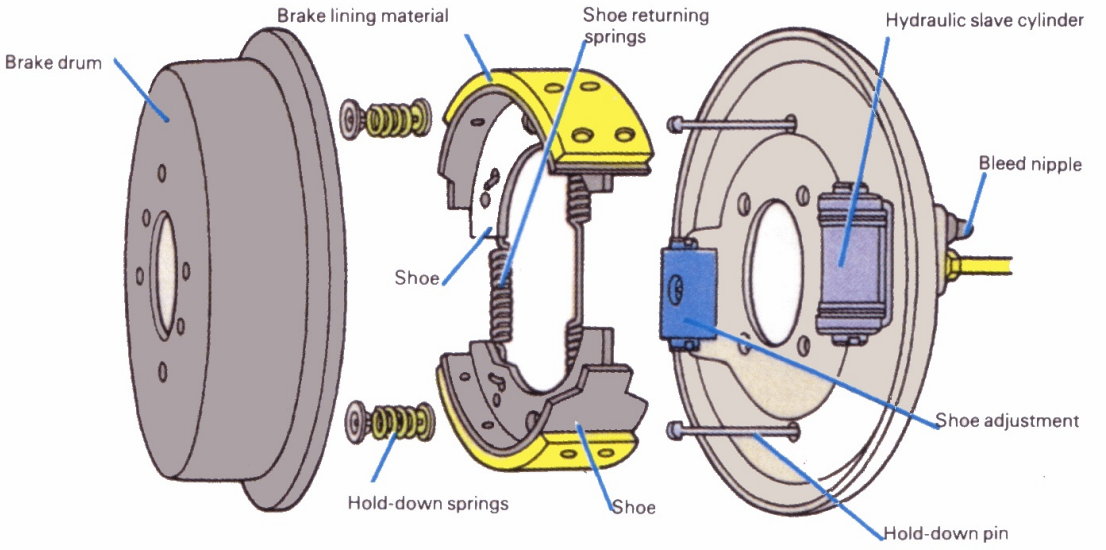


Figure 1.2: Drum Brake

- Uniform lining wear
- Uniform response
- Lower tendency to pull
- High fade resistance
- The wheel is free to rotate when braking is not performed; that results in saved power and fuel

Nevertheless, these braking systems have certain weaknesses, Such as *brake squeal*, *brake judder* and *brake dust*. *Brake squeal* is a loud high-pitched noise that occurs while braking action. It is mostly produced due to the vibration of brake components especially the pads and discs. This type of squeal should not negatively affect brake stopping performance.

Brake judder is the vibration transfer to the drive through chassis while braking [4]. The judder phenomenon can be classified into two distinct subgroups: hot and cold judder. Hot judder is usually produced as a result of longer, more moderate braking from high speed where the vehicle does not come to a complete stop [13]. Cold judder, on the other hand, is the result of uneven disc wear patterns or disc thickness variation (DTV). These variations in the disc surface are usually the result of extensive vehicle road usage.

Brake Dust occurs when braking force is applied; the act of abrasive friction between the brake pad and the rotor wears both the rotor and pad away. The "brake dust" that is seen deposited on wheels, calipers and other braking system components consists mostly of rotor material. Brake dust can damage the finish of most wheels if not washed off. Generally brake pad that aggressively abrades more rotor material away, such as metallic pads, will create more brake dust.

1.2.1 Brake Caliper

The core of disk brakes is the caliper body (Figure 1.1). This U-shaped casting that wraps around the rotor and is mostly made of cast iron or aluminum alloys. Especially for the race cars and high performance vehicles it should be light and stiff. The stiffness results in short pedal strokes. Furthermore, the performance of the car due to mass distribution highly depends on the weight of caliper which should be as low as possible. Material used is usually regulated by the champion rules; therefore, these two characteristics can be revised only be altering the design and optimizing it. In racing competitions the material

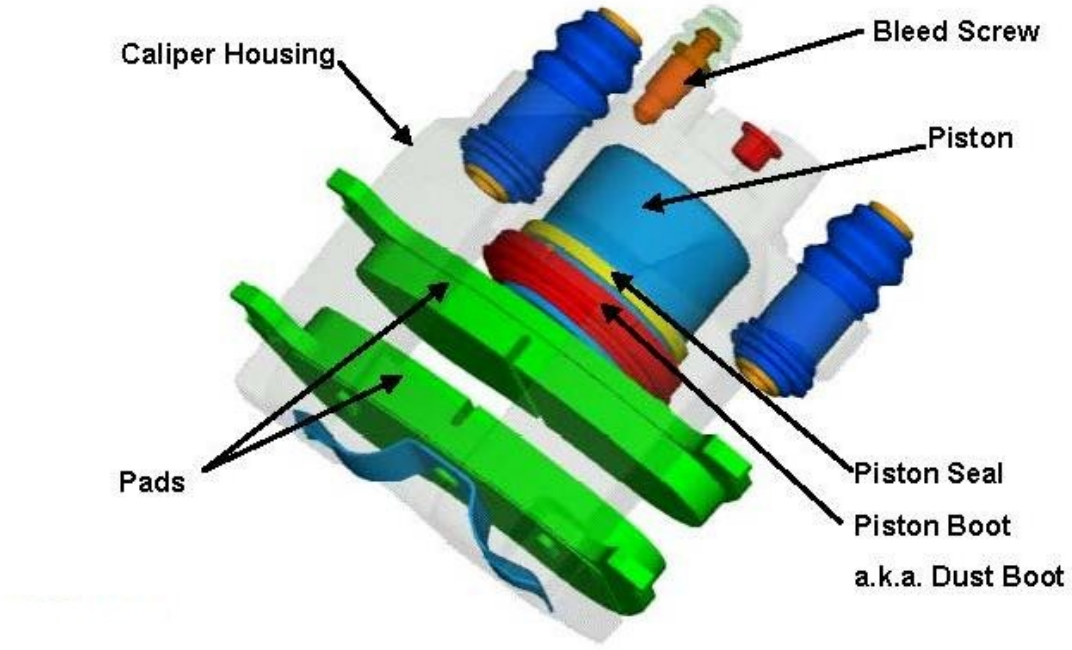


Figure 1.3: Basic Disk Brake Components- Reprinted SAE source

used for manufacturing these calipers mostly have modulus of elasticity less than 80 GPa. And indeed the engineer will opt for the material with the lowest density possible.

Different parts of a standard caliper are (Figure 1.3):

Caliper body Can be manufactured from one or two pieces that is bolted together. One piece calipers are usually forged and two piece ones are regularly machined from an aluminum billet. All types are surfaced treated to further increase the hardness. This extra hardness comes handy if there collision with wheel rim (during wheel changing) or if other object hit the caliper.

Pistons Made of aluminum and completed with a titanium insert that insulate them from heat coming from the pad [2].

Crossover pipe The brake fluid should reach both sides of a caliper. This is carried out through stainless steel pipes called crossover pipes (Figure 1.4).

Seal At high temperatures it is highly crucial to prevent leaking of brake fluid, this calls for an efficient seal. Yet the seal should allow an appropriate "roll-back" of piston; otherwise, the pads would retain their contact with the disk even when braking is

not applied. Furthermore, excessive "roll-back" will result in fact that pedal stroke could exceed the comfort level of the driver.

Bleed screws Whenever there is an air gap or bubbles in the fluid inside the caliper, bleed screws can be used to relieve them.

Disk caliper can be categorized as *fixed caliper*, *frame caliper* and *fist caliper*.

Fixed Calipers

Gets its name from the fact that it is rigidly mounted to the knuckle; no part of the caliper body moves when the brakes are applied. It has cylinders on both sides of the brake disc and frame (Figure 1.5). The caliper is composed of two halves that are usually bolted together. The pistons on both side of friction surface are connected to each other through a duct that is in the caliper (Figure 1.4). Advantages of these brakes are:

- Rigidity ensures low fluid displacement
- Strength and heat dissipating ability ideal for heavy duty
- Rigid mounting and low flexibility
- Firm and linear pedal feel

Some Drawbacks are:

- High weight of caliper
- High cost
- Complexity
- Higher chances of leaks

Fist Caliper and Frame Caliper

Only has cylinder on one side, as it mounts in such a way that it is free to move axially (Figure 1.1). Furthermore, it is usually located on the inboard side. This means only a small amount of space is required on the outboard side, so that the fist caliper also

Figure 1.4: Fixed caliper and the Connecting Duct

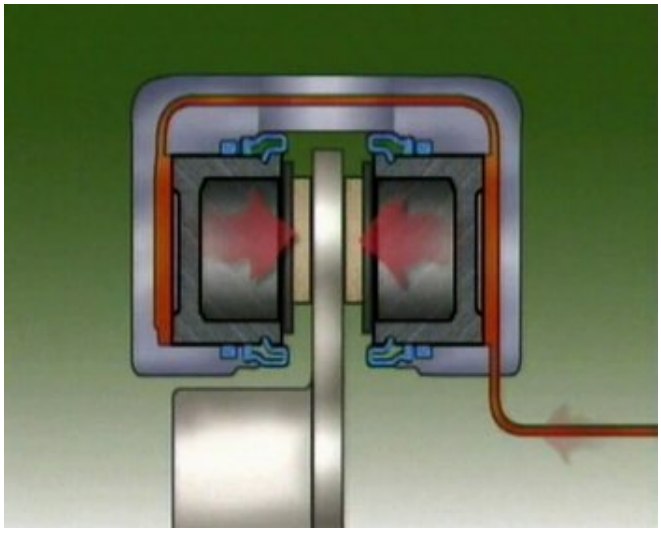


Figure 1.5: Fixed Caliper with Four Pistons

permits a negative scrub radius[3], its compact construction even allows front-wheel drive to be combine with negative scrub radius (vehicles with negative scrub radius require that the brake be mounted further into the rim, meaning that there is less design space) [2]. During braking the caliper piston moves out of its bore and forces the inner pad against the rotor while the pressure on the closed end of the bore moves the caliper body in the opposite direction forcing the outer pad against the disk at the same time. The caliper body moves every time the brakes are applied. *Frame Caliper brake* is also widely used for vehicles with negative scrub radius. It also has piston on one side. Chief benefits of these two types of brake are:

- Compact design dimension
- Low weight
- Simple construction
- Low Prices

The associated drawbacks are:

- Allows some flex freedom in caliper suspension that might deteriorate pedal feel
- Caliper suspension flex allows body to twist slightly during braking that can lead to taper lining wear
- Slow transfer of heat

1.3 Simulation of Brake System

There are numerous advantages in using a simulation technique before an actual manufacturing of a product. By testing the product in preliminary design stages, swapping time consuming test by simulated ones, reducing number of built prototypes and analyzing different aspects of piece behavior such as static, dynamic, vibration and so forth, one can reduce the costs and development time, as well as to increase the product quality. Also, the same rules stated above would apply to design and manufacturing of a brake system.

[3]The scrub radius is the distance in front view between the king pin axis - is the line between the upper and lower ball joints of the hub-and the center of the contact patch of the wheel, where both would theoretically touch the road.

Computer aided models can be applied to different physical areas of a brake system, including:

- Mechanical analysis
- Hydraulic analysis
- Pneumatic analysis
- Electromagnetic analysis
- Thermal analysis
- Investigation of control algorithms

For this project mechanical analysis is the main interest. For further analysis of the designed piece other aspects could be studied and developed. However, for now these are not the chief concerns.

In modeling of a component, various levels of model resolution can be distinguished:

Detailed model: All the possible details and effects are modeled and taken into consideration.

Optimized model: Discarding the less substantial factors and effects and less detailed modeled than "detailed model" can be obtained. Still this model contains more details than the "simple" one. Based on the substantial factors some design variables are defined and utilizing these variables the modeled can be optimized. The aimed features are the objective function that monitored to observe and control the optimization process based on them.

Simplified model: Here merely the most crucial effects and factors and considered. All those ones that would make the calculations and simulation cumbersome are excluded. This model is the baseline for all other models. It contains the higher flexibility in design freedom and is the most innovative step.

Transfer functions: Input and output variables are connected through simple functions or maps, which typically have no connection anymore to the physical effects. Such functions are often used in control algorithms.

The first step to design a brake system is to create a simplified system model. In the next step, the more detailed designs are considered. Following this, optimization should

be performed. The data of these steps are used for further design of high resolution model. For real time simulation, the models must be simplified or they are substituted by a combined development environment consisting of the parts and mathematical models [2]. One of the basic principles of brake system design is the calculation of the brake forces distribution.

As mentioned before, 3D CAE (Three dimensional computer-aided engineering) is used for both preliminary phase and for component development phase. For this thesis the tool used is finite element method (FEM). A brake system should provide the braking power, hold against stresses, and last but not least a good comfort and vibration behavior. Commonly in preliminary design the components have a virtual simple form (likewise in our case of design as will be demonstrate later simple beams are used to form the brake caliper). In theory all parts designed by a FEM code should exhibit the same behavior in real tests as in the simulation. An essential component of virtual description is the geometry defined using CAD system. Besides, the physical properties of the material in their operation temperature range should be considered. If problems still occurs during the validation of components, then the use of computer-aided method will contribute to a better understanding of the whole process and to find an efficient solution for the problem at hand [2].

The type of methods that are used can be divided into procedures that are generally used in machine and vehicle construction and special methods that are tailored for brake systems. The first includes strength and stiffness calculation of the components. The second type of method concerns especially the analysis of vibration in the frequency range from 0 to 15000 Hz, which includes low frequency effects of comfort as well as high frequency noises. With high-frequency commonly the limits of computer aided methods are reached. For the analysis of the squeal noise (1000 to 15000 Hz), the complex intrinsic value calculation has been established. Coulombs‘s law of friction is a component of the equation system that must be solved to understand usable vibrations arising in the system [2]. One of the fundamental design calculations is the static analysis of the brake caliper. The stiffness of this component contributes to the overall stiffness of the brake system and, consequently, influences the pedal feel.

1.4 Research Goals

As depicted above, caliper plays a vital role in brake performance of any vehicle. Any improvement of the caliper could have substantial impact on vehicles performance,

even more so in racing cars. Therefore, this research project will attempt to optimize a high performance brake caliper for a racing car. This is done by altering the general shape of the caliper and questioning various details to obtain lower mass and higher stiffness. The stiffness is assessed through comparison of displacement of the actual caliper and the optimized one due to identical forces. Although having a lower mass is one the chief goals of the project, due to the fact that this project is mostly concerned with preliminary design of a caliper and not a detailed one, exact comparison between the real and optimized model could not be carried out. Nonetheless, the final model should not be heavier than the actual caliper by a great degree, so that after creating the detailed model and eliminating the redundant parts, a lighter caliper could be gained. The specific purposes of this research are as follows:

- Propose an analytical model that can predict the displacement of any frame under a certain force field.
- Model a general shape of the caliper by the analytical model.
- Strive to the make model as simple as possible yet resembling the actual caliper.
- Alter the specified design variables and their ranges to gain the desired objective functions.
- Find the optimal solutions that minimize mass and compliance.
- Study these optimal solutions and search for patterns.
- Generalize the trends seen in the optimized cases.
- Utilize those patterns to create new a caliper design.
- Perform a detailed model study and topology optimization and compare the results with the preliminary optimization results.
- Assess the design to reassure that the general goals are indeed yielded.

Chapter 2

Development of the Model

2.1 Original Caliper

The caliper that is going to be assessed belongs to a high performance racing vehicle. These calipers are highly sophisticated and already well developed. This fact can be noticed in Figure 2.1. These calipers are stiffer than even normal high performance vehicles and are the leading edge of their technology.

Figure 2.1: Sophisticated Racing Brake Caliper

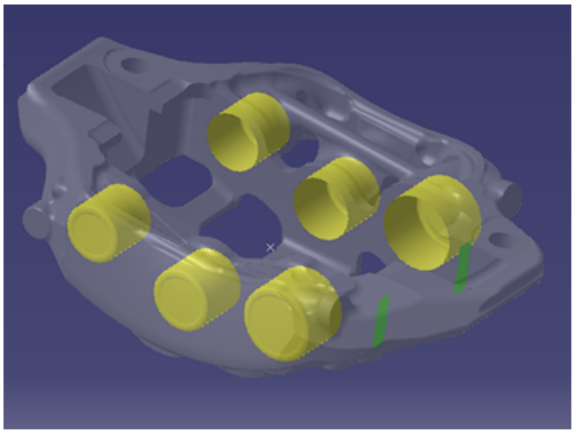


Figure 2.2: The Caliper Original Design

2.1.1 Shape and Configuration

Figure 2.2 depicts a view of the caliper. It is a *Fixed Caliper* with six cylinders. A regular caliper contains two to six pistons and those ones with six pistons are considered heavy duty caliper brakes. The counter cylinders on each side are identical. On the other hand, on each side the three cylinders have different diameters. Indeed, the different diameters are not accidental. During braking it is obvious that the disk is rotating; hence, when the pads are squeezed on the disk, they are pushed forward perpendicular to the axis of rotor. The pads are stopped by the caliper, inserting a force on the caliper. To secure the balance of pads and hinder extra rotational movement, a bigger force is required at locations closer to where this lateral force is being inserted. In Figure 2.2 these two lateral forces are marked by green straps.

For optimization of the caliper two main domains can be defined; the "frozen" or "un-designed" domain and "optimization" or "design" domain. The "un-design" domain refers to those ones that are fixed and not being changed. Here we will go through them extensively.

Figure 2.3 and 2.4 exhibits the entire "design domain". The cross section of the caliper can be altered in all points of the caliper. The front and back of the caliper (Figure 2.7 on page 16) can also be revised to enhance the response of the caliper to pressure and forces.

On the other hand, the position and number of pistons are fixed and cannot be

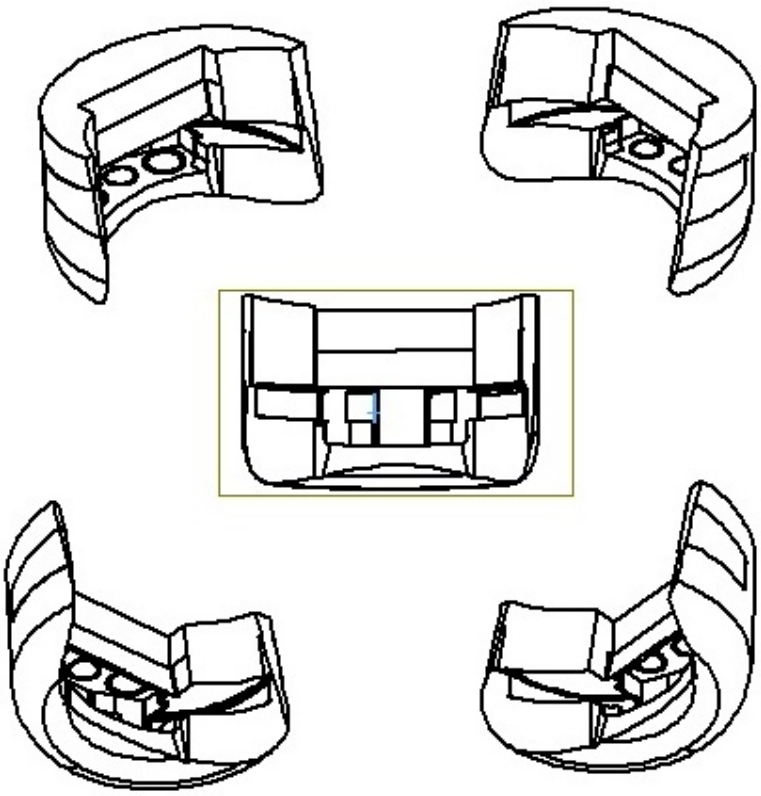


Figure 2.3: Caliper Design Domain Different Angles

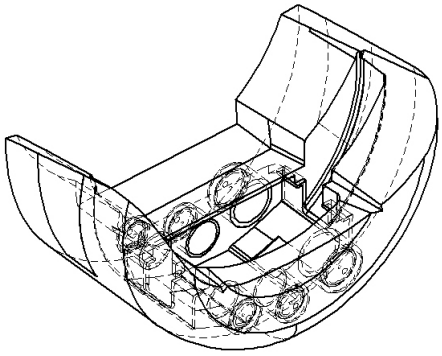


Figure 2.4: Caliper Design Domain- Detailed View

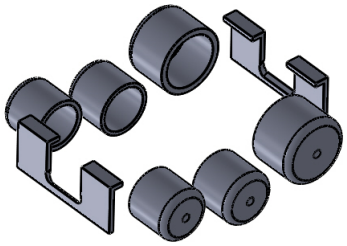


Figure 2.5: The Six Cylinders of the Caliper-"Frozen Domain"

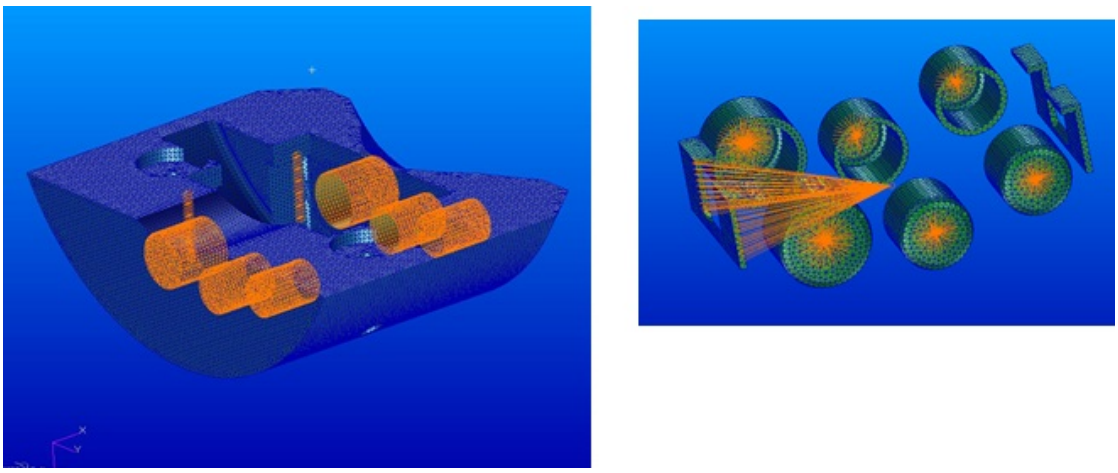


Figure 2.6: Forces Locations in the Original Caliper

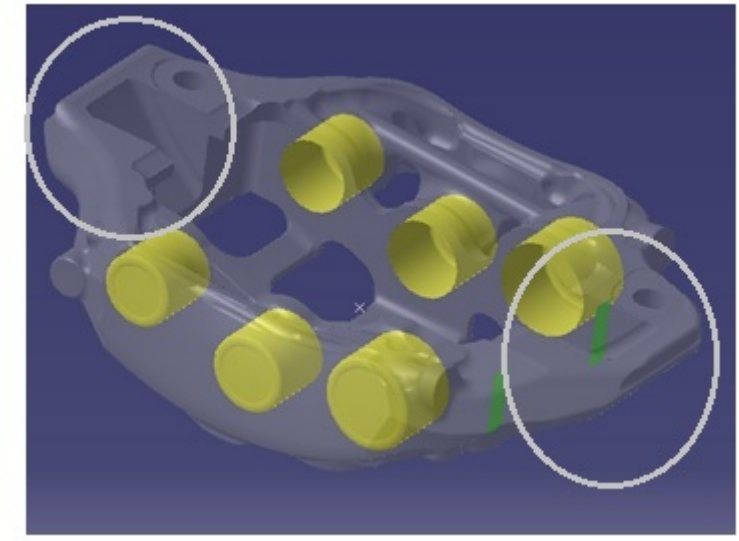


Figure 2.7: The Part of Caliper that Can Be Modified

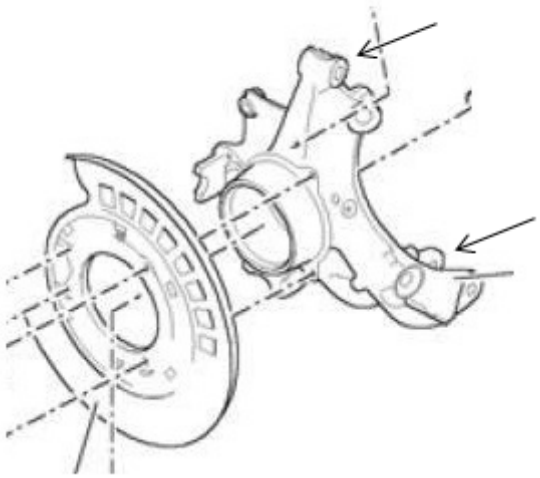


Figure 2.8: The Hub Carrier-The connection locations are marked

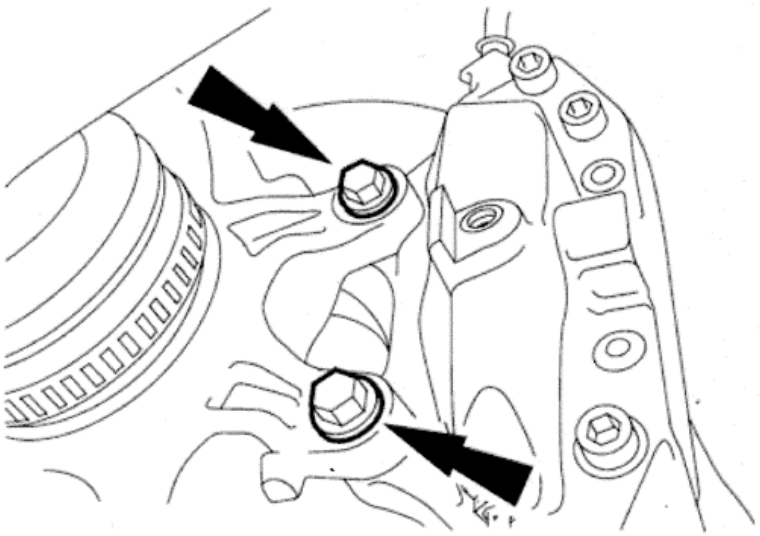


Figure 2.9: Caliper Fixed Via Two Screws

changed; in addition, the place where the lateral forces of the pads are exerted on the caliper is also fixed. Therefore, these create the "frozen domain" (Figure 2.5 on page 15).

The caliper is connected to hub carrier (Figure 2.8) via two M10 screws, as regulations describes: "11.2.2 No more than two attachments may be used to secure each brake caliper to the car." Also, between two parts there are two titanium rings. Figure 2.9 depicts how the caliper is connected to hub carrier. In modeling the caliper the two points of connection with the hub carrier acts as boundary conditions with fixed displacement; nonetheless, the assembly introduces a spring effect that should be taken into consideration.

2.1.2 Caliper Material

The weight of caliper is about 1.4 kg. It is manufactured from 7075 aliminium alloy. Aluminium alloy 7075 is an aluminium alloy, with zinc as the primary alloying element. It is strong, with a strength comparable to many steels, and has good fatigue strength and average machinability, but has less resistance to corrosion than many other Al alloys. Its relatively high cost limits its use to applications where cheaper alloys are not suitable. 7075 aluminum alloy's composition roughly includes 5.6 % to 6.1 % zinc, 2.1 % to 2.5 % magnesium, 1.2 % to 1.6 % copper, and less than half a percent of silicon, iron, manganese, titanium, chromium, and other metals. It is produced in many tempers, some of which are 7075-O, 7075-T6, 7075-T651. Aluminium 7075 has a density of 2.810 gr/cm^3. The mechanical properties of 7075 depend greatly on the temper of the material. Un-heattreated 7075 (7075-O temper) has maximum tensile strength no more than 276 MPa, and maximum yield strength no more than 145 MPa. The material has an elongation (stretch before ultimate failure) of 9 % to 10 %. T6 temper 7075 has an ultimate tensile strength of 510 MPa to 538 MPa and yield strength of at least 434 MPa to 476 MPa. It has a failure elongation of 5 % to 8 %. T651 temper 7075 has an ultimate tensile strength of at least 462 MPa to 538 MPa and yield strength of 372 MPa to 462 MPa. It has a failure elongation of 3 % to 9 % [1]. 7000 series alloys such as 7075 are often used in transport applications, including marine, automotive and aviation, due to their high strength-to-density ratio. Their strength and light weight is also desirable in other fields [8].

The material of caliper is fixed and is dictated by regulation. Specifically the RACING COMPETITION TECHNICAL REGULATIONS" mentions:"11.2.1 All brake calipers must be made from aluminium materials with a modulus of elasticity no greater than

80 MPa". Consequently, in the present research project no assessment of the material is going to be made.

2.2 Preliminary Model

For solving any engineering problem, the engineer should always neglect the unnecessary details. Optimizing this caliper is no exception either; it would be very cumbersome to optimize the caliper including all the details right off the bat. Conversely, if the caliper is stripped down to its base and take the simplest shape possible, the process would be more manageable. Furthermore, The FE models of the brake disc and axle components utilized are usually very complex, the solution of equation systems with more than 300.000 degrees of freedom demands especially efficient mathematical process [2]. Thus, even with limited computation resources it would be impossible to perform the optimization.

Furthermore, to have real sense and control over the solution and process, it is not possible just to leave the model to commercial software for optimization, without knowing which would be the right direction to steer the software. Hence, with the simplified geometry, an effective code should be conducted so that comprehensive knowledge of every stage of the optimization would never be lost.

