Czech Technical University in Prague
Faculty of Civil Engineering
Department of Concrete and Masonry Structures

Master's Thesis

Quality Evaluation of Finite Element Models with Applications on Concrete Structures

Aneta Bulíčková

Supervisors: Ing. David Krybus, Ph.D.
Ing. Radek Štefan, Ph.D.

Study Program: Civil Engineering

Branch of Study: Building Structures

2020

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta stavební
Thákurova 7, 166 29 Praha 6

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: Bulíčková Jméno: Aneta Osobní číslo: 439107

Zadávající katedra: K133 - Katedra betonových a zděných konstrukcí

Studijní program: Stavební inženýrství

Studijní obor: Konstrukce pozemních staveb

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce: Quality Evaluation of Finite Element Models with Applications on Concrete Structures

Název diplomové práce anglicky: Quality Evaluation of Finite Element Models with Applications on Concrete Structures

Pokyny pro vypracování:
Rešerše literatury.
Volba řešení kontroly kvality sítě konečných prvků.
Algoritmizace řešení a implementace v programovacím jazyce C++.
Aplikace navrženého řešení na vybraných příkladech betonových konstrukcí.

Seznam doporučené literatury:
P. J. Frey, P.-L. George - Mesh Generation application to finite elemens.
B. Eckel - Thinking in C++.
Normy: ČSN EN 1990, ČSN EN 1991-1-1, ČSN EN 1992-1-1
Procházka, J., a kol. Navrhování železobetonových konstrukcí. Příklady a postupy.
Procházka, J., Šmejkal, J. Betonové stropní a schodišťové konstrukce.

Jméno vedoucího diplomové práce: Ing. Radek Štefan, Ph.D.

Datum zadání diplomové práce: 19. 9. 2019 Termín odevzdání diplomové práce: 5.1.2020
Údaj uveďte v souladu s datem v časovém plánu příslušného ak. roku

Podpis vedoucího práce Podpis vedoucího katedry

III. PŘEVZETÍ ZADÁNÍ

Beru na vědomí, že jsem povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je nutné uvést v diplomové práci a při citování postupovat v souladu s metodickou příručkou ČVUT „Jak psát vysokoškolské závěrečné práce" a metodickým pokynem ČVUT „O dodržování etických principů při přípravě vysokoškolských závěrečných prací".

30. 9. 2019
Datum převzetí zadání Podpis studenta(ky)

Acknowledgments

I would like to express great appreciation to Dr. David Krybus for his guidance, valuable advice and encouragement. I am particulary grateful for his time he devoted me to discuss the topic and assist me in the process of learning C++ language. Furthermore, I am obligated to his colleagues Dr. Burkhard Bornemann and Dr. Roland Sauer from company RIB Software SE for their opinions and suggestions regarding the topic. Also, I would like to acknowledge the company for providing me computational equipment.

I much appreciate advice given by Dr. Radek Štefan. I am grateful that he initiated and encouraged my interest in computational science.

Last but not least, I would like to thank my family for supporting me throughout the entire study period. In particular, my heartiest appreciation belongs to my fiancé for his love, understanding and patience.

Declaration

I hereby affirm that this master's thesis has been written by myself, under the supervision of Dr. David Krybus and Dr. Radek Štefan. All sources of information that have been used in the thesis are acknowledged in the text and listed in the Bibliography, in accordance with the requirements given by the CTU Guideline - Metodický pokyn č. 1/2009 O dodržování etických principů při přípravě vysokoškolských závěrečných prací (in Czech).

Prague,
6 January 2020 .....

Abstract

Mesh quality plays an important role in the accuracy and stability of Finite Element Analysis. Many structural engineering softwares are able to automatically generate meshes on the specified model and nowadays this feature is assumed to be well developed. However, it is always encouraged to check the mesh quality prior to calculating since poor-quality elements and other inconsistencies may cause an inaccurate or misleading solution.

The main objective of the thesis is to create a computational tool applicable for evaluating the mesh quality of a plane structures. The tool determines quality metrics for each element of the mesh and provides an overall outlook on the solved structure. Based on that, a user is able to locate areas, where the mesh is of poor quality. The tool was developed in C++ programming language.

In addition, the thesis analyzes the theory regarding the mesh quality and possible means used for its evaluation. Moreover, various examples of concrete structures are analyzed in order to investigate the influence of the mesh quality on the results.

The thesis was written in corporation with the company RIB Software SE.

Key words: mesh quality, Finite Element Analysis, computational tool, C++, concrete

Abstrakt

Kvalita sítě zaujímá zásadní roli v přesnosti a stabilitě metody konečných prvků. Mnoho stavebních softwarů dokáže na analyzovaném modelu automaticky vygenerovat síť a zároveň lze předpokládat, že je tato funkce v dnešní době dobře vyvinutá. Nicméně kvalita sítě by měla být vždy zkontrolována, protože nekvalitní prvky sítě a další nekonzistence mohou způsobit nepřesné nebo zavádějící řešení.

Hlavním cílem této práce je vytvoření výpočetní pomůcky, která slouží k hodnocení kvality sítě plošných konstrukcí. Pomůcka stanovuje metriky kvality pro každý prvek sítě a poskytuje celkový náhled na kvalitu sítě řešené konstrukce. Na základě toho je uživatel schopen lokalizovat oblasti, kde je síť špatně vygenerována. Pomůcka byla vyvinuta v programovacím jazyce C++.

Dále práce analyzuje teorii týkající se kvality sítě a možných způsobů jejího vyhodnocení. Kromě toho jsou analyzovány příklady betonových konstrukcí za účelem zjištění vlivu kvality sítě na výsledky analýzy.

Diplomová práce byla napsána ve spolupráci se společností RIB Software SE.

Klíčová slova: kvalita sítě, metoda konečných prvků, výpočetní pomůcka, C++, beton

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

AR	aspect ratio
CST	constant strain triangle
DOF	degree of freedom
FE	finite element
FEA	Finite Element Analysis
JR	Jacobian ratio
LST	linear strain triangle
Q4	bilinear quadrilateral
Q8	quadratic quadrilateral
Q9	biquadratic quadrilateral
UML	Unified Modeling Language
VTK	Visualization Toolkit
XML	Extensible Markup Language

List of Symbols

$\mathbf{A}$	area
$\mathbf{b}$	width of the cross-section
$\mathbf{c_i}$	constant
$\mathbf{E}$	Young's modulus
$\mathbf{F}$	acting force
$\mathbf{G}$	shear modulus
$\mathbf{h}$	height of the cross-section / thickness
$\mathbf{I_y}$	moment of inertia with respect to y-axis
$\mathbf{J}$	Jacobian matrix
$\mathbf{K}$	flexural rigidity of the slab
$\mathbf{L_i}$	length
$\mathbf{L_{max}}$	maximum side length
$\mathbf{L_{min}}$	minimum side length
$\mathbf{m_{xx}}$	bending moment in x direction
$\mathbf{m_{yy}}$	bending moment in y direction
$\mathbf{M}$	bending moment
$\mathbf{N_i}$	shape function
$\mathbf{P_i}$	point
$\mathbf{p}$	uniformly distributed load
$\mathbf{S_i}$	surface area
$\mathbf{u}$	horizontal displacement
$\mathbf{v}$	vertical displacement
$\mathbf{w}$	deflection of the slab
$\mathbf{x, y, z}$	spatial coordinates
$\boldsymbol{\gamma_{xy}}$	shear strain

$\triangle \mathbf{L}$	length difference
$\boldsymbol{\delta}$	deflection
$\boldsymbol{\nu}$	Poisson's ratio
$\boldsymbol{\sigma}$	principal stress
$\boldsymbol{\theta_e}$	angle of equi-angular element
$\boldsymbol{\theta_{max}}$	maximum angle
$\boldsymbol{\theta_{min}}$	minimum angle
$\boldsymbol{\epsilon_x}$	strain in x direction
$\boldsymbol{\epsilon_y}$	strain in y direction
$\boldsymbol{\xi, \eta}$	mapped coordinates
$\triangle$	triangle
$\square$	quadrilateral

C++ class description is explained in the terms of UML below:

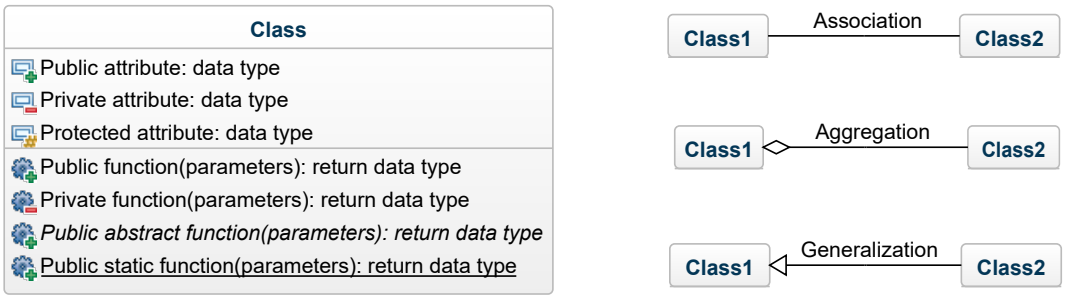


Figure 1: C++ Class diagram explanation

Meaning of the attribute/function denotation is as follows:

Public	is accessible outside the class defining it
Private	is only accessible within the class defining it
Protected	is accessible in the class defining it and in classes that inherit from that class

Meaning of the mutual relation between classes is as follows:

Association	relationship between two objects
Aggregation	special case of association - an object "has-a" another object
Generalization	inheritance - "is-a" relationship between objects

Chapter 1

Introduction

This thesis investigates the topic of the mesh quality, which is an integral part of Finite Element Analysis (FEA). Firstly, the theoretical background is given. Then, mesh related problems, which may arise while executing FEA in commercial software, are pointed out.

Solution is suggested in the practical part, which introduces the computational tool **MeshEvaluator** created as a part of the thesis. The tool was developed in C++ programming environment. It uses data generated by *RIB* software applicable to plane structures. Furthermore, it calculates mesh quality metrics for each element of the mesh and evaluates their overall quality. *XML* output compatible with the graphical software *ParaView* enables visualization of the analyzed plane problem.

Chapter 2 provides a brief introduction to the topic. It summarizes the principle of FEA and points out remarks related to the mesh. Furthermore, element types are described in order to outline their behavior and encourage correct usage.

Chapter 3 gives a theoretical background about the mesh quality assessment. The following metrics used for element quality evaluation are introduced: configuration, relative midpoint difference, aspect ratio, skewness, Jacobian ratio and distortion. Their application and behavior in dependence on the element shape are further analyzed. The Jacobian matrix for quadrilateral elements is derived in Appendix A. The Chapter also sets criteria for the overall quality evaluation, which is further used in the practical part.

Chapter 4 describes the computational tool **MeshEvaluator**, which was created as a part of the thesis in C++ programming language. The internal structure of the tool is listed as well as its internal flow during its execution. The processes of reading input, computing and writing output are summarized. The input and output files are illustratively shown in Appendix B.

Chapter 5 shows practical use of the developed tool. It analyzes two concrete plane structures: a cantilever subjected to a point load and a two-way slab subjected to a distributed load. Different geometries of the mesh are used in order to study correlation between the mesh quality and the accuracy of the results. Additional considerations that arose during elaborating this Chapter are attached in Appendix C.

Taking everything into account, conclusion is given in **Chapter 6**. The ideas for possible future extensions are included.

Chapter 2

Mesh and its elements

A brief introduction to the FEA, the mesh and the types of the elements is given in this Chapter.

2.1 FEA with the relation to the mesh

FEA is one of the most common processes, which is used to find an approximate solution to a complex engineering problem. In practice the whole process is usually solved by computational softwares due to its complexity. Figure 2.1 illustrates steps of the procedure.

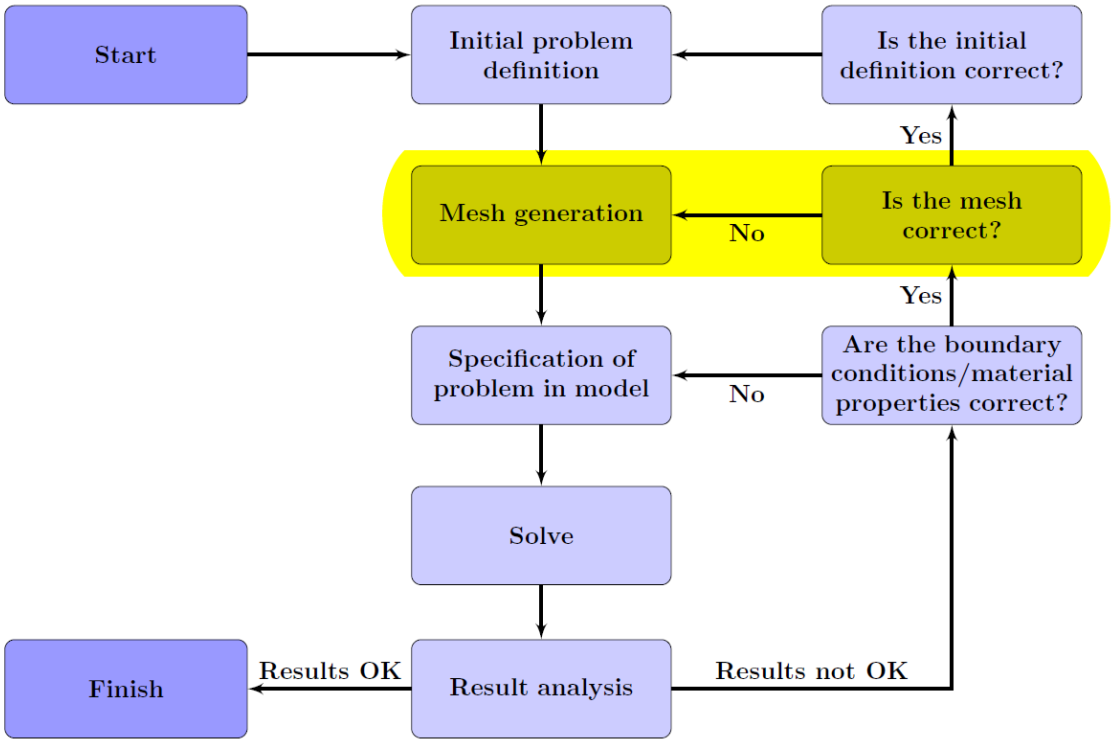


Figure 2.1: Process of FEA, according to [1]

In structural engineering it is necessary to create models of given structures to analyze their mechanical behavior. To do so, it involves breaking down the model into a mesh of discrete elements. The elements, which are connected in nodes, create a continuous space. The material properties, the restrains and the loading scenario are further assigned to the model. Each element is analyzed individually in its nodes. After that, all elements are assembled to form a complete system of equations (*stiffness matrix*). Taking into account the acting load, the system of equations is solved in order to get *a displacement field*. Other values can be calculated based on the displacement field (e.g. internal forces, reactions, stresses).

It is important to mention that FEA gives only an approximate solutions and that the degree of the result accuracy depends on each step. An incorrect interpretation of any step may result in misleading solutions. If the results are not acceptable, changes need to be done and the follow-up process needs to be repeated. All in all, the mesh is a fundamental part of FEA. It predestinates how the model is divided and where the resulting values are calculated. A poor quality mesh may cause inaccurate results and misunderstanding of the structure.

For simple problems like a regular structure, the mesh can be easily generated by hand. However, if a complex or irregular geometry is analyzed, application of Finite Element preprocessor is required to generate the mesh. Furthermore, the shape of FE mesh should be adapted to the problem that is to be calculated. The mesh should be finer at the areas where *stress peaks* are expected. These lie, for example, at the transition points of a big concentrated loads, at points where the support conditions change or in the re-entrant corners [4]. Usually finer mesh ensures more accurate results, but it also increases computational cost as well. Therefore, a high-quality mesh means that there is an optimal balance between the computational cost and the achieved fineness.

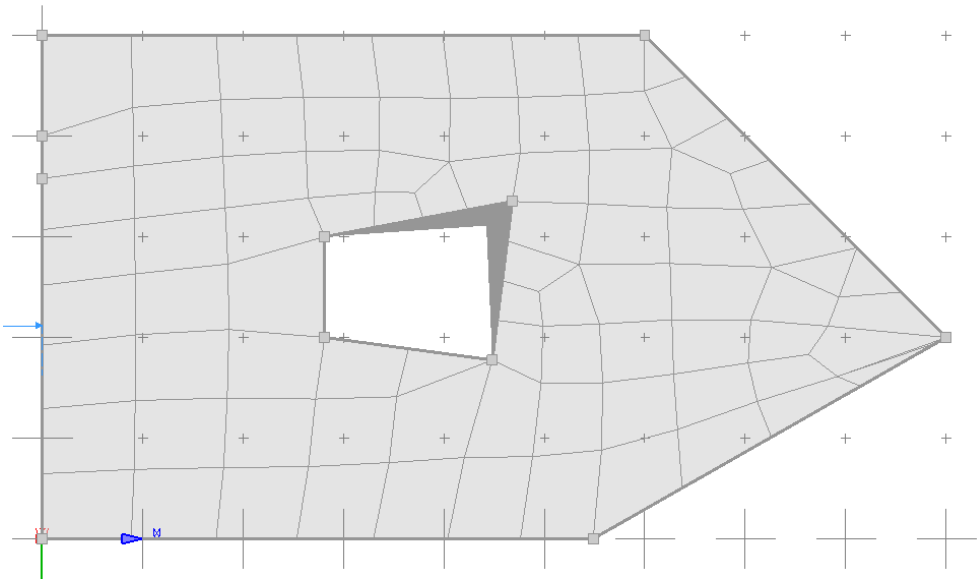


Figure 2.2: Example of automatically generated mesh in software *RIB iTWO structure fem*

2.2 2D element types

The most common 2D element types applied in linear solid mechanics are introduced according to [5] and [9] in this Section. The summary is presented in Table 2.1.

The elements can be sorted by their geometric nature to triangles and quadrilaterals. Moreover, they can be divided by their interpolation function into linear and quadratic. Figure 2.3 shows examples of the element types in 2D plane: **(a)** *linear triangle (CST)*, **(b)** *quadratic triangle (LST)*, **(c)** *bilinear quadrilateral (Q4)*, **(d)** *quadratic quadrilateral (Q8)* and **(e)** *biquadratic quadrilateral (Q9)*.

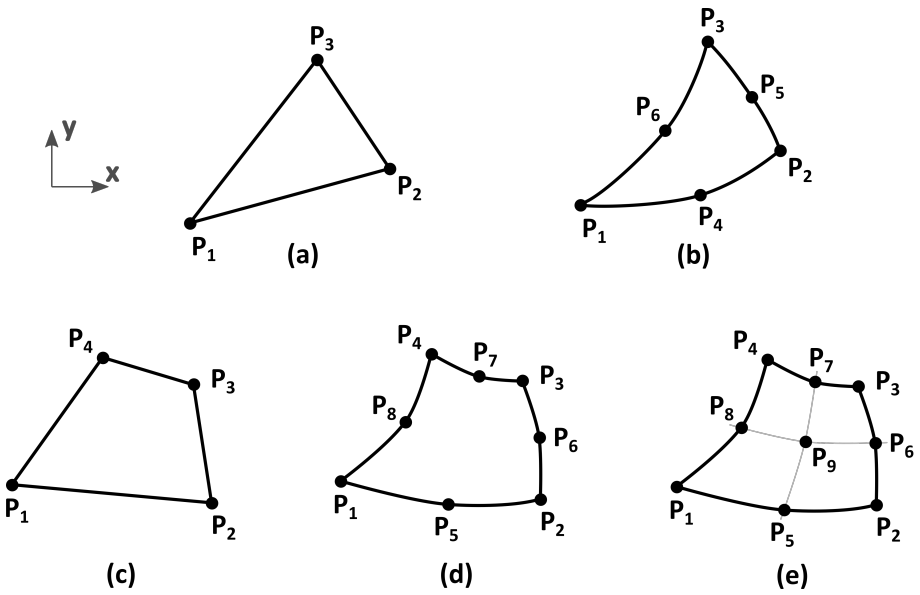


Figure 2.3: 2D element types, according to [5]

Linear triangle (CST)
The linear triangle is the simplest element for planar analysis. The element is denoted by the abbreviation *CST*, which stands for *constant strain triangle*. It refers to the property of having constant strain across the element. It is due to the fact that the displacement field is linear and its derivative, which is the strain, is constant. CST element exhibits undesirably stiff behavior in bending. That can cause inaccurate results, which can be improved by mesh refinement, however convergence is rather slow.

Quadratic triangle (LST)
The quadratic triangle has three midside nodes in addition to the vertex nodes. That allows the displacements to be expressed by a quadratic functions. Consequentially, the strain as a derivative is linear (*LST* as *linear strain triangle*). LST elements (and quadratic elements in general) perform better in bending because they can describe the real deformation more precisely. Furthermore, modeling nonlinear boundaries (e.g. circular hole, rounded edge) is substantially better by using quadratic elements, which fit the shape more accurately.

Bilinear quadrilateral (Q4)
The bilinear quadrilateral is a four-node plane element. The displacement field is expressed by the terms x, y and the quadratic term xy. Therefore the strain in x direction varies linearly with the coordinate y and the strain in y direction varies linearly with the coordinate x. That is why the element is denoted *bilinear*. Because of strains (and stresses) are not restricted to a constant value, the Q4 element is more efficient and accurate then the linear triangle (CST). However, similarly like CST, Q4 element is unable to describe the bending deformation well. In the state of pure bending, a fictitious shear strain is introduced because of an inadequate approximation, which results in overrated stiffness of the element. This defect is called *shear locking* (described in more detail in Section 3.3.1).

Quadratic quadrilateral (Q8)
The quadratic quadrilateral Q8 is obtained by adding nodes to the sides of the bilinear quadrilateral. The element is also called *serendipity element*. The quadratic quadrilateral preforms well in bending. Displacement shape of an initially straight side can be expressed as a straight line or a quadratic curve, therefore the element can describe the real deformation in bending sufficiently (for rectangles exactly).

Biquadratic quadrilateral (Q9)
A ninth node is added within the element's center in case of Q9 element. This node is internal and it is not directly connected to any other element. The Q9 element is also denoted as *Lagrange element*. The displacement field consists of an extra x^2y^2 term. The behavior is slightly better then the Q8 element. Unlike Q8, it can exactly represent the state of pure bending for non-rectangular elements. In general, Q9 elements are less sensitive to nonrectangularity, curvature of sides and placement of side nodes away from the midsides.

To summarize, quadratic elements are preferred in the analysis because of the higher accuracy and the flexibility in modeling complex geometry such as curved boundaries. An application of these elements ensures more accurate solution to be achieved with fewer elements. Also, an unwanted error caused by shear locking (Section 3.3.1) is avoided by using quadratic elements. On the other hand, development of higher order elements requires additional nodes, which results in more degrees of freedom and therefore more complex calculation.

In addition, triangular elements in general provide poorer results then quadrilateral elements, which are more robust and efficient. It is therefore preferable to use primarily quadrilateral elements in the mesh. However, in some regions of the model (e.g. sharp corners) it may be necessary to use triangles. Both shapes can be combined together under the condition that the compatibility is ensured. The approximation of the displacement is valid within the element as well as along the edge, which connects two elements together. Therefore only the same order elements can be combined (the linear triangle CST with the linear quadrilateral Q4 and the quadratic triangle LST with the quadratic quadrilateral Q8 or Q9).

Element	**CST**	**LST**	**Q4**
Nodes	3	6	4
DOFs	6	12	8
Displacement field	$u = c_1 + c_2x + c_3y$ $v = c_4 + c_5x + c_6y$	$u = c_1 + c_2x + c_3y + c_4x^2 + c_5xy + c_6y^2$ $v = c_7 + c_8x + c_9y + c_{10}x^2 + c_{11}xy + c_{12}y^2$	$u = c_1 + c_2x + c_3y + c_4xy$ $v = c_5 + c_6x + c_7y + c_8xy$
Strains	$\varepsilon_x = c_2$ $\varepsilon_y = c_6$ $\gamma_{xy} = c_3 + c_5$	$\varepsilon_x = c_2 + 2c_4x + c_5y$ $\varepsilon_y = c_9 + c_{11}x + 2c_{12}y$ $\gamma_{xy} = c_3 + c_8 + (c_5 + 2c_{10})x + (2c_6 + c_{11})y$	$\varepsilon_x = c_2 + c_4y$ $\varepsilon_y = c_7 + c_x$ $\gamma_{xy} = c_3 + c_6 + c_4x + c_8y$
Pros	simplicity in formulation	flexibility in modeling curved boundaries, fast convergence	simplicity in formulation, not constant strain
Cons	poor accuracy, slow convergence, constant strain, shear locking	computational complexity	poor accuracy, slow convergence, shear locking

Q8	**Q9**
8	9
16	18
$u = c_1 + c_2x + c_3y + c_4x^2 + c_5xy + c_6y^2 + c_7x^2y + c_8xy^2$ $v = c_9 + c_{10}x + c_{11}y + c_{12}x^2 + c_{13}xy + b_{14}y^2 + c_{15}x^2y + c_{16}xy^2$	$u = c_1 + c_2x + c_3y + c_4x^2 + c_5xy + c_6y^2 + c_7x^2y + c_8xy^2 + c_9x^2y^2$ $v = c_{10} + c_{11}x + c_{12}y + c_{13}x^2 + c_{14}xy + c_{15}y^2 + c_{16}x^2y + c_{17}xy^2 + c_{18}x^2y^2$
$\varepsilon_x = c_2 + 2c_4x + c_5y + 2c_7xy + c_8y^2$ $\varepsilon_y = c_{11} + c_{13}x + 2c_{14}y + c_{15}x^2 + 2c_{16}xy$ $\gamma_{xy} = c_3 + c_{10} + (c_5 + 2c_{12})x + (2c_6 + c_{13})y + c_7x^2 + 2(c_8 + c_{15})xy + c_{16}y^2$	$\varepsilon_x = c_2 + 2c_4x + c_5y + 2c_7xy + c_8y^2 + 2c_9xy^2$ $\varepsilon_y = c_{12} + c_{14}x + 2c_{15}y + c_{16}x^2 + 2c_{17}xy + 2c_{18}x^2y$ $\gamma_{xy} = c_3 + c_{11} + (c_5 + 2c_{13})x + (2c_6 + c_{14})y + c_7x^2 + 2(c_8 + c_{16})xy + c_{17}y^2 + 2c_9x^2y + 2c_{18}xy^2$
well approximation of bending deformation, fast convergence, efficient shape, flexibility in modeling curved boundaries	perfect approximation of bending deformation, less sensitive to imperfections, fast convergence, efficient shape, flexibility in modeling curved boundaries
computational complexity	computational complexity

Table 2.1: Summary of the element types

Chapter 3

Mesh Quality Assessment

Many structural design softwares contain packages, which are able to automatically generate FE meshes. In order to provide acceptable results when an analysis is carried out, a high-quality mesh is necessary. A low quality mesh may lead to misinterpretations of an analyzed model and further errors in the solution. However, there are no precise criteria how to define a quality of the mesh. It is a relative term and it always depends on a specific problem. It is also noteworthy that FEA provides only an approximate solutions. Therefore, quality alone cannot guarantee the accuracy of the analysis. Generally, it can be claimed that the mesh quality is good if the resulting solution quality is good. [11]

The geometry of the mesh is a good indicator of determining its quality. A regular mesh, which contains elements with same size and shape will probably exhibit good quality. But this kind of mesh is most likely generated on a structure, which is also regular (e.g. rectangular slab). However, common structures contain irregular areas (e.g. holes, sharp corners, connections between horizontal and vertical members). These regions are prone to containing ill-quality elements. Furthermore, in these areas there is a higher probability of stress peaks. In combination with the poor mesh it may lead to an incorrect results.

In order to ensure computational accuracy and correctness of the results it is highly recommended to check the mesh quality first. To do so, metrics, which evaluate different quality parameters of the mesh elements are presented in this Chapter. These metrics evaluate individually each element; therefore, a mutual interaction of the elements is not taken into account. An acceptable range can be set for each metric to point out ill-quality elements. But as it was mentioned earlier, this range may vary case by case and it is always up to an engineer to make the judgment. Attention needs to be paid to the areas of interest or the areas with high concentration of poor-quality elements.

3.1 Configuration

The configuration of an element refers to the arrangement of its nodes. Just by looking at the layout of the element we are able to point out poorly formed elements. Various configurations

of a quadrilateral in 2D plane are shown in Figure 3.1: **(a)** *convex positive*, **(b)** *convex negative*, **(c)** *non-convex*, **(d)** *degenerated*, **(e)** *self-intersecting*.

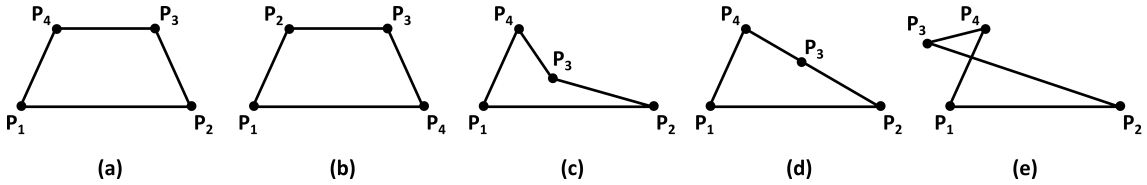


Figure 3.1: Various configurations of quadratic element, according to [11]

In order to categorize the configuration of a quadrilateral, the parameter called *surface area* is introduced. This parameter represents the area of the element and by its definition it can be obtained only for a triangle. In case of a quadrilateral, it is a sum of surface areas of two triangles formed by considering one of its diagonal (Figure 3.2).

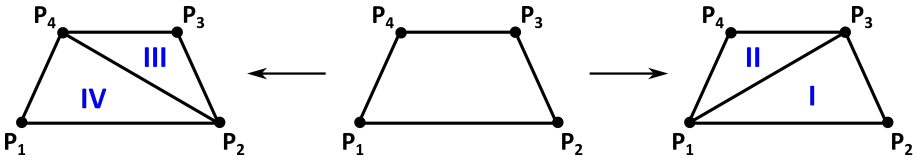


Figure 3.2: Partial surface areas of a quadrilateral, according to [11]

The surface area of corresponding triangles is calculated as a half of the determinant of the matrix, which considers vertex coordinates:

$$S_{I} = \frac{1}{2}\begin{vmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{vmatrix}, \qquad S_{II} = \frac{1}{2}\begin{vmatrix} x_3 - x_4 & x_3 - x_1 \\ y_3 - y_4 & y_3 - y_1 \end{vmatrix}, \tag{3.1}$$

$$S_{III} = \frac{1}{2}\begin{vmatrix} x_3 - x_4 & x_4 - x_2 \\ y_3 - y_4 & y_4 - y_2 \end{vmatrix}, \qquad S_{IV} = \frac{1}{2}\begin{vmatrix} x_2 - x_1 & x_4 - x_2 \\ y_2 - y_1 & y_4 - y_2 \end{vmatrix}.$$

Two triangles allow the surface area calculation of the quadrilateral, while four triangles are necessary to check the configuration of the element [11] (see Figure 3.1):

(a) **convex positive** - all four surface areas are positive,

(b) **convex negative** - all four surface areas are negative (reverse numbering of the nodes),

(c) **non-convex** - one surface area is negative (or one surface area is positive in case of reverse numbering of the nodes),

(d) **degenerated** - one surface area is equal to zero,

(e) **self-intersecting** - two surface areas are negative.

In order to ensure the quality of the mesh it is always desirable to use convex quadrilaterals [4]. Other mentioned configurations bring a distortion into the element.

3.2 Relative midpoint difference

In case of quadratic elements it is recommended to make sure that additional side-nodes are located in the middle of the corresponding side (and the internal node of Q9 element is in the intersection of straight lines connecting the opposite midpoints). Quadratic elements should not be initially deformed in most cases. Exceptions are regions where quadratic elements describe curved edges (Figure 3.3). In such case the side node of the side adjacent to the edge adjusts to the shape. That leads to better approximation of the real shape. However, it should be ensured that the difference or the curvature is not too large. It is always preferable to use dense mesh around curved edges in order to avoid this problem.

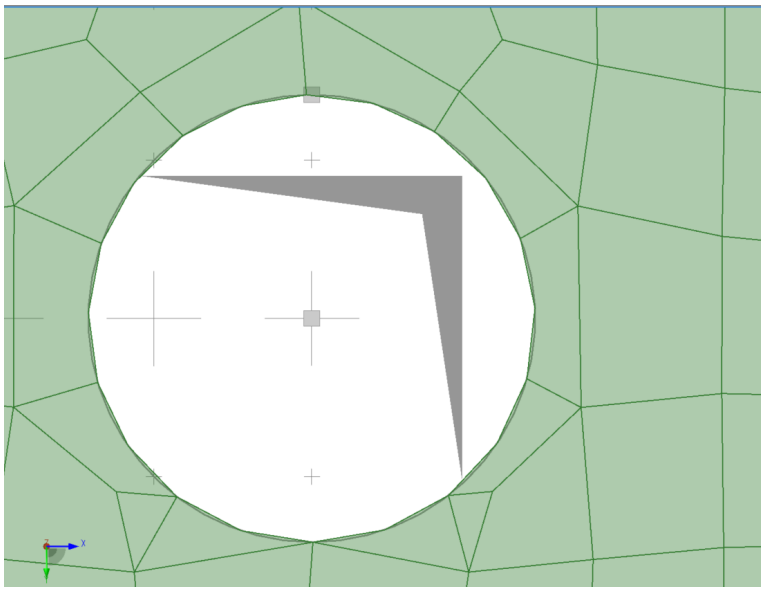


Figure 3.3: Quadratic elements describing curved boundary of a hole

Relative midpoint difference is calculated as:

$$\mathrm{RelativeMidpointDifference} = \frac{\triangle \mathrm{L}}{\mathrm{L}}, \tag{3.2}$$

where $\triangle L$ stands for difference between additional node and actual middle-point of corresponding side. The length of the corresponding side is denoted L.

3.3 Aspect Ratio

The aspect ratio describes the difference between characteristic dimensions of the element. There are several approaches of how to calculate it. The simplest one defines the aspect ratio as a ratio between the longest and shortest dimension of the element:

$$\text{AspectRatio} = \frac{L_{max}}{L_{min}} \tag{3.3}$$

According to [14], the aspect ratio is calculated as

$$\text{AspectRatio} = \frac{L_{max}(L_1 + L_2 + L_3)}{4\sqrt{3}A} \qquad \text{for triangles} \tag{3.4}$$

and

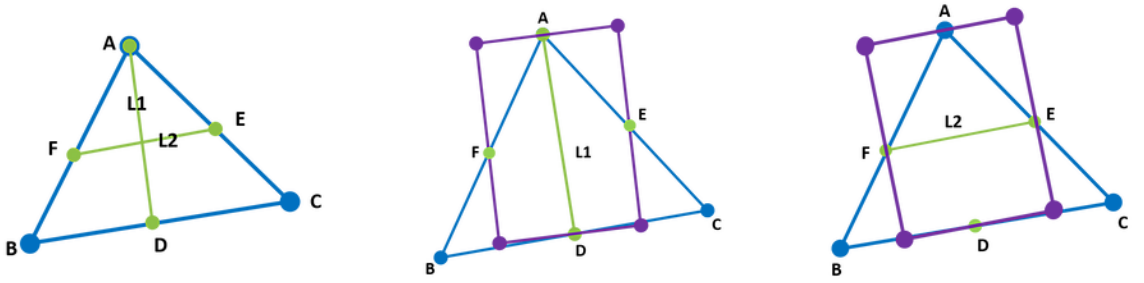
$$\text{AspectRatio} = \frac{L_{max}(L_1 + L_2 + L_3 + L_4)}{4A} \qquad \text{for quadrilaterals,} \tag{3.5}$$

where A stands for the area of the element and L_i corresponds to a side, $i = 1, ..., 4$.