2.2.1 General Form of the Model

The caliper is simplified to a frame with beams as its elements; the frame should be as simple as possible yet resemble the true form of the caliper. Figure 2.10 on the following page and figure 2.11 on page 21 demonstrate this simple model. Each piston is modeled by two elements (Figure 2.12 on page 21) with the pressure acting on the central node that these two elements have in common (Figure 2.6 on page 16). Plus, the two lateral forces due to pad movements are apparent in the figure. The moment due to pressure of pistons create a minor error and can be neglected. Three dimensional, 3D, beam element has been chosen, because it is simple, the response of the members can be anticipated quite exactly, and it can account for axial force, transverse shear force in each of two directions, bending about each principal axis of the cross section, and torque about the longitudinal axis of the member.

The same type of material as the original caliper has been utilized (AL 7075). The mechanical properties density (ρ), modulus of elasticity (E),and Poisson's ratio (ν) has

been introduced to the model.

In the beginning, the spring effect of the hub carrier connection is neglected, the caliper is assumed to be completely constrained to ground. In addition, connecting elements that relate the front to back and left to right side of caliper has not been introduced to the model. Nevertheless, all these elements have been addressed in the further stages of project.

A MATLAB code is generated that is able to thoroughly assess, the various frame shapes with beam elements, from stress and displacement point of view. The code should be able to calculate the stress at each node of the model. This code will be used as a basis for the optimization stage, where the caliper is changed continuously and the corresponding displacements and volume are computed to determine the optimized model.

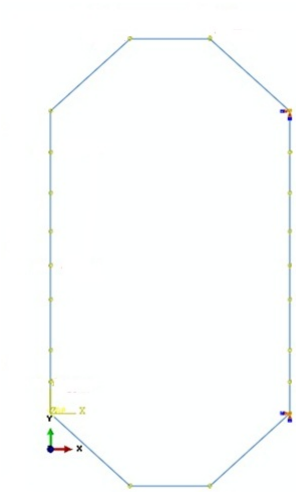


Figure 2.10: Simplified Model Top View

2.3 Theory Behind the Model

After knowing how the general model looks like, how it discretizes the structure, how the structure is loaded, the material data and what B.C. is supporting the structure, the "pre-process stage" has been covered. It is now time to consider the stiffness of the element and perform the "solution stage". As Cook [3] perfectly explains, to perform calculations, the software must

- generate the stiffness matrix $\mathbf{k}$ of each element,

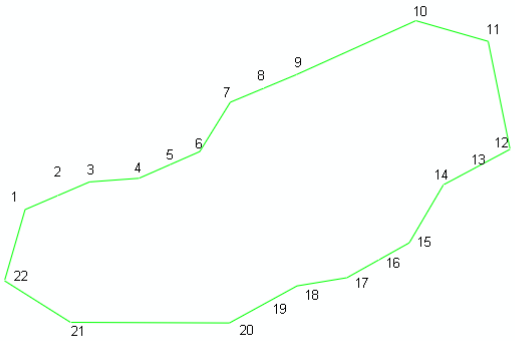


Figure 2.11: Simplified Model Side View

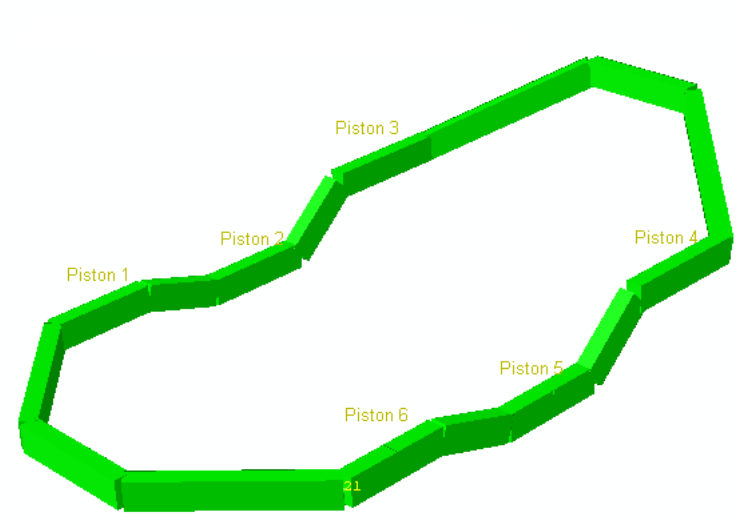


Figure 2.12: Simplified Model Pistons Positions

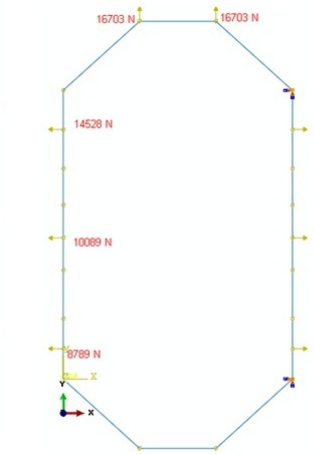


Figure 2.13: Simplified Model Force Locations

- connect elements together, that is, assemble the element **k** matrices to obtain the structure or "global" matrix **K**,
- assemble loads into a global load vector **R**,
- impose boundary conditions, and
- solve the global equations $\mathbf{KD} = \mathbf{R}$ for the vector **D** of unknowns. Indeed **D** contains displacement components of nodes.

D is also used for stress analysis to compute the strains and stresses. Obviously, next stage would be "post-processing", where **D** would be analyzed, plotted and printed.

2.3.1 Generating Stiffness Matrix

Producing the corresponding stiffness matrix for each element is the cornerstone of FE analysis. For most elements a general formula can be used.

$$\mathbf{k} = \int \mathbf{B^T E B} \mathrm{dV} \tag{2.3.1}$$

Where **B** is the *strain-displacement matrix*, **E** is the *material property matrix* (it may also be called *the constitutive matrix*), and dV is an increment of the element volume V. Equation 2.3.1 can be derived by realizing that the work done by nodal loads that

are applied to create nodal displacements, and that this work is stored in the element as elastic strain energy.

The *strain-displacement* can be obtained from the strain value of the element. For the sake of simplicity, here just a two dimensional element is illustrated, the solution can be generalized to a three dimensional one.

2D Beam Elements

First let us just consider axial displacement of the beam element. To obtain **B** the axial displacement, u, of an arbitrary point on the beam should be written as linear interpolation between its nodal values. The term that relates the displacement along the beam with the nodal one is called *shape function* **N**. As an example, if the beam's length is L and the position along the beam is demonstrated by x then u can be seen as

$$u = \begin{bmatrix} \frac{L-x}{L} & \frac{x}{L} \end{bmatrix} \begin{Bmatrix} u1 \\ u2 \end{Bmatrix} \tag{2.3.2}$$

$$u = \mathbf{N}\mathbf{d} \tag{2.3.3}$$

Indeed, these two equations are identical.

$$\mathbf{N} = \begin{bmatrix} \frac{L-x}{L} & \frac{x}{L} \end{bmatrix}$$

And,

$$\mathbf{d} = \begin{Bmatrix} u1 \\ u2 \end{Bmatrix}$$

From Mechanics of Material,

$$\epsilon_x = \frac{du}{dx}$$

consequently,

$$\epsilon_x = \left[\frac{d}{dx}\mathbf{N}\right]\mathbf{d} = \mathbf{B}\mathbf{d} \qquad \text{Thus} \qquad B = \begin{bmatrix} -\frac{1}{L} & \frac{1}{L} \end{bmatrix} \tag{2.3.4}$$

Again for the sake of simplicity, we should consider the in-plane bending and transverse shear force, and neglect other degrees of freedom. Starting from the general Equation 2.3.1 and adopting to a beam element

$$\mathbf{k} = \int_0^L \mathbf{B}^{\mathbf{T}} E I \mathbf{B} dx \tag{2.3.5}$$

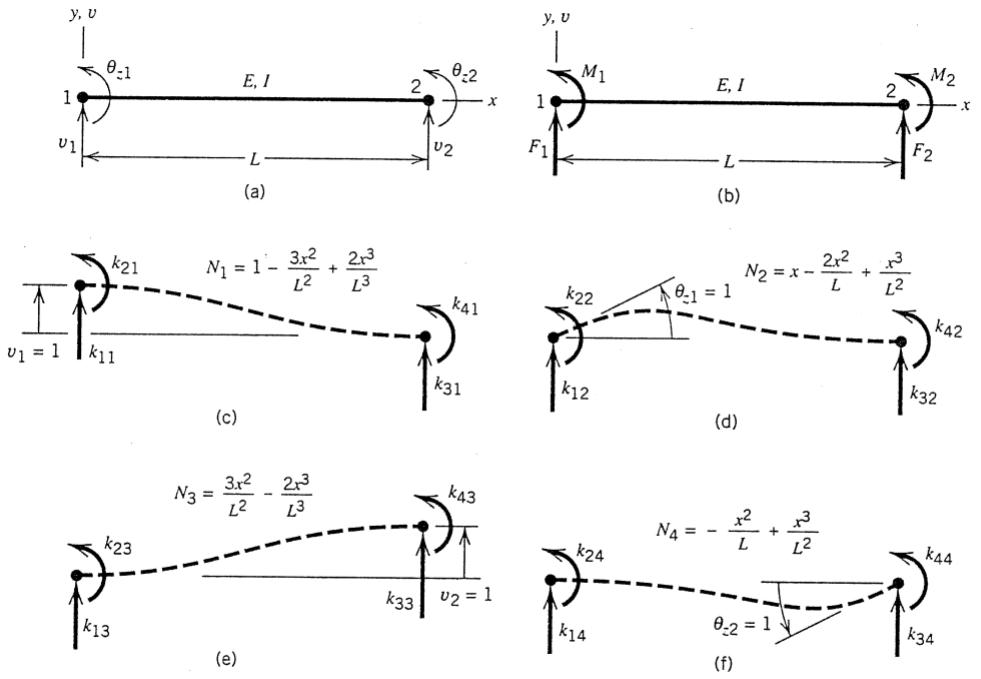


Figure 2.14: (**a**) Simple plane beam element and its nodal d.o.f. (**b**) Nodal load associated with the d.o.f. (**c-f**) Deflected shapes and shape functions associated with activation of each d.o.f. in turn. Nodal loads are labeled according to their position in **K**. (Reprinted from [3])

This time **Bd** corresponds to curvature d^2v/dx^2 of the beam element. Obtained from strain energy of the element under nodal displacement - $\mathbf{d^T kd}/2$ - which now depends on curvature. The curvature of any beam element in plane can be exhibited by:

$$v = \beta_1 + \beta_2 x + \beta_3 x^2 + \beta_4 x^3 \tag{2.3.6}$$

Using boundary conditions the coefficients β_i can be achieved. After obtaining these coefficients in terms of nodal displacements the equation can be rewritten as,

$$v = [N_1 N_2 N_3 N_4] \begin{Bmatrix} v_1 \\ \theta_{z1} \\ v_2 \\ \theta_{z2} \end{Bmatrix} = \mathbf{Nd} \tag{2.3.7}$$

Figure 2.14 depicts the value of **N** for a simple beam element. The curvature of a beam

$$\mathbf{k} = \begin{bmatrix} AE/L & 0 & 0 & -AE/L & 0 & 0 \\ 0 & 12EI/L^3 & 6EI/L^2 & 0 & -12EI/L^3 & 6EI/L^2 \\ 0 & 6EI/L^2 & 4EI/L & 0 & -6EI/L^2 & 2EI/L \\ -AE/L & 0 & 0 & AE/L & 0 & 0 \\ 0 & -12EI/L^3 & -6EI/L^2 & 0 & 12EI/L^3 & -6EI/L^2 \\ 0 & 6EI/L^2 & 2EI/L & 0 & -6EI/L^2 & 4EI/L \end{bmatrix} \begin{matrix} u_1 \\ v_1 \\ \theta_{z1} \\ u_2 \\ v_2 \\ \theta_{z2} \end{matrix}$$

Figure 2.15: Stiffness Matrix of 2D Beam Element

element is

$$\frac{d^2v}{dx^2} = \left[\frac{d^2}{dx^2}\mathbf{N}\right]\mathbf{d} = \mathbf{Bd} \tag{2.3.8}$$

After including the axial d.o.f. as well, and using the Equation 2.3.1 one can come to the stiffness matrix (Figure 2.15), remembering that this element type can resist axial stretching, traverse shear force, and bending in one plane.

3D Beam Elements

Now the most general case is introduced. To deal with this kind of elements a "global" coordinate axes XYZ is defined. On the other hand, the element will have its own "local" axis, within which the longitudinal axis of element x coincide with the one of the "local" axis. The x is defined by the 2 node coordinates. Still for defining the coordinate system, we need another node. The third node is either an extra node or another node of the structure, whose coordinate defines the orientation of xy. Nodes 1 and 2, each one has six degree of freedom, which are three displacements and three rotations. In total, consequently, there is 12 degree of freedom defined for each element. Nodal coordinates, elastic modulus E, shear modulus G, cross-sectional area A, principal moment of inertia I_{yy} and I_{zz} of A, torsional constant J, and transverse shear factors ϕ_y and ϕ_z are required for generating the stiffness matrix. In our case the cross sections are rectangular therefore J is not the polar moment of the cross-sectional area A. For a rectangular cross section

$$J = a.b^3\left[\frac{16}{3} - 3.36\frac{b}{a}\left(1 - \frac{b^4}{12\cdot a^4}\right)\right] \tag{2.3.9}$$

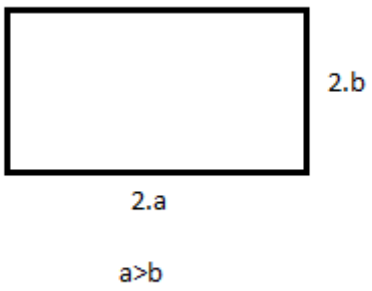


Figure 2.16: Dimensions of J

Figure 2.16 depicts the values of "a" and "b".

For defining the stiffness of a 3D beam, one must determine the slenderness of the beam. Depending if the beam is considered slender or not, two types of beams with different stiffness matrices exits. The first type if called Euler-Bernoulli beams, these elements do not allow for transverse shear deformation; plane sections initially normal to the beam's axis remain plane (if there is no warping) and normal to the beam axis. They should be used only to model slender beams: the beam's cross-sectional dimensions should be small compared to typical distances along its axis (such as the distance between support points or the wavelength of the highest mode that participates in a dynamic response). For beams made of uniform material, typical dimensions in the cross-section should be less than about 1/15 of typical axial distances for transverse shear flexibility to be negligible. (The ratio of cross-section dimension to typical axial distance is called the slenderness ratio). Consequently, for thin-walled structures the following can be assumed to hold:

> Straight line, normal to the mid surface (mid axis in the case of beams), remains **straight and normal** to the deformed mid surface (axis) throughout deformation.

For beams, this assumption is associated with the name of Bernoulli. The restriction of cross sectional lines to remain normal has been repealed by Timoshenko. The assumption then reads:

> Straight line, normal to the mid surface (mid axis in the case of beams), remains **straight** throughout deformation.

This relaxation allows the approximate consideration of transverse shear deformations. Thus, slightly thicker structures, but also e.g. space frames can be analyzed more accurately. They can be used for thick ("stout") as well as slender beams. For beams made from uniform material, shear flexible beam theory can provide useful results for cross-sectional dimensions up to 1/8 of typical axial distances or the wavelength of the highest natural mode that contributes significantly to the response. Beyond this ratio the approximations that allow the member's behavior to be described solely as a function of axial position no longer provide adequate accuracy. The Timoshenko beams can be subjected to large axial strains. The axial strains due to torsion are assumed to be small. In combined axial-torsion loading, torsional shear strains are calculated accurately only when the axial strain is not large.

In the current project both stiffness matrices are generated and the results are studied. However, as it will be observed in the following chapters, the modeled beams are quite short and can be considered stout; therefore, the assumption of using Bernoulli beam will not warranty realistic results. Furthermore, Timoshenko can account for both thick and slender beams. Thus, after verifying stages that both beam types were used, we opt for using Timoshenko beam for the final computations. Timoshenko stiffness matrix is as follows

$$
\begin{aligned}
\alpha^1 &= \frac{6}{5} \\
\phi_z &= \frac{12EI_z}{\frac{GA}{\alpha L^2}} \\
\phi_y &= \frac{12EI_y}{\frac{GA}{\alpha L^2}} \\
k_1 &= \frac{EA}{L} \\
k_2 &= \frac{12EI_z}{L^3(1+\phi_z)} \\
k_3 &= \frac{6EI_z}{L^2(1+\phi_z)} \\
k_4 &= \frac{4EI_z}{L(1+\phi_z)} + \frac{\phi_z EI_z}{L(1+\phi_z)}
\end{aligned}
$$

$$
\begin{aligned}
k_5 &= \frac{2EI_z}{L(1+\phi_z)} - \frac{\phi_z EI_z}{L(1+\phi_z)} \\
k_6 &= \frac{12EI_y}{L^3(1+\phi_y)} \\
k_7 &= \frac{6EI_y}{L^2(1+\phi_y)} \\
k_8 &= \frac{4EI_y}{L(1+\phi_y)} + \frac{\phi_y EI_y}{L(1+\phi_y)} \\
k_9 &= \frac{2EI_y}{L(1+\phi_y)} - \frac{\phi_y EI_y}{L(1+\phi_y)} \\
k_{10} &= \frac{GJ}{L}
\end{aligned}
$$

$$
\begin{bmatrix}
k_1 & 0 & 0 & 0 & 0 & 0 & -k_1 & 0 & 0 & 0 & 0 & 0 \\
0 & k_2 & 0 & 0 & 0 & k_3 & 0 & -k_2 & 0 & 0 & 0 & k_3 \\
0 & 0 & k_6 & 0 & -k_7 & 0 & 0 & 0 & -k_6 & 0 & -k_7 & 0 \\
0 & 0 & 0 & k_1 0 & 0 & 0 & 0 & 0 & 0 & -k_1 0 & 0 & 0 \\
0 & 0 & -k_7 & 0 & k_8 & 0 & 0 & 0 & k_7 & 0 & k_9 & 0 \\
0 & k_3 & 0 & 0 & 0 & k_4 & 0 & -k_3 & 0 & 0 & 0 & k_5 \\
-k_1 & 0 & 0 & 0 & 0 & 0 & k_1 & 0 & 0 & 0 & 0 & 0 \\
0 & -k_2 & 0 & 0 & 0 & -k_3 & 0 & k_2 & 0 & 0 & 0 & -k_3 \\
0 & 0 & -k_6 & 0 & k_7 & 0 & 0 & 0 & k_6 & 0 & k_7 & 0 \\
0 & 0 & 0 & -k_1 0 & 0 & 0 & 0 & 0 & 0 & k_1 0 & 0 & 0 \\
0 & 0 & -k_7 & 0 & k_9 & 0 & 0 & 0 & k_7 & 0 & k_8 & 0 \\
0 & k_3 & 0 & 0 & 0 & k_5 & 0 & -k_3 & 0 & 0 & 0 & k_4
\end{bmatrix}
\tag{2.3.10}
$$

Whereas the Bernoulli beam stiffness matrix look like

$$
\begin{aligned}
k_1 &= \frac{EA}{L} \\
k_2 &= \frac{12EI_z}{L^3} \\
k_3 &= \frac{6EI_z}{L^2} \\
k_4 &= \frac{4EI_z}{L} \\
k_5 &= \frac{2EI_z}{L} \\
k_6 &= \frac{12EI_y}{L^3} \\
k_7 &= \frac{6EI_y}{L^2} \\
k_8 &= \frac{4EI_y}{L} \\
k_9 &= \frac{2EI_y}{L} \\
k_{10} &= \frac{GJ}{L}
\end{aligned}
$$

$$
\begin{bmatrix}
k_1 & 0 & 0 & 0 & 0 & 0 & -k_1 & 0 & 0 & 0 & 0 & 0 \\
0 & k_2 & 0 & 0 & 0 & k_3 & 0 & -k_2 & 0 & 0 & 0 & k_3 \\
0 & 0 & k_6 & 0 & -k_7 & 0 & 0 & 0 & -k_6 & 0 & -k_7 & 0 \\
0 & 0 & 0 & k_10 & 0 & 0 & 0 & 0 & 0 & -k_10 & 0 & 0 \\
0 & 0 & -k_7 & 0 & k_8 & 0 & 0 & 0 & k_7 & 0 & k_9 & 0 \\
0 & k_3 & 0 & 0 & 0 & k_4 & 0 & -k_3 & 0 & 0 & 0 & k_5 \\
-k_1 & 0 & 0 & 0 & 0 & 0 & k_1 & 0 & 0 & 0 & 0 & 0 \\
0 & -k_2 & 0 & 0 & 0 & -k_3 & 0 & k_2 & 0 & 0 & 0 & -k_3 \\
0 & 0 & -k_6 & 0 & k_7 & 0 & 0 & 0 & k_6 & 0 & k_7 & 0 \\
0 & 0 & 0 & -k_10 & 0 & 0 & 0 & 0 & 0 & k_10 & 0 & 0 \\
0 & 0 & -k_7 & 0 & k_9 & 0 & 0 & 0 & k_7 & 0 & k_8 & 0 \\
0 & k_3 & 0 & 0 & 0 & k_5 & 0 & -k_3 & 0 & 0 & 0 & k_4
\end{bmatrix}
\tag{2.3.11}
$$

where G is modulus of rigidity. A is the surface area of the cross section of the beam. I_z and I_y are the second moment of inertia. For the cross section depicted in Figure 2.18 these values are

$$
I_y = \frac{ba^3}{12} \quad \& \quad I_z = \frac{ab^3}{12} \tag{2.3.12}
$$

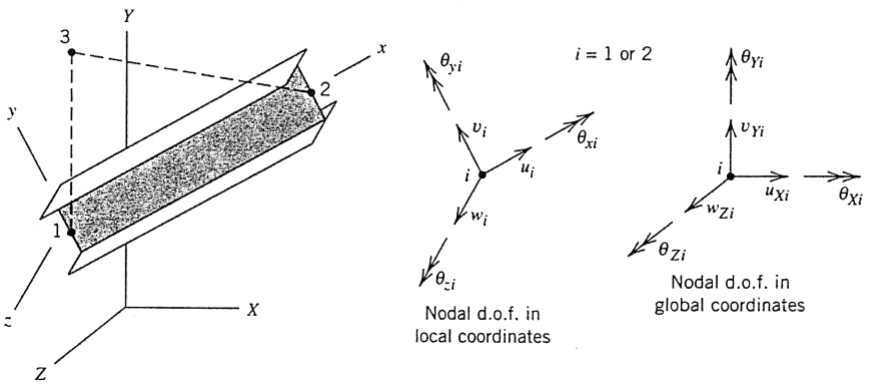


Figure 2.17: 3D beam element arbitrarily oriented in global coordinated XYZ, with nodal d.o.f. in local and global systems.(Reprinted from [3])

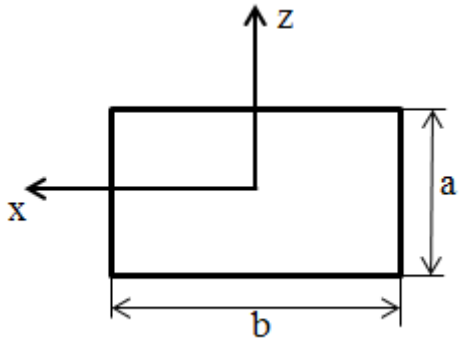


Figure 2.18: Cross Section of the Beam in Global Coordinate

where y and z are the "local" coordinate axes.

It should be reminded after creating these stiffness matrices, they should be transformed into the "global" one. This is done via a "transformation matrix" that is obtained by determining the relative rotation between the "local axis" and "global axis" (Figure 2.19). The stiffness matrix is most easily written in a local coordinate system. However, each element is oriented in a certain direction in the XYZ. Rather than creating the stiffness formula in *global coordinate system*, that is could be a tedious task, it is easier to transform an element initially formulated in local coordinates. Figure 2.19 on the following page, reprinted from [3], shows a rather easy and trivial example for the sake of further explanation.

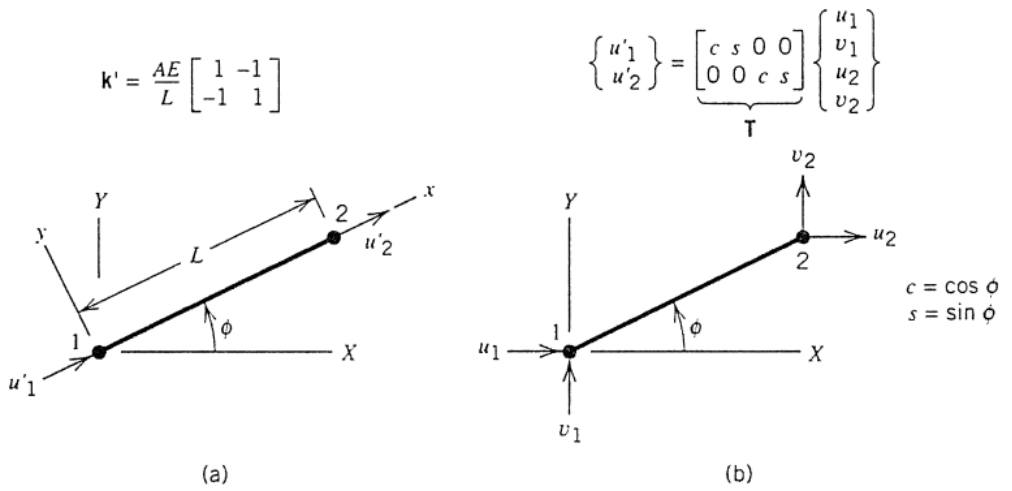


Figure 2.19: (**a**) Stifness matrix of a bar element in local coordinate xy. Local d.o.f. are u_1' and u_2'. (**b**) Transformation from local to global d.o.f. in the same plane.(Reprinted from [3])

2.3.2 Loads and Displacements

Mechanical Loads that are applied on a structure are the followings:

- Concentrated force or moment directly at a node.
- Load line or distributed force or moment along a line
- Surface pressure
- Body force loading, acts on every point of material in the body like weight or inertia forces
- Thermal loading

and all these loads are characterized by magnitude, direction, and applied node.

Limitation of the element can be viewed as follows

- The beam should be initially straight, linearly elastic, without taper
- Judging from the shape function a beam that is subjected to forces and moments only at nodes, would have a deflection shape that is cubic with respect to beam‘s axial direction (x), therefore:
  - A FE model built of beam elements provides an exact solution when forces/moments are applied to the nodes only

  - In the case that the load is a uniformly distributed one, the beam deflection would be of fourth degree in terms of x. Hence, the solution with beam elements would be somehow inexact, however, exact results can be approached if the number of elements is increased continuously (2.22 on page 37).

- If line loads are applied to the element, which includes axial forces, weight or inertia forces, then the load should be transformed to corresponding nodal forces. As an example, if the line load is q and applied on a beam along the main axis x with total length of L then total load would be q.L , with half of that applied to each node.

- If a transversal load is applied on the beam, then the load should be converted to equivalent forces and moments on the nodes; remembering that if these nodal load are equal in two adjacent elements the moments will cancel each other out at the common node.

After generating the stiffness matrix and keeping in mind the limitations of the FE code, the load vector **R** is introduced. This vector is composed of all the forces and moments at each node. This vector is introduced in the global coordinate.

Boundary conditions have their impact on the displacement vector **D** as predetermined displacement value on the corresponding nodes. For the rest of the nodes, which are "active nodes", applying $\mathbf{D} = \mathbf{K^{-1}R}$, would reveal the values of displacements at nodes. The achieved vector is the nodal displacement in global coordinates.

2.3.3 Calculating Stresses

Other than calculation of displacement, computing stress is one the main goals of a FE code. To that end based on the knowledge of the displacement vector, one should transfer it from global coordinate to the local one, and this is done by the "transformation matrix".