The third method uses graphical procedures in order to determine the aspect ratio [2]. For triangles the following approach needs to be conducted:

- From the triangle ABC the vertex A is joined with the opposite side midpoint by line $L1$. Two remaining sides are connected at their midpoints by line $L2$ (Figure 3.4a).
- A parallel line to $L1$ is drawn at midpoints E and F. A perpendicular line to $L1$ is drawn at the midpoint D and the vertex A. Rectangle-1 is created (see Figure 3.4b).
- The same approach is repeated for line $L2$. A parallel line to $L2$ is drawn at the midpoint D and the vertex A. A perpendicular line to $L2$ is drawn at midpoints E and F. Rectangle-2 is created (see Figure 3.4c).
- The length ratio is calculated as a ratio between longest and shortest side for Rectangle-1 and Rectangle-2.
- The same procedure from the beginning is repeated for corners B and C. In total six rectangles are examined.
- The maximum length ratio out of these six rectangles is found.
- The aspect ratio of the triangle ABC is calculated as:

$$\text{AspectRatio} = \frac{\max(\text{Length ratio})}{\sqrt{3}} \tag{3.6}$$

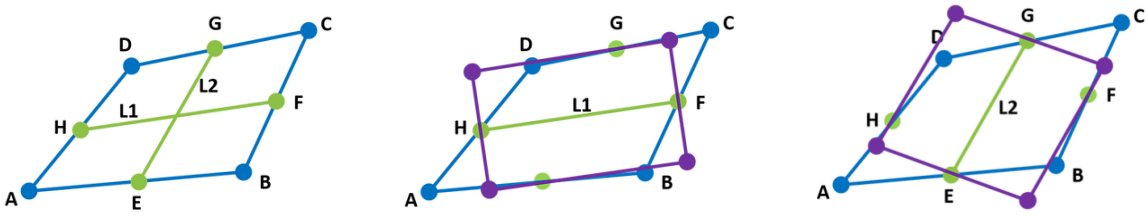


(a) (b) (c)

Figure 3.4: The approach calculating the aspect ratio of a triangle. (a) $L1$ and $L2$; (b) Rectangle-1; (c) Rectangle-2.[2]

A similar method is applicable to quadrilaterals:

- From the quadrilateral $ABCD$ the opposite midpoints are connected by lines $L1$ and $L2$ (Figure 3.5a).
- A perpendicular line to $L1$ is drawn at midpoints H and F. A parallel line to $L1$ is drawn at midpoints G and E. Rectangle-1 is created (see Figure 3.5b).
- The same approach is repeated for the line $L2$. A perpendicular line to $L2$ is drawn at midpoints G and E. A parallel line to $L2$ is drawn at midpoints H and F. Rectangle-2 is created (see Figure 3.5c).
- The length ratio is calculated as a ratio between longest and shortest side for Rectangle-1 and Rectangle-2. The maximum length ratio out of these two rectangles is found. The aspect ratio of the quadrilateral $ABCD$ is calculated according to Equation 3.6.

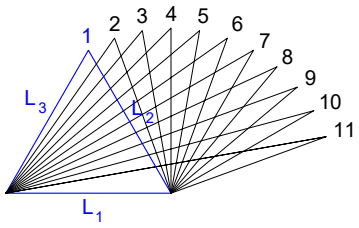


(a) (b) (c)

Figure 3.5: The approach calculating the aspect ratio of a quadrilateral. (a) $L1$ and $L2$; (b) Rectangle-1; (c) Rectangle-2. [2]

The aspect ratio of different isosceles triangles shown in Figure 3.6 is calculated according to three methods mentioned above (Equations 3.3, 3.4 and 3.6). Graphical display can be seen in Figure 3.7. The dependence of the aspect ratio on the length of the third side is analyzed. It is evident from the results that the ideal aspect ratio is equal to 1, which corresponds to the equilateral triangle (respectively square for quadrilateral elements). The same value is reached by all three methods. The aspect ratio then grows as the third dimension of the triangle is increasing. After a certain point the

deviation between the analyzed approaches can be observed. The aspect ratio calculated by Equations 3.4 and 3.6 has a similar shape and it grows exponentially. On the other hand, Equation 3.3 indicates gradual increase.



△	L_1, L_2	L_3	**AR1 (Eq. 3.3)**	**AR2 (Eq. 3.4)**	**AR3 (Eq. 3.6)**
1	1	1.00	1.00	1.00	1.00
2	1	1.15	1.15	1.11	1.12
3	1	1.29	1.29	1.24	1.26
4	1	1.41	1.41	1.39	1.46
5	1	1.53	1.53	1.59	1.67
6	1	1.64	1.64	1.83	1.96
7	1	1.73	1.73	2.15	2.33
8	1	1.81	1.81	2.60	2.82
9	1	1.88	1.88	3.27	3.62
10	1	1.93	1.93	4.39	4.89
11	1	1.97	1.97	6.60	7.39

Figure 3.6 & Table 3.1: Aspect ratio of isosceles triangles

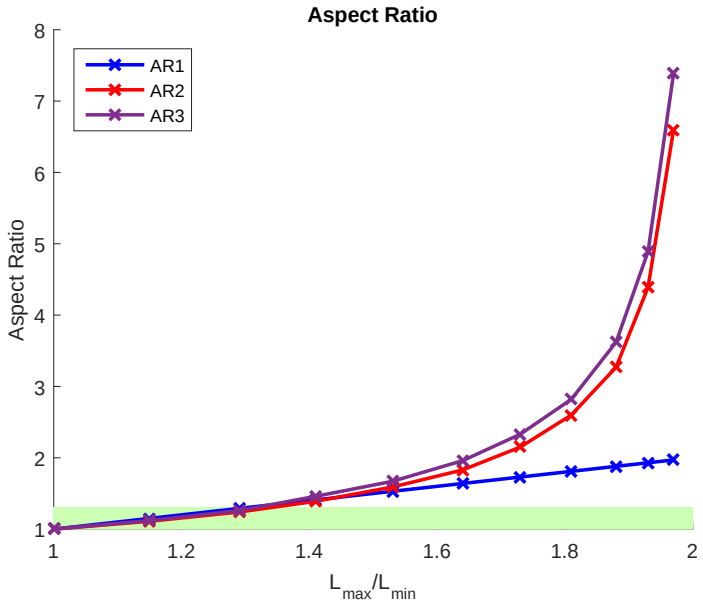


Figure 3.7: Aspect ratio for triangles (Figure 3.6) calculated by different approaches

According to the second method (Equation 3.4), the acceptable value of the aspect ratio is assumed to be 1.3 and lower for both triangles and quadrilaterals (Figure 3.7 in green) [14]. It can be noticed that the acceptable range for all three approaches is rather similar, therefore any approach can be used. Deviation starts to be obvious from value 1.5 and then it grows rapidly. However, that is beyond the range of acceptance.

3.3.1 Shear locking

In linear solid mechanics, an error called shear locking may affect linear elements with high aspect ratio. The defect is caused by the linear nature of these elements, which are not able to describe the state of bending properly. In FEA, deformation caused by bending can be interpolated only linearly by linear elements, even though the real behavior differs (Figure 3.8). By linear approximation the top and bottom edges remain straight although they are bent. This incorrect approximation introduces parasitic shear strain into the element, which is not actually there. As a result, the element appears stiffer then it is expected and the resulting displacements are smaller then they should be.

(a) Real behavior **(b)** Linear interpolation

Figure 3.8: Deformation in pure bending, according to [9]

This defect is significant for elements with high aspect ratio (when the bending acts over the longer side). In such cases, the deviation between the real and the assumed shape of the bent element is bigger, which can results in more fictitious shear strain introduced into the element and bigger error in the displacements.

Shear locking can be avoided by using higher order elements. Quadratic elements can interpolate bending well. Another option is to use denser meshes with smaller elements. [5]

3.4 Skewness

The skewness is an angular quality measure of the element with respect to the angles of an ideal element (square and equilateral triangle). The method of normalized angle deviation defines the skewness as the maximum of two values:

$$\text{Skewness} = \max\left[\frac{\theta_{\max} - \theta_e}{180^\circ - \theta_e}, \frac{\theta_e - \theta_{\min}}{\theta_e}\right], \tag{3.7}$$

where θ_{max} and θ_{min} stand for the maximum and the minimum angle of the element, respectively. The equi-angular element yields the angle θ_e to 90° for a quadrilateral and 60° for a triangle. [2]

The skewness ranges from 0, which is represented by the equi-angular element, to 1. It is assumed that acceptable elements have the skewness in the range from 0 to 0.5.

Figure 3.9 shows the skewness of triangles in relation to the size of the maximum and the minimum angle. Value 0 refers to the ideal equi-angular triangle, whereas increasing skewness relates to more tapered triangle. From the graph can be seen that the skewness of a triangle is solely governed by the size of the minimum angle. For example all triangles with the minimum angle equal to 30° have the skewness 0.5, which is also recommended as a limit of the acceptable quality (in green in Figure 3.9) .

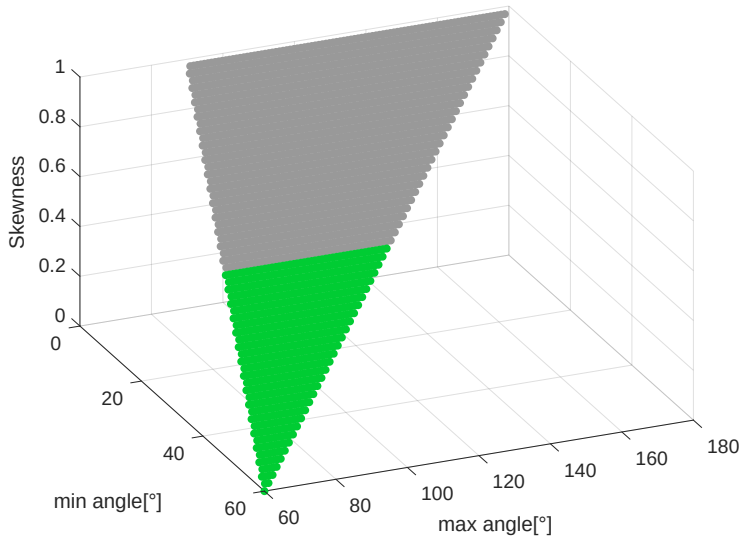


Figure 3.9: Skewness of triangles in dependence on the maximum and the minimum angle

Figure 3.10 represents a relation between the skewness and the maximum and the minimum angle of convex quadrilaterals. Value 0 belongs to the ideal square-shaped element. Increasing skewness corresponds to more tapered elements with a bigger difference between the internal angles. Skewness is here dependent on both the minimum and the maximum angles.

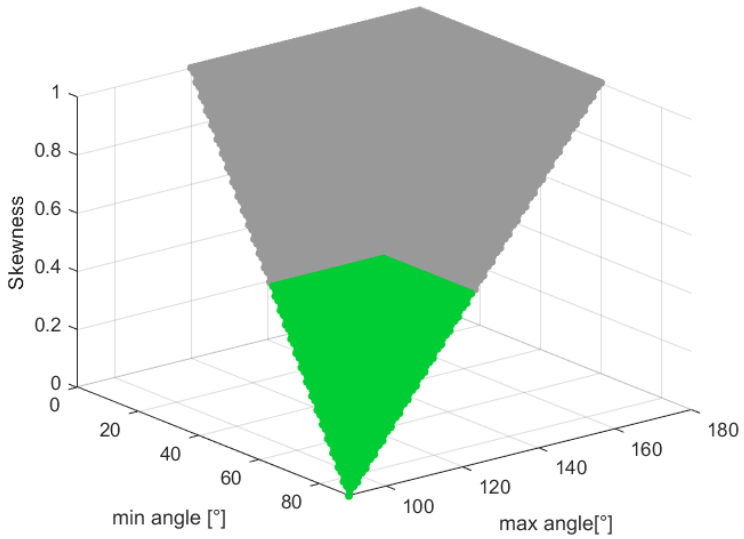


Figure 3.10: Skewness of quadrilaterals in dependence on the maximum and the minimum angle

Figure 3.11 demonstrates the dependency of the skewness on the minimum angle of both triangles and convex quadrilaterals. The acceptable range is depicted in green. In case of triangles there is a linear relation between these two properties and no effect of the maximum angle. With regard to quadrilaterals, skewness depends specifically on the element. It is different for each case.

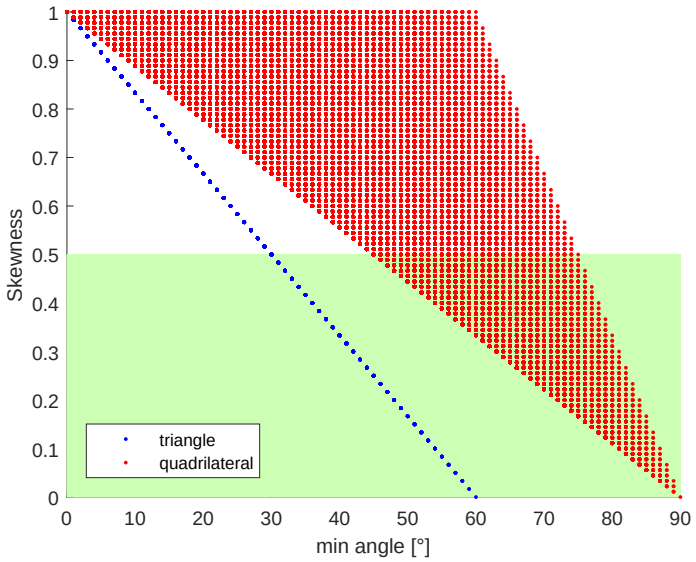


Figure 3.11: Skewness of quadrilaterals in dependence on the minimum angle

3.5 Jacobian ratio

The Jacobian ratio describes the element's deviation from a perfect shape. The perfect shape is represented in the *natural* (mapped) coordinate system, which ranges from -1 to 1 in all directions and depends on the element type. The transformation of a Q4 element is shown in Figure 3.12. The Jacobian matrix is necessary in order to perform the mapping. Its determinant represents the stretching of the parametric space required to fit the natural coordinate space.

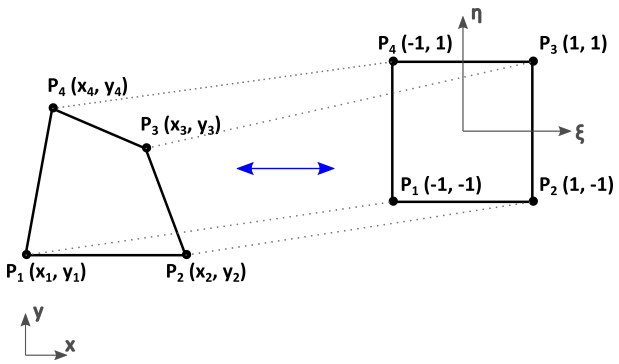


Figure 3.12: The Q4 element mapped into the square element in coordinate system $\xi - \eta$, according to [16]

The Jacobian matrix is used as a scale factor to transform from one coordinate system to another (e.g. physical length dx to $d\xi$). The Jacobian matrix for 2D elements is formulated as:

$$\mathrm{J} = \begin{pmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{pmatrix} = \begin{pmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{pmatrix} \tag{3.8}$$

The components on the main diagonal represent the relation between the physical and the mapped coordinates: x vs. ξ and y vs. η. The off-diagonal components express the skewness of the element. The determinant of the Jacobian matrix $det(J)$ describes the ratio of the area measured in the physical xy-coordinate system to the area measured in the natural $\xi\eta$-coordinate system. [16]

The components of the Jacobian matrix are calculated as:

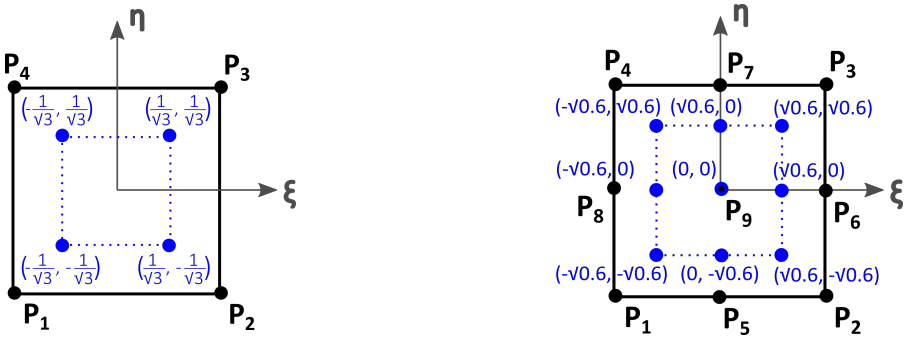
$$\begin{aligned} J_{11} &= \frac{\partial x}{\partial \xi} = \frac{\partial}{\partial \xi} \sum_{i=1}^{n} N_i x_i, \\ J_{12} &= \frac{\partial y}{\partial \xi} = \frac{\partial}{\partial \xi} \sum_{i=1}^{n} N_i y_i, \\ J_{21} &= \frac{\partial x}{\partial \eta} = \frac{\partial}{\partial \eta} \sum_{i=1}^{n} N_i x_i, \\ J_{22} &= \frac{\partial y}{\partial \eta} = \frac{\partial}{\partial \eta} \sum_{i=1}^{n} N_i y_i, \end{aligned} \tag{3.9}$$

where $N_i(\xi, \eta)$ is a shape function associated with the node i of the element. The shape functions of Q4 and Q9 elements are listed in the Table 3.2.

Q4	Q9
$N_1 = \frac{1}{4}(1-\xi)(1-\eta)$ $N_2 = \frac{1}{4}(1+\xi)(1-\eta)$ $N_3 = \frac{1}{4}(1+\xi)(1+\eta)$ $N_4 = \frac{1}{4}(1-\xi)(1+\eta)$	$N_1 = \frac{1}{4}(\xi-1)(\eta-1)\xi\eta$ $N_2 = \frac{1}{4}(\xi+1)(\eta-1)\xi\eta$ $N_3 = \frac{1}{4}(\xi+1)(\eta+1)\xi\eta$ $N_4 = \frac{1}{4}(\xi-1)(\eta+1)\xi\eta$ $N_5 = -\frac{1}{2}(\xi^2-1)(\eta-1)\eta$ $N_6 = -\frac{1}{2}(\xi+1)(\eta^2-1)\xi$ $N_7 = -\frac{1}{2}(\xi^2-1)(\eta+1)\eta$ $N_8 = -\frac{1}{2}(\xi-1)(\eta^2-1)\xi$ $N_9 = (\xi^2-1)(\eta^2-1)$

Table 3.2: The shape functions of Q4 and Q9 elements (numbering of the nodes corresponds to Figure 3.13)

The Jacobian matrix is evaluated at the integration points i.e. *Gauss points*. The numerical integration is carried out at these points in order to obtain the stiffness matrix. The number of Gauss points depends on the element type. It is assumed that they are located at $\xi, \eta = \pm\frac{1}{\sqrt{3}}$ for the Q4 element and $\xi, \eta = \pm\sqrt{0.6},\ 0$ for the Q9 element (see Figure 3.13). [5]



(a) Q4 guadrilateral

(b) Q9 quadrilateral

Figure 3.13: The location of the Gauss points

The examples of the different quadrilateral elements (Figure 3.14) are analyzed. The resulting Jacobian matrices, their determinants and the Jacobian ratios (which is explained later in Equation 3.10) are summarized in Table 3.3.

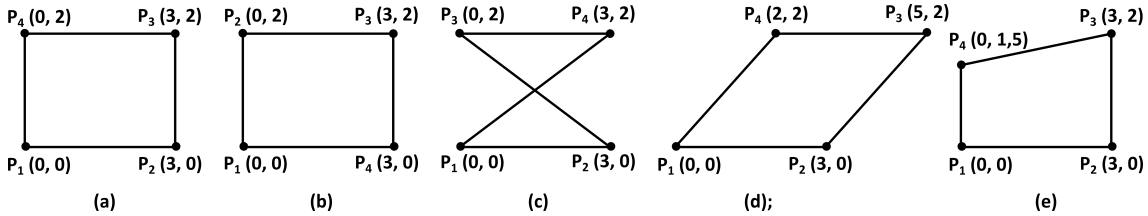


Figure 3.14: The examples of the quadrilateral elements, according to [16]

Case **(a)** displays a rectangular element with a counterclockwise numbering of the nodes. The Jacobian matrix transfers the rectangle in the global coordinate system into a square in the coordinate system $\xi\eta$. The main diagonal components are a scale factor between the coordinate systems and the $\xi\eta$ off-diagonal members are equal to 0. The determinant of the Jacobian matrix is independent of the values of ξ and η and it is a positive number. It shows that the area of the rectangle is 1.5 times larger than the area of the mapped square.
Case **(b)** shows the same element but with a clockwise numbering of the nodes. The main diagonal and the off-diagonal of the Jacobian matrix components are switched and the determinant is negative. It indicates that the nodes are not arranged with respect to the element's formulation.
Case **(c)** is again represented by the negative determinant of the Jacobian matrix. The nodes are not properly arranged.
In case **(d)** a parallelogram is shown. Two pairs of parallel sides are equivalently transferred into the mapped coordinate system independently of the values ξ and η. The skewness is indicated by the off-diagonal term. Hence, the other is equal to zero, therefore it has no influence on the determinant, which again shows that the area is 1.5 times larger than the area of the mapped square.
Case **(e)** represents a distorted element. The numbering of the nodes is in the correct order and the element has off-diagonal terms in the Jacobian matrix. Additionally, the determinant now depends on the values of the mapped coordinates ξ, η (either or both, depends on the skew side of the element). This affects the magnitude of the determinant, which varies for different points of the element.

The examples shown above sum up how the distortion of an element affects the Jacobian matrix. That can be used as an indicator of the mesh quality. It is desirable for the element to be defined in the proper order (negative determinant indicates reverse numbering of the nodes). For other cases, however, the Jacobian matrix depends on the element size (a larger rectangular element will have larger determinant, yet it is mapped to the idealized isoparametric element equally as a smaller rectangle with the same side ratio). As a result, using solely the determinant of the Jacobian matrix as a mesh quality parameter is not appropriate. Instead, the *Jacobian ratio* is used. [16]

The Jacobian ratio is calculated as:

$$\text{JacobianRatio} = \frac{\min(\det(\mathrm{J}))}{\max(\det(\mathrm{J}))} \tag{3.10}$$

It is defined as a ratio between the smallest and the largest determinant of the Jacobian matrix evaluated in the Gauss points of the element.

The third row in Table 3.3 evaluates the Jacobian ratio for the examples shown in Figure 3.14. The Jacobian Ratio for cases **(a)** and **(b)** is 1 regardless of the order of the node numbering. The value 1 describes the perfectly mapped element.

Case **(c)** shows a negative Jacobian ratio, which indicates a distorted element shape.
At the first glance it is evident that the parallelogram in case **(d)** is of lower quality then the previously mentioned rectangle in case (a). However, the Jacobian ratio of both elements is 1, which corresponds to all parallelograms regardless of the sizes of angles or the side lengths. For example a square and a rectangle with one side much longer then the other will have the same Jacobian ratio although the quality parameters are not the same. A different quality check is required for these cases.
A skewed element is illustrated in case **(e)**. The Jacobian ratio is smaller then 1, which indicates a distortion. Even more skewed elements will exhibit even smaller value of the Jacobian ratio.

	(a)	**(b)**	**(c)**	**(d)**	**(e)**
J	$\begin{pmatrix} 1.5 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 \\ 1.5 & 0 \end{pmatrix}$	$\begin{pmatrix} -1.5\eta & 0 \\ -1.5\xi & 1 \end{pmatrix}$	$\begin{pmatrix} 1.5 & 0 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 1.5 & 0.125 + 0.125\eta \\ 0 & 0.875 + 0.125\xi \end{pmatrix}$
det(J)	1.5	-1.5	-1.5η	1.5	$1.3125 + 0.1875\xi$
JR	1	1	-1	1	0.8476

Table 3.3: The Jacobian matrix, determinant and Jacobian ratio of analyzed examples (Figure 3.14)

In conclusion, the Jacobian ratio ranges between -1 to 1 where 1 corresponds to a perfectly mapped element. As mentioned before, if the element is well-formed and not heavily distorted, the Jacobian ratio should be greater then 0. Ill-formed elements have negative Jacobian ratio. In general, it is recommended to ensure that the Jacobian ratio is higher then 0.5. An attention needs to be paid to the quadrilaterals with two parallel sides and big difference in the side lengths or the sizes of the internal angles. All parallelograms have the Jacobian ratio equal to 1 although they may be far from a perfectly shaped element. The quality should be evaluated by using different metrics for these cases (e.g. aspect ratio or skewness).

Furthermore, this analysis is valid only for the quadrilateral elements. The triangular CST elements are constant strain in nature. The Jacobian Ratio for them is assumed to be 1 in all cases [16]. For the LST triangles it is possible to calculate the Jacobian ratio using the isoparametric formulation and three Gauss points inside the element; however, for many cases the Jacobian ratio results also in 1 regardless of its quality. Therefore other parameters for the quality checks are recommended.

3.5.1 Q4 and Q9 elements

It is proven in Appendix A that the effect of the additional nodes 5, 6, 7, 8 and 9 can be neglected if the Q9 element is "properly" formed (i.e. nodes 5, 6, 7 and 8 are located in the middle of each side and node 9 lies in the intersection of straight lines, which connect the opposite midpoints). The influence of these nodes will reduce the formula for calculating each member of the Jacobian matrix to the form, which corresponds to the Q4 element.

However, it is assumed that the position of the Gauss points differs for both element types (Figure 3.13). As a result, the Jacobian ratio for a "properly" formed Q9 element is different compared to a Q4 element. The trend can be seen in Figure 3.16 where the trapezoids with different fourth

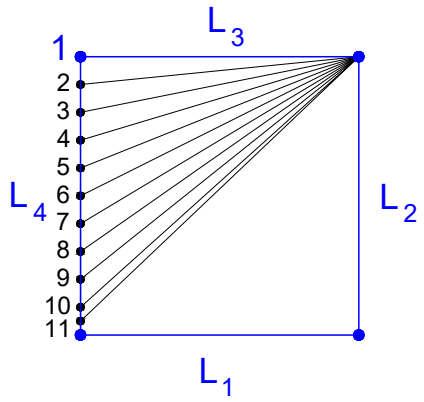
vertices are compared. The Jacobian ratios are calculated for both element types (Q4 and Q9) and displayed in the graph in relation with the side ratio L_4/L_1. The square is represented by the Jacobian ratio equal to 1 for both cases. The value of the Jacobian ratio varies for the distorted elements. Larger Gauss point coordinates for Q9 elements cause larger differences in evaluated determinants and therefore smaller ratios of the minimum and the maximum value. The Jacobian ratio of a Q9 element decreases faster.

A similar metric to the Jacobian ratio is the *distortion*. This quality parameter also describes how well-behaved the mapping is from the physical to the natural coordinates. The distortion is calculated as:

$$\text{Distortion} = \frac{4\min(\det(\mathrm{J}))}{\mathrm{A}}, \tag{3.11}$$

where 4 stands for the area of the natural quadrilateral and A is the physical area of the element. The ratio of areas is multiplied by the minimum determinant of the Jacobian matrix evaluated in the Gauss points. The distortion, as well as the Jacobian ratio ranges between -1 to 1 where 1 refers to the square. The acceptable range is assumed to start from 0.5. [14]

The distortion of the quadrilaterals Q4 and Q9 from Figure 3.15 was evaluated. Comparison with the Jacobian ratio is shown in Figure 3.16. The Jacobian ratio seems to be the more strict parameter because it decreases faster. Green area depicts the acceptable range of the both mesh quality metrics.



□	L_4/L_1	JR (Q4)	JR (Q9)	D (Q4)	D (Q9)
1	1	1.00	1.00	1.00	1.00
2	0.9	0.94	0.92	0.97	0.96
3	0.8	0.88	0.84	0.94	0.91
4	0.7	0.82	0.76	0.90	0.86
5	0.6	0.75	0.68	0.86	0.81
6	0.5	0.67	0.59	0.81	0.74
7	0.4	0.60	0.50	0.75	0.67
8	0.3	0.52	1.41	0.69	0.58
9	0.2	0.44	0.32	0.62	0.48
10	0.1	0.36	0.22	0.53	0.37
11	0.05	0.31	0.18	0.48	0.30
12	0.01	0.28	0.14	0.43	0.24

Figure 3.15 & Table 3.4: The Jacobian ratio and the distortion of the Q4 and the Q9 elements

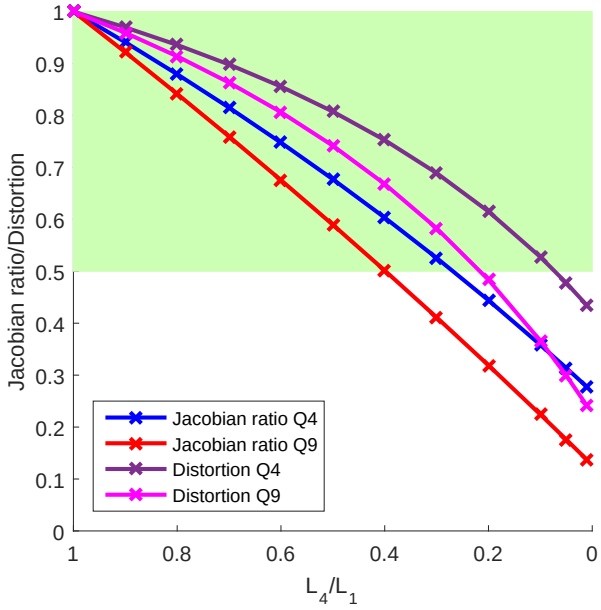


Figure 3.16: The Jacobian ratio and the distortion in dependence on L_4/L_1 (Figure 3.15)

3.6 Summary

Figure 3.17 summarizes the distorted element shapes that might reduce the accuracy of the analysis.

(a) large aspect ratio - intercepted by the aspect ratio (Section 3.3)

(b) parallel deviation - intercepted by the Jacobian ratio, the distortion (Section 3.5)

(c) high skewness - intercepted by the skewness (Section 3.4)

(d) triangular quadrilateral [1] - intercepted by the configuration (Section 3.1)

(e) self-intersecting quadrilateral [1] - intercepted by the configuration (Section 3.1)

(f) off-center node [1] - intercepted by the relative midpoint difference (Section 3.2)

(g) strongly curved side [1] - intercepted by the relative midpoint difference (Section 3.2)

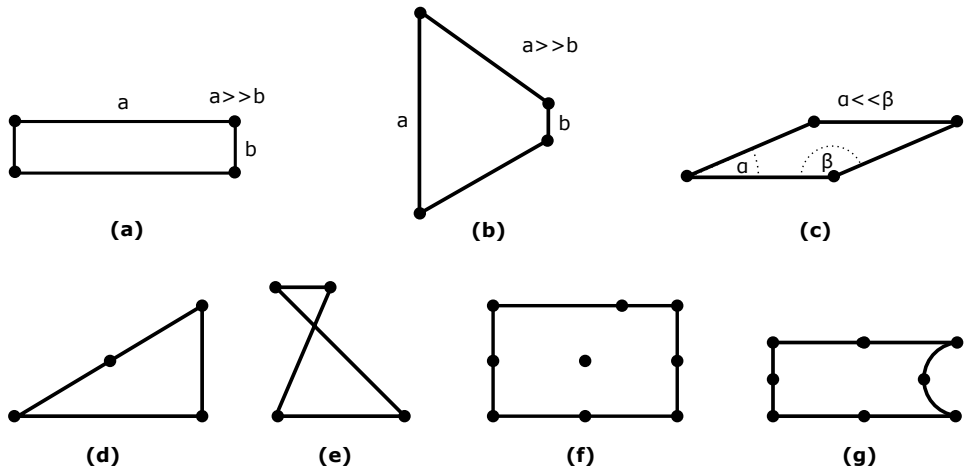


Figure 3.17: Element shape distortion, according to [9]

Table 3.5 summarizes the presented mesh quality metrics and their possible application, full range, acceptable range, relatively-acceptable range and unacceptable range. The ideal value or the state of the element is denoted in bold. Presented values were chosen with respect to the listed literature and they are used in the practical part of the thesis to provide an overall evaluation of the mesh quality according to the quality of its elements. The amount of distortion, however, varies with the element type, the mesh arrangement and the analyzed physical problem. Numerical limits are somewhat arbitrary. What is unacceptable in one situation may be acceptable in another [9]. Therefore, the classification according to these ranges serves rather as a warning to the user than as a rule that has to be complied with.

[1] Intercepted also by the Jacobian ratio and the distortion (Section 3.5).

Metric	Applicable to	Range	Acceptable range	Relatively-acceptable range	Unacceptable range
Configuration	□	**convex**, non-convex, degenerated, self-intersecting	**convex**	-	non-convex, degenerated, self-intersecting
Aspect ratio	□, △	$\langle \mathbf{1}, \infty \rangle$	$\langle \mathbf{1}, 1.3 \rangle$	$\langle 1.3, 1.5 \rangle$	$(1.5, \infty)$
Skewness	□, △	$\langle \mathbf{0}, 1)$	$\langle \mathbf{0}, 0.5 \rangle$	$\langle 0.5, 0.6 \rangle$	$(0.6, 1\rangle$
Jacobian ratio	□	$\langle -1, \mathbf{1} \rangle$	$\langle 0.5, \mathbf{1} \rangle$	$\langle 0.4, 0.5 \rangle$	$\langle -1, 0.4)$
Distortion	□	$\langle -1, \mathbf{1} \rangle$	$\langle 0.5, \mathbf{1} \rangle$	$\langle 0.4, 0.5 \rangle$	$\langle -1, 0.4)$
Relative midpoint difference	quadratic □, △	$\langle \mathbf{0}, \infty \rangle$	$\langle \mathbf{0}, 0.1 \rangle$	$\langle 0.1, 0.2 \rangle$	$(0.2, \infty)$

Table 3.5: The summary of the mesh quality metrics

Chapter 4

MeshEvaluator tool

This Chapter introduces a computational tool called **MeshEvaluator**, developed for the evaluating mesh quality of the plane structures. The tool was designed and implemented in the C++ programming language. The information about C++ code development were drawn from [10] and [15]. The programming conventions and the coding style comply with the recommendations introduced in [3].

The tool takes a *text* file of the mesh geometry generated by *RIB iTWO structure fem SLAB* as an input. Afterwards it calculates the above-mentioned mesh quality metrics and generates a *XML* output, which can be imported into the software *ParaView* in order to get a graphical visualization of the problem. Figure 4.1 shows the conceptual structure of the developed tool. The diagram represents classes, which were created and their interconnections.