Now using the shape function and with the knowledge of the local nodal displacement is it possible to gain the displacement on desired locations of each element. Shape functions are interpolations that determine what the value of displacements along the beam would be if the corresponding two nodes‘ displacements are known. Their value depends on the type of modeling element type. For each element type, there exists a specific shape function matrix. For a 3D beam element the shape function is as follows

$$
\begin{aligned}
N_1 &= -\frac{1}{L}(x - x_j) \\
N_2 &= \frac{1}{L}(x - x_i) \\
N_3 &= 1 - 3\frac{x^2}{L^2} + 2\frac{x^3}{L^3} \\
N_4 &= x\left(-1 + 2\frac{x}{L} - \frac{x^2}{L^2}\right) \\
N_5 &= \frac{x^2}{L^2}\left(3 - 2\frac{x}{L}\right) \\
N_6 &= \frac{x^2}{L}\left(1 - \frac{x}{L}\right)
\end{aligned}
$$

$$
\begin{bmatrix}
N_1 & 0 & 0 & 0 & 0 & 0 & N_2 & 0 & 0 & 0 & 0 & 0 \\
0 & N_3 & 0 & 0 & 0 & -N_4 & 0 & N_5 & 0 & 0 & 0 & N_6 \\
0 & 0 & N_3 & 0 & N_4 & 0 & 0 & 0 & N_5 & 0 & N_6 & 0 \\
0 & 0 & 0 & N_1 & 0 & 0 & 0 & 0 & 0 & N_2 & 0 & 0
\end{bmatrix} \tag{2.3.13}
$$

$$
\begin{bmatrix} u_x(x) \\ u_y(x) \\ u_z(x) \\ \theta_x(x) \end{bmatrix} = \mathbf{N}(x)\mathbf{u} \tag{2.3.14}
$$

However for obtaining the stress along the element the strains are needed. To that end strains are gained via

$$
\begin{bmatrix} \frac{\partial u_x}{\partial x} \\ \frac{\partial u_y}{\partial x} \\ \frac{\partial u_z}{\partial x} \\ \frac{\partial \theta_x}{\partial x} \end{bmatrix} = \frac{\partial \mathbf{N}(x)}{\partial x}\mathbf{u} \qquad \begin{bmatrix} \frac{\partial^2 u_x}{\partial x^2} \\ \frac{\partial^2 u_y}{\partial x^2} \\ \frac{\partial^2 u_z}{\partial x^2} \\ \frac{\partial^2 \theta_x}{\partial x^2} \end{bmatrix} = \frac{\partial^2 \mathbf{N}(x)}{\partial x^2}\mathbf{u} \qquad \begin{bmatrix} \frac{\partial^3 u_x}{\partial x^3} \\ \frac{\partial^3 u_y}{\partial x^3} \\ \frac{\partial^3 u_z}{\partial x^3} \\ \frac{\partial^3 \theta_x}{\partial x^3} \end{bmatrix} = \frac{\partial^3 \mathbf{N}(x)}{\partial x^3}\mathbf{u} \tag{2.3.15}
$$

In general the relations between strains and stresses are evaluated by “constitutive equation”. The most popular form of the constitutive relation for linear elasticity is the

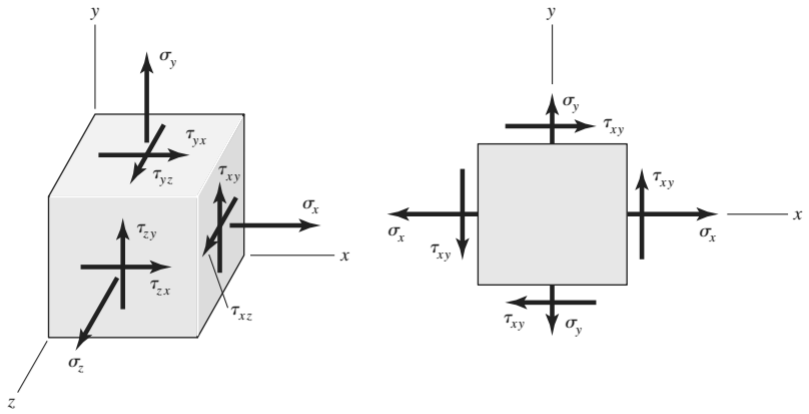


Figure 2.20: (**a**) General three-dimensional stress.(**b**) Plane stress with "cross-shears" equal.

following relation that holds for isotropic materials:

$$\begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \\ \gamma_{yz} \\ \gamma_{zx} \\ \gamma_{xy} \end{bmatrix} = \begin{bmatrix} \frac{1}{E} & \frac{-\nu}{E} & \frac{-\nu}{E} & 0 & 0 & 0 \\ \frac{-\nu}{E} & \frac{1}{E} & \frac{-\nu}{E} & 0 & 0 & 0 \\ \frac{-\nu}{E} & \frac{-\nu}{E} & \frac{1}{E} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{G} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{G} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{G} \end{bmatrix} \begin{bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{yz} \\ \tau_{zx} \\ \tau_{xy} \end{bmatrix} \tag{2.3.16}$$

However, that is the general case; for the present study many of the strains, stresses are negligible or not present. The present stresses and strains are

- Normal tensile or compression stress σ_x, σ_y or σ_z which can be due to normal forces or the flexural bending moment. Here just σ_x is considered, therefore, is obtained from

$$\sigma_x = E\epsilon_x \tag{2.3.17}$$

ϵ_x is the *longitudinal normal strain* of the element or the *deformation of the member per unit length.* Where the *normal strain* is

$$\epsilon_x = \frac{\partial u_x}{\partial x} + \frac{\partial^2 u_y}{\partial x^2} y + \frac{\partial^2 u_z}{\partial x^2} z \tag{2.3.18}$$

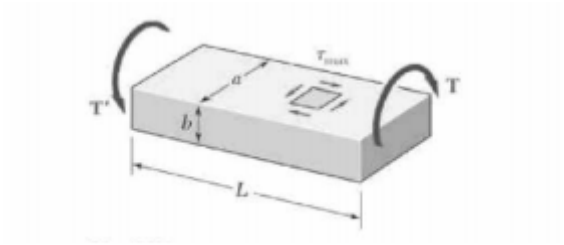


Figure 2.21: Torsional shear stress on a beam with a rectangular cross section

y and z are the distances from the cross-section neutral axis, that is obviously linearly varying across the cross section. As it will be explained further, in the present study the critical cases, therefore, just the maximum stresses are of interest. Thus, instead of y and z half of parameters of cross-sections are replaced.

- Shear stress τ_{xy}, τ_{xz} and τ_{yz} can occur due to either shear force or torsional moment. With the help of subsequent equations it is possible to determine the maximum stress that occurs along the center line of the wider face of the beam with a uniform rectangular cross section. Denoting by L the length of the beam, by a and b, respectively, the wider and narrower side of its cross section, and by T the magnitude of the torques applied to the beam (Figure 2.21), we find that the maximum shearing stress [6]

$$\tau_{max} = \frac{T}{c_1 a b^2} \quad , \quad \theta_x = \frac{TL}{c_2 a b^3 G} \quad \text{Thus} \quad \tau_{max} = \frac{c_2 b G \theta_x}{c_1 L} \tag{2.3.19}$$

The coefficients c_1 and c_2 depends only upon a/b and are given by Table 2.1 for a number of values of that ratio. Note that Equation 2.3.19 are only valid in the elastic range.

From Table 2.1 we can recognize that c_1 and c_2 will hold the same values if $a/b \geqslant 5$. It may be shown that for such values of a/b, we have

$$c_1 = c_2 = \frac{1}{3}(1 - 0.630 b/a) \qquad (\textbf{\textit{for}}\ \mathbf{a/b \geqslant 5}\ \textbf{\textit{only}}) \tag{2.3.20}$$

As for the transverse shear stress goes, for the Timoshenko beam we have

$$\begin{aligned} \gamma_{xy} &= \frac{\partial u_z}{\partial x} \\ \gamma_{xz} &= \frac{\partial u_y}{\partial x} \end{aligned}$$

Table 2.1: Coefficients for Rectangular Beams in Torsion

a/b	c_1	c_2
1	0.208	0.1406
1.2	0.219	0.1661
1.5	0.231	0.1958
2	0.246	0.229
2.5	0.258	0.249
3	0.267	0.263
4	0.282	0.281
5	0.291	0.291
10	0.312	0.312
∞	0.333	0.333

Therefore the transverse shear stress is

$$\tau_{xy} = kG\gamma_{xy} \qquad \tau_{xz} = kG\gamma_{xz} \tag{2.3.21}$$

G being the shear modulus

$$G = \frac{E}{2(1+\nu)} \tag{2.3.22}$$

and k the shear correction factor. This factor is dependent on the cross-sections and on the type of problem. Some authors use 5/6 for the static problems [7]. We will opt for the same value.

In the case stresses should be calculated, the FE software will compute displacement first and then stresses will be gained. Owing to the fact that stresses are proportional to strains, and of course strain is the derivative of displacements, nodal stresses are more accurate than stresses. In fact, stresses are well approximated at the center of the element, as well as the mean stress on the element. However, stresses at a boundary are not accurate [3]. Figure 2.22 compares the accuracy of the exact solution with a FE one under load line.

So far, all the stresses are computed separately, and all the values of each stress vary depending on the location. Therefore, in order to treat the element and define the critical point from stress point of view, we need to introduce an equivalent stress that takes into

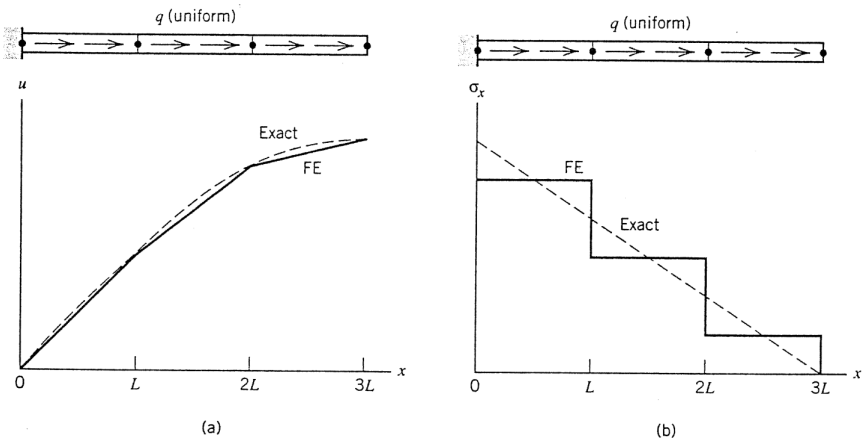


Figure 2.22: Exact solution VS FE one under load line (Reprinted from [3])

consideration all other stresses. In addition, the location with maximum stress should be determined.

The *von Mises stress*, σ', can be thought as a *single, equivalent, or effective stress* for the entire general state of stress given by $\sigma_x, \tau_{xy}, \tau_{xz}$

$$\sigma' = \frac{1}{\sqrt{2}} \left[(\sigma_x - \sigma_y)^2 + (\sigma_y - \sigma_z)^2 + (\sigma_z - \sigma_x)^2 + 6(\tau_{xy}^2 + \tau_{yz}^2 + \tau_{zx}^2) \right]^{1/2} \qquad (2.3.23)$$

As mentioned before, each of the stresses in Equation 2.3.23 alter in different positions along the beam and across the cross section; therefore, to discover the critical equivalent stress that is the maximum one, one should identify the locations that can be fair candidates for having the maximum stresses. By studying these locations, calculating and comparing the "von Mises stress" at different locations all along the element, we can determine the maximum stress at the element. In our case the critical points across the cross section of the element where the stress can be at its highest level are:

- At the center, where the transverse shear stress is at its highest level
- At the corners, here the all the shear stresses are equal to zero, but the bending stresses are at their maximum level. Additionally, there exists one corner where flexural stress due to moment along x and y are both positive.
- At each midface of the profile, namely at mid-a and mid-b (2.18 on page 30), here one of the transverse shear stresses is at its max, whereas the other one is zero. Equally important, the torsional shear stress is also maximum; furthermore,

bending moment creates a normal stress. For further explanation, in Figure 2.18, if the point is located at the middle of edge a, we have τ_{yz} due to torsion along y axis, T_y, plus the transverse one. The moment along z, M_z, generates the curvature $\frac{\partial^2 u_x}{\partial y^2}$ so the normal stress along the y axis. Whereas, for the mid-face of edge b we will have τ_{yz} due to T_y and the transverse shear. M_x is responsible for $\frac{\partial^2 u_z}{\partial y^2}$ so the normal stress σ_y.

2.4 Finite Element Code

Conducting a capable FE code is the foundation of every design optimization project. A capable code must be precise, nimble and accurate, that can account for all loads and various configurations. In developing the present code, a great deal of attention was paid to hold these standards and to assure the robustness of the results.

An overview of the stages of the code is depicted in the Figure 2.23. Every finite element analysis has these stages in common.

2.4.1 Pre-Processing

As in any CAE software the first step in performing a finite element analysis is the pre-processing stage. The stage describes any type of processing performed on raw data to prepare it for another processing procedure. Pre-processing transforms the data into a format that will be more easily and effectively processed for the purpose of other stages of modeling.

Chores done during the pre-processing stage of the code are

1. Definition of Material Property
2. Generation of Geometry of the Frame
3. Calculation of Mass and Area Properties
4. Setting the Input Loads
5. Description of Boundary Conditions

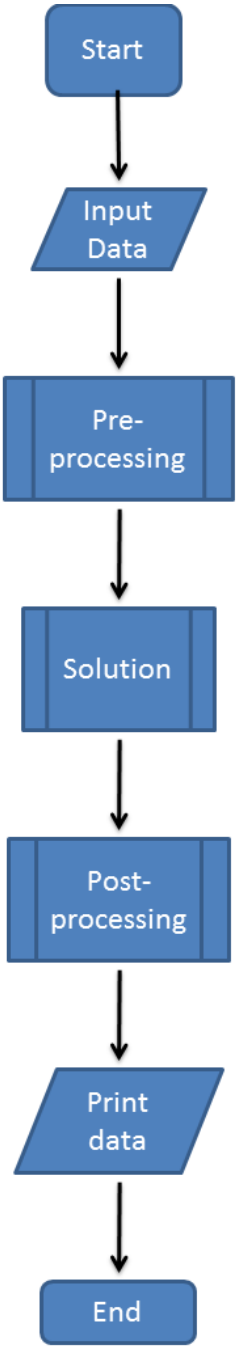


Figure 2.23: Over View of the Code

Defining Material Properties

The main material properties (subsection 2.1.2) are introduced to the code, these properties include: density, ρ, modulus of elasticity, E, Poisson's ratio, ν, modulus of

rigidity, G. The values of these properties are listed in Table 2.2.

Table 2.2: Input Material Properties

Property	Value	Dimension
ρ	2700	$\mathrm{kg\,m^{-3}}$
E	7.10E+10	Pa
ν	0.33	*non dimensional*
G	2.84E+10	Pa

These data are used in the next stages for calculation of stiffness of elements and the mass of the system.

Geometry of the Frame

The main task that is done during pre-processing stage is modeling the physical geometry of the frame. In other words, how the simplified model looks like and what the dimensions of the elements are. Indeed the caliper can have numerous shapes; plus, the chief goal of the research is to discover the optimal one. Therefore, for each analysis the geometry and dimensions must be determined.

For creating the shape, starting from a reference point, the coordinates of the nodes in the "global coordinate" are inserted in the code. The coordinates are inserted in a matrix that each row describes one node; furthermore, the matrix contains three columns, that are x, y and z respectively. An instance of this

$$\textit{Node Coordinate} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 14 & 0 \\ 0 & 28 & 0 \\ \vdots & \vdots & \vdots \\ 95 & 80 & 7 \\ \vdots & \vdots & \vdots \end{bmatrix}$$

As for the cross section another matrix is defined in which each row of the matrix is one element. *Cross section matrix* contains two columns, which describe the dimensions

of the cross sections of the elements. For instance

$$\textit{Cross Section} = \begin{bmatrix} 35 & 45 \\ 40 & 42 \\ \vdots & \vdots \end{bmatrix}$$

Here the values are inserted in mm.

However, so far we haven't actually defined for the code how the elements are connected. For this, *Element Nodes* matrix is introduced. Each row of this matrix defines an element. In each row the two columns are the nodes numbers. As an example

$$\mathsf{Element\,Nodes} = \begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 8 \\ \vdots & \vdots \end{bmatrix}$$

Therefore, the $\mathsf{element}\,1$ is connected to $\mathsf{element}\,2$ via $\mathsf{node}\,2$ and the $\mathsf{element}\,2$ is connected to $\mathsf{element}\,3$ though $\mathsf{node}\,3$.

The number of nodes is required by reading the number of rows of *node coordinate matrix*; on the other hand, the number of elements is the number of row in *Element Nodes matrix*. It is worth pointing out that we are modeling the hub carrier connection to the caliper by 6 springs. However, in the code two springs that each has stiffness in x, y, and z directions are used. In constructing the code springs are considered as elements; however, they should not be blended with other elements, since they behave completely different throughout the solution.

The number of degree of freedom, *d.o.f.*, for each node in a 3D beam modeling like the present one is equal to *6* (Figure 2.17).

$$\textit{Node}\ \mathsf{d.o.f.} = \begin{bmatrix} u_x \\ u_y \\ u_z \\ \theta_x \\ \theta_y \\ \theta_z \end{bmatrix} \tag{2.4.1}$$

Therefore, for the whole structure number of d.o.f. is

$$\mathsf{Structure\ d.o.f.} = 6 \times \textit{Number of Nodes} \tag{2.4.2}$$

Accordingly, $\mathrm{d.o.f.}_1$ is u_{1_x}, $\mathrm{d.o.f.}_2$ is u_{1_y}, $\mathrm{d.o.f.}_7$ is u_{2_x} and so forth. Each element has twelve d.o.f. and based on the element identity it contains two nodes, such that these two nodes determine the corresponding degrees of freedom of the element within the whole structure set, e.g., in

$$\mathrm{Element\,Nodes} = \begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 8 \\ \vdots & \vdots \end{bmatrix}$$

$\mathrm{element}\,3$ contains $\mathrm{node}\,3$ and $\mathrm{node}\,8$; hence, the degrees of freedom associated with this element are

$$\begin{Bmatrix} \mathrm{d.o.f.}_{13} \\ \mathrm{d.o.f.}_{14} \\ \mathrm{d.o.f.}_{15} \\ \mathrm{d.o.f.}_{16} \\ \mathrm{d.o.f.}_{17} \\ \mathrm{d.o.f.}_{18} \\ \mathrm{d.o.f.}_{43} \\ \mathrm{d.o.f.}_{44} \\ \mathrm{d.o.f.}_{45} \\ \mathrm{d.o.f.}_{46} \\ \mathrm{d.o.f.}_{47} \\ \mathrm{d.o.f.}_{48} \end{Bmatrix} = \begin{bmatrix} u_{3_x} \\ u_{3_y} \\ u_{3_z} \\ \theta_{3_x} \\ \theta_{3_y} \\ \theta_{3_z} \\ u_{8_x} \\ u_{8_y} \\ u_{8_z} \\ \theta_{8_x} \\ \theta_{8_y} \\ \theta_{8_z} \end{bmatrix}$$

For each element a vector, *Element Dof*, like the one above, is defined that describes the

associated degrees of freedom with the element. In general the matrix can be defined by

$$\mathtt{Element_i\,Dof} = \begin{bmatrix} 6\times \mathtt{node_i} - 5 \\ 6\times \mathtt{node_i} - 4 \\ 6\times \mathtt{node_i} - 3 \\ 6\times \mathtt{node_i} - 2 \\ 6\times \mathtt{node_i} - 1 \\ 6\times \mathtt{node_i} - 0 \\ 6\times \mathtt{node_{i+1}} - 5 \\ 6\times \mathtt{node_{i+1}} - 4 \\ 6\times \mathtt{node_{i+1}} - 3 \\ 6\times \mathtt{node_{i+1}} - 2 \\ 6\times \mathtt{node_{i+1}} - 1 \\ 6\times \mathtt{node_{i+1}} - 0 \end{bmatrix} \tag{2.4.3}$$

Now our geometry is fully described.

Mass and Area Properties

As noted before, the chief aim of pre-processing stage is to get the necessary data ready for the *Solution* stage (Figure 2.23). Element's stiffness is the foundation of the *solution* stage. For computing the stiffness matrix second moment of inertia and torsional constant is very much required. In equation 2.3.1 on page 22 the relation for the torsional constant is given by 2.3.9 on page 25, and the equation for second moment of inertia is given by 2.3.12 on page 29. After receiving the dimensions of the cross section of the element, the code must identify which edge is the larger one in calculating torsional constant.

Indeed the area is equal to $\mathtt{a}\times\mathtt{b}$, $\mathtt{a}$ and $\mathtt{b}$, being the dimensions of the cross section of the element. For calculating the mass of the structure the length of each element is needed. From *Element Node matrix* we know what nodes are located on each element. Plus, the *Node Coordinate matrix* describes the position of each node. Using these two matrices one can compute the length of each element from

$$\mathtt{L_e} = \sqrt{(\mathtt{x_2}-\mathtt{x_1})^2 + (\mathtt{y_2}-\mathtt{y_1})^2 + (\mathtt{z_2}-\mathtt{z_1})^2} \tag{2.4.4}$$

Where $\mathtt{x_2}$ and $\mathtt{x_1}$ are the X position of second and first nodes of element respectively in global coordinate system. In calculating the displacement along the element from the nodal displacement vector (2.3.14 on page 33), the local $\mathtt{x}$ along the element should be

specified. The element should be divided into a number of pieces; where these pieces are the resolution of x along the element. This is another input that is specified during pre-processing stage.

The Mass of the structure will be

$$M_{tot} = \sum_{e=1}^{n} \rho \times L_e \times A$$

n being the total number of elements.

Setting the Loads

On the real caliper loads appear in form of forces, moments, pressures, load line, transverse forces. Nonetheless, as discussed before for the sake simplicity of manipulability the smaller loads are neglected and the larger ones are modeled with equivalent normal forces. Indeed before this consideration certain verification has been performed to make sure that this assumption will not significantly affect the results of the research. These will be presented in the subsequent chapters of the present report.

The size of *force vector* is the same as the *Structure d.o.f.* number. Meaning that for every degree of freedom, there can exist a force acting on that direction, e.g. for hexagon structure, with six elements and nodes, the *force vector* will look like this

$$\mathbf{R} = \begin{bmatrix} F_{1_x} \\ F_{1_y} \\ F_{1_z} \\ \vdots \\ F_{4_z} \\ M_{4_x} \\ \vdots \\ M_{6_z} \end{bmatrix}$$

F_{1_x} and M_{1_x} are the normal force and moment in X global coordinate system.

Boundary Conditions

In describing the boundary conditions, B.C., we make changes to the global displacement vector. The global displacement vector, **D**, is consist of all nodes d.o.f.; i.e., each node introduces its own d.o.f. and the vector representing all of them is **D**.

$$\mathbf{D} = \begin{bmatrix} u_{1_x} \\ u_{1_y} \\ \vdots \\ \theta_{6_x} \\ \vdots \\ \theta_{n_x} \\ \theta_{n_y} \\ \theta_{n_z} \end{bmatrix}$$

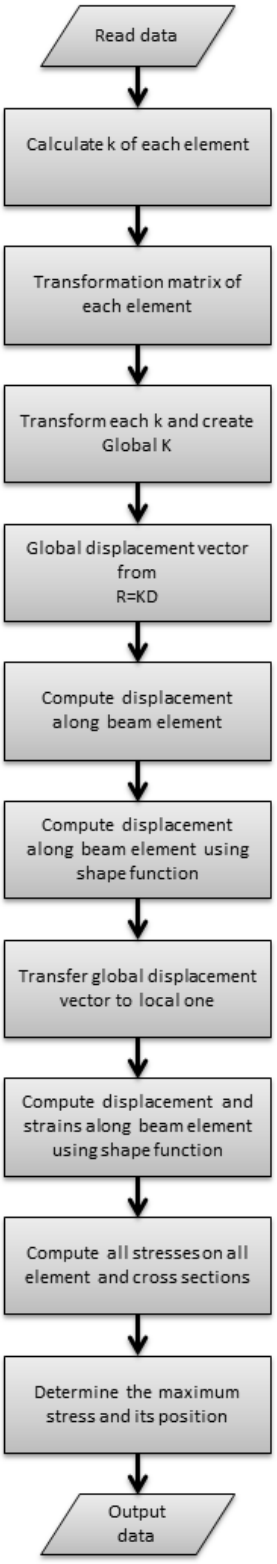
where n is the number of nodes.

When a node is fixed in some specific direction, the corresponding d.o.f. is exempted from participating in *solution equation* $\mathbf{D} = \mathbf{K}^{-1}\mathbf{R}$, and the value of it is set to zero.

2.4.2 Solution

The processor reads the data prepared in the preprocessing stage, generates matrices that describe the behavior of each element and combines these matrices into a large matrix equation that represents the whole structure. Applying the appropriate B.C. described in previous stage, it will precede by solving $\mathbf{KD} = \mathbf{R}$ to get the field displacement values at nodes. Element and node value of strain and stresses are computed from each solution [11].

Figure 2.24: Solution Stage Flow Chart



Global Stiffness Matrix

The matrix depending on which beam element we opt for, will differ. As explained in subsection 2.3.1; moreover, specifically by equation 2.3.10 on page 28 and 2.3.11 on page 29, depending on the slenderness of the beam element we should decide to use either Timoshenko beam theory or Bernoulli beam element. The stiffness matrix obtained by those two equations is in local coordinate system.

Therefore for each element a *rotation* or *transformation matrix* is defined. This will transform each local stiffness matrix to a global one. For creating the *global stiffness matrix* of the whole structure these 12×12 matrices should be assembled in one enormous matrix. This is done by first creating an empty global square matrix that its size *structure d.o.f.*, then for each element the calculated stiffness matrix is transformed and inserted in the global square matrix with indexes that correspond to the $\mathsf{Element_i\ Dof}$ (Equation 2.4.3). Note that for considering springs that are acting as our boundary conditions, the described stiffness is in global coordinates and there is no need to use *transformation matrix* for those stiffness matrices. They just have to be inserted in the correct spot in the global stiffness matrix.

$$\mathsf{Structure\ Dof} = \mathsf{n}, \qquad \mathsf{Global}\ \textit{Stiffness} = [\quad]_{\mathsf{n}\times\mathsf{n}}$$

$$\mathsf{Global}\ \textit{Stiffness}(\mathsf{Element_i\ Dof}, \mathsf{Element_i\ Dof}) = \mathsf{T}^{-1} \times \mathsf{k} \times \mathsf{T}$$

For springs

$$\mathsf{Global}\ \textit{Stiffness}(\mathsf{Element_i\ Dof}, \mathsf{Element_i\ Dof}) = \mathsf{k}_{\mathsf{s}_{12\times12}}$$

This should be repeated for all the elements in the structure until all the elements add their contribution to the *Global Stiffness matrix* .

Global Displacement Vector

It is achieved by the simple yet very powerful equation $\mathbf{KD = R}$. At this point all the terms of this equation except $\mathbf{D}$ is known. The solution is calculated for those activated node‘s d.o.f. that are not described as B.C.

Strains and Stresses

In subsection 2.3.3 and section 2.3.1 it was explained how shape function can be used to achieve the displacement and strains across each element. Equation 2.3.14 on page 33

describes the displacement along the beam element and Equation 2.3.15 can be used to obtain the strains. Equations are repeated here for convenience.

$$\begin{bmatrix} N_1 & 0 & 0 & 0 & 0 & 0 & N_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & N_3 & 0 & 0 & 0 & -N_4 & 0 & N_5 & 0 & 0 & 0 & N_6 \\ 0 & 0 & N_3 & 0 & N_4 & 0 & 0 & 0 & N_5 & 0 & N_6 & 0 \\ 0 & 0 & 0 & N_1 & 0 & 0 & 0 & 0 & 0 & N_2 & 0 & 0 \end{bmatrix} \tag{2.3.13}$$

$$\begin{bmatrix} u_x(x) \\ u_y(x) \\ u_z(x) \\ \theta_x(x) \end{bmatrix} = \mathbf{N}(x)\mathbf{u} \tag{2.3.14}$$

$$\begin{bmatrix} \frac{\partial u_x}{\partial x} \\ \frac{\partial u_y}{\partial x} \\ \frac{\partial u_z}{\partial x} \\ \frac{\partial \theta_x}{\partial x} \end{bmatrix} = \frac{\partial \mathbf{N}(x)}{\partial x}\mathbf{u} \qquad \begin{bmatrix} \frac{\partial^2 u_x}{\partial x^2} \\ \frac{\partial^2 u_y}{\partial x^2} \\ \frac{\partial^2 u_z}{\partial x^2} \\ \frac{\partial^2 \theta_x}{\partial x^2} \end{bmatrix} = \frac{\partial^2 \mathbf{N}(x)}{\partial x^2}\mathbf{u} \qquad \begin{bmatrix} \frac{\partial^3 u_x}{\partial x^3} \\ \frac{\partial^3 u_y}{\partial x^3} \\ \frac{\partial^3 u_z}{\partial x^3} \\ \frac{\partial^3 \theta_x}{\partial x^3} \end{bmatrix} = \frac{\partial^3 \mathbf{N}(x)}{\partial x^3}\mathbf{u} \tag{2.3.15}$$

In the code, for each portion of the element $\mathbf{N}$ is calculated from equation 2.3.13. The size of the portions depends on the specified resolution of x along the element. Solving equation 2.3.14, the displacement vector for the first portion is gained. Then we move on to the next portion and repeat the cycle until the whole length of the element is covered. Then the whole cycle is repeated for the next element. To shed light on how exactly this is done here a part of the code is depicted.

```
for e=1:numberElements;

SH=[];

for x=linspace(0,L(e),elementPieces);

N1=-1/L(e)*(x-L(e));
N2=1/L(e)*(x);
N3=1-3*(x/L(e))^2+2*(x/L(e))^3;
N4=x*(-1+2*x/L(e)-(x/L(e))^2);
N5=(x/L(e))^2*(3-2*x/L(e));
N6=x^2/L(e)*(1-x/L(e));
a=[N1 0 0 0 0 0 N2 0 0 0 0 0];
b=[0 N3 0 0 0 -N4 0 N5 0 0 0 -N6];
c=[0 0 N3 0 N4 0 0 0 N5 0 N6 0];
d=[0 0 0 N1 0 0 0 0 0 N2 0 0];
F=[a;b;c;d];
Ux=F*U(elementDof); % U is the global
%displacement vector AKA "R"
SH=[SH,Ux];



end
Ut1=[Ut1;SH];% each row present an element
%and each column is a position along that element
end
```

The exact same procedure is carried out for obtaining strain, except instead of Equation 2.3.14, Equation 2.3.15 is used in the cycle.

As for stress, in the beginning the global displacements of nodes are transformed to the local ones. Remembering figure 2.19 on page 31, and if $\mathbf{T}$ would be the global *transformation matrix*, then in MATLAB code we have,

$$\mathrm{R} = \mathbf{T}\left((\mathrm{element}_{\mathrm{number}} - 1) \times 12 + 1 : \mathrm{element}_{\mathrm{number}} \times 12, :\right)$$

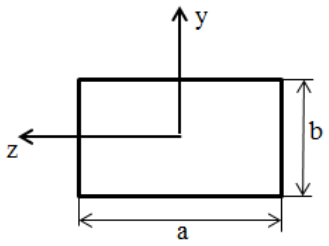


Figure 2.25: Elements Cross Section

$$\text{Local Nodal Displacement} = U_L = R \times \mathbf{D}(\text{Element Dof})$$

With the same procedure that calculates global displacement\strain along the beam element, we shall carry out to compute local ones along the beam element, except here U or $\mathbf{R}$ is replaced by U_L.