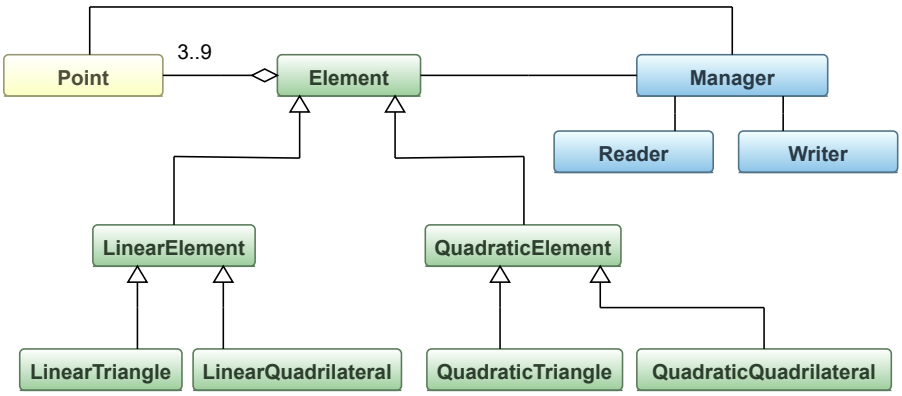


Figure 4.1: Class diagram of the MeshEvaluator

The class ***Point*** holds the data of the nodes of the mesh. By grouping the nodes, particular elements are created and they are operated by the base class ***Element***. The number of nodes (3 to 9) firstly assigns the elements to the derived classes ***LinearElement*** or ***QuadraticElement*** and then to ***LinearTriangle***/***LinearQuadrilateral*** or ***QuadraticTriangle***/***QuadraticQuadrilateral*** subclasses, respectively. The class ***Reader*** manages reading of the input *text* file and ensures the correct classification of found objects (points and elements). The class ***Manager*** creates both elements and points

according to their classification and provides an access to them at the later stage. Whereas the class ***Writer*** calculates the mesh quality metrics on each element and generates the output into the *XML* file. In addition, a ***UnitTest*** was developed in order to continuously check correctness of the results while the code was being developed.

Further specification of classes is introduced in the following Sections. Short description of their attributes and methods is given according to UML class representation presented in Figure 1.

4.1 Starting the program

Firstly, an explanation of a function main(), which starts the program, is given.

```
1  #include "stdafx.h"
2  #include "MeshEvaluator/Writer.h"
3  #include "MeshEvaluator/Manager.h"
4  #include "MeshEvaluator/Reader.h"
5
6  int _tmain(int argc, _TCHAR* argv[])
7  {
8    rtt::String inputFilePath =
9    _T("C:\\Users\\bulickova\\Projects\\MeshEvaluator\\subs");
10   MeshEvaluator::Manager manager;
11   MeshEvaluator::Reader reader(inputFilePath);
12   reader.readInput(manager);
13   rtt::String outputFilePath =
14   _T("C:\\Users\\bulickova\\Projects\\MeshEvaluator\\some.xml");
15   if (argc > 2)
16     outputFilePath = argv[2];
17   MeshEvaluator::Writer writer(outputFilePath, manager);
18   writer.writeOutput();
19
20   return 0;
21 }
```

Figure 4.2: main() function

The MeshEvaluator tool was created in *Microsoft Visual Studio Professional 2013* using the C++ programming language. When any C++ program starts, it executes initialization code and calls a special function main(), where the primary-executed code for the program is. The function main() for the developed tool is depicted in Figure 4.2. The following algorithm introduces the main ideas behind the code:

Algorithm 1 main() function

1: set the input file path
2: create the manager object of the **Manager** class
3: create the reader object of the **Reader** class with the input file path as a parameter
4: execute readInput() function on object reader with the manager as a parameter
5: set the output file path
6: create the writer object of the **Writer** class with the output file path and manager as parameters
7: execute writeOutput() function

The sequence diagram of the developed tool MeshEvaluator is depicted in Figure 4.3. It shows how the operations are carried out when the tool starts to run. The operations between classes and the mutual collaboration is captured in the sequential order. The diagram indicates processes, which are governed by the main() function, but carried out elsewhere.

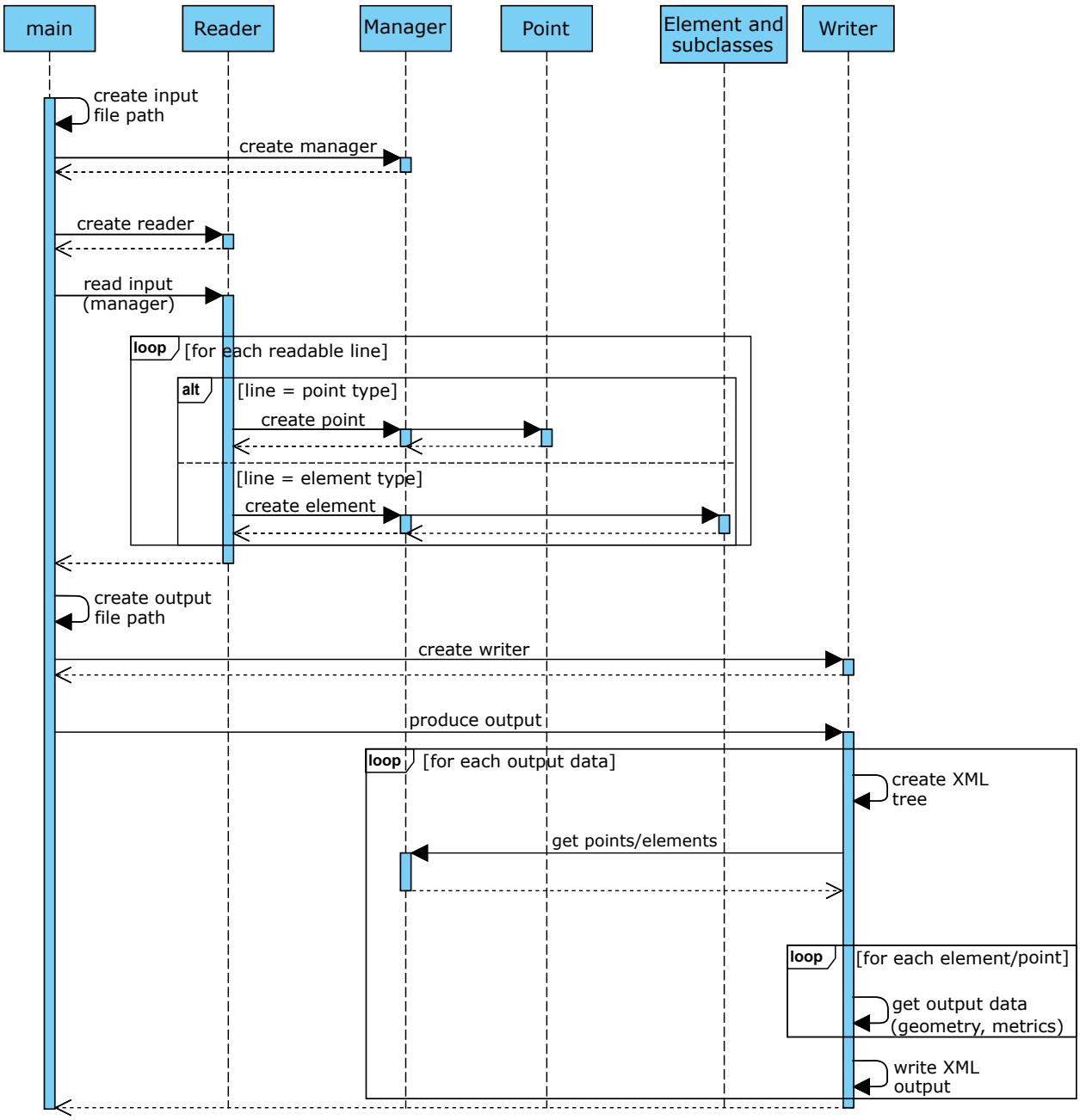


Figure 4.3: Sequence diagram of the MeshEvaluator

4.2 Class Point

The class ***Point*** is used as a container for the nodes of the mesh. Each point is represented by its ID (identifier) and spatial coordinates x, y and z. These values express private attributes of the class, not settable outside of it (m_id, m_coordinateX, m_coordinateY, m_coordinateZ). The public functions getId(), getCoordinateX(), getCoordinateY() and getCoordinateZ() are used to obtain these properly encapsulated values. The representation of the class is shown in Figure 4.4.

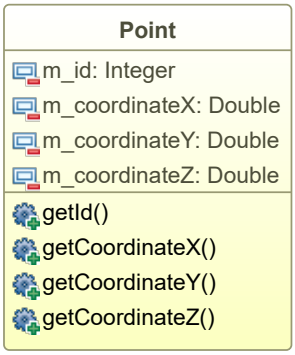


Figure 4.4: Class ***Point*** representation

An object of the class ***Point*** has following parameters:

Point*(ID, coordinate x, coordinate y, coordinate z)*,

where *ID* is expressed by an integer and *coordinates* by a double-precision floating-point number.

4.3 Class Element and its inherited classes

The base class ***Element*** and the inherited classes are displayed in Figure 4.5. The inheritance is a concept used in object oriented programming, which allows to define a class in terms of another class by using the same interface. In this case, the elements can be sorted into the linear and the quadratic elements, which can further be divided into the triangles and the quadrilaterals. The inherited classes ***LinearTriangle***, ***LinearQuadrilateral***, ***QuadraticTriangle*** and ***QuadraticQuadrilateral*** use the data members and the member functions of the base class ***Element***. Each element has attributes: m_id, which represents the element's ID (identifier) and m_points, which describe the points forming the element. The attributes are protected, therefore they can be accessed only from the base class itself and its subclasses.

Figure 4.5: Class ***Element*** hierarchy representation

More specifically, an object of the ***Element*** subclass has following parameters depending on its classification:

LinearTriangle$(ID,\ point_1, point_2, point_3)$,
LinearQuadrilateral$(ID,\ point_1, point_2, point_3, point_4)$,
QuadraticTriangle$(ID,\ point_1, point_2, point_3, point_4, point_5, point_6)$,
QuadraticQuadrilateral$(ID,\ point_1, point_2, point_3, point_4, point_5, point_6, point_7, point_8, point_9)$,

where *ID* of the element is expressed by an integer and *points* refer to the class ***Point*** where they are stored and further specified.

The public functions of the class ***Element*** are of various types, as explained in Figure 1:

- **General** - the function is both declared and defined in the base class and therefore the implementation is the same in all the subclasses - considerTolerance(...), getPointDistance(...), getSideMidpoint(...), getSurfaceArea(...), getTriangleAngle(...), getElementPoints(). These functions have the auxiliary meaning and they are used in order to simplify other functions implemented on the specific elements. Some of these auxiliary functions are **static**, which means that they are independent of any particular object of the class (i.e. element).
- **Abstract** - the function is declared in the base class but defined individually in the subclasses (the implementation for each subclass is different) - getNumberOfNodes(), getArea(), getMidpointDifference(), getConfiguration(), getAspectRatio(), getSkewness(), getJacobianRatio(), getDistortion().

In addition, each subclass consists of private functions getSideLength...(). Also, subclasses of quadrilaterals are using function getJacobianDeterminant(). These functions are only for the internal usage in the subclass itself.

The functions used for calculating the mesh quality metrics are further described in following Subsections.

4.3.1 Configuration

The configuration of the element is determined according to Section 3.1. The classification is represented by a numerical values due to its easier graphical display. The possible outcomes are:

Indication	Configuration
1	convex quadrilateral
2	triangle
3	non-convex quadrilateral
4	degenerated quadrilateral
5	self-intersecting quadrilateral

If the element is a triangle, the configuration is automatically set to 2. For the quadrilaterals, Algorithm 2 is used in order to specify the configuration.

Algorithm 2 getConfiguration() function for quadrilaterals

get surface areas S (Equation 3.1)
if $\forall s \in S > 0$ **or** $\forall s \in S < 0$ **then**
    configuration $\leftarrow 1$
else if $(\exists s \in S < 0$ **and** $\forall s' \in \{S \setminus s\} > 0)$ **or** $(\exists s \in S > 0$ **and** $\forall s' \in \{S \setminus s\} < 0)$ **then**
    configuration $\leftarrow 3$
else if $\exists s \in S = 0$ **then**
    configuration $\leftarrow 4$
else if $\exists s_1, s_2 \in S < 0$ **and** $\forall s' \in \{S \setminus \{s_1 \ \ s_2\}\} > 0$ **then**
    configuration $\leftarrow 5$
end if

4.3.2 Relative midpoint difference

The position of the midside nodes is checked for quadratic elements. The relative midpoint difference is further described in Section 3.2. Function getMidpointDifference() checks all midpoints of the element and returns the maximum value of the relative deviation (Algorithm 3).

Algorithm 3 getMidpointDifference() function for quadratic elements

get middle points of all sides (and middle point of the element for quadrilateral)
get relative deviations of the middle points (Equation 3.2)
get maximum relative deviation

4.3.3 Aspect ratio

The aspect ratio is described in Section 3.3. The function is applicable for all plane elements. The process of the corresponding function is summarized by Algorithm 4.

Algorithm 4 getAspectRatio() function

get side lengths,the area
get maximum side
get aspect ratio (Equations 3.4 or 3.5)

4.3.4 Skewness

The description of mesh quality parameter skewness is given in Section 3.4. The calculation is applicable for triangles and convex quadrilaterals. Remaining distorted plane elements have the skewness assigned to 1 (the worst value). Corresponding function follows steps of Algorithm 5.

Algorithm 5 getSkewness() function

if the element is a triangle **or** a convex quadrilateral **then**
    get internal angles
    get maximum and the minimum angle
    get skewness (Equation 3.7)
else
    skewness $\leftarrow 1$
end if

4.3.5 Jacobian ratio

The Jacobian ratio is further specified in Section 3.5. The usage of this parameter is restricted to quadrilaterals with the constant coordinate z. Algorithm 6 describes the process of the calculation.

Algorithm 6 getJacobianRatio() function for quadrilaterals

1: **for** $i = 1$ to *the number of Gauss points*
2: get members of the Jacobian matrix (Equation 3.9)
3: get Jacobian determinant
4: **end for**
5: return Jacobian determinant in Gauss points
6: get maximum and minimum determinant of the Jacobian matrix
7: get Jacobian ratio (Equation 3.10)

4.3.6 Distortion

The quality parameter distortion is similar to the Jacobian ratio. The same restrictions are valid. The procedure of corresponding function is described by Algorithm 7.

Algorithm 7 getDistortion() function for quadrilaterals

1: get area
2: **for** $i = 1$ to *the number of Gauss points*
3: get members of Jacobian matrix (Equation 3.9)
4: get Jacobian determinant
5: **end for**
6: return Jacobian determinant in Gauss points
7: get minimum determinant of the Jacobian matrix
8: get distortion (Equation 3.11)

4.3.7 Overall Evaluation

The function getOverallEvaluation() provides an evaluation for all elements according to Table 3.5. It calculates all above mentioned metrics and evaluates the conditions given in Algorithm 8. The classification of the element is done by numerical values. Possible outcomes are:

Indication	Classification
1	acceptable quality
0.5	relatively acceptable quality
0	unacceptable quality

Algorithm 8 getOverallEvaluation() function

1: get configuration, aspect ratio, skewness, Jacobian ratio, distortion
2: **if** (configuration $\leftarrow$ 1 **or** 2) **and** aspect ratio $\leq$ 1.3 **and** skewness $\leq$ 0.5 **and** Jacobian ratio $\geq$ 0.5 **and** distortion $\geq$ 0.5
3: overall evaluation $\leftarrow$ 1
4: **else if** aspect ratio $\leq$ 1.5 **and** skewness $\leq$ 0.6 **and** Jacobian ratio $\geq$ 0.4 **and** distortion $\geq$ 0.4
5: overall evaluation $\leftarrow$ 0.5
6: **else**
7: overall evaluation $\leftarrow$ 0

4.4 Class Reader

The class **Reader** consists of the private attribute m_inputFilePath, which holds the path to the input *text* file generated by *RIB iTWO structure fem SLAB* and the public function readInput(Manager&), which reads the input file (Figure 4.6). Appendix B shows how the input file is created in more detail. The input *text* file, which describes the mesh geometry is captured in Figure B.4.

Figure 4.6: Class ***Reader*** representation

The main purpose of the class ***Reader*** is to go through the input file line by line and evaluate condition for each line in order to classify containing information into following:

- line, which is skipped,
- line, which describes **Point**,
- line, which describes **Linear quadrilateral**,
- line, which describes **Linear triangle**,
- line, which describes **Quadratic quadrilateral**,
- line, which describes **Quadratic triangle**.

Algorithm 9 describes the function readInput(Manager&), which is continuously applied on each line of the input file. The binary decision variable `doRead` is created in order to exclude unneeded lines and it is set to the value *false* at the beginning. The string `line` is declared in order to represent the content of each line.

Algorithm 9 readInput(Manager&) function, which is applied on each line

```
1:  if line ← "ND-END-COOR" then
2:      doRead ← false
3:  if line ← "EL-END-TOP" then
4:      break
5:  if line ← "ND-COOR" or line ← "EL-TOP" then
6:      doRead ← true
7:      continue
8:  if line ≠ ∅ and doRead ← true then
9:      if "P04Q" ⊆ line then
10:         create linear quadrilateral
11:     else if "P03T" ⊆ line then
12:         create linear triangle
13:     else if "P09Q" ⊆ line then
14:         create quadratic quadrilateral
15:     else if "P06T" ⊆ line then
16:         create quadratic triangle
17:     else
18:         create point
19:     end if
20: end if
```

An illustrative description of the algorithm is shown in Figure 4.7. Due to the template form of the input file, initial lines can be excluded straightaway. When a line consists of string "ND-COOR", boolean `doRead` switches to *true* until it reaches string "ND-END-COOR". The same procedure is repeated for string "EL-TOP". When string "EL-END-TOP" is identified, the whole cycle breaks. According to the content of lines, which are read, points and elements are created.

Figure 4.7: Illustrative description of reading-file algorithm

4.5 Class Manager

The creation of points and elements is done by class ***Manager***. This class consists of the private attributes m_points and m_elements, which are used to store specific points and elements according to their classification while reading the input file. Points and elements are gradually appended to the vector by functions createPoint(...), createLinearQuadrilateral(...), createLinearTriangle(...), createQuadraticQuadrilateral(...) and createQuadraticTriangle(...). Vector is a container with an arbitrary size. Therefore, the number of points and elements can alter depending on the solved problem. Functions getPoints() and getElements() are used to access stored points and elements when calculating mesh quality metrics and generating the final output.

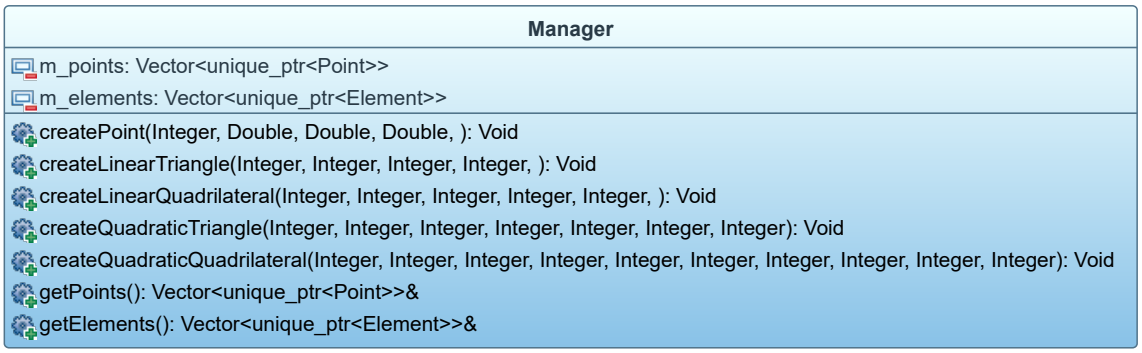


Figure 4.8: Class ***Manager*** representation

4.6 Class Writer

The final steps of the program are carried out in the ***Writer*** class where the calculation of the mesh quality metrics is done and the *XML* output is created. The attributes and functions of the class are shown in Figure 4.9.

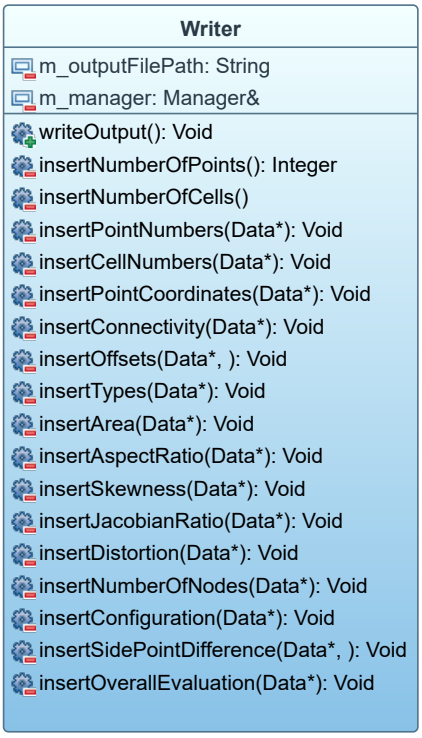


Figure 4.9: Class ***Writer*** representation

The class uses the private attribute m_outputFilePath in order to specify the path of the created output file. The attribute m_manager is used to obtain points and elements, which are necessary for determining the output data.

The public function writeOutput() is called from main() and consists of the private functions shown in Figure 4.9. Each private function inserts the information about mesh geometry or the calculated mesh quality metrics to the final *XML* output. General procedure of insert...(Data*) function summarizes Algorithm 10. In addition, libraries and functions provided by *RIB Software* were used.

Algorithm 10 General procedure of insert...(Data*) function

```
1: set XML attributes
2: get elements or points
3: initialize string for writing
4: for each element or point
5:     get value (the geometry information or the mesh quality metrics)
6:     write value on a new line
7: end for
```

The output file is depicted in Appendix B in Figure B.5. The output is written as *UnstructuredGrid* data in *XML*-based *VTK* format. *VTK* (*Visualization Toolkit*) is an open-source, object-oriented software system used for computer graphics, visualization and image processing [6]. *VTK Unstructured dataset* was chosen because topologically irregular set of points and cells are displayed. *XML* is a markup language used for encoding documents in a format that is both human-readable and machine readable. This form of output is compatible with the graphical software *ParaView*, which is later used for the visualization of the solved problem. *XML* documents have a hierarchical structure, which can be interpreted as a tree structure.

The *XML* tree of the output file is shown in Figure 4.10. The branches *PointData*, *Points* and *Cells* hold data about geometry of the mesh. Whereas, the *CellData* branch contains information about each element, including calculated mesh quality metrics, which can be chosen to be visualized in *ParaView*.

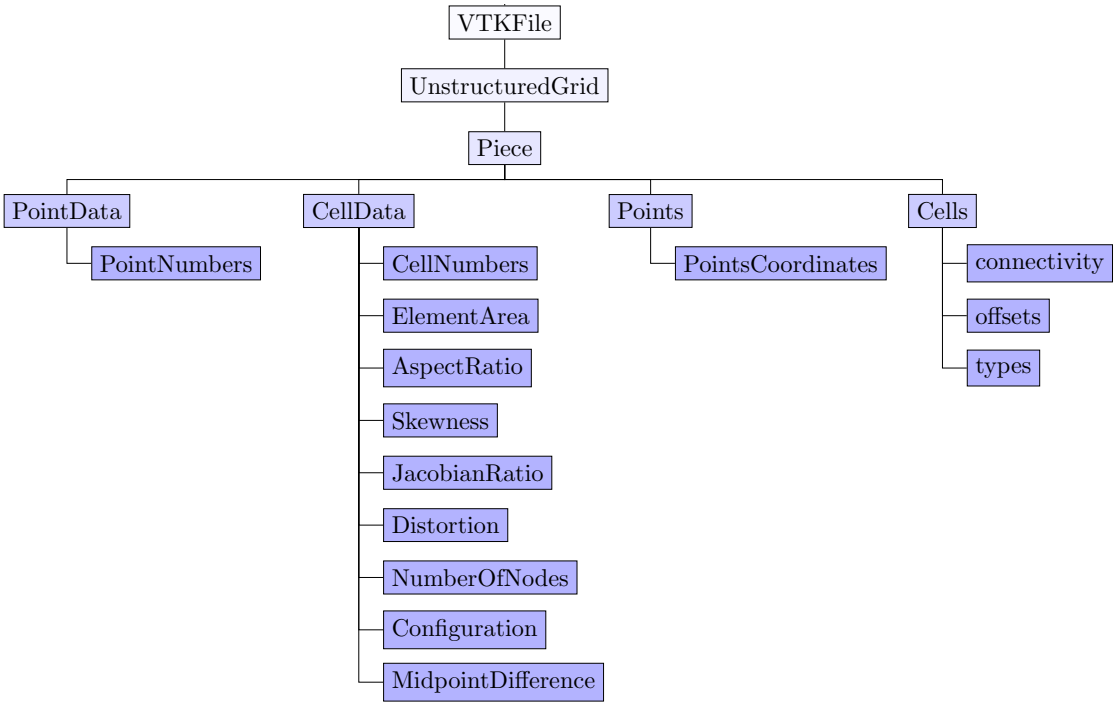


Figure 4.10: Structure of *XML* output file

4.7 Unit testing

The process of developing a code consists of creating the code itself and then its refactoring. Refactoring is a procedure, which restructures existing computer code without changing its external behavior. The main motivation behind refactoring is to make the code more readable and simpler to use. All in all, the whole process of code development is complex and errors may unknowingly occur. Some code errors are pointed out by the compiler; however, others may be overlooked without any warning and the code may not behave as intended. Unit tests are therefore used in order to make sure that the code is fit to use and that it generates desired results.

The separate project **MeshEvaluatorTest** was created in *Microsoft Visual Studio Professional 2013*. The project is interconnected with MeshEvaluator tool in order to be able to use its methods. Two tests for linear and quadratic elements are established using functions testLinearElements() and testQuadraticElements() (Figure 4.11). In addition, libraries and functions provided by *RIB Software* were used.

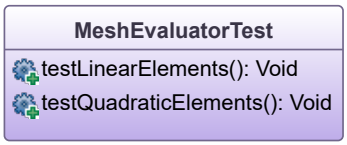


Figure 4.11: ***MeshEvaluatorTest*** representation

Specifically shaped elements are tested:

- Linear elements (Figure 4.12),
- Quadratic elements (Figure 4.13).

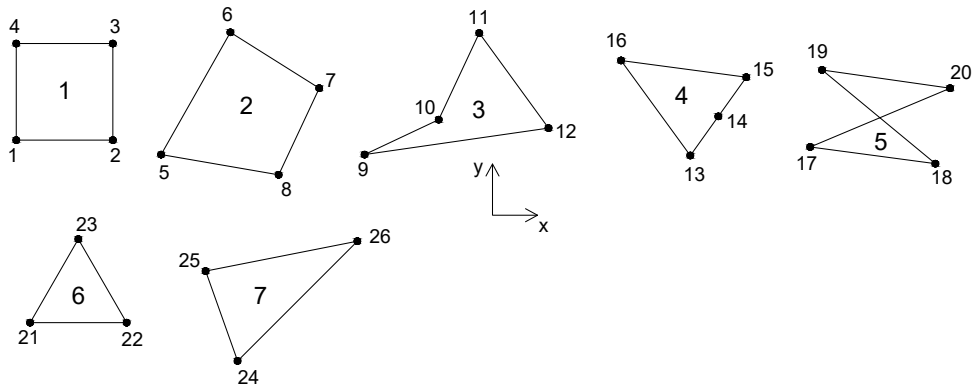


Figure 4.12: Tested linear elements

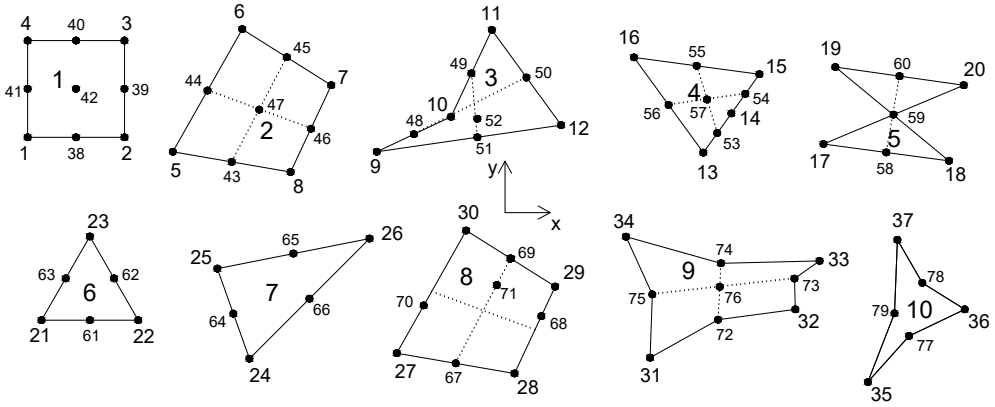


Figure 4.13: Tested quadratic elements

Algorithm 11 shows the procedure of testing. Firstly, the input *text* file with the tested elements is uploaded. Mesh evaluating metrics are calculated and the output *XML* file is created. These steps follow the same procedure as the MeshEvaluator tool. Afterwards, the output *XML* file is compared with a reference *XML* file, which was originally created and which contains the desired output. If all lines of both *XML* files are the same, the Unit test is successful. In case of difference, the Unit test fails and lines which are not identical are pointed out.

Algorithm 11 Unit tests

```
set the input tested file path
create the manager object of the Manager class
create the reader object of the Reader class with the input tested file path as a parameter
execute readInput() function on object reader with the manager as a parameter
set the output tested file path
create the writer object of the Writer class with the output file path and manager as parameters
execute writeOutput() function
set the output compared file path
open the output tested file and the output compared file
for i = 1 to number of lines of output file
    compare the line of the output tested file and the output compared file
    if the lines are identical
        continue
    else
        write the number of the line, the expected content and the real content
    end if
end for
if all lines are identical
    the Unit test is successful
else
    the Unit test fails
end if
```

Chapter 5

Comparison of different meshes

This Chapter analyzes practical examples of two plane structures of a different type, a loading scenario and points of interest:

- a cantilever beam subjected to a point load (Section 5.1),
- a two-way rectangular slab subjected to an uniformly distributed load (Section 5.2),

The examples were analyzed in softwares *RIB Trimas® fem* and *RIB iTWO structure fem*. An element type can be chosen in both softwares either as a linear, which uses Q4 quadrilateral and CST triangle or as a quadratic, which uses Q9 quadrilateral and LST triangle. The automatically generated mesh on those structures was deliberately distorted in order to observe the influence of its quality on the final results. The accuracy of several distorted geometries was analyzed and compared with the regular mesh and the analytical solution.

In most cases denser meshes provide more accurate solutions. However, a denser mesh will increase a computational time, which is also a significant aspect of FEA with a relation to the mesh quality. A mesh is assumed to be of higher quality if a more accurate solution is calculated more quickly. Assuming that, an investigation of the density, which leads to the acceptable results was carried out for both examples.

The solved examples provide an overall outlook on FEA with relation to the quality of the mesh. Even though the considered examples are fundamental and they are not complicated to solve, their analysis reveals behavior of such structures, which can be extrapolated to more complex problems. For these problems, there is a higher probability of automatic mesh generation, which is somehow or somewhere distorted. Furthermore, for larger and more complicated models, users may not notice the problem at the first glance.

5.1 Cantilever beam

The example in Figure 5.1 depicts a cantilever beam subjected to a point load F = 100 kN at the tip of the beam. The cantilever with the cross-sectional dimensions 0.2 × 1 m is 6 m long. The used material is concrete C25/30 (E = 31.5 GPa, G = 13.125 GPa).

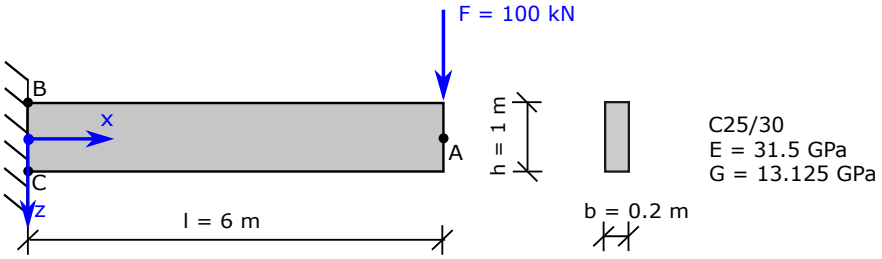


Figure 5.1: Cantilever beam example

The cantilever beam was analyzed using software *RIB Trimas® fem*. The different geometries of the mesh were used in order to obtain following results:

- vertical displacement at the tip (point A),
- principal stress at the upper edge of the support (point B),
- principal stress at the bottom edge of the support (point C).

The analytical results are obtained according to the following procedure. The displacement is calculated with respect to the formula derived for cantilever beams. It is recommended to use *Timoshenko beam* theory if the ratio $l/h < 10$ [13] . The theory assumes that the cross-section does not stay perpendicular to the center line of the beam after the deformation. That induces shear deformation apart from the bending affect. The total displacement is calculated as:

$$I_y = \frac{1}{12}bh^3 = \frac{1}{12} \cdot 0.2 \cdot 1^3 = 0.016667 \text{ m}^4$$

$$\delta_{A,bending} = \frac{FL^3}{3EI_y} = \frac{100 \cdot 6^3}{3 \cdot 31.5 \cdot 10^3 \cdot 0.016667} = 13.714 \text{ mm}$$

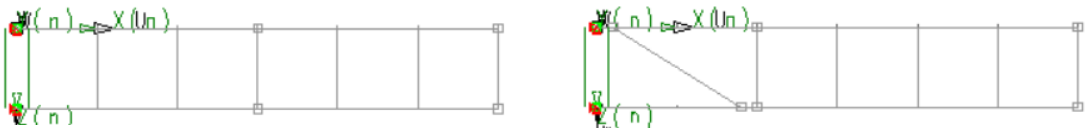
$$\delta_{A,shear} = \frac{FL}{kGA} = \frac{100 \cdot 6}{\frac{5}{6} \cdot 13.125 \cdot 10^3 \cdot 0.2 \cdot 1} = 0.2743 \text{ mm}$$

$$\delta_A = \delta_{A,bending} + \delta_{A,shear} = \mathbf{13.9883} \text{ mm}$$

The principal stress at the upper and bottom edge takes into account only bending. The shear stress is assumed to be equal to zero at the edges of the cross-section.

$$\sigma_{B,C} = \pm\frac{M_y}{I_y}z = \pm\frac{100 \cdot 6}{0.016667} \cdot \frac{1}{2} \cdot 10^{-3} = \pm\mathbf{18} \text{ MPa}$$

The cantilever beam in Figure 5.1 firstly demonstrates the influence of poorly shaped elements on the results of FEA by a fundamental example. Two geometries of meshes (Figure 5.2) are analyzed - a regular mesh with square elements and almost the same mesh with a distortion close to the fixed support. The calculation is performed with both linear (Q4) and quadratic (Q9) elements.



(a) Regular mesh **(b)** Distorted

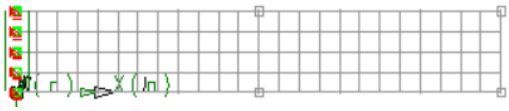
Figure 5.2: Comparison of two meshes

Table 5.1 summarizes the results obtained from calculations and compares them to the analytical solution. It is demonstrated that quadratic elements provide better accuracy. They are less sensitive to the mesh quality, giving almost the same results for both geometries. Similar results are also obtained from analysis using regular mesh and linear elements. All these results are comparable with the analytical solution, which serves as a correctness check. However, mesh distortion has a significant effect on the analysis, which uses linear elements. Displacement drops to 67% of the value obtained by the analytical solution. Furthermore, stresses at the upper and bottom edge of the cross-section are underestimated to 72% and 48%. Different values also indicate non-symmetry of the mesh with respect to the x-axis.

Mesh	δ_A [mm]		σ_B [MPa]		σ_C [MPa]	
analytical	13.99	100%	18.00	100%	-18.00	100%
regular Q9	13.90	99%	17.92	100%	-17.92	100%
distorted Q9	13.89	99%	17.98	100%	-17.92	100%
regular Q4	13.85	99%	16.51	92%	-16.51	92%
distorted Q4	9.40	67%	12.89	72%	-8.56	48%

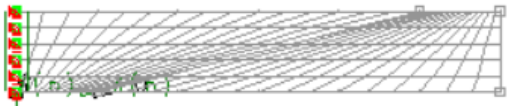
Table 5.1: Comparison between analytical solution and the example from Figure 5.2

Examples of different mesh geometries are shown in Figure 5.3. The overall mesh quality obtained from MeshEvaluator is displayed next to its corresponding geometry. Blue color denotes acceptable quality of the elements, grey stands for relatively acceptable quality and red color means unacceptable quality (evaluation criteria specified in Table 3.5). Examples below depict rather dense mesh (80 to 114 elements for each case).