Transforming these strains in the local coordinate system, it now possible to determine what the stresses are along the beam elements. The first one to be calculated is the normal stress by equation 2.3.17 on page 34 that is present all over the cross section of the element. The normal flexural stress is achieved from

$$\sigma_x = \frac{b}{2} E \frac{\partial^3 u_y}{\partial x^3}, \qquad \sigma_x = \frac{a}{2} E \frac{\partial^3 u_z}{\partial x^3} \tag{2.4.5}$$

where a and b are presented in figure 2.25.

Next in line is shear stress, which is known from equations 2.3.21 on page 36 and 2.3.19 on page 35. Finally, von Mises stress is calculated at center, corners, and midface of cross sections along the each element beam is formulated. For each position in cross section the maximum value along all elements is extracted. Comparing all the maximums of center, midfaces, and corner would reveal the highest stress value in the entire structure. For more clarification piece of the code is attached here

```
function [SVMco,num2,di2,dj2]= cornerVM(SigX,...
SigByx,SigBzx,tauxy,...
tauxz,tautzx,tautyz,numberElements,elementPieces)
tauxy=zeros(numberElements,elementPieces);
tautyx=zeros(numberElements,elementPieces);
tautzx=zeros(numberElements,elementPieces);
tauxz=zeros(numberElements,elementPieces);
SX=SigX+abs(SigByx)+abs(SigBzx);
tXY=tauxy+tautyx;
tXZ=tauxz+tautzx;
SVMco=1/(2^0.5).*(2.*SX.^2+6.*(tXY.^2+tXZ.^2)).^0.5;
[num2,ind1]=max(SVMco(:));
[di2,dj2]= ind2sub(size(SVMco),ind1);
end
```

The function calculates von Mises stress at corner point of cross section:

```
 %% Von Mises in different locations and the critical stress

%at beams center
[SVMce,num1,di1,dj1]=...
    centerVM(SigX,SigByx,SigBzx,tauxy,tautyx,tauxz,...
    tautzx,numberElements,elementPieces);
% at edge corner
[SVMco,num2,di2,dj2]=...
    cornerVM(SigX,SigByx,SigBzx,tauxy,tautyx,tauxz...
    ,tautzx,numberElements,elementPieces);
%at midface in the local y direction on face b
[SVMmfy,num3,di3,dj3]= ...
    MidfaceYd(SigX,SigByx,SigBzx,tauxy,tautyx,tauxz...
    ,tautzx,numberElements,elementPieces);
%at midface in the local z direction on face a
[SVMmfz,num4,di4,dj4]=...
    MidfaceZd(SigX,SigByx,SigBzx,tauxy,tautyx,tauxz...
    ,tautzx,numberElements,elementPieces);

VM=[num1
    num2
    num3
    num4];
Ind=[di1 dj1
    di2 dj2
    di3 dj3
    di4 dj4];
[VMmax,gg]=max(VM);
```

2.4.3 Post-Processing

This processor takes the results and creates graphic display of the structural deformation and stress components. The node displacements are usually very small for most

engineering structures. Particularly in the present code, the displacement in each d.o.f. at each node is printed. This is easily done like

```
U=Global_displacement
for II =[1:6:GDof]

    u(j)=U(II);
    j=j+1;
end
```

where GDof is of course *Structure d.o.f.* The same cycle holds for the other d.o.f. Then these values along with Maximum stress are printed for the user. Note that in the optimization section some plotting and graphical features are added to this stage.

2.5 Remarks

Although the code is developed with a great precision and all the formulations are validated against robust references, to make sure that the code is healthy and the results contains enough accuracy, it must be tasted against some commercial FEM software. In the next chapter this will be carried out. Different scenarios are painted for the code, and all the stresses and displacements are compared against the results of the software. Only after this, we can move on to the optimization stage with great confidence.

Chapter 3

Validation

3.1 Introduction

Like any other engineering design, the present model didn't come about in one shot. The model was started with a very simple shape and slowly developed to a more perfect one, meanwhile some unnecessary details were cut off to make the model simple and manageable. The MATLAB code that is thoroughly explained in chapter 2 realizes this model. However, this code should be calibrated and validated against solid reference software as well. We chose ABAQUS for this mission. Subsection 3.1.1 gives a brief overview of the software. For verification some points and outputs are of more importance for the design. These results were watched more closely and every time the model is revised an equivalent model is developed in ABAQUS to certify MATLAB output against the ABAQUS one.

3.1.1 About ABAQUS

ABAQUS is a suite of software applications for finite element analysis and computer-aided engineering. Abaqus is used in the automotive, aerospace, and industrial products industries. The product is popular with academic and research institutions due to the wide material modeling capability, and the program's ability to be customized. Abaqus also provides a good collection of multiphysics capabilities, such as coupled acoustic-structural, piezoelectric, and structural-pore capabilities, making it attractive for production-level simulations where multiple fields need to be coupled. Abaqus/CAE is capable of pre-processing, post-processing, and monitoring the processing stage of the solver; however, the first stage can also be done by other compatible CAD software, or

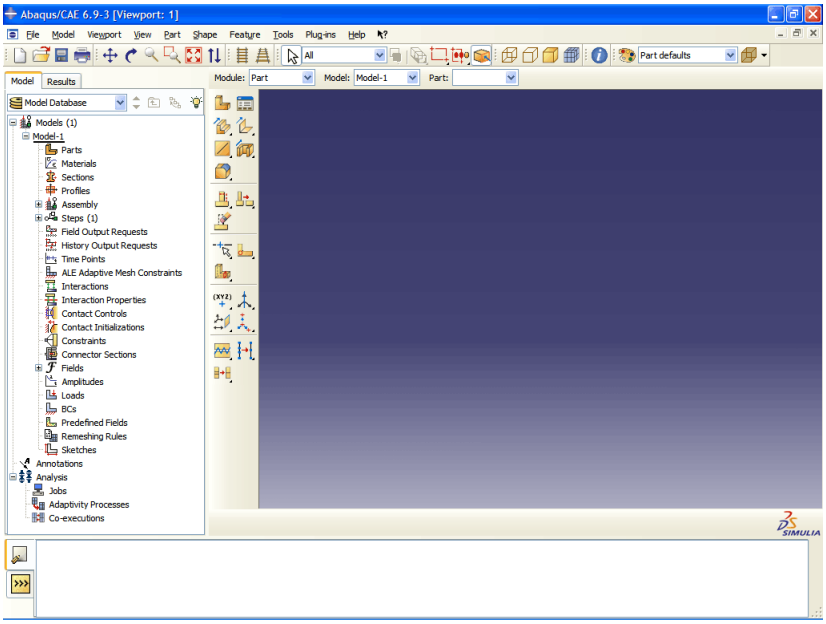


Figure 3.1: ABAQUS GUI

even a text editor. The product suite consists of four core software products [14]:

1. Abaqus/CAE, or "Complete Abaqus Environmen". It is a software application used for both the modeling and analysis of mechanical components and assemblies
2. Abaqus/CFD, a Computational Fluid Dynamics software application
3. Abaqus/Standard, a general-purpose Finite-Element analyzer that employs implicit integration scheme (traditional).
4. Abaqus/Explicit, a special-purpose Finite-Element analyzer that employs explicit integration scheme to solve highly nonlinear systems with many complex contacts under transient loads.

In the present study Abqus/CAE is employed to certify the outputs of MATLAB code.

Abaqus/CAE is a complete Abaqus environment that provides a simple, consistent interface for creating, submitting, monitoring, and evaluating results from Abaqus/Standard and Abaqus/Explicit simulations. Abaqus/CAE is divided into modules, where each module defines a logical aspect of the modeling process; for example, defining the geometry, defining material properties, and generating a

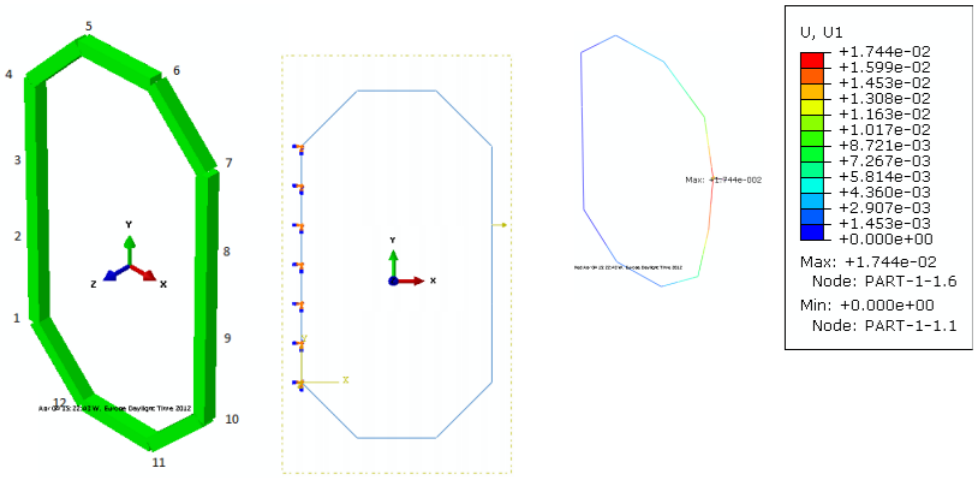


Figure 3.2: The Very First Frame

mesh. As you move from module to module, you build the model from which Abaqus/CAE generates an input file that you submit to the Abaqus/Standard or Abaqus/Explicit analysis product. The analysis product performs the analysis, sends information to Abaqus/CAE to allow you to monitor the progress of the job, and generates an output database. Finally, you use the Visualization module of Abaqus/CAE (also licensed separately as Abaqus/Viewer) to read the output database and view the results of your analysis.

Abaqus/Viewer provides graphical display of Abaqus finite element models and results. Abaqus/Viewer is incorporated into Abaqus/CAE as the Visualization module.

3.2 Evolution of the Model

The very first attempt to model the caliper is a very simple 2D frame with 12 node and 12 elements, this model was developed more so to test the MATLAB code. Figure 3.2 depicts this model and the results from ABAQUS model

As you can note the model is very simple; however, it's valuable in the sense that it backs up the precision of our code, of course it wasn't by no means the end of validation of the code, the code was put to test in various scenarios and at each step of the project for warrantying a sound result at the end of optimization processes.

The first thing to be introduced to this simple model to make it more realistic is the piston locations with their exact locations in 3D space. Figure 3.3 is reprinted here,

Table 3.1: Matlab Results of First Model. See Figure 3.2 for comparison

Node	Displacement
1	0
2	0
3	0
4	0
5	0.002663
6	0.003606
7	0.009460
8	0.017441
9	0.014537
10	0.006308
11	0.002666
12	0.002121

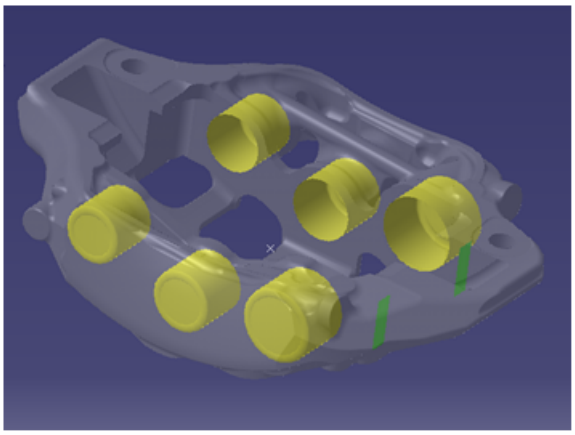


Figure 3.3: Caliper Original Design

as can be observed there are three pistons on each side. Therefore, the model should contain an element for each of these pistons; furthermore, we need an element to connect each of these pistons together. Equally important, there should be the spot where to insert lateral concentrated force on the caliper due to rotation of disk (shown by green straps in Figure 3.3). A node should be located at each of these points to carry the load.

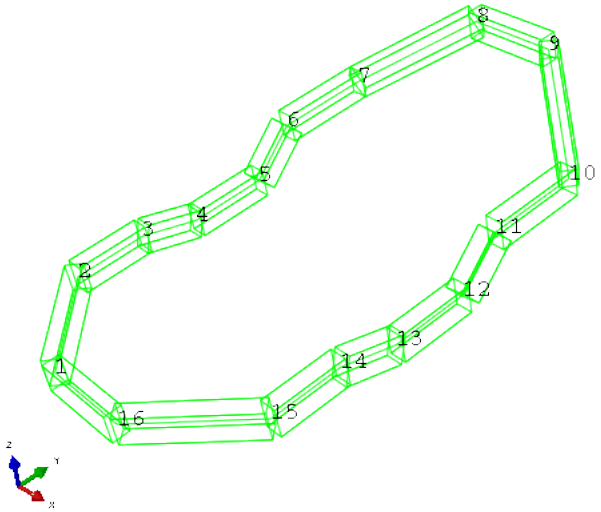


Figure 3.4: Second Model with Node Numbers

Thus, the caliper model should look like the following so far (3.4 & 3.5 on the next page)

Nevertheless, modeling the pistons with just one element means that the pressure distribution that fluid exerts on the cylinder bottom must be known. Since we do not have such information, an alternative solution shall be sought. One method is to model each piston by two elements instead of one. In this manner a good approximation is to model the pressure at the bottom of the cylinder as concentrated force acting on the middle node in between those two elements. This concentrated force is obtained by multiplying the pressure in the cylinder by the area of the bottom. Thus, the model now has evolved into (3.7 & 3.6 on the following page).

Third model is almost fully grown, and after validation of the accuracy against ABAQUS counterpart model, which shall be discussed in the subsequent section, we can introduce more details in to the model. If figure 3.3 on the previous page is watched closely, one can notice that the frame contains some connecting elements. These connecting elements have not been modeled yet. Luckily these connecting elements highly resemble beam elements in reality; consequently, it is possible to insert the actual dimensions of these elements in the model. Additionally, as expressed in subsection 2.1.1, the caliper is mounted on the hub carrier via two M10 screws that cannot be altered. However, these

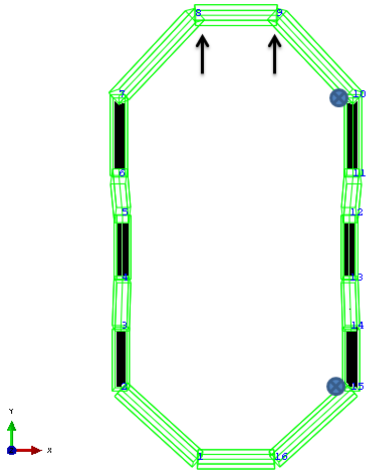


Figure 3.5: Second Model top view, the pistons are colored in black. The lateral forces are shown by arrows, and the B.C. is depicted by small blue circles

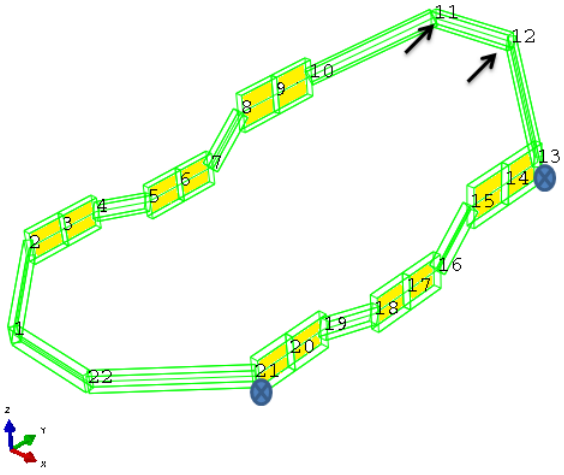


Figure 3.6: Third Model, the pistons are colored in yellow color. The nodes are numbered, the lateral forces are shown by arrows, and the B.C. is depicted by small blue circles

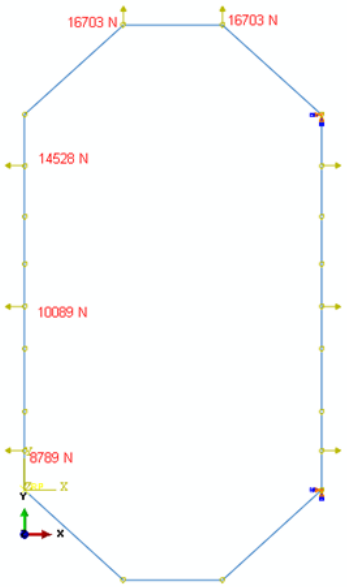


Figure 3.7: Third Model, all the forces and B.C. are presented in the figure

connections will introduce a sort of spring effect on the B.C. of caliper. Presently, the model is sophisticated enough to introduce that spring effect into the B.C. of caliper. The values of the stiffness are reported in the preceding section. Next model includes the spring as B.C. and connecting element.

3.3 Validation

All these models that have been developed so far are conducted to be a basis for the optimization stage, therefore the precision of these models are of great importance. And these models can serve as the most perfect tool to test the MATLAB FE code that has been generated. To this end, for the preliminary models-the first, second and third models-all the node displacement were put to the test. Here for being brief just the results of the third model, that is the most developed one, are reported.

The pressure inside the cylinder is $14.273\,\mathrm{MPa}$, converting this to concentrated force, the value of force over each piston would be, $14\,528\,\mathrm{N}$, $1089\,\mathrm{N}$ and $8789\,\mathrm{N}$. The positions of these concentrated forces are exhibited in Figure 3.7. As mentioned in subsection 2.3.1, there are two different stiffness matrices that can be used for getting the displacement,

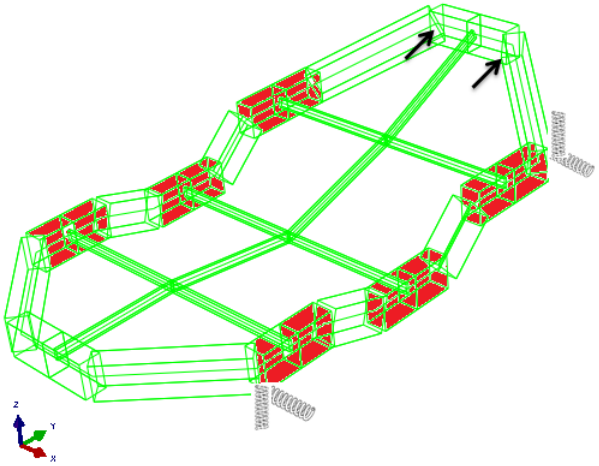


Figure 3.8: Forth Model, the pistons are colored in red color. The lateral forces are shown by arrows, the connecting elements can be observed and the springs as the B.C. can be noted

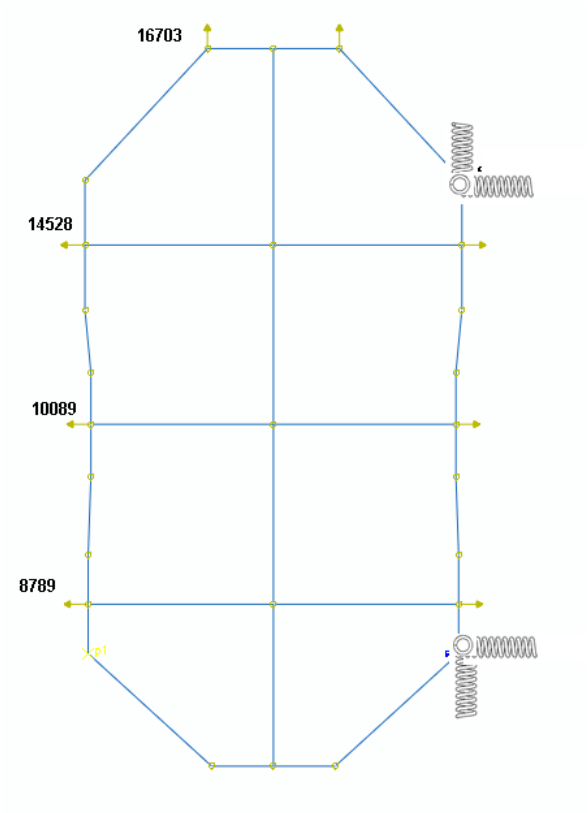


Figure 3.9: Forth Model, as noted, the springs act on XYZ directions. Forces and their values are present along with the connecting elements

namely, Bernoulli and Timoshenko. Here the structure is solved by both beam element types and the results are compared against Standard, Linear Geometry, cubic formulation element type of ABAQUS for Bernoulli beams; and Standard, Linear Geometry, standard quadratic element type of ABAQUS for Timoshenko beams.

Table 3.2: Bernoulli Beam Element Verification-u_x

NODE#	u_x[mm] *ABAQUS*	u_x[mm] *MATLAB*	*%error*
1	-0.023517	-0.023513	0.0128
2	-0.029793	-0.029788	0.0191
3	-0.037174	-0.037165	0.0246
4	-0.052605	-0.052577	0.0531
5	-0.057621	-0.057589	0.0566
6	-0.059474	-0.059435	0.0645
7	-0.047011	-0.046995	0.0329
8	-0.039073	-0.039058	0.0388
9	-0.027430	-0.027413	0.0613
10	0.008737	0.008732	0.0550
11	0.010449	0.010445	0.0466
12	0.000000	0.000000	0.0000
13	0.001773	0.001773	0.0128
14	0.004573	0.004572	0.0203
15	0.008024	0.008017	0.0862
16	0.008301	0.008296	0.0668
17	0.007039	0.007035	0.0651
18	0.003059	0.003058	0.0270
19	0.001003	0.001003	0.0208
20	0.000000	0.000000	0.0000
21	-0.020540	-0.020541	0.0022
22	-0.022539	-0.022540	0.0023
MAX error			0.0862

Table 3.3: Bernoulli Beam Element Verification-u_y

NODE#	u_y[mm] *ABAQUS*	u_y[mm] *MATLAB*	*%error*
1	0.051795	0.051800	0.0109
2	0.052357	0.052362	0.0106
3	0.052918	0.052924	0.0105
4	0.054232	0.054238	0.0102
5	0.054834	0.054839	0.0101
6	0.055436	0.055441	0.0099
7	0.057714	0.057719	0.0083
8	0.058438	0.058442	0.0082
9	0.059161	0.059165	0.0080
10	0.044030	0.044023	0.0137
11	0.020207	0.020202	0.0264
12	0.000000	0.000000	0.0000
13	0.000000	0.000000	0.0000
14	0.000000	0.000000	0.0000
15	0.000000	0.000000	0.0000
16	0.000000	0.000000	0.0000
17	0.000000	0.000000	0.0000
18	0.000000	0.000000	0.0000
19	0.000000	0.000000	0.0000
20	0.000000	0.000000	0.0000
21	0.021406	0.021406	0.0015
22	0.047999	0.048001	0.0036
MAX error			0.0264

Table 3.4: Bernoulli Beam Element Verification-$\mathfrak{u}_z$

NODE#	$\mathfrak{u}_z$[mm] *ABAQUS*	$\mathfrak{u}_z$[mm] *MATLAB*	%*error*
1	0.038096	0.038022	0.1940
2	0.039760	0.039685	0.1871
3	0.041235	0.041161	0.1809
4	0.042593	0.042519	0.1744
5	0.042819	0.042746	0.1725
6	0.042183	0.042110	0.1738
7	0.040770	0.040697	0.1776
8	0.037844	0.037772	0.1893
9	0.034125	0.034054	0.2075
10	0.005881	0.005860	0.3547
11	-0.003887	-0.003888	0.0067
12	0.000000	0.000000	0.0000
13	0.000000	0.000000	0.0000
14	0.000000	0.000000	0.0000
15	0.000000	0.000000	0.0000
16	0.000000	0.000000	0.0000
17	0.000000	0.000000	0.0000
18	0.000000	0.000000	0.0000
19	0.000000	0.000000	0.0000
20	0.000000	0.000000	0.0000
21	0.003312	0.003311	0.0231
22	0.016066	0.016039	0.1669
MAX error			0.3547

Table 3.5: Bernoulli Beam Element Verification-θ_x

NODE#	θ_x[rad] *ABAQUS*	θ_x[rad] *MATLAB*	%*error*
1	0.000124	0.000124	0.0371
2	0.000113	0.000113	0.0207
3	0.000097	0.000097	0.0038
4	0.000042	0.000042	0.0541
5	-0.000013	-0.000013	0.2800
6	-0.000073	-0.000073	0.0602
7	-0.000144	-0.000144	0.0350
8	-0.000183	-0.000183	0.0264
9	-0.000232	-0.000232	0.0175
10	-0.000482	-0.000481	0.1848
11	-0.000451	-0.000450	0.1406
12	0.000000	0.000000	0.0000
13	0.000000	0.000000	0.0000
14	0.000000	0.000000	0.0000
15	0.000000	0.000000	0.0000
16	0.000000	0.000000	0.0000
17	0.000000	0.000000	0.0000
18	0.000000	0.000000	0.0000
19	0.000000	0.000000	0.0000
20	0.000000	0.000000	0.0000
21	0.000156	0.000155	0.4885
22	0.000198	0.000197	0.3550
MAX error			0.4885

Table 3.6: Bernoulli Beam Element Verification-θ_y

NODE#	θ_y[rad] *ABAQUS*	θ_y[rad] *MATLAB*	%*error*
1	0.000544	0.000542	0.2345
2	0.000584	0.000583	0.2515
3	0.000625	0.000623	0.2664
4	0.000660	0.000658	0.3000
5	0.000661	0.000659	0.3054
6	0.000662	0.000660	0.3106
7	0.000649	0.000648	0.2170
8	0.000640	0.000638	0.2194
9	0.000630	0.000629	0.2220
10	0.000348	0.000348	0.1706
11	0.000216	0.000215	0.2686
12	0.000000	0.000000	0.0000
13	-0.000043	-0.000043	0.3769
14	-0.000086	-0.000086	0.3771
15	-0.000110	-0.000109	0.4845
16	-0.000102	-0.000102	0.4811
17	-0.000095	-0.000095	0.4764
18	-0.000063	-0.000063	0.4377
19	-0.000032	-0.000031	0.4377
20	0.000000	0.000000	0.0000
21	0.000316	0.000316	0.2364
22	0.000406	0.000406	0.1827
MAX error			0.4845

Table 3.7: Bernoulli Beam Element Verification-θ_z

NODE#	θ_z[rad] *ABAQUS*	θ_z[rad] *MATLAB*	%*error*
1	0.000365	0.000365	0.0471
2	0.000507	0.000506	0.0432
3	0.000535	0.000535	0.0504
4	0.000419	0.000418	0.0668
5	0.000236	0.000235	0.1476
6	0.000013	0.000012	3.2910
7	-0.000317	-0.000317	0.0207
8	-0.000562	-0.000562	0.0068
9	-0.000695	-0.000695	0.0214
10	-0.000725	-0.000725	0.0173
11	-0.000601	-0.000601	0.0129
12	0.000000	0.000000	0.0000
13	0.000153	0.000153	0.0164
14	0.000147	0.000147	0.0356
15	0.000064	0.000064	0.1623
16	-0.000034	-0.000034	0.2302
17	-0.000125	-0.000125	0.0403
18	-0.000162	-0.000162	0.0357
19	-0.000123	-0.000123	0.0241
20	0.000000	0.000000	0.0000
21	-0.000910	-0.000910	0.0028
22	-0.000500	-0.000500	0.0123
MAX error			3.2910

As seen, the results hold spectacular accuracy, the highest $\%error$ is the θ_z at node 6 in Table 3.7, where the displacements are 1.28×10^{-5} and 1.24×10^{-5}; hence, the error is truly negligible.

Next the results of the comparison of Timoshenko beam element are presented.