(a) Regular mesh

(b) Quality evaluation in *ParaView*

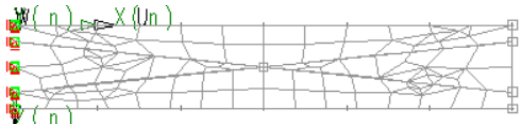


(c) Distorted mesh 1

(d) Quality evaluation in *ParaView*

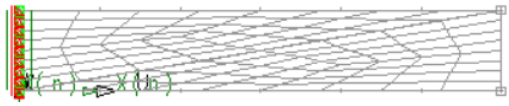
(e) Distorted mesh 2

(f) Quality evaluation in *ParaView*



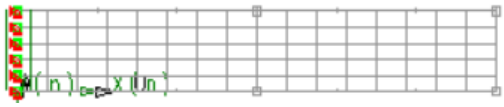
(g) Distorted mesh 3

(h) Quality evaluation in *ParaView*



(i) Distorted mesh 4

(j) Quality evaluation in *ParaView*



(k) Distorted mesh 5

(l) Quality evaluation in *ParaView*

Figure 5.3: Different mesh geometries

Figures 5.4 and 5.5 study behavior of the analyzed values - displacement and principal stresses in dependence on the mesh density. The analysis was carried out using linear elements. Each line represents convergence curve of geometries shown above when the density of the mesh is gradually

changed. The left side figures show convergence in large-scale. Assuming the analytical and the regular mesh solutions as a reference, an acceptable accuracy of results is reached even for coarser mesh. Therefore, it can be claimed that the coarser mesh is sufficient for this problem and that it ensures balance between computational time and the accuracy. The black rectangle in large-scale graphs borders the area, which is displayed as magnified in the right side figures.

Regarding the displacement, all curves converge to the analytical solution eventually. However, principal stresses are further increasing in both support corners with the denser mesh. It may be caused by the physical nature of the boundary condition, which does not allow a movement in the z-direction. That may lead to a stress peaks in the corners as the mesh gets denser (this effect is analyzed in Appendix C.1 in detail).

The non-symmetry of the elements with respect to the x-axis results in non-symmetric principal stresses at the upper and bottom edge of the cross-section (absolute value of stresses at the bottom edge are captured by the dashed line for the non-symmetrical cases in Figure 5.5).

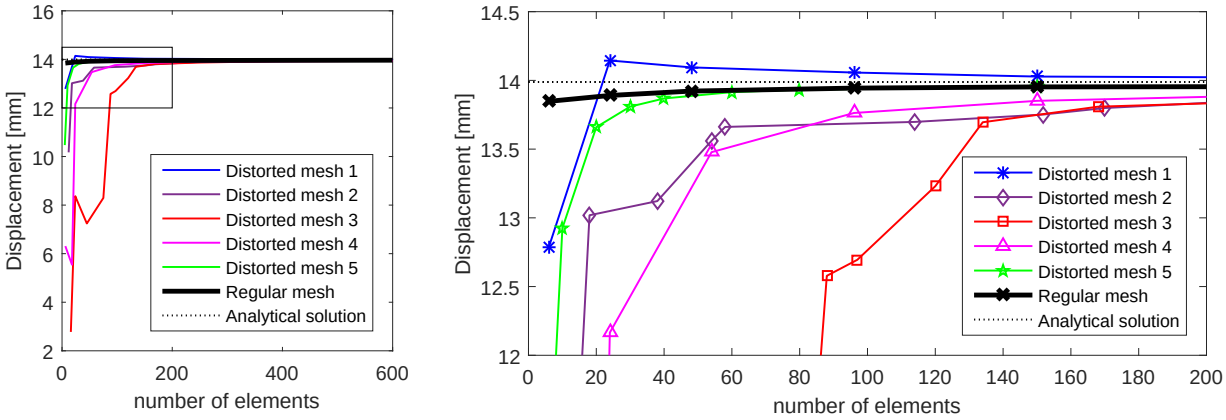


Figure 5.4: Displacement at the tip

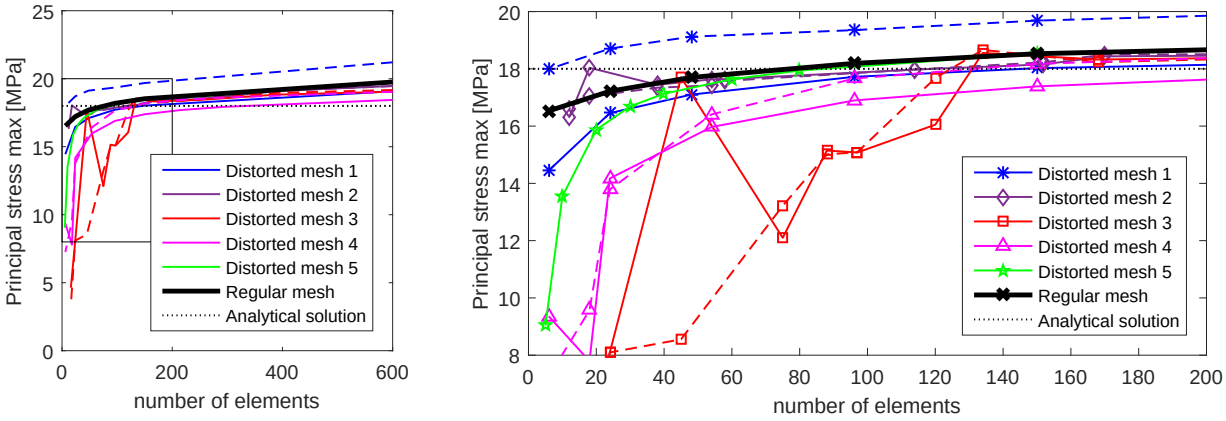


Figure 5.5: Principal stresses at the fixed edge (solid line stands for stresses at the upper edge, dashed line represents absolute values at the bottom edge if they are not equal)

Short comments describe each analyzed geometry of the mesh:

- **Regular mesh** (Figure 5.3a) consists of only square elements. The convergence curve of the displacement is smooth and converges to the analytical solution well. The accurate results are obtained even if a coarse mesh is used. The convergence curve of the principal stress is smooth as well, starting at the reasonably accurate value for the coarsest mesh.
- **Distorted mesh 1** - The mesh is misshaped by a line, which leads from the bottom left to the top right (Figure 5.3c). In a lower left corner (point C) elements are denser, which seemingly results in the higher stress then in the left upper corner (point B). The displacement from certain density converges, but it has decreasing trend unlike the remaining curves.
- **Distorted mesh 2** - The mesh is misshaped by the three skewed lines (Figure 5.3e). More distorted elements can be found in the center of the cantilever rather than at the fixed support or at the tip. The calculated displacement at the tip is affected by a mesh distortion gradually along the entire length. Which presumably causes slower convergence that is manifested by a bigger difference in the displacement when the mesh is coarse. Stresses, on the other hand, are rather balanced and close to the results obtained by using the regular mesh. It may be due to the fact that the mesh is almost regular for all analyzed densities in the region of the support. It is not strongly affected by the distortion in the middle.
- **Distorted mesh 3** - The mesh is governed by two almost diagonal lines, which intersect in the middle (Figure 5.3g). More distorted elements can be found at the fixed support and at the tip, whereas better-quality elements are in the center of the cantilever (unlike Distorted mesh 2). The displacement is influenced by the distortion along the entire length. Stresses are seemingly more affected by the distortion close to the support and they are non-symmetrical at the top and at the bottom. Both displacement and stresses exhibit slow convergence. Starting from density around 140 elements, both analyzed values become more stable and converge.
- **Distorted mesh 4** - The mesh is misshaped by the diagonal line (Figure 5.3i) and consists of elongated skewed quadrilateral elements. The distortion is reflected in both displacement and stresses, which have small convergence rate. Stresses at the top and at the bottom are not symmetrical.
- **Distorted mesh 5** (Figure 5.3k) uses only rectangular elements. Elements of the coarse mesh exhibit rather high aspect ratio because of the perceptible side length difference. The aspect ratio decreases as the mesh gets denser. The convergence curve is smooth, but with a slow rate, which might be caused by the effect called shear locking, which underestimates the displacement and stresses when the aspect ratio is high. It is caused by the fact that linear elements cannot describe bending accurately and they exhibit overstiff behavior by unintentionally introducing shear stress into the element. However, according to *RIB Trimas® fem* manual [4], this unwanted effect is substantially reduced in the version of the program used.

Another example of a distorted mesh is captured in Figure 5.6. The calculation was done using a mesh with linear elements. Although the mesh is rather dense (152 elements), it produces unsatisfactory results summarized in Table 5.2. Results are compared with the analytical solution. It is shown that all analyzed values reach only to approximately 46% of the desired solution. On the other hand, the displacement obtained for previously analyzed meshes (Figure 5.4) reached at least 98% of the analytical value if the same density of the mesh is considered.

This particular example demonstrates that high accuracy of the results cannot be ensured solely by a mesh density and that other influences must be taken into account - such as element type or mesh quality.

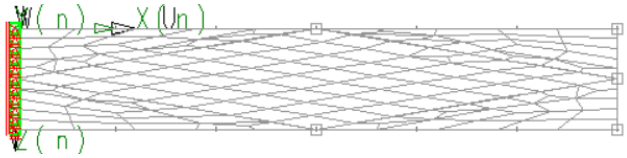
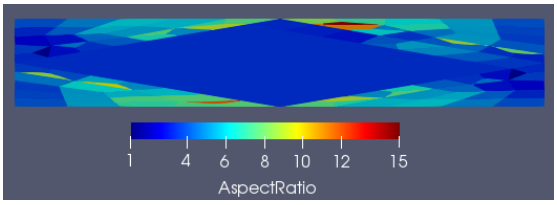


Figure 5.6: Example of highly distorted mesh

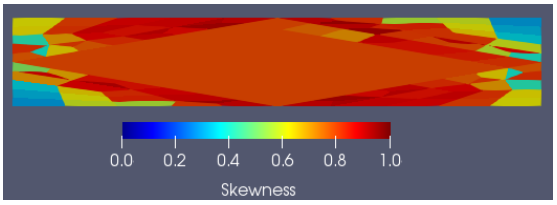
Case	δ_A [mm]		σ_B [MPa]		σ_C [MPa]	
analytical	13.99	100%	18.00	100%	-18.00	100%
distorted	6.45	46%	8.51	47%	-8.35	46%

Table 5.2: Comparison between analytical solution and example from Figure 5.6

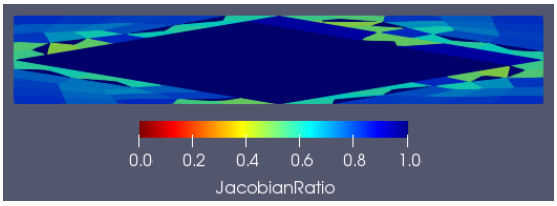
Figure 5.7 depicts mesh quality metrics of the example obtained from MeshEvaluator tool. All elements are fairly distorted and mostly characterized by high skewness and aspect ratio. Overall, the mesh is classified as unsatisfactory.



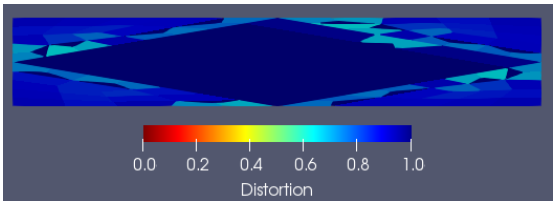
(a) Aspect ratio



(b) Skewness



(c) Jacobian ratio



(d) Distortion

Figure 5.7: Mesh quality metrics in *ParaView*

5.2 Two-way slab

A rectangular two-way concrete slab was analyzed in the second example in order to determine influence of the mesh distortion on the results. The problem is depicted in Figure 5.8. The rectangular concrete slab has dimensions of 6 × 4 m and a thickness of 0.2 m. It is made of concrete C20/25 ($\nu = 0.2$, E = 30 GPa). The slab is fixed along its longer sides and pinned along its shorter edges. An uniformly distributed load acts on the slab with magnitude of 10 kN/m^2.

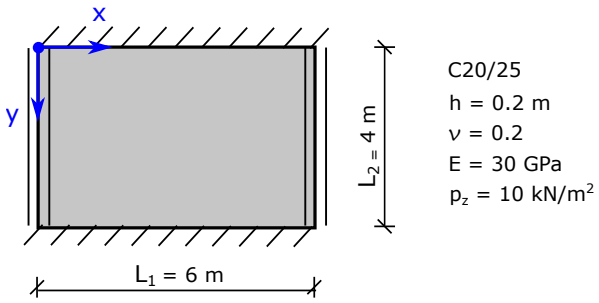


Figure 5.8: Rectangular slab example

The displacement in the center of the slab is calculated analytically according to [12]. It is derived from the differential plate equation for a pure bending of a thin plate:

$$\frac{\partial^4 w}{\partial x^4} + 2\frac{\partial^4 w}{\partial x^2 \partial y^2} + \frac{\partial^4 w}{\partial y^4} = \frac{p}{K}. \tag{5.1}$$

It complies with an assumption that the flexural rigidity of the slab is constant and it can be expressed as:

$$K = \frac{Et^3}{12(1-\nu^2)}. \tag{5.2}$$

Taking boundary conditions into account, the displacement in the center of the slab based on *Navier solution* using *Fourier series* is expressed as:

$$w = \frac{4p}{Ka}\sum_n \frac{1}{\alpha_n^5}\left\{1 - \frac{1}{\frac{1}{2}[sinh(\alpha_n b) + \alpha_n b]}\left[\left(sinh\frac{\alpha_n b}{2} + \frac{\alpha_n b}{2}cosh\frac{\alpha_n b}{2}\right)cosh(\alpha_n y) - \right.\right.$$
$$\left.\left. - sinh\frac{\alpha_n b}{2}\,\alpha_n y\, sinh(\alpha_n y)\right]\right\} sin(\alpha_n x), \tag{5.3}$$

where:

$\alpha_n = \frac{n\pi}{a}$,

$n = 1, 3, 5...$

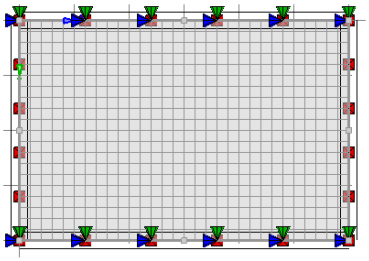
$a = 6, b = 4,$ which are the dimentions of the slab

$x = 3, y = 0$ for the center of the slab

The progression stabilizes when $n = 11$ and displacement is equal to:

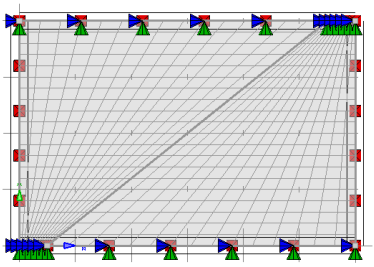
$$w = 0.3042 \text{ mm}$$

The slab was computed in *RIB iTWO structure fem* software using different geometries of the mesh as it is shown in Figures below. In the first case a regular mesh is used in order to get a reference results (Figure 5.9a). The next five cases represent distorted meshes, which are misshaped by the depicted lines governing the formation of the elements. Figures on the right side represent an overall quality of the corresponding mesh obtained from MeshEvaluator tool according to Table 3.5 (blue stands for acceptable quality, grey for relatively acceptable quality and red means unacceptable quality). The density of all depicted meshes is assumed to be sufficiently high (600 to 670 elements on each slab shown).

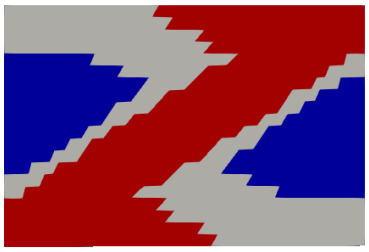


(a) Regular mesh

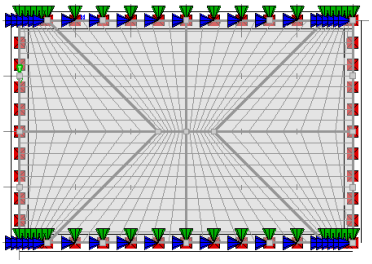
(b) Quality evaluation in *ParaView*



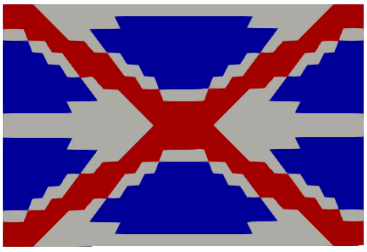
(c) Distorted mesh 1



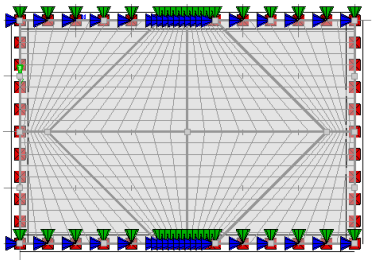
(d) Quality evaluation in *ParaView*



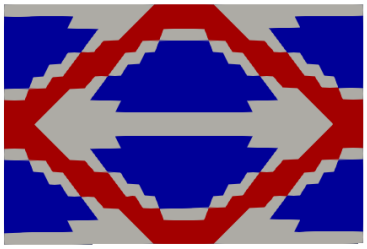
(e) Distorted mesh 2



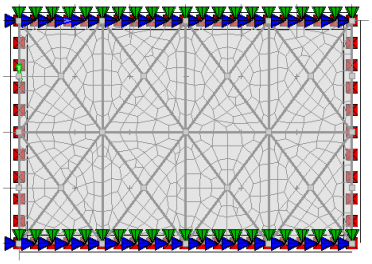
(f) Quality evaluation in *ParaView*



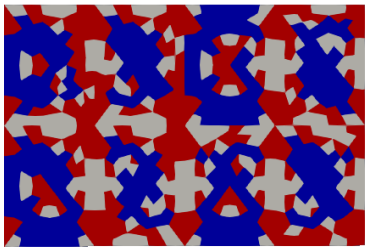
(g) Distorted mesh 3



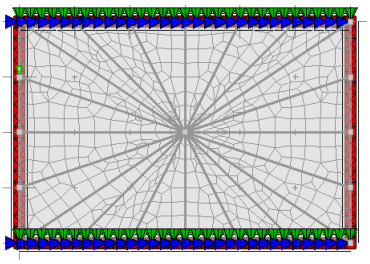
(h) Quality evaluation in *ParaView*



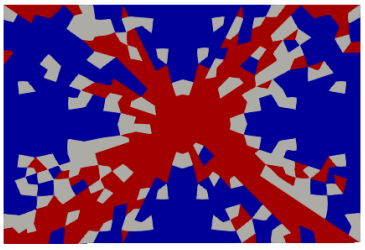
(i) Distorted mesh 4



(j) Quality evaluation in *ParaView*



(k) Distorted mesh 5



(l) Quality evaluation in *ParaView*

Figure 5.9: Different geometries of the mesh and their quality evaluation

The above-shown geometries of were further investigated. The mesh density of all cases was gradually changed, from which the convergence curves were obtained. The following values were studied:

- displacement in the center (Figure 5.10),
- m_{xx} in the center (Figure 5.11),
- m_{yy} in the center (Figure 5.12),
- m_{yy} at the fixed edge (Figure 5.13).