Table 3.8: Timoshenko Beam Element Verification-u_x

NODE#	u_x[mm] *ABAQUS*	u_x[mm] *MATLAB*	$\%error$
1	-0.030638	-0.030776	0.4503
2	-0.039070	-0.039248	0.4546
3	-0.047276	-0.047466	0.4024
4	-0.063841	-0.064035	0.3034
5	-0.069407	-0.069607	0.2882
6	-0.070230	-0.070404	0.2476
7	-0.056287	-0.056454	0.2976
8	-0.046835	-0.046973	0.2955
9	-0.031042	-0.031097	0.1766
10	0.009557	0.009569	0.1229
11	0.011281	0.011293	0.1054
12	0.000000	0.000000	0.0000
13	0.005032	0.005097	1.2879
14	0.008497	0.008574	0.9107
15	0.012762	0.012850	0.6876
16	0.013551	0.013650	0.7339
17	0.011332	0.011413	0.7183
18	0.005909	0.005965	0.9538
19	0.003009	0.003049	1.3304
20	0.000000	0.000000	0.0000
21	-0.026766	-0.026890	0.4631
22	-0.028800	-0.028879	0.2728
MAX error			1.3304

Table 3.9: Timoshenko Beam Element Verification-u_y

NODE#	u_y[mm] *ABAQUS*	u_y[mm] *MATLAB*	*%error*
1	0.061118	0.061309	0.3129
2	0.061665	0.061856	0.3097
3	0.062212	0.062403	0.3066
4	0.063654	0.063848	0.3035
5	0.064241	0.064434	0.3002
6	0.064827	0.065020	0.2970
7	0.068501	0.068720	0.3201
8	0.069206	0.069425	0.3164
9	0.069911	0.070129	0.3126
10	0.052804	0.052973	0.3210
11	0.025139	0.025233	0.3735
12	0.000000	0.000000	0.0000
13	0.000000	0.000000	0.0000
14	0.000000	0.000000	0.0000
15	0.000000	0.000000	0.0000
16	0.000000	0.000000	0.0000
17	0.000000	0.000000	0.0000
18	0.000000	0.000000	0.0000
19	0.000000	0.000000	0.0000
20	0.000000	0.000000	0.0000
21	0.028183	0.028318	0.4784
22	0.058376	0.058584	0.3562
MAX error			0.4784

Table 3.10: Timoshenko Beam Element Verification-u_z

NODE#	u_z[mm] *ABAQUS*	u_z[mm] *MATLAB*	%*error*
1	0.038808	0.038750	0.1486
2	0.040280	0.040218	0.1538
3	0.041558	0.041492	0.1590
4	0.043387	0.043330	0.1311
5	0.043412	0.043352	0.1396
6	0.042592	0.042528	0.1499
7	0.039558	0.039464	0.2397
8	0.036463	0.036366	0.2679
9	0.032601	0.032501	0.3070
10	-0.000093	-0.000093	0.3041
11	-0.009578	-0.009693	1.1979
12	0.000000	0.000000	0.0000
13	0.000000	0.000000	0.0000
14	0.000000	0.000000	0.0000
15	0.000000	0.000000	0.0000
16	0.000000	0.000000	0.0000
17	0.000000	0.000000	0.0000
18	0.000000	0.000000	0.0000
19	0.000000	0.000000	0.0000
20	0.000000	0.000000	0.0000
21	0.004147	0.004163	0.3795
22	0.017122	0.017116	0.0298
MAX error			1.1979

Table 3.11: Timoshenko Beam Element Verification-θ_x

NODE#	θ_x[rad] *ABAQUS*	θ_x[rad] *MATLAB*	%*error*
1	0.000092	0.000092	0.7092
2	0.000081	0.000080	0.8001
3	0.000064	0.000064	0.9947
4	0.000010	0.000009	6.3367
5	-0.000044	-0.000044	1.3424
6	-0.000103	-0.000104	0.5362
7	-0.000172	-0.000172	0.3014
8	-0.000210	-0.000210	0.2375
9	-0.000257	-0.000257	0.1842
10	-0.000500	-0.000499	0.1046
11	-0.000464	-0.000463	0.0765
12	0.000000	0.000000	0.0000
13	0.000000	0.000000	0.0000
14	0.000000	0.000000	0.0000
15	0.000000	0.000000	0.0000
16	0.000000	0.000000	0.0000
17	0.000000	0.000000	0.0000
18	0.000000	0.000000	0.0000
19	0.000000	0.000000	0.0000
20	0.000000	0.000000	0.0000
21	0.000144	0.000143	0.6784
22	0.000175	0.000174	0.6419
MAX error			6.3367

Table 3.12: Timoshenko Beam Element Verification-θ_y

NODE#	θ_y[rad] *ABAQUS*	θ_y[rad] *MATLAB*	%*error*
1	0.000537	0.000536	0.2559
2	0.000580	0.000578	0.2640
3	0.000622	0.000620	0.2712
4	0.000661	0.000659	0.2946
5	0.000665	0.000663	0.2917
6	0.000668	0.000666	0.2886
7	0.000659	0.000658	0.1824
8	0.000651	0.000650	0.1777
9	0.000643	0.000642	0.1730
10	0.000365	0.000365	0.0688
11	0.000230	0.000229	0.1246
12	0.000000	0.000000	0.0000
13	-0.000043	-0.000043	0.3953
14	-0.000085	-0.000085	0.3954
15	-0.000109	-0.000108	0.5028
16	-0.000102	-0.000101	0.4931
17	-0.000095	-0.000094	0.4817
18	-0.000063	-0.000063	0.4394
19	-0.000032	-0.000031	0.4394
20	0.000000	0.000000	0.0000
21	0.000305	0.000304	0.3072
22	0.000394	0.000393	0.2414
MAX error			0.5028

Table 3.13: Timoshenko Beam Element Verification-θ_z

NODE#	θ_z[rad] *ABAQUS*	θ_z[rad] *MATLAB*	%*error*
1	0.000351	0.000350	0.1448
2	0.000482	0.000481	0.1547
3	0.000501	0.000500	0.1939
4	0.000371	0.000370	0.3318
5	0.000180	0.000179	0.8014
6	-0.000050	-0.000051	3.3069
7	-0.000388	-0.000390	0.3366
8	-0.000639	-0.000641	0.2366
9	-0.000776	-0.000778	0.2190
10	-0.000802	-0.000804	0.1655
11	-0.000655	-0.000657	0.1679
12	0.000000	0.000000	0.0000
13	0.000150	0.000150	0.0564
14	0.000142	0.000141	0.1079
15	0.000057	0.000057	0.0393
16	-0.000041	-0.000041	0.1277
17	-0.000132	-0.000132	0.0541
18	-0.000167	-0.000167	0.0168
19	-0.000126	-0.000126	0.0154
20	0.000000	0.000000	0.0000
21	-0.000892	-0.000892	0.0322
22	-0.000488	-0.000488	0.0261
MAX error			3.3069

The worst %error occurs on node 4 for θ_x displacement as can be noted in table 3.11 on page 72. However, the value of the corresponding θ_x is just 9.66×10^{-6} and 9.04×10^{-6}. Therefore, the error is severely small and negligible. All in all, it is perfectly safe to say that our code is perfectly healthy.

Forth Model contains spring and the connecting element; with the new feature it is only safe to still check the results against the ABAQUS counterpart model. However, verifying all the nodes at every step against the ABAQUS one is a highly tedious task. As a result, five nodes that are more vital to the optimization- objective functions- are pointed out and selected for validation of each model. These five points are u_x of three pistons that are across the fixed side of caliper and the u_y of the two points that the pads insert lateral force on. These points are selected because they contain the high displacement values and are more sensitive from design point of view, in Figure 3.10 these points are pointed out by arrows. Thus, from now on, instead of comparing all nodes‘ values just 5 values will be certified as a representative of the all the nodes.

Forth Model (Figure 3.8 & 3.9) is going to be the base model of optimization process; hence, this model is very significant for the project. Although the code has been subjected to thorough inspection, the model is going to be certified against the counterpart ABAQUS model to assure that the results yielded in the MATLAB code are perfectly sound.

Table 3.14: Forth Model-Validation-Bernoulli Beam Element

NODE #	Displacement	ABAQUS	MATLAB	%error
2	x	-0.63272	-0.63272	3.16E-05
5	x	-0.02697	-0.02697	1.59E-02
8	x	0.57193	0.57193	2.80E-04
10	y	2.54505	2.54504	4.72E-04
12	y	2.11495	2.11494	2.84E-04

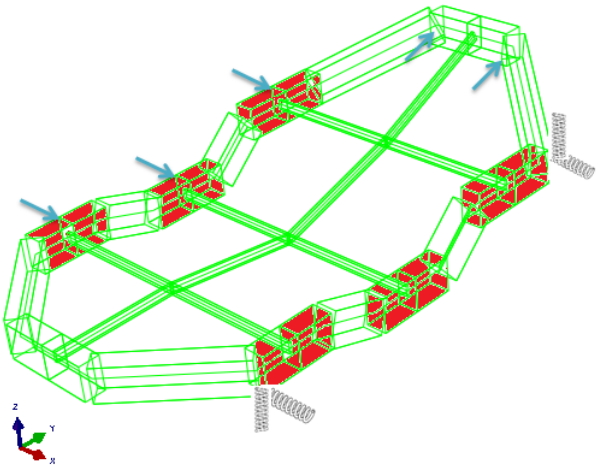


Figure 3.10: Forth Model- The observed points are pointed out by the arrows

Table 3.15: Forth Model-Validation-Timoshenko Beam Element

NODE #	Displacement	ABAQUS	MATLAB	%error
2	x	-0.07351	-0.07351	9.17E-03
5	x	-0.00937	-0.00934	3.89E-01
8	x	0.05747	0.05751	6.57E-02
10	y	0.32250	0.32296	1.41E-01
12	y	0.25667	0.25698	1.23E-01

3.4 Remarks

The accuracy of the results from MATLAB code are very pleasing, the worst error is about 0.1%. Therefore, we can conclude that the output is reliable. The FE code and simplified model is fully developed and verified; consequently, we can move to the next step, that is optimization. In that step real life values of force and spring stiffness are going to be put into action. The yield optimized cases are going to be compared against a real caliper to understand if a true progress has been made.

Chapter 4

Optimization

4.1 Introduction

So far, a simple mathematical model for a high performance racing vehicle caliper has been introduced, and a FE code was conducted in MATLAB to solve this (or any geometry with beam element) model. It has been completely certified against robust commercial software, ABAQUS, to warranty sound results. Now we are one step closer to the goal of this research project, which is optimization of design of a caliper. So we shall introduce an optimizer into the developed code. This optimizer would change the defined design variables, then handing them to the FE part of the code to calculate the objective functions. By finding the Pareto set of these objective functions, the optimized results would be known. The goal is to find new patterns in those optimized designs that can aid engineers in future designs and get higher performance from the caliper. Indeed, the optimized caliper behavior must be compared against a real caliper to have a sense if any improvement has been made.

To this end, in this chapter first a brief theoretical background of optimization is given. Then the objective functions and design variables are defined. The optimizer is added to the code and the code is put to a series of tests again and then tuned for clearer results. In the end the code is run to gain the final results.

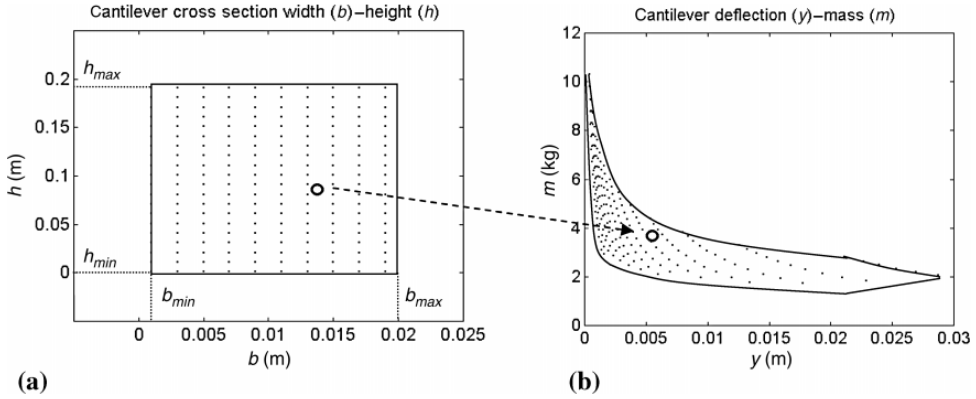


Figure 4.1: Design Variables and Objective Functions Domains. (**a**) A cantilever cross-section width b and height h that are considered for optimization. (**b**) Cantilever mass (m) and cantilever deflection at the free end y-Reprinted from [10]

4.2 Theoretical Background

4.2.1 Optimal Solution

Optimization is the definition of certain parameters to achieve the desired performance of a system, while considering a set of certain constrains [10]. The desired performances are always in contradiction; thereupon, it not possible to find one optimal solution to a problem that would minimize all objective functions contemporarily, as an example, making a structure stiffer and lighter. Removing mass would result in a lighter structure; however, at the same time, this would make the structure less stiff. These desired performances are named "objective functions" while the certain parameters that should be set are "design variables". All the possible solutions to a problem or all the combinations of design variables can be viewed in the "design variable domain"; furthermore, the resulted performances are depicted in the "objective function domain". Figure 4.1 depicts these two domains for optimization process of a cantilever beam to get a lighter and stiffer structure, by changing the cross section dimensions.

The chief question here is how to find the optimal solutions of all possible combinations of design variables and what an optimal solution is. The Pareto optimal set, is a set of solutions that contains all the optimal solutions, in theory is a set which contains all infinite solutions coming from minimization of vector of objective functions [10]. Pareto set can be connected or can have gaps.

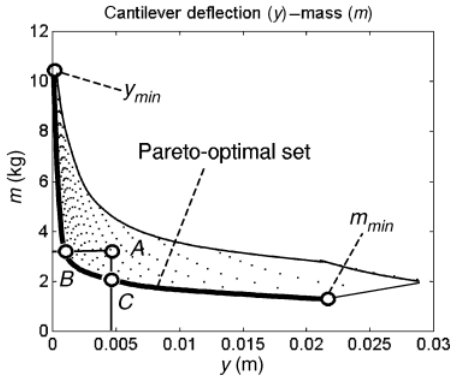


Figure 4.2: Definition of the Pareto-optimal set in the objective functions domain. Reprinted from [10]

If there are n_{dv} design variables and n_{ob} objective functions, then op_i is an optimal solution in Pareto set if and only if, one attempts to improve an objective function it would result in deterioration of another one; i.e., in order to improve one criterion, other criteria must get worse. For all other non-optimal solutions, at least one objective function can be reduced without increasing the other components [10]. This point is clearly illustrated in Figure 4.2, the Pareto set is the bold line. If we want to improve C from mass point of view, then displacement shall be increased, or vice versa. However, Point A is not an optimal one because if one decides to select this point as an optimal solution, why not selecting point C that lower mass and the exact displacement or point B that weights the same but it is stiffer. This means that one objective function is improving while the other one is still the same, not worsened. Hence, point A is clearly not an optimal point.

Since the Pareto set for more than one objective function is infinite (a line for 2D, a surface for 3D and so forth), and all the solutions are the eligible solutions, it is the task of engineer to select the best solution from his point of view.

Now that the definition of Pareto optimal solution is clear, it is obvious that there exist infinite solutions to investigate. By no means it is possible to cover all these solutions especially if the design variable vector is large. Years or thousands of years are needed to cover all solutions of a fairly simple engineering problem.

4.2.2 Methods of Generating and Obtaining Pareto-optimal set

Exhaustive Method

This method to find Pareto optimal set is the simplest one; however, this comes with a price, the method is highly unmanageable due to huge computations size required. In the method every possible combination of variables is assessed and minimized. if each design variable may assume n_v different values within its definition range, and design variable presented by n_{dv} then the number of all possible combinations is

$$n_v^{n_{dv}}$$

The volume of computations can be incredibly enormous; therefore, it is only useful when the system is very simple.

Consequently, it is only natural if we opt for not assessing all the solutions and only exploring some of the possible solutions.

Random Search

Exhaustive search considers every set of combinations of design parameters to construct the Pareto set; on the other hand, "Random search" selects random set of points and assesses them, then constructs the Pareto set based on minimization of the corresponding objective functions domain solution. In principle, the design variable values can be chosen according to a random sequence. Theoretically, it is not possible to generate a random sequence of points using a computer, which is a completely deterministic machine. In fact a deterministic procedure can generate only a pseudorandom sequence [10]. It sounds better than the Exhaustive method but still not perfect. Here the issue is that the selection of points in design domain is not disperse, in other words, they are not uniform (Figure 4.3); additionally, when adding points, they might be overlapping the regions that are already covered by previously existed points, and there exist regions that are not covered at all. This means that the generated Pareto-optimal set might miss important info.

Low Discrepancy Sequence

This method has been used in the present project and is one of the most efficient methods. The selection of points from the design parameter domain is much more uniform, Low discrepancy sequences differs from pseudo-random sequences due to the

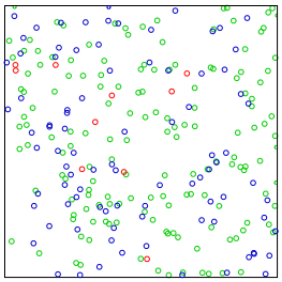


Figure 4.3: Random Search Design Domain with 256 points-(red=1,..,10, blue=11,..,100, green=101,..,256)-

fact that the points are more evenly distributed in the space [10]. Until recently uniform grid was the only way to select the points uniformly. However, a uniform grid requires a huge number of points.

Discrepancy is a quantitative measure for deviation of a sequence from a uniform distribution; i.e., measure of how evenly a given set of points is distributed in a given domain. Low Discrepancy sequence has the best uniformity known, the sequence can be constructed by N point and still be acceptable for practical applications. Figure 4.4 on the following page depicts an example with two different point sets. For comparison, 10000 elements of a sequence of pseudorandom points is plotted against the Low discrepancy one (Figure 4.5). The low discrepancy sequence shows evidently better uniformity characteristics.

Moreover, the uniformity of the low discrepancy sequence is independent of the number of points, that is even if N is small, the points are distributed evenly in the space [12]. Low discrepancy sequences are useful to explore the design variables space. However, being able to compute a function f at any given design variable vector does not imply that one comprehends the behavior of the function [5]. Evaluating f at a well-chosen set of locations $x_1, \cdots, x_i, \cdots, x_N$ by computing the responses $y_1, \cdots, y_i, \cdots, y_N$, the function f can be visualized. By plotting the responses versus the input variables, we may identify strong dependencies between x and y . The same procedure can be adopted to find correlations between two generic objective functions and between design variables and objective functions. We can use low discrepancy sequences to find points with desirable values of y that can be used to identify the most promising subregion of the design variables space [10].

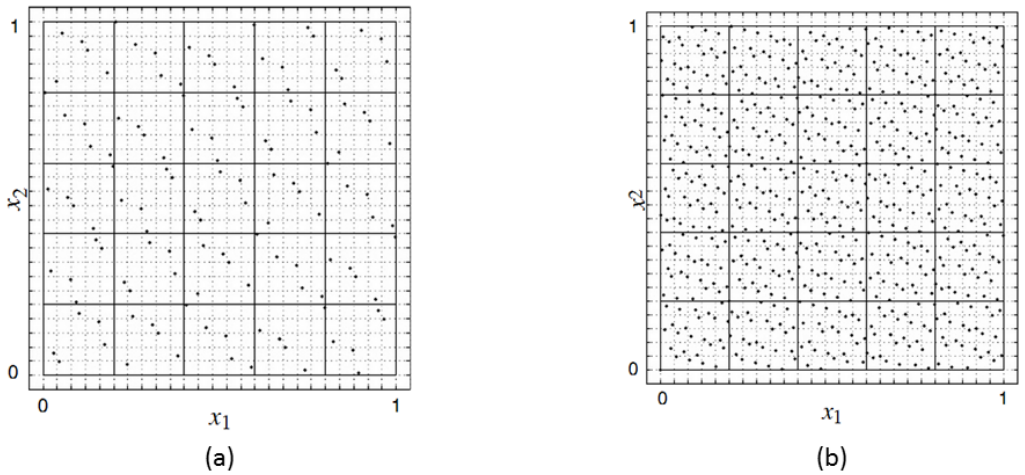


Figure 4.4: (**a**) 125 points of a low discrepancy (0,3,5)-net in base 5. For each two variables that are plotted, the results is a 5×5 grid of 5- point Latin hypercube samples. The points in a 3D projection can be split into 125 cubes having one point (**b**) 625 point of a low discrepancy (0,4,5)-net in base 5. For each of the two variables that are plotted, the square can be divided into 625 square of side 1/25 or into 625 rectangles of side $1/5 \times 1/5 \times 1/125$ and each square or rectangle has one of the points. Each variable is sampled once for all of the 625 one-dimensional subintervals. Reprinted from [10]

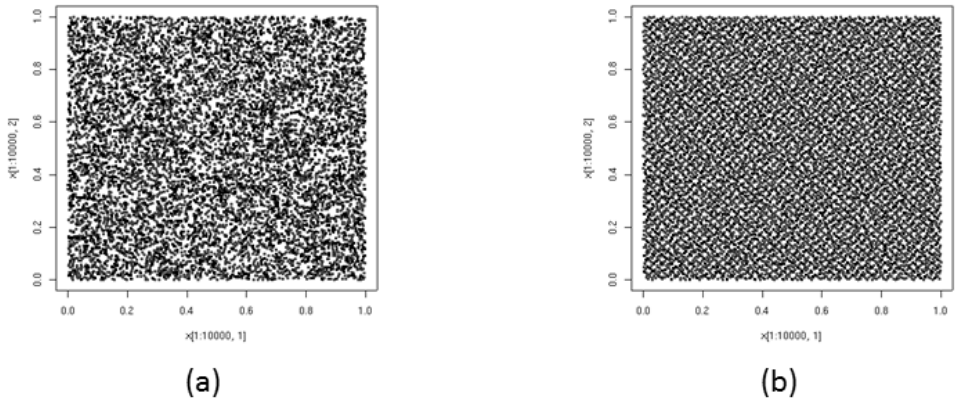


Figure 4.5: (**a**) Pseudorandom search with 10000 elements. (**b**) Low discrepancy sequence of the Sobol' type again 10000 elements

Weighted Sum

This method uses scalarization for gaining the Pareto-optimal set. The objective functions are weighted and then we shall minimize the weighted sum of objective functions for different weight settings. Note that the weights are not related to the importance of objective function but only they are factors that with their variation different solutions are obtained. Therefore, Pareto set is gained by altering the weighting function.

$$\min\{\lambda_1 f_1(x) + \lambda_2 f_2(x) + \cdots + \lambda_{n_{of}} f_{n_{of}}(x)\} \qquad \text{for} \quad x \in F, \tag{4.2.1}$$

$$0 \leqslant \lambda_i \leqslant 1 \qquad \sum_{i=1}^{n_{of}} \lambda_i = 1$$

It is noted that the objective functions are normalized. Changing the weights λ_i systematically would result in generation of points in Pareto-optimal set. Figure 4.6 on the next page depicts an optimization with two objective functions.

$$\phi(x) = \lambda_1 f_1(x) + \lambda_2 f_2 x$$

is the expression of a straight line in the objective function space, for different values of x, it is a family of lines. The solution of the scalar problem $\min_{x \in F} \phi(x)$ is a non-dominated solution for the original multi-objective problem. We consider multiple problems varying

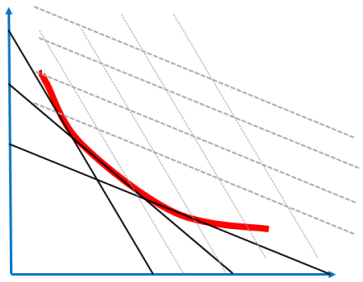


Figure 4.6: Weighted sum method, with bi-dimensional objective function domain. The dashed lines represent values that does not correspond to $\min_{x \in F} \phi(x)$

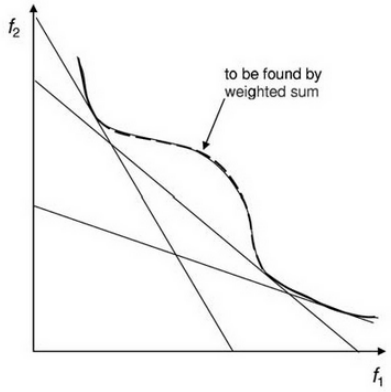


Figure 4.7: Weighted sum deficiency with two objective functions. The Pareto-optimal set being not convex, the weighted method fails to find the whole Pareto-optimal set- Reprinted from [10]

λ_1 and λ_2 to try construct the entire Pareto-optimal set. As Figure 4.6 shows each solution line is tangent to Pareto-optimal set. The limitation of the method is that the Pareto-optimal set should not be convex, the weighted method fails to find the whole Pareto-optimal set (Figure 4.7).

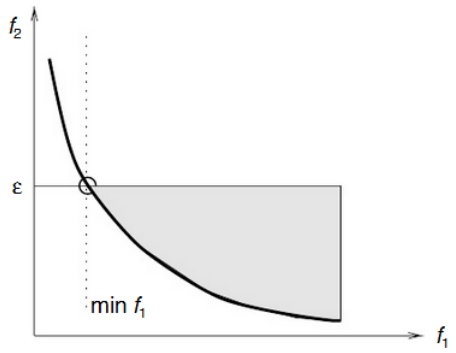


Figure 4.8: Constraints Method for 2D Objective Function Domain. Reprinted from [10]

Constraints Method

This method is considered to be one of the most effective techniques to compute the Pareto-optimal set. The problem is reformed somehow to get just one objective function and all the other objective functions are transformed to constraints. Then we just have to minimize that objective function.

$$\min_{x \in F} f_1(x) \tag{4.2.2}$$

$$f_2(x) \leqslant \epsilon_2 \quad f_3(x) \leqslant \epsilon_3 \quad \cdots \quad f_{n_{of}}(x) \leqslant \epsilon_{n_{of}}$$

The Pareto-optimal set is gained by altering the constraints ϵ_i for each objective function that acts as a constraint, and finding the solution for each level. If the problem is solved for all possible values of ϵ_i and the resulting solutions are unique, then these solutions constitute the entire Pareto-optimal set. If the solutions are not unique for some values of ϵ_i, then the Pareto points must be selected by direction comparison. For further explanation, we shall again consider the two dimensional objective function problem. Therefore

$$\min_{x \in F} f_1(x) \qquad f_2(x) \leqslant \epsilon$$

In Figure 4.8, ϵ value is set. The minimum value of f_1 is the bold line, that happens to be part of Pareto-optimal set, increasing ϵ the rest of the optimal solutions would start to appear.

Sorting of the Solutions

It is another highly efficient method to find the Pareto-optimal set. In the present work this method is implemented in the code. The method is based on re-sorting the

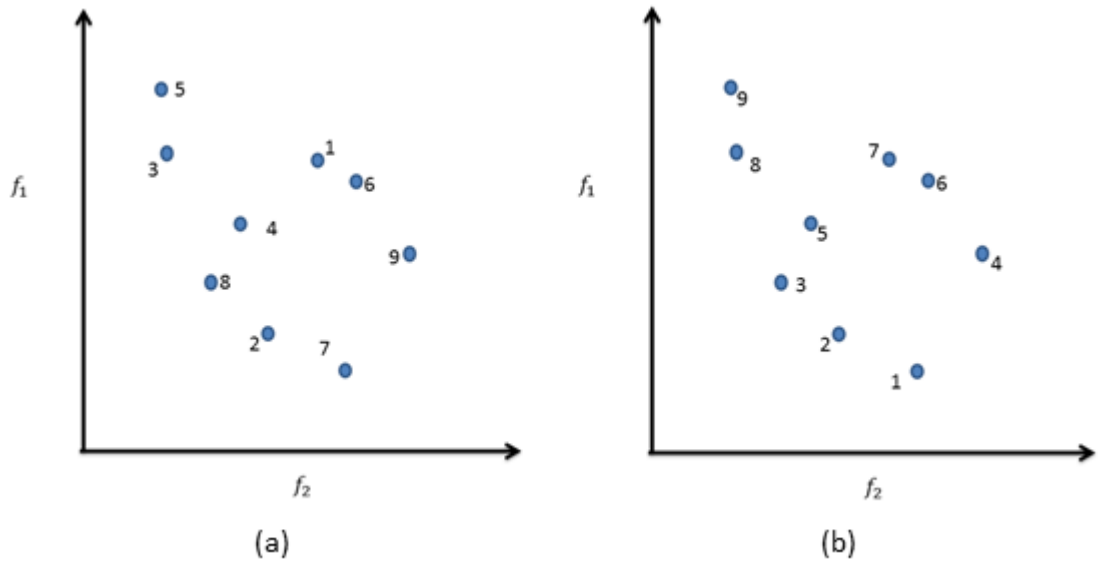


Figure 4.9: (**a**) The solutions unsorted in objective function domain. (**b**) The solutions sorted based on their f_1 function value

solutions with respect to one objective function. Then starting from the "lowest solution", we should mark or "Cross" those solutions that their value of other objective functions is higher than the corresponding objective function value of the "lowest solution". Then this process is repeated for the second lowest solution- however the previous point is left out of calculations- until all the solutions and all the objective functions have been gone through. Here to make things more clear an example is given, let us assume that the objective function domain is 2D and the solutions are nine solutions existing in the domain. Figure 4.9 exhibits the solutions first in original order, the first step is to order these solutions with respect to their f_1 value. In the second step, f_2 value of point 1 is considered, any point with f_2 value higher than one of point 1 is "crossed"(Figure 4.10(a)). This process is repeated for point 2 (Figure 4.10(b)), point 3 (Figure 4.10(c)) and so forth. In the end when all the points have been gone through the process, the "uncrossed" solutions are the Pareto-optimal set.

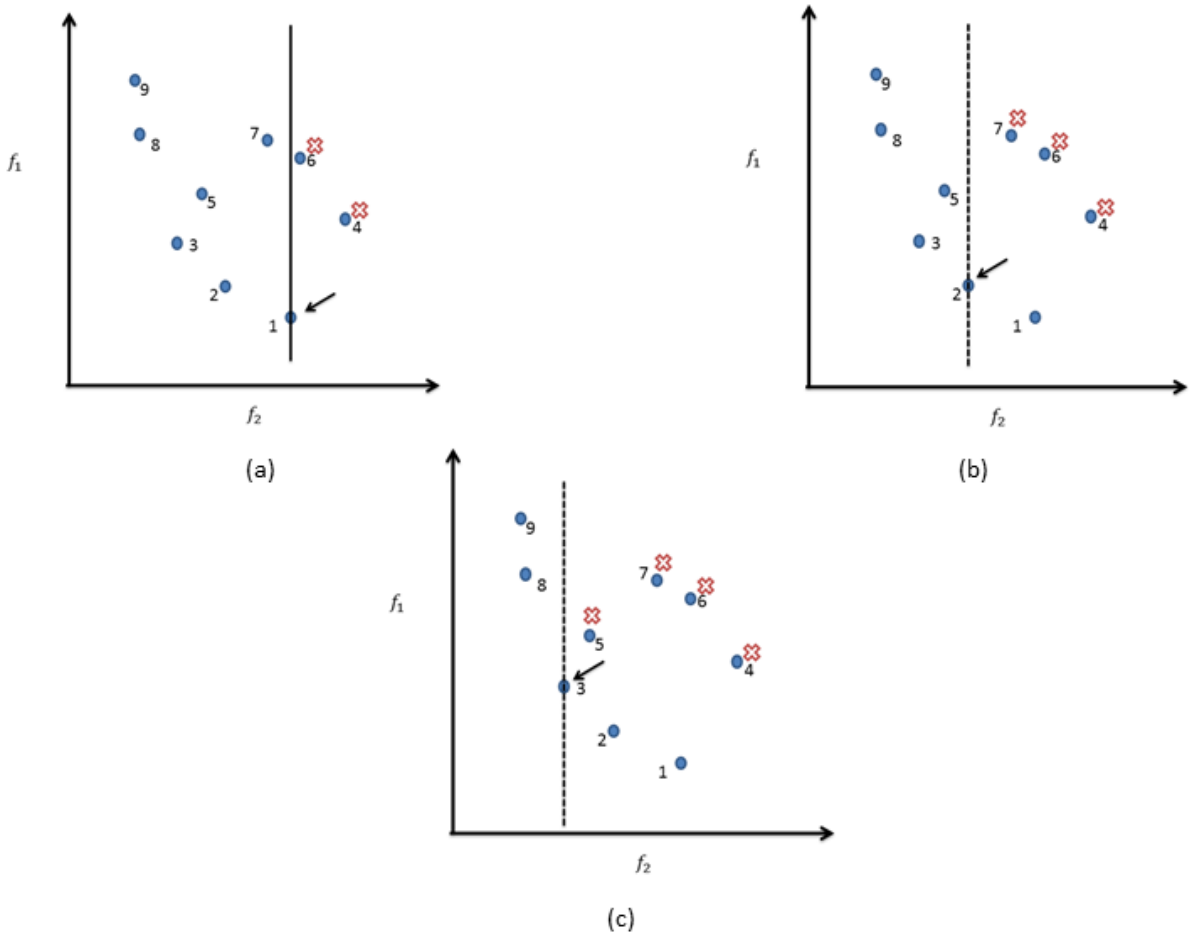


Figure 4.10: (**a**) Solutions are "crossed" based on point 1 (**b**) Solutions are "crossed" based on point 2. (**c**) Solutions are "crossed" based on point 3.

4.3 Arming the Code with Optimizer

For adding an optimizing ability to our code, the very first thing that should be done is to define the desired performances or objective functions, the manageable parameters or design variables, and the constraints. Following this, it should be decided which method is going to be used to select a certain combination of design variables from all possible solutions. A certain number of sets is going to be selected since as pointed out in 4.2.2 on page 81 Exhaustive search method that assesses all possible combination is just useful for very simple problems, but here as will be noted soon, even the simplified problem is quite complex. Selecting the right combinations then one should turn his attention on how to find the Pareto-optimal set from the corresponding the objective function values, in objective function domain.

4.3.1 Objective Function & Design Variables

Defining these two concepts has a huge impact on the final result and even the main course of conclusion drawn from the optimization process. The definition of objective function depends on the designer ability and of course the part requirements. In the present project, it is clear that the general aim is to reduce the weight of the caliper and increase its general stiffness.

Objective Function Vector

The weight can be considered quite straight forward. By assessing the volume of each element, summing these volumes over the entire structure, the total volume is gained. The density being fixed, one can easily know if mass is increasing or decreasing by observing the total volume. Therefore, total volume is set as of one the objective functions undisputedly.

On the other hand, investigating stiffness is a bit more complex and requires some decision making. For assessing the stiffness of the caliper under a certain fixed load configuration, displacement is a good representative. Obviously it not possible to consider all the nodes displacements in all d.o.f.; therefore, a certain point must be selected as indicator of the whole structure stiffness. Hence, we should decide how many nodes, which nodes, and which d.o.f to select for indicating the stiffness of our structure. To this end, it is a good rule of thumb to go with the points that are directly under the loads, or would have high displacement, away from constraints. In the simplified model

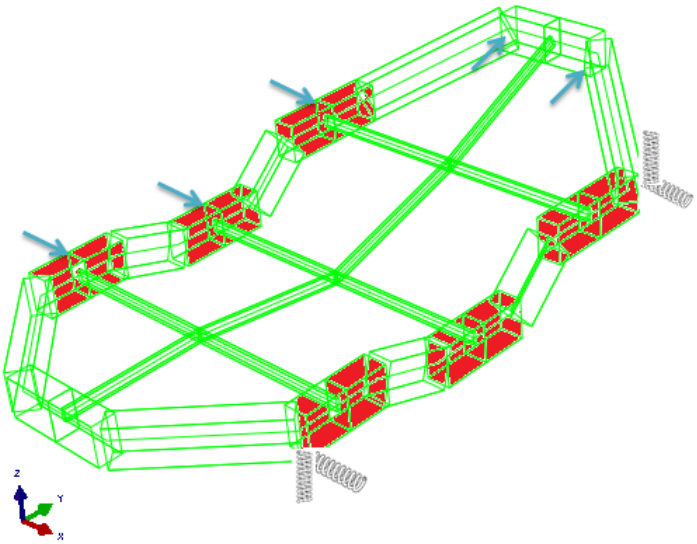


Figure 4.11: The nodes that their displacements construct the objective function vector are pointed out by arrows. Three of these arrows point to X direction and two of them to Y. The pistons are colored in red. The springs emphasize that here hub carrier compliance is taken into consideration and has been modeled by springs.

as exhibited in Figure 4.11, five node are selected. Above that, the d.o.f. that has the same direction as the applied load on that node is going to be the objective function. This can be a fair and solid selection of objective functions vector. As it will be noted in the subsequent section even this very simple selection of objective function vector is not that tangible, and it might be necessary to make it even simpler by summing all these displacement objective functions as one value.

$$Objective\ Fucntion = \begin{bmatrix} u_{2_x} \\ u_{5_x} \\ u_{8_x} \\ u_{10_y} \\ u_{12_y} \\ volume \end{bmatrix}$$

Design Variables Vector

More challenging than selection of objective functions is the definition of design variable that can change the main course of problem solution. They are actually our means of control over the possible solutions. Fixing one the design variables or adding one can change the results to a problem dramatically. As an aid in defining these variables let us first specify what is actually fixed in the design of the caliper.

As pointed out in chapter 2 the positions and the diameters of the pistons are fixed ("non-design domain"). Therefore, the nodes coordinates that model the cylinders cannot be altered. In figure 4.12 on the next page nodes positions 1, 2, 3, 4, 5, 6, 7, 8, 9, 13, 14, 15, 16, 17, 18, 19, 20 and 21. Plus the positions of nodes 25, 26 and 27 are also fixed. That leaves us with the positions of nodes 10, 11, 12, 22, 23 and 24. For the sake of simplicity node 11 and node 23 is considered to be the mean value of node 10 and 11, 22 and 24 respectively. This way without having a major impact on the final results the number of design variables is reduced; which helps in managing the final results and having a better sense over them.

Considering that the collecting node element has fixed dimensions, they will be completely excluded from design variables. After considering these assumptions and facts, the remaining possible design variables are the cross section dimensions of the beams plus the node position of nodes 10, 12, 22 and 24. Elements 10 and 11 actually represent one piece of the caliper, so we can assign the same cross section for them. By comparing Figure 4.12 and Figure 4.11, it can be noted that elements $1, 2-4, 5-7, 8-$

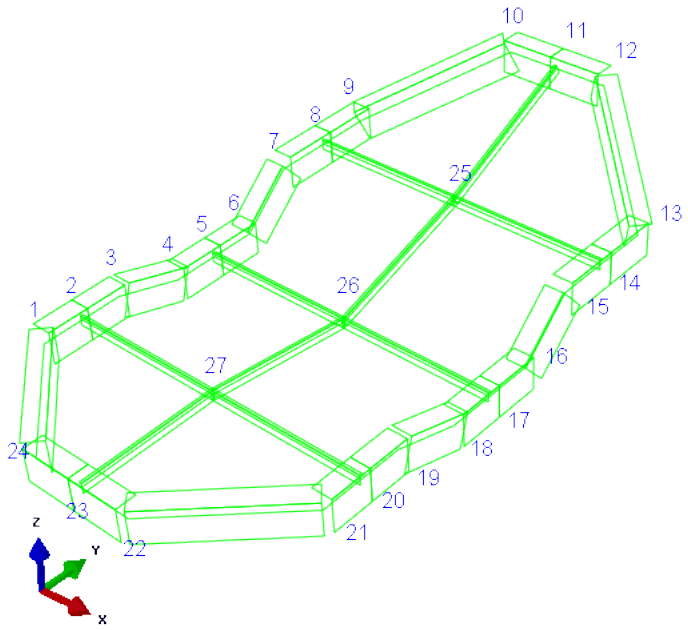


Figure 4.12: Modeled Frame with Node Numbers

$13, 14-16, 18-19, 20$, each group represents one piston, so they cannot have different cross section parameters. Therefore; for each group one cross section is assigned. After all these simplifications and realizations

$$\textit{Total number of Design Variable} = 16 \times 2 + 4 \times 3 = 44$$

where 16 is the number of elements with variable cross section, 2 is the cross section dimensions, 4 is the nodes 10, 12, 22 and 24, and 3 is their coordinate in space. Although, severely simplified, still 44 is a very large design variable number. Later it will be explained how this is again simplified for smaller vector size.

Design variables has been defined; however, still the ranges over which these values can vary should be determined. The ranges are evaluated first by assessing the design domain space, and considering the normal range of values possible. Later based on the results of preliminary optimization they have been refined and re-tuned to gain fine results and exploit the run samples more efficiently.

4.3.2 Setting Up the Code

Up to this point a FE code has been developed that can account for the displacements and stresses in a structure with any shape and configuration. Now the code is going to have the capability to perform optimization as well. The FE code plus the optimizer is depicted in flow chart of figure 4.13 on the following page. The added processes will be discussed in this part of the report.

Reading Data

The meaning of design variables and which factors of the frame are going to serve for that end was discussed. For each cross section and node coordinate the ranges that their value can vary within is determined and dictated to the code. This range should take good advantage of the whole design domain. After this, the number of samples or solutions that we want to generate should be indicated. This corresponds to the number of samples in the design variable domain that will be extracted from all the possible combinations. For preliminary stages this value is around 20,000 samples, for mid-study is about 100,000 samples and in the final run 10,005,000 samples is selected.

Sobol Sequence

Having selected a number of samples and design variables, using MATLAB for creating Sobol grid we are in great luck, because MATLAB makes it truly easy to find the grid, using only two commands as printed below. It should be reminded that Sobol search method is a low discrepancy sequence that makes it possible to explore our grid in a uniform way without leaving unexplored regions. This is well explained in section 4.2.2.

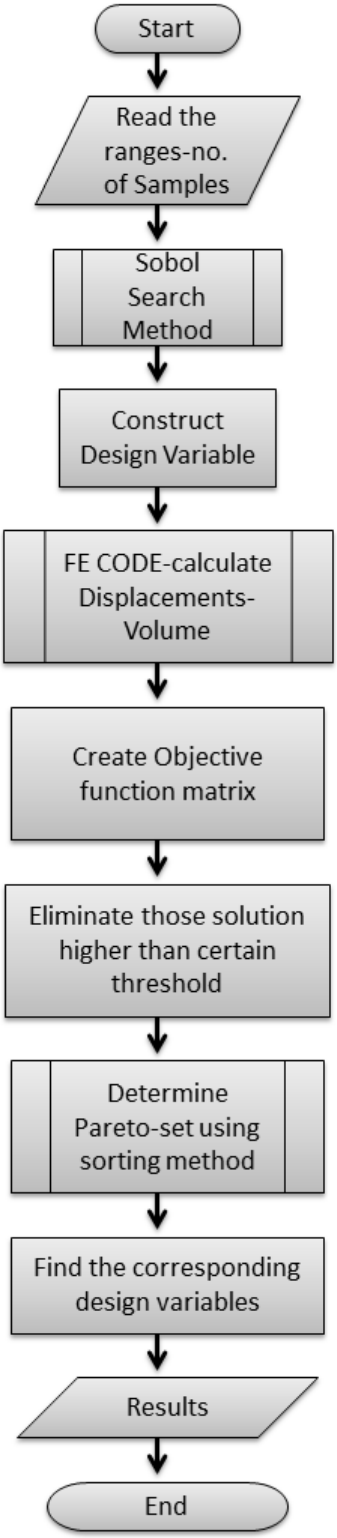
```
SO=sobolset(number_of_design_variables);
Data=net(SO,number_of_samples); % creating the grid
```

For more clarification, if 1000 samples and 48 design variables is selected, then

$$\textit{Formed gird} = \mathtt{Data}_{1000\times 48}$$

This will be mixed with other data that are fixed, like the piston‘s node position, to fully describe the caliper. From this point the cycle would be identical to the one of chapter 2. It should be noted that, the matrix “Data” contains values between 0 and 1. So for each design variable the values should be de-normalized.

Figure 4.13: Entire Code OverView



```
Design_variable1 = min_value+Data(:,1)*(max_value-min_value)

Design_variable2 = min_value+Data(:,2)*(max_value-min_value)
```

And so forth for all forty eight design variables. `min_value` and `max_value` are the minimum and maximum values in the range bracket for each design variable. Then these de-normalized values should be inserted in the appropriate position of their corresponding matrix, e.g., the node coordinates of node number 10, should be the inserted in the node coordinate matrix in the tenth row.

$$\text{node coordinates} = \begin{bmatrix} x_1 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_{10} & y_{10} & z_{10} \\ \vdots & \vdots & \vdots \\ \\ x_{27} & y_{27} & z_{27} \end{bmatrix}$$

The same goes for the cross sections.

Objective Function Matrix

After constructing design variables and inserting them in the FE code, the next steps are identical to the ones of chapter 2. Following that, the objected functions for each solution should be extracted from the outputs of FE stage. Five of them are extracted from displacements and the last one is the volume is that gained from the cross sections and node positions. All objective functions are inserted in one matrix where each row represents one solution and each column would be an objective function, e.g., for a thousand samples and six objective function

$$\text{Objective Function Matrix} = \begin{bmatrix} ob_1 & ob_2 & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}_{1000\times 6}$$

Elimination round

The chief aim of the search project is to generate a caliper design that is more efficient that the current designs. However, the simplified caliper actually is a frame constructed with beam elements; therefore, the weight of the caliper would be higher than an actual

one, still this does mean that finding trends in the design might not hold for a refined built version.

To make sure that the only solutions with a better behavior than the original caliper would be studied, an elimination round is added to process. This elimination round can be regarded as extra set of constraints. Acquiring the actual data of displacement of the caliper under the same load configuration and B.C., the solutions that have higher displacements than those are omitted from the solution pool. Indeed, these displacements are the same as our objective functions.

Each objective function of all solutions is compared against the original caliper value, and if even one of six objective functions is higher, the solution is eliminated, the associated design variables that have generated this objective function is also omitted. This way, just those solutions would make it to Pareto-optimal search stage that have better objective functions than the original caliper. As an example

```
counter= true(number_of_samples,1);
for i=1:(number_of_samples)
if abs(OBJ_F(i,5))>0.046
    counter(i)=0;
    end
end

OBJ_F=OBJ_F(counter,:); % Corrected OF domain by constrains
D_V=D_V(counter,:); % Corrected design variable
```

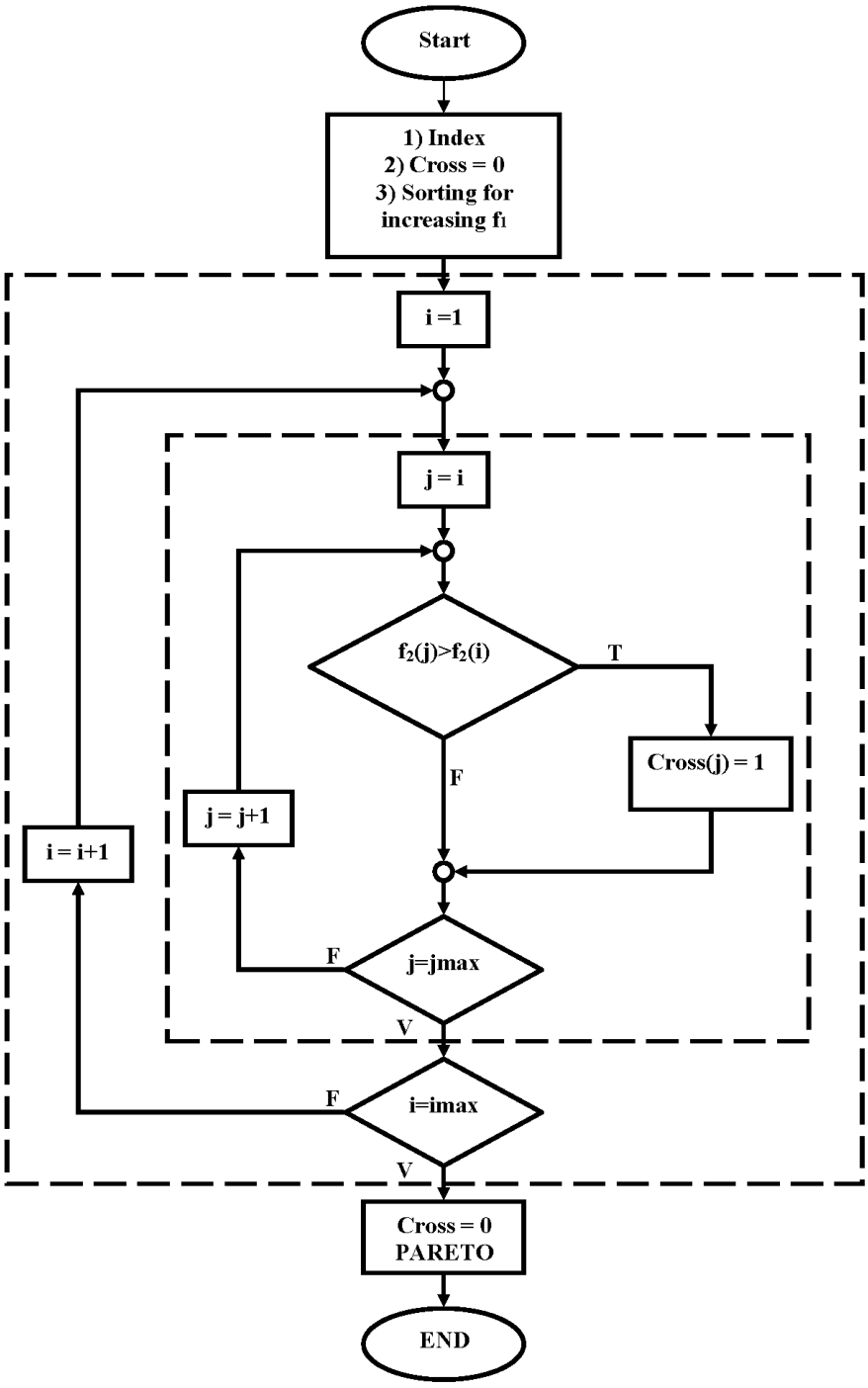
Bear in mind, due to the fact that bulky beam elements are being handled here, a volume with almost two times the original one is defined as constraint. Next step is finding the Pareto-optimal solutions.

Pareto-optimal Set

In section 4.2.2 some methods were explained that would result in effective determination of Pareto-optimal set. In the code the “sorting method” is chosen for finding the Pareto-optimal set. In that section the method is well presented, in Figure 4.14 a clear flowchart of how the actual code is searched for the optimal set is shown.

Knowing the Pareto-optimal set, the respective design variables are extracted. These

Figure 4.14: Sorting Flow Chart



results are printed; the corresponding frames are plotted and studied closely. From those we will strive to obtain a trend in the solutions that can give a general conclusion about how to change the designs for better results.

4.4 Preliminary Results

Here the results must be validated again, but there is no software that is able to check the results just like the FE part of the code. To know if the optimizer is working properly, the objective function domain should be plotted. However; it is not possible to visualize a 6D domain. Consequently, the displacement objective functions are summed up as one objective function, and the domain is plotted again volume objective function.

$$\textit{Simplfied objective function}_1 = \sum_{i=1}^{5} |\mathtt{obf}_i|$$

$$\textit{Simplfied objective function}_2 = \sum_{i=1}^{el} \rho \times V_i = \mathtt{obf}_6$$

where the first four objective functions are the displacements and $\mathtt{el}$ is the total number of elements.

The results are plotted for 15,000 solutions in figure 4.15 on the next page. Two of the Pareto-optimal solutions are chosen and plotted, plus the whole set is studied to understand how to refine the design variable ranges. For the first trial the springs are deactivated and the connecting elements are not considered.

As can be noted from Figure 4.15 "sorting" method is working correctly.

$$\textit{Objective Fucntion} = \begin{bmatrix} u_{2_x} \\ u_{5_x} \\ u_{8_x} \\ u_{10_y} \\ u_{12_y} \\ volume \end{bmatrix}$$

Note that in the following tables those are the values of main objective functions and not the "simplified objective functions". However, the Pareto-set was calculated based on the summation of main objective function.

Figure 4.15: 2D objective Function Domain with 15000 samples- Two cases Pointed out to be furthered study.-top view

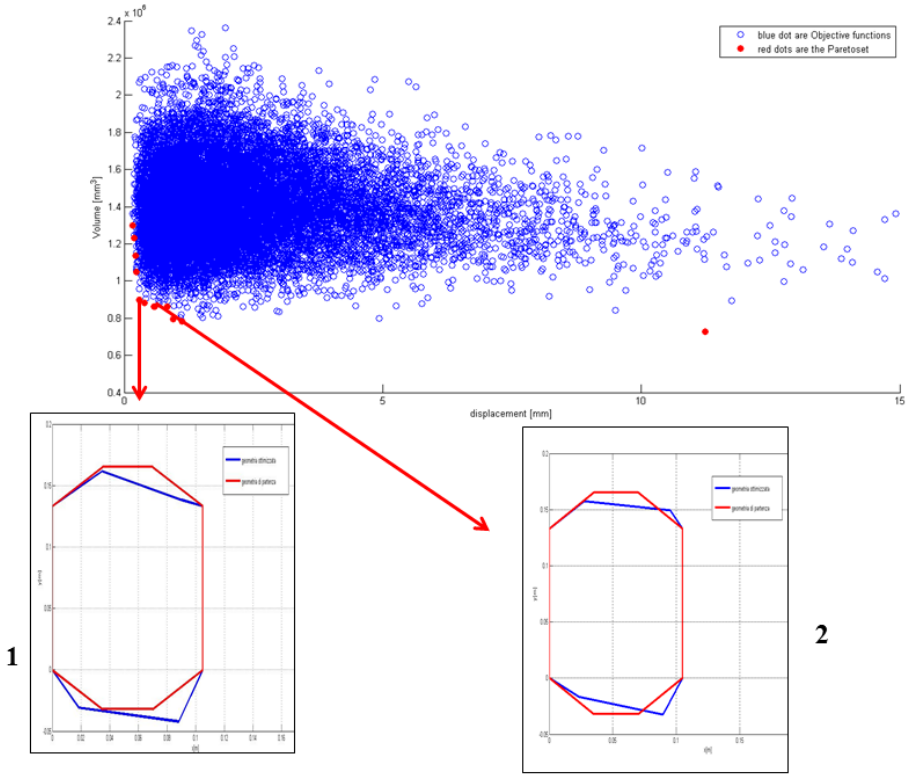
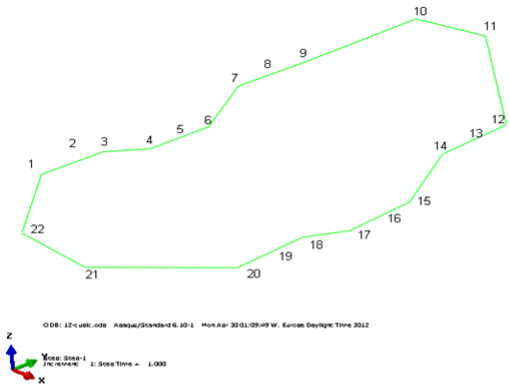


Table 4.1: Case 1- For element numbers refer to fig: 4.17

no. elemento	$J_{xx}[mm^4]$	$J_{zz}[mm^4]$	$area[mm^2]$
1	699051	174763	2048
2	121945	522000	1740
3	169189	160401	1406
4	254248	51710	1173
5	277333	468693	2080
6	86848	67070	957
7	51750	30418	690
8	240000	303750	1800

Figure 4.16: Preliminary Ranges



Nodo 10	Nodo 11	Nodo 21	Nodo 22
x = 0 ÷ 36 mm	x = 69 ÷ 104 mm	x = 69 ÷ 104 mm	x = 0 ÷ 36 mm
y = 134 ÷ 220 mm	y = 134 ÷ 220 mm	y = -85 ÷ 0 mm	y = -85 ÷ 0 mm
z = 10 ÷ 70 mm	z = 10 ÷ 70 mm	z = 10 ÷ 70 mm	z = 10 ÷ 70 mm

Table 4.2: Case 1 Objective Function Values

Mass [kg]	obf_1	obf_2	obf_3	obf_4	obf_5
2.33	0.13	0.24	0.13	0.07	0.004

Table 4.3: Case 2 Objective Function Values

Mass [kg]	obf_1	obf_2	obf_3	obf_4	obf_5
2.32	0.23	0.34	0.14	0.1	0.008

Figure 4.17: (1)Case 1. (2)Case 2. (0)The symmetric case

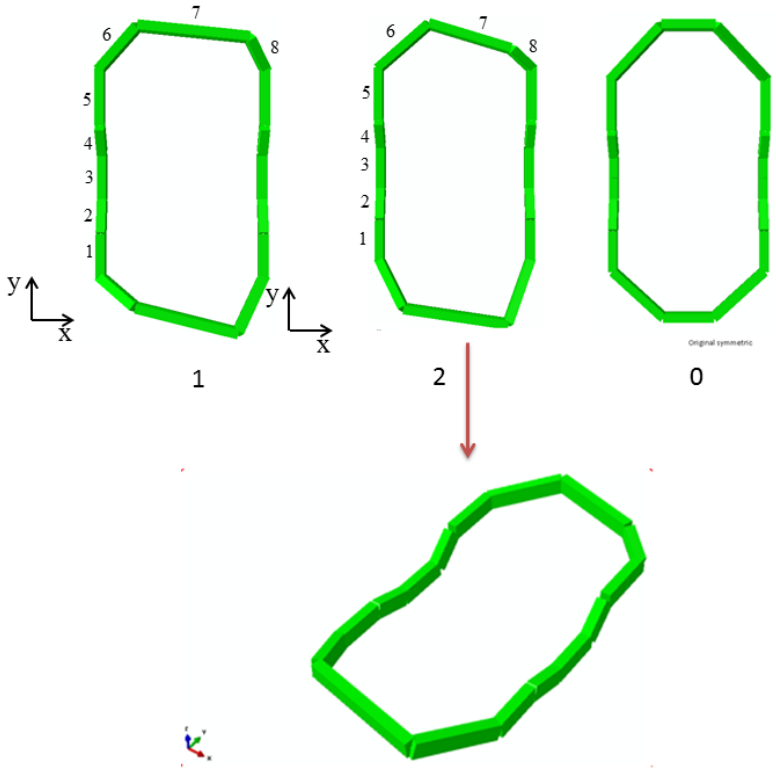


Table 4.4: Case 2- For element numbers refer to fig: 4.17

no. elemento	$J_{xx}[mm^4]$	$J_{zz}[mm^4]$	$area[mm^2]$
1	468693	277333	2080
2	130854	91856	1147
3	221616	555579	2052
4	1180929	1615169	4071
5	1095355	1288875	3776
6	32667	196082	980
7	132512	28667	860
8	168843	197333	1480

Based on these results, the following conclusions can be made:

- The code is healthy and the Pareto searching function is working properly.
- The first observation is that optimized frames are not symmetrical anymore. For reassurance the two observed cases are constructed in ABAQUS and the same results are yielded, therefore there might be some pattern here.
- The optimized design variable ranges are investigated to realize which the appropriate range for each design variable is.
- The spring and connecting elements now can be added to the model.
- Studying the Pareto-set these cases intrigued some ideas on how to downsize the design variable number.

4.5 Improving the Model

After these observations, attempts for enhancing are commenced.

- The hub carrier effect is included in the model, and the connecting elements are added. With this, the results are one step closer to the ones of actual caliper, and direct comparison makes more sense.
- The model is revived back to 6D objective function domain, taking into consideration all six objective functions.
- The preliminary observation of the code, showed that a better result would yield if the Z value of the nodes 10, 12, 22 and 24 in figure 4.12 on page 92 are identical. Furthermore, designing a caliper means that it should be manufactured. Manufacturability is a big factor on deciding if a design is superior to another one. Selecting equal Z values would definitely help the manufacturability of the product. Plus, equating their Z value would decrease the number of design variables by three.
- Another great opportunity to downsize the design variable number is eliminating the cross section design variables of elements between two pistons. The values of the cross sections are adapted from the adjacent “piston element” dimensions. Looking at Figure 4.18 would shed light on the matter.

Figure 4.18: Elements With the Same Cross Sections Have the Same Color.

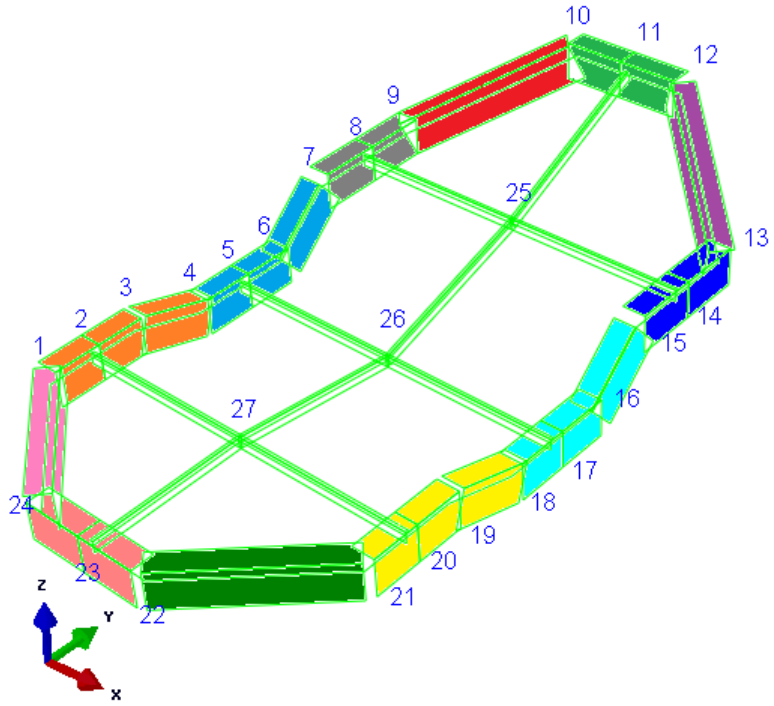
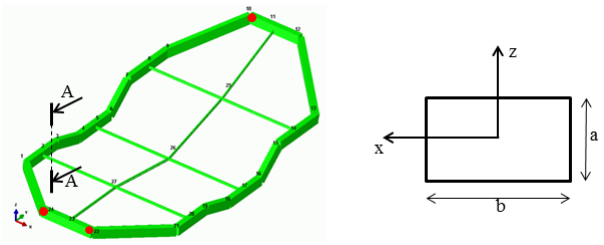


Figure 4.19: Elements Cross Section Nomenclature in Global Coordinate



This way the new number of design variables is

$$Number\ of\ Design\ Variables = 12 \times 2 + 4 \times 2 + 1 = 33$$

Where 12 is the different colors in Figure 4.18. 4 is the number of points, and 2 is X, Y d.o.f. and 1 is Z d.o.f. in the global coordinate.

- The constraints, which would eliminate the solutions with values higher than them, are set to be the original caliper displacement under the same load configuration.
- Code was run for multiple times with sample value of 20,000 assessing the output Pareto-set, design variables ranges and objective functions. The design variable ranges that do not seem to be participating in the optimal solutions are cut off for better exploitation of sample numbers. The new design variable ranges are printed in tables 4.5 to 4.10. Note that the reference is set at node 1 (Figure 4.18).

At this stage the model is mature enough. It is possible to run the code for higher number of samples and have some impressions about what the optimized results would suggest. To this end, the code is run with 100,000 samples.

4.6 Trials and Results

Solving the code for 100,000 different caliper configurations, the Pareto-optimal set is gained and studied closely. Pareto-optimal set has the population of 93. Therefore, the first conclusion that can be drawn from the simulation is that a much greater number of samples is needed to great a robust Pareto-optimal set population.

The population is studied closely, and based on the general shape, and objective functions‘ values are sorted from “more interesting” to “less interesting”. Here because we are not following a specific design aim, all the objective functions are equally valuable for us.

Table 4.5: Ranges for "a"

Elements	**Min**[mm]	**Max**[mm]
1,2,3	30	65
4,5,6	32	65
7,8	38	70
9	25	65
10,11	20	65
12	25	65
13,14	38	70
15,16,17	32	65
18,19,20	30	65
21	20	65
22,23	20	65
24	20	65

Table 4.6: Ranges for "b"

Elements	**Min**[mm]	**Max**[mm]
1,2,3	20	65
4,5,6	20	65
7,8	25	65
9	17	65
10,11	17	65
12	17	65
13,14	25	65
15,16,17	20	65
18,19,20	20	65
21	17	60
22,23	17	60
24	17	60

Table 4.7: Node 10 Ranges in [mm]

	MIN	MAX
X	10	36
Y	150	220
Z	15	42

Table 4.8: Node 12 Ranges in [mm]

	MIN	MAX
X	69	94
Y	150	220

Table 4.9: Node 22 Ranges in [mm]

	MIN	MAX
X	69	94
Y	-85	-15

Table 4.10: Node 24 Ranges in [mm]

	MIN	MAX
X	10	36
Y	-85	-15

In the case a specific aspect of the caliper was of more interest, then that corresponding objective function would have higher impact on the selection of "interesting cases". Therefore, those solutions that have a moderate distance from the threshold objective function values are the more interesting ones. Some of these interesting cases are selected for further investigations.

Although the code has been subjected to numerous verification during the project, here after selecting a few solutions, the MATLAB results are validated against ABAQUS ones, printed in Table 4.11. Notice that cases 18, 1, 2 and 19 are just some trivial choices in the optimal solutions that are viewed and investigated to give us better impression of the result's meaning.

Table 4.11: Comparison of results of a case generated by the optimizing code with the counterpart in ABAQUS. The original values of the caliper displacement are printed to exhibit the improvements

CASE	Original	Case 18	Case 18
	N\A	Matlab	Abaqus
$\mathbf{obf_4}$[mm]	0.9850	0.6246	0.6240
$\mathbf{obf_5}$[mm]	0.9850	0.4482	0.4479
$\mathbf{obf_1}$[mm]	-0.2576	-0.1115	-0.1114
$\mathbf{obf_2}$[mm]	-0.1679	-0.0986	-0.0986
$\mathbf{obf_3}$[mm]	-0.0456	-0.0397	-0.0397
$\mathbf{obf_6}$[mm^3]	5.19E+05	854785.5	8.55E+05
Mass[kg]	1.4	2.31	2.31

Assessing these cases reveals that there is not a perfect sense of symmetry in the optimized design. To further explore this, some other solutions are studied. Therefore the new idea is, if asymmetric configurations in the Pareto-optimal set always yield better results that symmetric ones under the same load configuration. For further testing this idea, the associated symmetric case is generated and the objective functions are compared. The associated symmetric case means that for each case all the cross section data are extracted and inserted in the built model; however, the node positions are maintained from the original design.

A close look at all these cases, reveals that in some cases the symmetric configuration yields a better result and in other cases the asymmetric one holds a pleasing outcome; while there are some cases with identical performances. For instance, in "case 18" symmetric design has lower $\mathtt{obf}_1$ and $\mathtt{obf}_2$; $\mathtt{obf}_4$ and $\mathtt{obf}_6$ are almost identical while $\mathtt{obf}_3$

Figure 4.20: Case 18 Top View- B.C. located on the right side

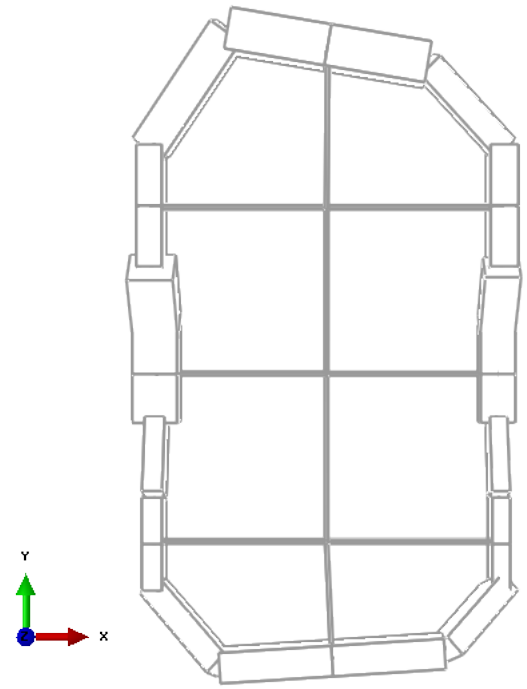


Figure 4.21: Case 18- B.C. located on the right side

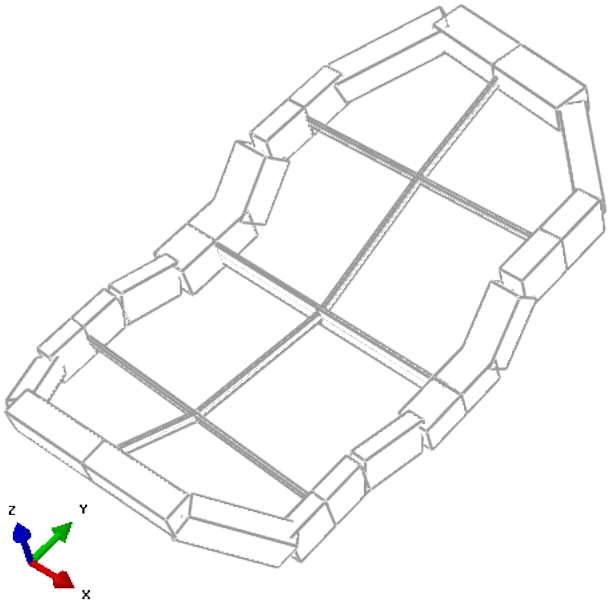


Figure 4.22: Case 18 Side View

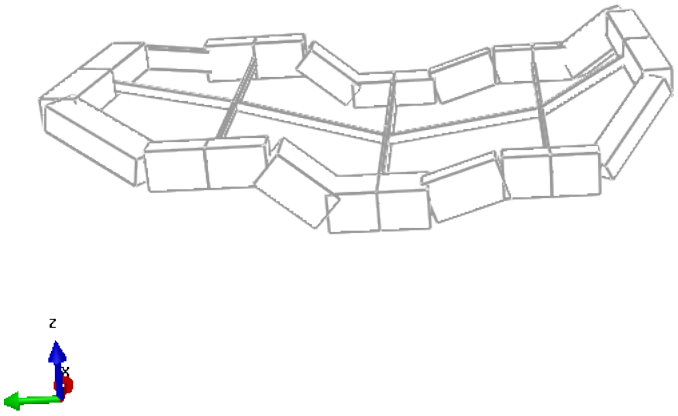


Table 4.12: Comparison of Case 18 Against Symmetric Version

	Case 18	Case 18	Symmetric
	Matlab	Abaqus	Abaqus
$\mathbf{obf_4}[mm]$	0.6246	0.6240	0.0632
$\mathbf{obf_5}[mm]$	0.4482	0.4479	0.5240
$\mathbf{obf_1}[mm]$	-0.1115	-0.1114	-0.0900
$\mathbf{obf_2}[mm]$	-0.0986	-0.0986	-0.0850
$\mathbf{obf_3}[mm]$	-0.0397	-0.0397	-0.0420
$\mathbf{obf_6}[mm^3]$	854785.5	8.55E+05	8.45E+05
Mass	2.31	2.31	2.3

Table 4.13: Case 1 Comparison with Symmetric Case

	Case 1	Case 1	Symmetric
	Matlab	Abaqus	Matlab
$\mathbf{obf_4}[\mathbf{mm}]$	0.4664	0.4659	0.4275
$\mathbf{obf_5}[\mathbf{mm}]$	0.4066	0.4064	0.4710
$\mathbf{obf_1}[\mathbf{mm}]$	-0.0909	-0.0909	-0.0831
$\mathbf{obf_2}[\mathbf{mm}]$	-0.0829	-0.0828	-0.0796
$\mathbf{obf_3}[\mathbf{mm}]$	-0.0377	-0.0376	-0.0449
$\mathbf{obf_6}[\mathbf{mm^3}]$	1020768	1.02E+06	1037527
$\mathbf{Mass}[\mathbf{kg}]$	2.76	2.76	2.81

Figure 4.23: Case 1 Side View- B.C. located on the right side

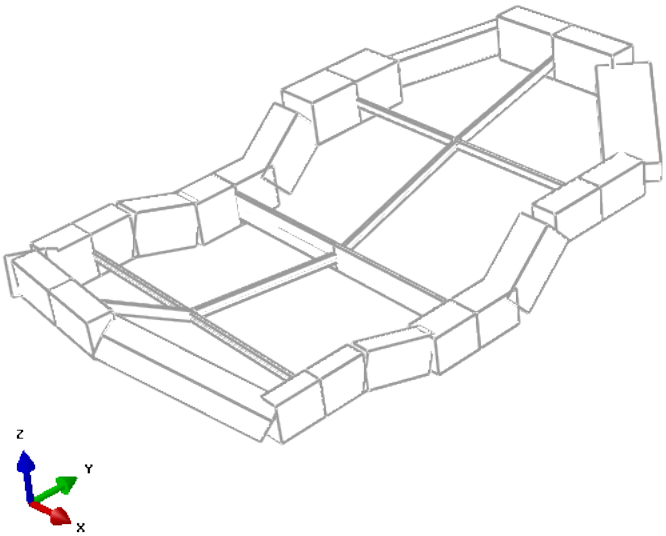


Figure 4.24: Case 1 Top View- B.C. located on the right side

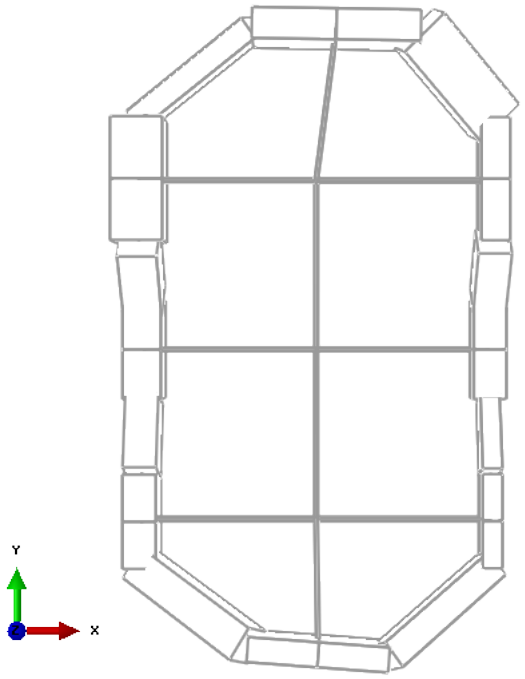


Table 4.14: Case 2 Comparison with Symmetric Case

	Case2	Symmetric
	Matlab	Matlab
$\mathbf{obf_4}[\mathbf{mm}]$	0.4670	0.4155
$\mathbf{obf_5}[\mathbf{mm}]$	0.3877	0.4559
$\mathbf{obf_1}[\mathbf{mm}]$	-0.0639	-0.0645
$\mathbf{obf_2}[\mathbf{mm}]$	-0.0708	-0.0810
$\mathbf{obf_3}[\mathbf{mm}]$	-0.0364	-0.0529
$\mathbf{obf_6}[\mathbf{mm^3}]$	934196.2	961292
$\mathbf{Mass}[\mathrm{kg}]$	2.52	2.6

Figure 4.25: Case 2 Top View- B.C. located on the right side

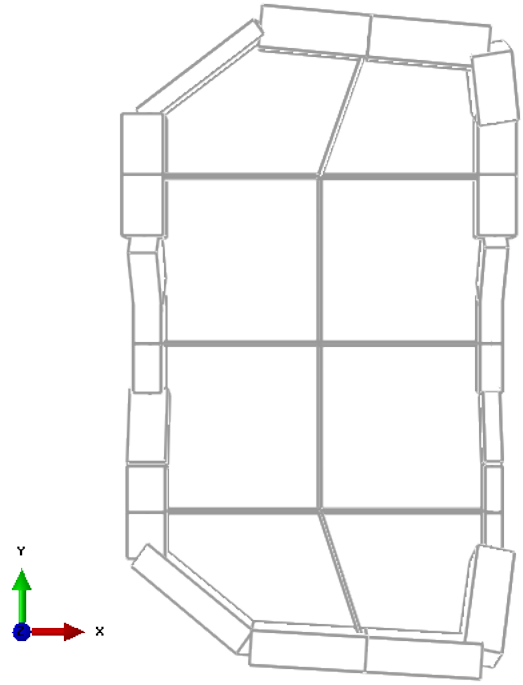


Figure 4.26: Case 2 Side View- B.C. located on the right side

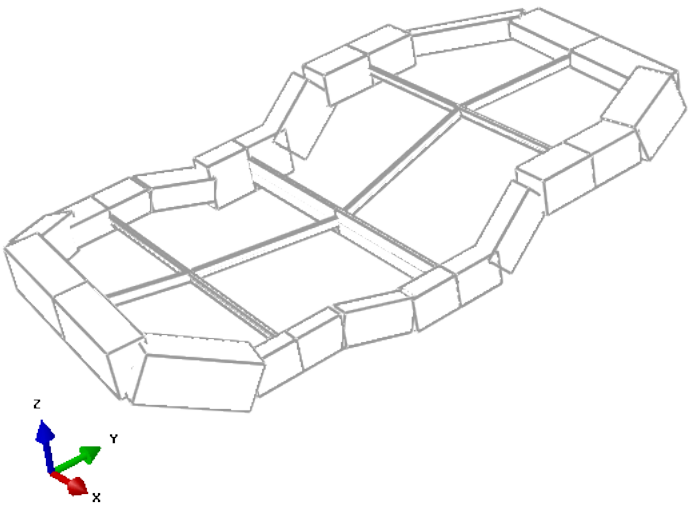
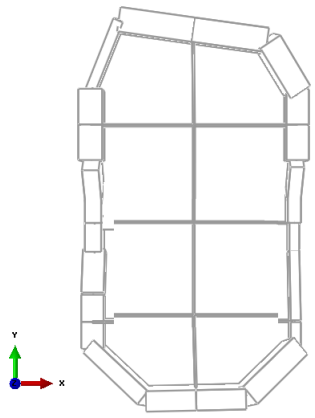


Table 4.15: Case 19 Comparison with Symmetric Case

	Case19	Symmetric
	Matlab	Matlab
$\mathbf{obf_4}[\mathrm{mm}]$	0.6299	0.4299
$\mathbf{obf_5}[\mathrm{mm}]$	0.3890	0.5188
$\mathbf{obf_1}[\mathrm{mm}]$	-0.1339	-0.0991
$\mathbf{obf_2}[\mathrm{mm}]$	-0.1036	-0.1005
$\mathbf{obf_3}[\mathrm{mm}]$	-0.0357	-0.0531
$\mathbf{obf_6}[\mathrm{mm^3}]$	994952.7	1077497
$\mathbf{Mass}[\mathrm{kg}]$	2.69	2.91

Figure 4.27: Case 19 Top View- B.C. located on the right side



and $\texttt{obf}_5$ are higher. Whereas in "case 2" other than $\texttt{obf}_4$ all other objective functions in the asymmetric case has lower value. "case 1" depicts another draw between two solutions and in "case 19" the asymmetric one seems to be the winner. Here are some factors that should be considered.

- These so called symmetric cases are not genuinely created by an optimizer code and just same asymmetric solution forces to be symmetric.
- Not all the Pareto-optimal solutions were compared with their symmetric counterparts.
- Overall impression of this trial move is that there should be an interesting relation between these two configurations.

4.7 Final Run

Based on these observations, it is very intriguing to know what would be the results if a genuine symmetric case is compared with the case without constraints on symmetry. Two symmetric cases are defined. The first model has no constraint of cross section dimensions; therefore, the cross section of the "left" and "right" side (Figure 4.28) could be different. However, the shape of the caliper from node position point of view is symmetric. The second case is completely symmetric, meaning that the cross sections and node positions are completely symmetric with Y axis the passes through mid-caliper.

In a nutshell, another version of the model is generated where nodes 10 and 24 (Figure 4.18 on page 103) is free to have any position in their ranges, that happens to be the same as the previous ranges of tables 4.7 to 4.10 on page 106. On the other hand , node 22, and 12 are the mirror images of nodes 10 and 24 with respect to the center line that nodes 11 and 23 are now located on and fixed on it (Figure 4.28); meaning that the design variables number is reduced by four in this version of model. The third model to be created is called "com sym" that points to complete symmetry of the caliper; the cross sections are also symmetric. And fourteen design variables are sufficient to cover all the cross sections.

$$\textit{Number of Design Variables}_{\texttt{asym}} = 12 \times 2 + 4 \times 2 + 1 = 33$$

$$\textit{Number of Design Variables}_{\texttt{sym}} = 12 \times 2 + 2 \times 2 + 1 = 29$$

$$\textit{Number of Design Variables}_{\texttt{comsym}} = 7 \times 2 + 2 \times 2 + 1 = 19$$

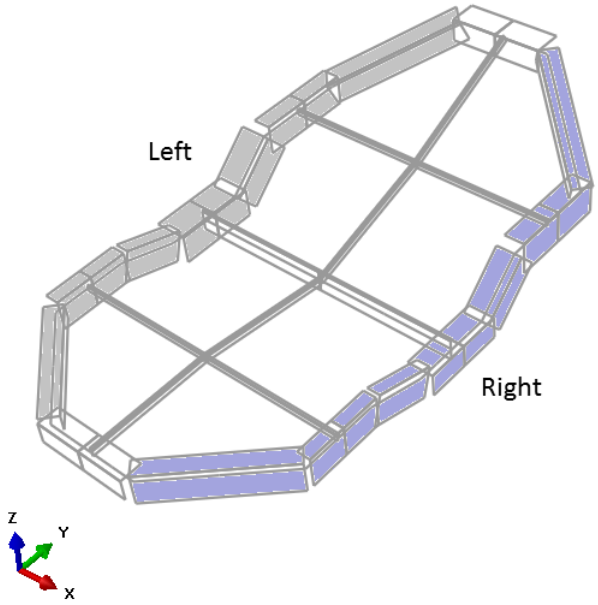


Figure 4.28: Definition of Symmetric

Nevertheless, to gain a legitimate result the samples should exceed hundreds of thousands and be in order of million. With such a high number of samples it is practically impossible to compare them one by one. Therefore, to get an idea over the relative stand between these two scenarios we should plot them. Nonetheless, it is not possible to plot a 6D objective function. Hence, the problem should be again simplified to a 2D one, so that both objective function domains and the Pareto-optimal set of both cases can be plotted and compared in one graph.

Furthermore, the code that can handle a millions samples should be redesigned that would run fast enough on university serves. When the number of samples increases the sizes of the matrices that are handled is increased as well, and CPU, in each step should save these bulky matrices and could occupy a great proportion of RAM and slow down the process by a great deal. The redesigned code now contains three parts; “part 1” generates design variables, but breaks them into smaller design variables matrices with smaller number of samples, like 1500. Therefore, we have 2600 matrices each one containing 1500 samples. “part 2” of code solves each package to get the corresponding objective functions. At last, “ part 3” of code will gather and assemble all these objective functions and generate the Pareto-optimal set. This way the computation time will decrease dramatically.

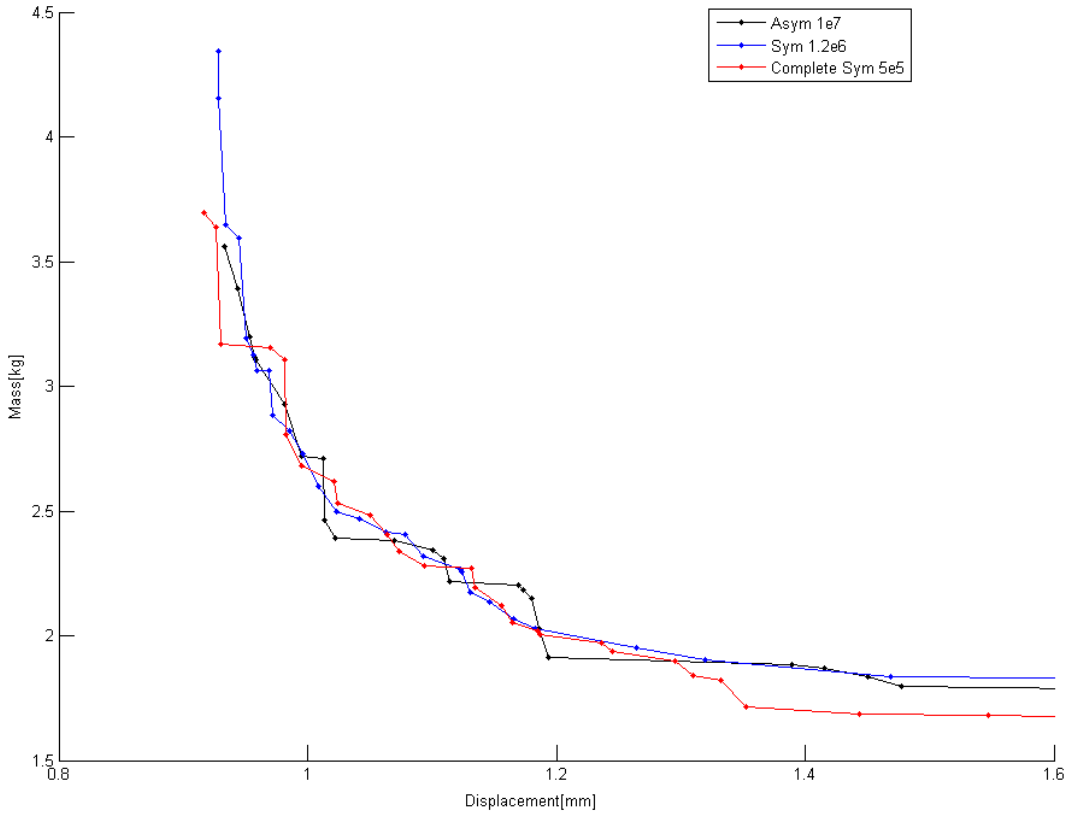


Figure 4.29: The Comparison of Pareto-optimal sets

The Pareto optimal set is obtained for 10,005,000 samples in case of asymmetric design. It is compared to the Pareto sets of symmetric case with 1,500,000 initial samples and the one of "complete symmetry" with 500,000 initial samples.

Final Remarks

As pointed out in the theory section, in exhaustive search method if each design variable may assume n_v different values within its definition range, and design variable presented by n_{dv} then the number of all possible combinations is

$$n_v^{n_{dv}}$$

Although here we are not using exhaustive search method, this gives some insights on how the required number of samples increases dramatically as design variables increase. Therefore, for the asymmetric case such a high number of design variables with respect to symmetric one is selected.

Obviously the asymmetric case is not always the best choice; however, there exist certain regions of the objective function domain, where the asymmetric one is undoubtedly the better choice. As can be noted, in these regions the asymmetric model holds a smaller volume than the symmetric counterpart. This could be very interesting, since the start of the project the general aim was to generate a lighter caliper with identical or better stiffness.

Here one solution in that region is selected and the data is published below as "final case". The comparison with the original design (Table 4.16) exhibits that all the objective functions has improved, expect volume. However, as pointed out before, here is just a coarse design stage with bulky beam elements, by refining the design; this objective function will decrease as well.

With these comparisons, we can conclude the preliminary study part. Now we have reached our goal to have an insight over the correct design direction of the brake caliper. It is established that base on simplified model, to achieve a lighter caliper with the same stiffness, and asymmetric design could be a great candidate. Moreover, by further study of the asymmetric model Pareto-optimal set, one can have a good sense over the right patterns to approach in the detailed design stage.

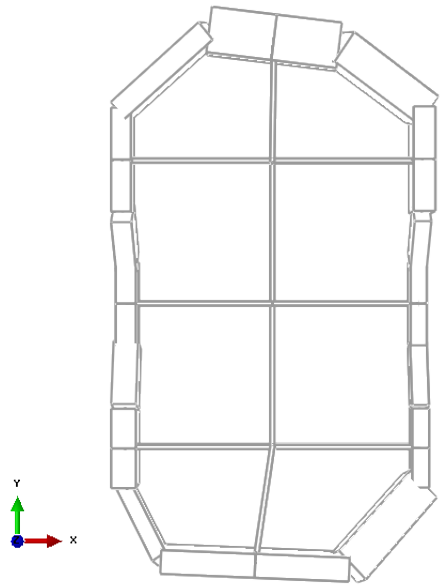


Figure 4.30: One of the solutions in the interesting zone that asymmetric design far surpasses the symmetric one- Top View

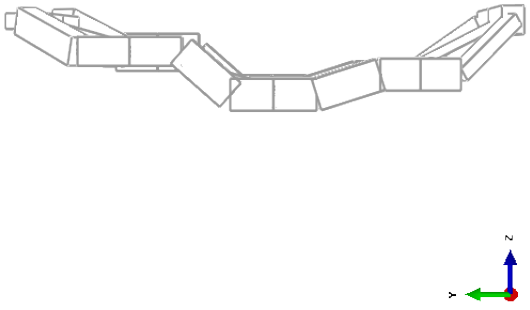


Figure 4.31: "Final Case" side View. The caliper is connected to hub from the opposite side

Table 4.16: "Final Case" Objective Functions

	Final Case	Original Value
$\mathbf{obf_4}$[mm]	0.4877	0.985
$\mathbf{obf_5}$[mm]	0.4213	0.985
$\mathbf{obf_1}$[mm]	-0.0894	-0.2576
$\mathbf{obf_2}$[mm]	-0.1168	-0.1679
$\mathbf{obf_3}$[mm]	-0.0404	-0.0456
$\mathbf{obf_6}$[mm^3]	750653.14	5.19E+05
Mass[kg]	2.03	1.4

Table 4.17: "Final Case" element's cross section values. Refer to fig 4.18 & 4.19

	$\mathfrak{a}[\mathrm{mm}]$	$\mathfrak{a}[\mathrm{mm}]$
$\mathfrak{el}_1$	44.31	34.94
$\mathfrak{el}_2$	44.31	34.94
$\mathfrak{el}_3$	44.31	34.94
$\mathfrak{el}_4$	44.49	28.39
$\mathfrak{el}_5$	44.49	28.39
$\mathfrak{el}_6$	44.49	28.39
$\mathfrak{el}_7$	39.99	27.75
$\mathfrak{el}_8$	39.99	27.75
$\mathfrak{el}_9$	41.03	30.22
$\mathfrak{el}_{10}$	23.54	62.19
$\mathfrak{el}_{11}$	23.54	62.19
$\mathfrak{el}_{12}$	52.41	47.62
$\mathfrak{el}_{13}$	64.37	29.97
$\mathfrak{el}_{14}$	64.37	29.97
$\mathfrak{el}_{15}$	40.02	23.10
$\mathfrak{el}_{16}$	40.02	23.10
$\mathfrak{el}_{17}$	40.02	23.10
$\mathfrak{el}_{18}$	33.44	23.86
$\mathfrak{el}_{19}$	33.44	23.86
$\mathfrak{el}_{20}$	33.44	23.86
$\mathfrak{el}_{21}$	28.26	53.34
$\mathfrak{el}_{22}$	41.47	32.32
$\mathfrak{el}_{23}$	41.47	32.32
$\mathfrak{el}_{24}$	20.77	21.84

Table 4.18: "Final Case" Node Position

	X[mm]	Y[mm]	Z[mm]
Node 10	31.07	158.39	19.60
Node 11	53.16	155.71	19.60
Node 12	75.24	153.03	19.60
Node 22	78.55	-24.79	19.60
Node 23	46.69	-23.48	19.60
Node 24	14.83	-22.17	19.60

Chapter 5

Detailed Study

5.1 Introduction

So far we have used a greatly simplified model to optimize the caliper. Now it is time to test the obtained results against a highly detailed model. For that, another study should be carried out on detailed model in the entire design domain space with the help of commercial CAE software. In the current research project MSC/Nastran was chosen. Moreover, SimLab due to its intuitive and user friendly design was chosen for the mesh preparation.

The approach used is the topological optimization. Topology optimization is a relatively new but extremely rapidly expanding research field of structural and continuum mechanics. It has interesting theoretical implications in mathematics, mechanics, multi-physics and computer science, but also important practical applications in the manufacturing (in particular, car, aerospace and machine) industries, and is likely to have a significant role in micro- and nano-technologies. Topology optimization achieves much higher savings than cross-section or shape optimization.

5.1.1 About Software

MSC Nastran

MSC Nastran is built on work done by NASA scientists and researchers, and is trusted for the design of mission critical systems in every industry, and demanding products such as spacecraft, aircraft, and vehicle.

In recent years, several extensions to its capabilities have resulted in a single multi-

disciplinary solver providing users with solutions to simulate everything from a single component to complex assemblies under diverse conditions.

MSC Nastran offers a complete set of linear static and dynamic analysis capabilities along with unparalleled support for superelements enabling users to solve large, complex assemblies more efficiently. MSC Nastran also offers a complete set of implicit and explicit nonlinear analysis capabilities, thermal and interior/exterior acoustics, and coupling between various disciplines such as thermal, structural, and fluid interaction. The capabilities of the software are:

- Structural solutions
- Efficient dynamic analysis
- Composites analysis to validate materials
- High performance FEA for fast results
- Multidisciplinary optimization
- Multidisciplinary Solution

Structural Solution. MSC Nastran has a comprehensive element library, including specialty elements like welds, bushings, and fasteners that accurately model complete assemblies. Solution options include linear and nonlinear statics, 3D contact, dynamics, acoustics, thermal analysis, and more. Built in capabilities for sensitivity analysis, parameter studies, and optimization enable you to find optimal sizing, shape, topology, topography and topometry optimization simultaneously to find better designs.

Multidisciplinary Optimization. Design optimization in MSC Nastran can combine analysis results from a number of disciplines, including statics, normal modes, buckling, direct and modal frequency, modal transient, acoustic, direct and modal complex Eigenvalue analysis, static Aeroelastic response and flutter analysis. In addition, design models can also employ superelements.

SimLab

SimLab is a process oriented, feature based finite element modeling software that allows you to quickly and accurately simulate engineering behavior of complex assemblies.

SimLab automates simulation- modeling tasks to reduce human errors and time spent manually creating finite element models and interpreting results. SimLab is not a traditional off the shelf Pre and Post processing software but a vertical application development platform to capture and automate simulation processes.

5.2 The Underlying Theory

Topology optimization is distinct from shape optimization since typically shape optimisation methods work in a subset of allowable shapes which have fixed topological properties, such as having a fixed number of holes in them. Therefore topology optimisation is used to generate concepts and shape optimisation is used to fine-tune a chosen design topology.

Topology optimisation is a mathematical approach that optimises material layout within a given design space, for a given set of loads and boundary conditions such that the resulting layout meets a prescribed set of performance targets. Using topology optimisation, engineers can find the best concept design that meets the design requirements.

Topology optimisation has been implemented through the use of finite element methods for the analysis, and optimisation techniques based on the method of moving asymptotes, genetic algorithms, optimality criteria method, level sets, and topological derivatives.

Topology optimisation is used at the concept level of the design process to arrive at a conceptual design proposal that is then fine-tuned for performance and manufacturability. This replaces time consuming and costly design iterations and hence reduces design development time and overall cost while improving design performance.

Two types of topology optimization exist: discrete or continuous, depending on the type of a structure.

For inherently discrete structures, the optimum topology or layout design problem consists in determining the optimum number, positions, and mutual connectivity of the structural members.

In this research the continuum method has been used. In topology optimization of continuum structures, the shape of external as well as internal boundaries and the number of inner holes are optimized simultaneously with respect to a predefined design objective. It is assumed that the loading is prescribed and that a given amount of structural material is specified within a given 2D or 3D design domain with given boundary conditions.

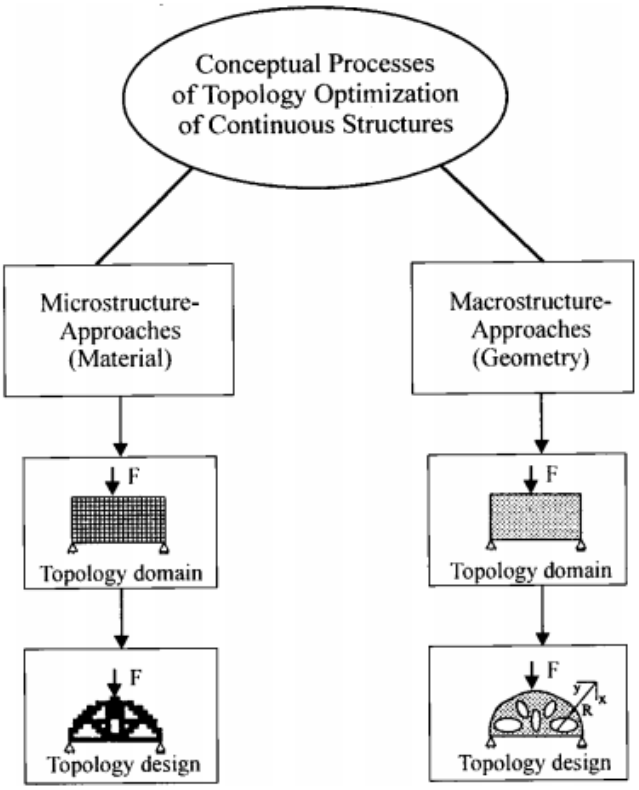


Figure 5.1: Conceptual process of topology optimization. Reprinted from [9]

There are several research activities going on throughout the world concentrating on these problems, and different solution procedures have been developed. Very roughly, one can distinguish between two classes of approaches, the so-called Material- or Micro-approaches vs. the Geometrical or Macro-approaches.

Microstructure-approaches (Material) It is our design objective to find that structural topology which renders a given design objective an optimum value subject to a prescribed amount of structural material. It is assumed that in solid form the amount of material is less than the amount that would be needed to cover the entire admissible domain for the continuum. Hence, for the initial design it is normally chosen to distribute the material evenly in some porous, microstructural form over the admissible design domain.

In the Microstructure-approach to topology optimization, it is customary to use a fixed finite element mesh to describe the geometry and the mechanical response

fields within the entire admissible design domain [9]. The optimization consists of determining whether each element should contain material or not. The density of the material in each element is used as a design variable defined between 1(solid material) and 0(void). This is the underlying approach in SIMP (Solid Isotropic Material with Penalization) that is used is this research project- and other methods based on homogenization technique.

Macrostructure-approaches (Geometry) Optimization is performed in conjunction with a shape optimization, the finite element mesh cannot be a fixed one, but must change with the changes of the boundaries of the design. Within the Macrostructure approach, the topology of a solid body can be changed by growing or degenerating material or by inserting holes. Therefore, it is based on geometry changes and works on finite element mesh which changes at each iteration.

5.2.1 Problem Statement

As any other optimization the goal is to minimize the objective functions, the objective function can be formulated as

$$\int_{\Omega} \phi(\rho) d\Omega$$

Therefore, the optimization is

$$\mathtt{min}_{\rho} \int_{\Omega} \phi(\rho) d\Omega$$

That is subjected to $\rho \in \{0, 1\}$, Design Constraints, and governing differential equations.

For the current problem the goal of optimization is to minimize the mass and maximize the stiffness; thus, minimizing the compliance of a structure to increase structural stiffness.

Ω is the design space, which is the allowable volume within which the design can exist. Assembly and packaging requirements and tool accessibility are some of the factors that need to be considered in identifying this space. With the definition of the design space, regions or components in the model that cannot be modified during the course of the optimisation are considered as "non-design" or "frozen" regions.

The Discrete Selection Field (ρ) , this is the field over which the discrete optimisation is to be performed. It selects or deselects a point on the design space to further the design objective. By selection it has to take the value **1** and by de-selection it has to take the value **0**. Here the density is considered to be the discrete selection field.

In continuous optimization it is assumed that ρ varies continuously over the domain (0,1).

As it is clear, the aim of research is to make the caliper stiffer. A stiff structure is one that has the least possible displacement when given certain a set of boundary conditions. A global measure of the displacements is the strain energy (also called compliance) of the structure under the prescribed boundary conditions. The lower the strain energy the higher the stiffness of the structure would be. So, the problem statement involves the objective functional of the strain energy which has to be minimized. The objective function should be chosen as a function of selection field. Therefore, the material property should be interpolated in terms of selection field.

5.2.2 SIMP

A widely used interpolation scheme is called the Solid Isotropic Material with Penalization (SIMP). This interpolation is essentially a power law that interpolates the Young's modulus of the material to the scalar selection field. The value of varies between (0,1) in general. This has been shown to confirm to micro-structure of the materials. So one could view topology optimization to be a process of selection of micro-structure at every point in space so that an objective function is minimized.

The elasticity tensor E_{ijkl} and the volume of a structure made of the material are given by

$$E_{ijkl}(x) = \rho(x)^p E^0_{ijkl}, \qquad p > 1; \quad \textit{Volume} = \int_\omega \rho(x)dx$$

where E_{ijkl} is the elasticity tensor of a given solid isotropic reference material and E^0_{ijkl} is the stiffness of the element when ρ is equal to one . The density enters the stiffness relation in a power $p > 1$ which has an effect of penalizing intermediate densities $0 < \rho < 1$, since at such densities the SIMP material has lower stiffness that the reference material at the same cost [9]. Figure 5.2 displays the relative stiffness E/E^0 versus density. It suggests that the use of SIMP material model will force the topology design toward limiting values $\rho = 0$ (void) and $\rho = 1$ (solid) and thereby prompt the creation of more distinctive 0-1 designs.

It is a drawback of the SIMP model that the topology designs obtained not only exhibit dependence on the value of p, but also on the finite element mesh applied (Figure 5.3). This drawback disappears when perimeter or surface area constraint is included in the formulation of the problem. Even if such a constraint is not included in the formulation, the SIMP model can yield very nice results and the simplicity of the model greatly facilitates the implementation of topology design in commercial finite element codes.

In general for the method these steps should be taken:

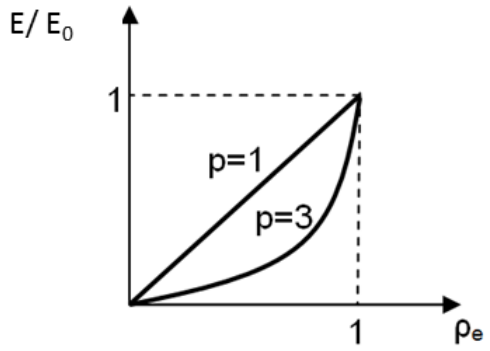


Figure 5.2: Relative Stiffness vs Volume Density for SIMP material

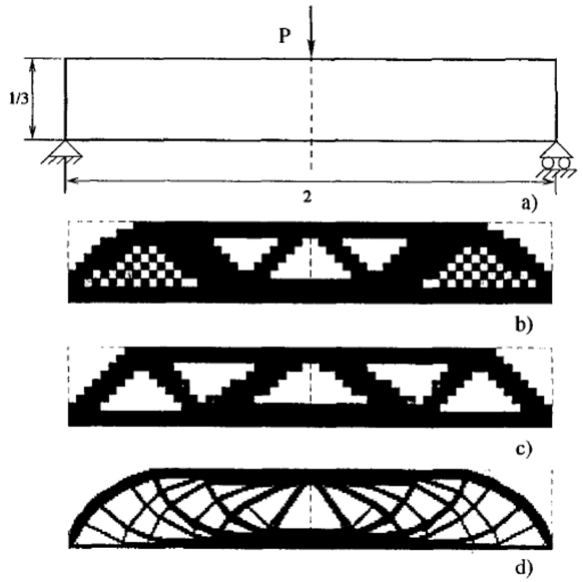


Figure 5.3: Mesh Refinement Effect on SIMP Model Results- (**d**) Has the higher mesh quality

1. Solving the "most relaxed" problem with p=1
2. Increasing the value of p
3. Resolving the new problem with the new value of p by using the optimal solution of previous problem as a starting point
4. Repeating 2 and 3 until p is sufficiently large

5.3 Optimization of the Caliper

The caliper is modeled and statically analyzed, and topology optimization is performed. As the flowchart in figure 5.4 depicts meshing the part is conducted by SimLab, and the solution is carried out by MSC Nastran (5.1.1).

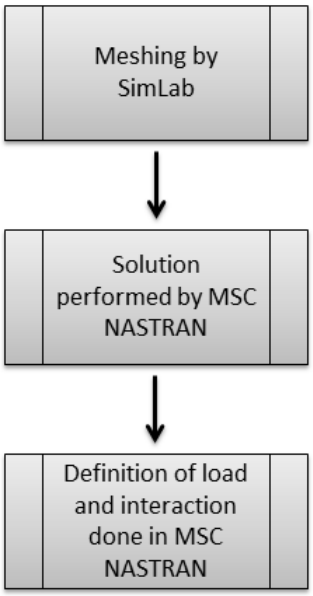


Figure 5.4: Analysis Flow Chart

Using SimLab it is possible to prepare the mesh for the component being analyzed. To prepare the mesh, it is preferable to use SimLab instead of the tool in PATRAN due to the fact that, the former is more intuitive and user-friendly, and it allows obtaining a more accurate meshing. Once completed the mesh, it is imported to the internal environment of PATRAN in order to define the load, the interactions and all other parameters needed for the solution.

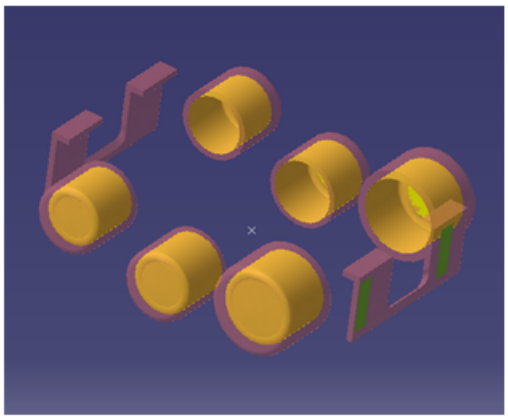


Figure 5.5: The Un-designed Zone

As explained in the theory section, in the component, two different domains are included: 1) The optimization domain, or the design domain, where the material distribution occurs during topological optimization, and the material could exist in any location in that space. 2) The frozen or un-designed zone is fixed during optimization, where in our case it is piston components and the supporting points of pads. Figure 5.6 and 5.5 demonstrate these two zones. In Figure 5.7 in the design domain, the location of B.C. or the place where caliper is connected to hub carrier via the screws are marked.

During the optimization these two domains are kept completely separate by creating groups corresponding to the two domains, so that they would be easily distinguishable and available to select in PATRAN.

As for the loads, two different scenarios have been painted. First just considering the forces due to the pressure acting on the pistons and neglecting the tangential force due to pad movement when disk is rotating (Figure 5.8). Second, when all these forces are present in the caliper and modeled (Figure 5.9).

Considering all forces to be present, the results are published below.

Results

Figures 5.10 and 5.11 on page 133 clearly indicate that the caliper should not be designed in a symmetric manner. This design may point to utopia region in the Pareto-optimal set gained by the MATLAB code. If compared with the solution that is depicted in figure 4.30 on page 118, we witness a great resemblance between these two designs

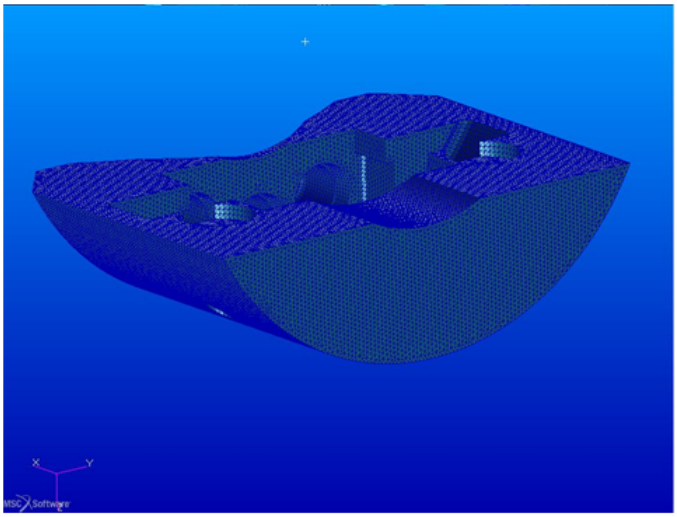


Figure 5.6: The Design Domain of the Caliper

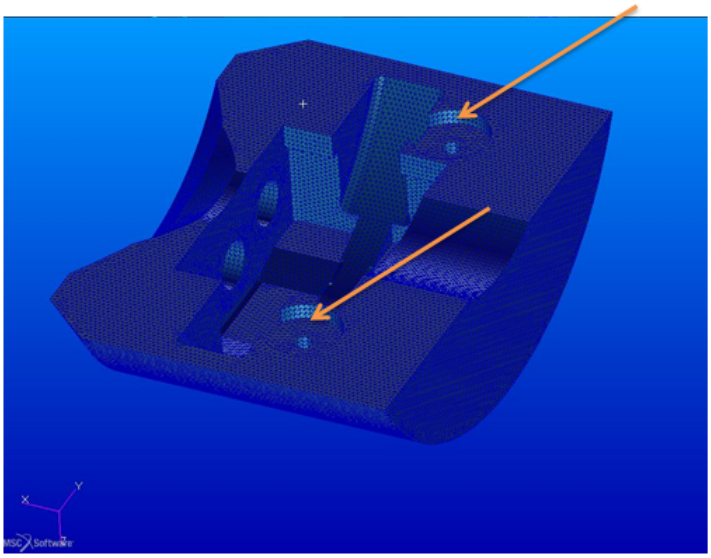


Figure 5.7: The Design Domain of the Caliper-Constraint Positions are marked by the arrows, where the screws secure the caliper on the hub carrier

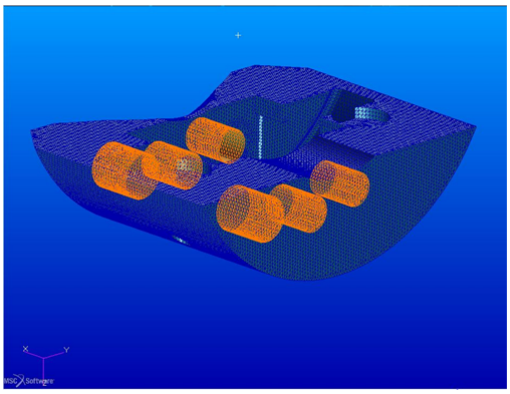


Figure 5.8: Load Case 1, where just uniform pressure in the cylinders are considered

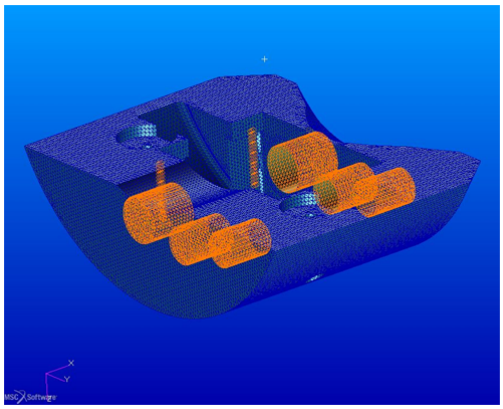


Figure 5.9: Load Case 2, the tangential forces are marked by orange straps. Along with that, the pressure in the cylinders is evident

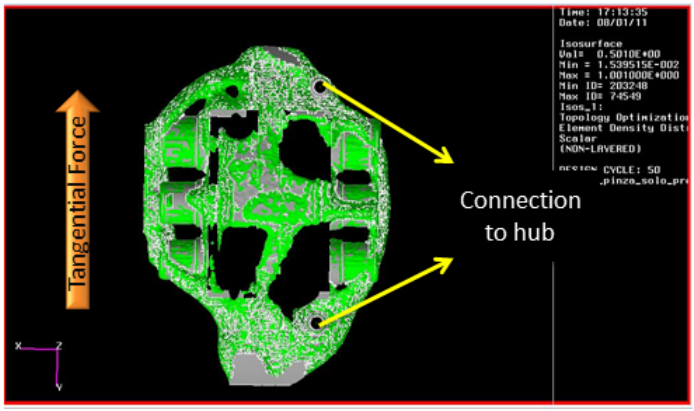


Figure 5.10: Topology Optimization Results. Elements with density greater than 0.2 is considered as "solid"

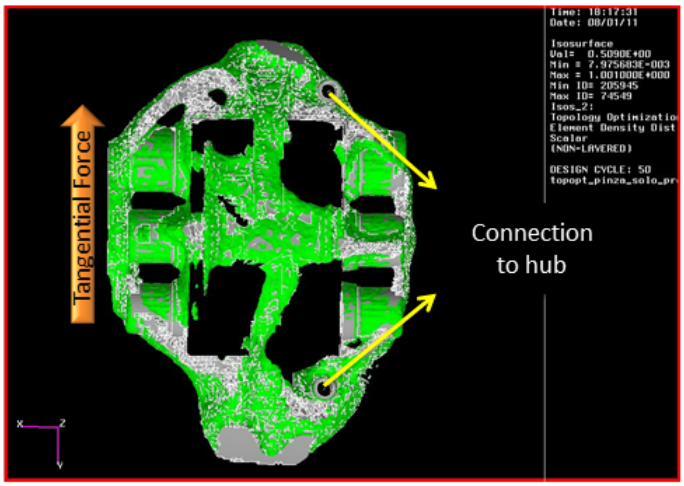


Figure 5.11: Topology Optimization Results. Elements with density greater than 0.5 is considered as "solid"

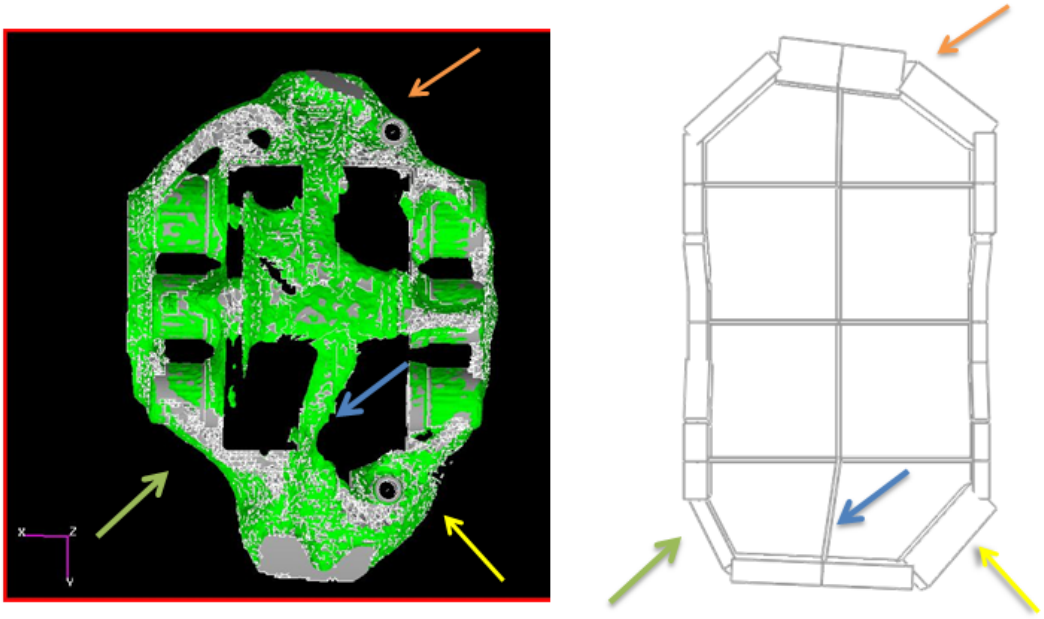


Figure 5.12: Comparison Between Topology Optimization & Matlab One. Some of the details are pointed out by arrows

(Figure 5.12). This asymmetric code ensures higher stiffness and lower mass of the caliper.

Chapter 6

Conclusions

This project is focused on the optimization and assessment of a heavy duty brake caliper. The project started by studying a current caliper design. The standard caliper design is based on a symmetric structure and the chosen racing caliper is no exception to this rule.

For the optimization of a caliper, or any engineering optimization, it would be cumbersome or even impossible to start the task without a deep understanding of the theory underlying the problem. A simple validated model against a robust commercial software proved to be an acceptable representative of the complicated shape of the caliper. This simplified model, that is based on beam elements, can be enhanced by further eliminating redundant factors and including those details would assist the model‘s behavior to approach the actual one.

During the optimization stage, a sobol sequence proved to be an effective method for sampling solutions in the design variables domain, even though the design variables vector is very large (44 design variables). Even with a small set of samples acceptable results are yielded; they are taken advantage of to further improve the design variables and their ranges. Downsizing the design variable vector, severely aided to decrease the required number of samples to achieve rigorous results.

After the first look at the optimization results, it is noted that the caliper design should not be necessarily symmetric. For validating this theory, the optimal solutions of two symmetric cases are compared against the general case with no constraints on symmetry. Investigating the results reveals that, the asymmetric designs are not the best ones in all regions of objective function domain. However, in certain design areas, where a reduced value of mass is required, the asymmetric designs are better than the

symmetric ones.

By examining both the optimized designs and the original caliper side by side, it is clear that all the objective functions except the volume have significant improvements. Increment of mass is expected because we are still in a preliminary design stage, where bulky beams have been considered for modeling the system. After refining the design the mass would notably decrease.

Next, the obtained results of the MATLAB model were compared against the ones of a detailed model study, which has been optimized by applying topology optimization by means of sophisticated commercial software. This comparison demonstrated that not only the conclusion drawn from our simple MATLAB model was correct, but the proposed optimal shapes are highly similar.

For future studies, now that is established that asymmetric design could be a promising one, a deep and detailed study of the design is suggested. Furthermore, an experimental study should be the next step towards creating a caliper that overcomes the current design performance.

Bibliography

[1] Asm. *Properties and Selection : Nonferrous Alloys and Special-Purpose Materials*. Asm.

[2] B. Breuer, K.H. Bill, et al. *Brake technology handbook*. 2008.

[3] R.D. Cook. *Finite element modeling for stress analysis*. Wiley, 1995.

[4] A. de Vries and M. Wagner. The brake judder phenomenon. Technical report, Society of Automotive Engineers, 400 Commonwealth Dr, Warrendale, PA, 15096, USA,, 1992.

[5] A. Dey and R. Mukerjee. *Fractional Factorial Plans*. Wiley Series in Probability and Statistics. Wiley, 2009.

[6] JR John T. Dewolf David F. Mazurek Ferdinand P.Beer E.Russel Johnston. *Mechanics of materials*. Mc Graw Hill higer education, fifth edition edition, 2007.

[7] A.J.M. Ferreira. *MATLAB Codes for Finite Element Analysis*. Springer, 2009.

[8] J.E. Hatch, Aluminum Association, and American Society for Metals. *Aluminum: Properties and Physical Metallurgy*. Aluminum / J.E. Hatch [Hrsg.]. American Society for Metals. American Society for Metals, 1984.

[9] X. Huang and M. Xie. *Evolutionary Topology Optimization of Continuum Structures: Methods and Applications*. John Wiley & Sons, 2010.

[10] G. Mastinu, M. Gobbi, and C. Miano. *Optimal Design of Complex Mechanical Systems: With Applications to Vehicle Engineering*. Springer, 2010.

[11] Zahit Mecitoglu. Finite element analysis in structures. page 16, 2008.

[12] R.B. Statnikov and J.B. Matusov. *Multicriteria optimization and engineering.* Mechanical engineering: Industrial Engineering. Chapman & Hall, 1995.

[13] E. Thoms. *Disc brakes for heavy vehicles.* 1988.

[14] Wikipedia. Abaqus — wikipedia, the free encyclopedia, 2012.

Index