In order to verify the correctness of the analysis, results were compared with values obtained from calculation tool *SlaFoR* [8]. The tool uses a *finite difference method* to analyze rectangular two-way slabs. Regular mesh was used and its density was changed in the same way. Displacement and moments were obtained.

Furthermore, the analytical value of the displacement is depicted in Figure 5.10.

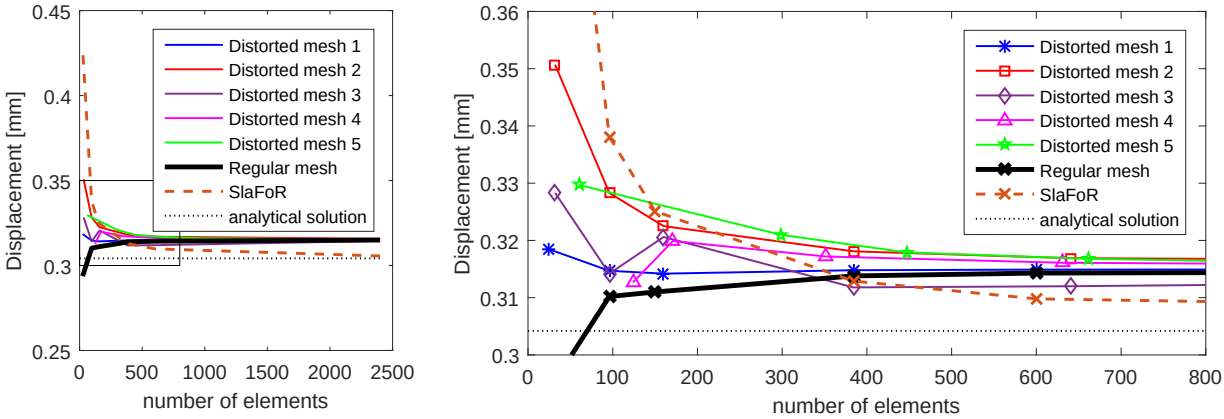


Figure 5.10: Displacement in the center of the slab

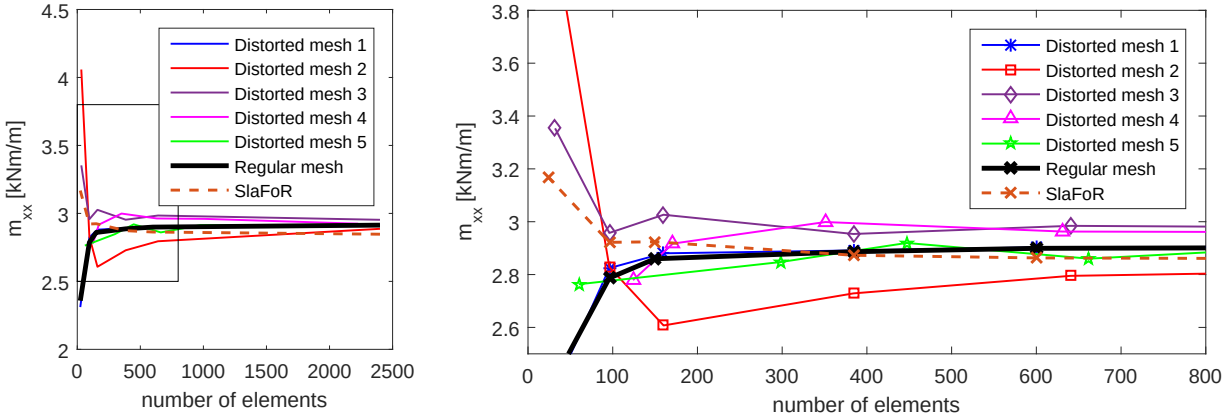


Figure 5.11: m_{xx} in the center of the slab

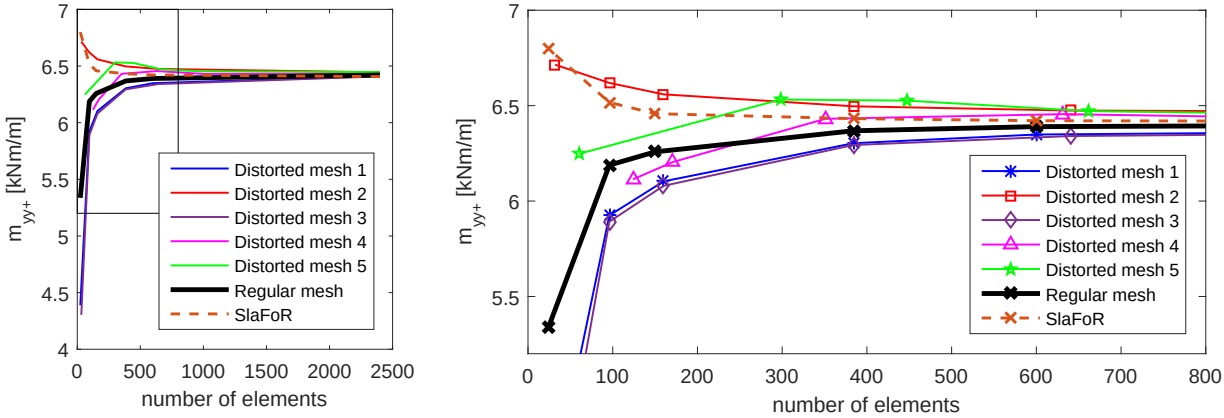


Figure 5.12: m_{yy} in the center of the slab

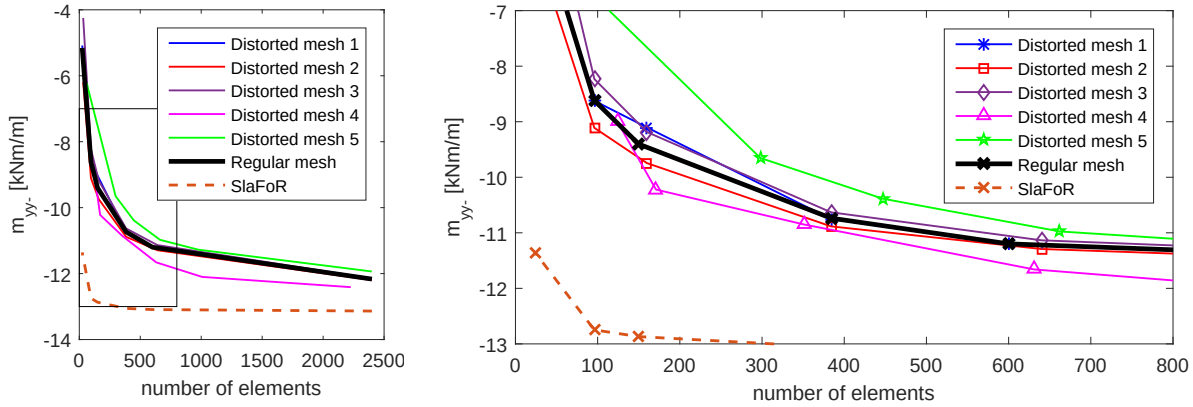


Figure 5.13: m_{yy} at the fixed edge

Noticeable deviations in the limit values can be observed by comparing the results obtained from different programs. Dashed curve stands for the solution calculated in *SlaFoR* tool. Displacement calculated in *SlaFoR* converges to the analytical solution (dotted line). The analytical solution as well as the calculation by *SlaFoR* is based on *Kirchhoff theory* suitable for thin slabs. Kirchhoff theory assumes, that there is no shear effect present. Whereas, the calculation performed by *RIB iTWO structure fem* uses *Mindlin theory* (further elaborated in Appendix C.2) suitable for thick slabs that includes effect of the shear. That results in the displacement of a slightly larger value (by 3 %). Moments obtained from both programs seem to reach the similar value, except for m_{yy} at the support. Moment from *RIB iTWO structure fem* is however still converging so it may lead to the similar value or the deviation is caused by using different methods.

Considering only the results obtained from *RIB iTWO structure fem*, all analyzed values converge to the same results if the density of the used mesh is sufficient (as can be seen in Figures 5.10, 5.11, 5.12 and 5.13). The convergence curves of distorted meshes are less smooth and sometimes volatile. It is noteworthy that in some cases of distorted meshes, the curves are approaching the limit from the opposite side then the curve representing the regular mesh. For example, the displacement increases with the regular mesh density, however it decreases for most of the distorted meshes.

The empirical recommendation for the sufficient mesh density of the slab is to use the element size equal to the slab thickness [7]. In this case it corresponds to 600 elements per a regular mesh. Since the solved problem is not complex, it might be satisfactory to use even a coarser mesh. Figures above on the right show such a region.

By comparing calculated values in the sufficient mesh density region, it can be concluded that even distorted mesh provides rather similar results. They vary only in percent units from the solution obtained by using regular mesh if the sufficient density is assumed. The biggest difference is in the shape of the convergence curve, especially when the mesh is coarse. However, all curves tend to converge to the similar limit.

All in all, performed analysis indicates that the accuracy of the results is not as strongly affected by poor mesh quality as it was in the previous example of the cantilever. It might be caused by the physical nature of the problem, when a 2D structure is loaded perpendicularly to the plane. The mesh generated on the slab is perpendicular to the load and it may not be as sensitive as in the case of the cantilever, when the load was acting in the same plane as the orientation of the elements.

Chapter 6

Conclusion

The main objective of the thesis was to analyze the mesh quality of FEA. First, the thesis provides a general overview of this topic and then introduces several metrics used for evaluating the quality of the mesh elements. It is generally known, that preprocessors of many commercial softwares are able to automatically generate mesh on the analyzed structure; however, the mesh may include regions where it is not properly arranged. FEA with errors in the mesh formation may produce misleading results. Therefore, in practical part of the thesis, computational tool **MeshEvaluator** was developed in the C++ programming environment.

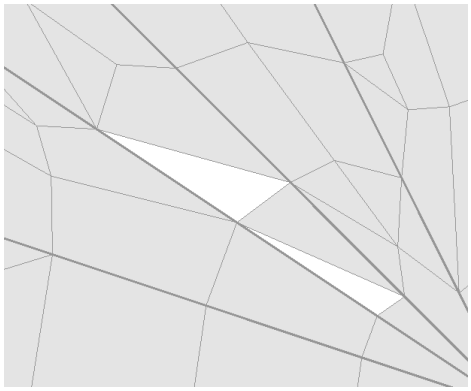
The MeshEvaluator tool is applicable to plane structures analyzed in software *RIB iTWO structure fem*. It calculates mesh quality metrics for each element of the mesh and provides the graphical outlook on an analyzed structure as a whole. Individual metrics, which include configuration, relative midpoint difference, aspect ratio, skewness, Jacobian ratio and distortion, can be displayed. Alternatively, the overall evaluation of elements according to Table 3.5 can be performed. The developed tool serves as a convenient aid for checking the quality of the generated mesh.

The severity of errors caused by poor-quality mesh was analyzed on two examples of concrete 2D structures - a bent cantilever and a bent slab. The mesh generated on those structures was deliberately distorted in several cases. Results were compared with results obtained while using regular mesh and the analytical solution. The influence of mesh quality on the results was observed. In general, the linear elements are more sensitive to the mesh quality then the quadratic elements. Furthermore, worse effect of poor-quality elements on the results was found in problems, where load acted in the same plane as the generated mesh. The bent cantilever exhibited greater deviations in the results then the bent slab. The effect of shear locking can be also present in in-plane loading.

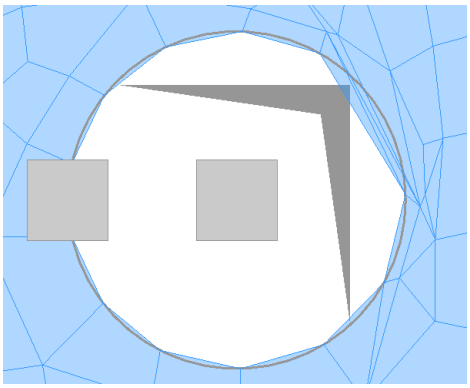
6.1 Outlook

While investigating the topic and analyzing the structures in structural design software, other imperfections concerning the mesh quality were observed. In this thesis, the mesh quality check is done only by checking each element individually and then extrapolate the evaluation of single elements continuously on the whole structure. However, other errors may appear because of the bad interaction between elements, such as:

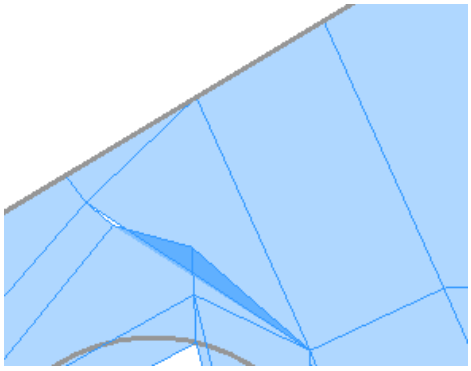
- missing element in the mesh (Figure 6.1a),
- element, which does not follow the boundary of the structure (Figure 6.1b),
- overlapping elements (Figure 6.1c),
- badly connected elements (Figure 6.1d).



(a) Missing element



(b) Element not following boundary



(c) Overlapping elements

(d) Badly connected elements

Figure 6.1: Errors in interaction between elements

The problems described above have also a negative influence on the results of FEA. Therefore, it provides a suggestion for possible extension of developed MeshEvaluator tool. In the future, the tool could intercept poor interaction between elements and alert user if such situation occurs.

Another possible goal is to extend applicability of the tool to the three-dimensional problems. 3D models tend to be more complex and it is harder for the user to notice poorly shaped elements or discontinuities in the mesh.

Finally, the tool could be incorporated to *RIB software* in order to provide better user-friendly interface. Related to that, the developed code should be further refactored to ensure maintainability for the further use.

Bibliography

[1] Finite Element Analysis: How to create a great model, retrieved from "https://coventivecomposites.com/".

[2] Mesh Quality Parameters in Finite Element Analysis, retrieved from "https://www.engmorph.com/mesh-quality-parameters".

[3] *RIBTEC® Common Coding Style*. Compiled on July 7, 2017.

[4] *TRIMAS® General FE-System - Basics*. FE-Systems for structural and bridge engineering, RIB Software SE.

[5] F. Alonso-Marroquin. Finite Element Modelling for Civil Engineering, 2016.

[6] L. S. Avila, S. Barre, R. Blue, B. Geveci, A. Henderson, W. A. Hoffman, B. King, C. C. Law, K. M. Martin, and W. J. Schroeder. *The VTK user's guide*. Kitware New York, 2010.

[7] P. Bílý. Návrh stropní desky v programu SCIA Engineer. *Vyztužování poruchových oblastí železobetonové konstrukce*, 2017.

[8] A. Bulíčková. Tvorba a aplikace vypočetní pomůcky pro stanovení vnitřních sil na deskách. B.S. thesis, Czech Technical University in Prague, 2018.

[9] R. D. Cook. *Concepts and Applications of Finite Element Analysis*. 1981.

[10] B. Eckel. *Thinking in C++*. Prentice-Hall, 2000.

[11] P. J. Frey and P.-L. George. *Mesh Generation: application to finite elements*. ISTE, 2000.

[12] K. Girkmann. Flächentragwerke, Wien. *Springer-Verlag*, 1963.

[13] C. Pacoste. Lecture notes in Finite Element Methods in Analysis and Design. KTH Royal Institute of Technology in Stockholm, March 2019.

[14] C. Stimpson, C. Ernst, P. Knupp, P. Pébay, and D. Thompson. The Verdict Library Reference Manual. *Sandia National Laboratories Technical Report*, 9(6), 2007.

[15] M. Virius. *Programovací jazyk C++*. Czech Technical University in Prague, 2016.

[16] K.-H. Yang. *Basic Finite Element Method as Applied to Injury Biomechanics*. Academic Press, 2017.

Used programs

- Microsoft Visual Studio Professional 2013 (RIB Software SE license)
- RIB iTWO structure fem 20.0 (RIB Software SE license)
- RIB Trimas® fem 19.0 (RIB Software SE license)
- TortoiseSVN 1.12.0.28568 (RIB Software SE license)
- Microsoft Office 365 ProPlus (RIB Software SE license)
- MATLAB R2015b (academic license)
- Autodesk AutoCAD 2018 22.0.49.0 (academic license)
- Inkscape 0.92.4
- ParaView 5.6.1
- Texmaker 5.0.3
- https://app.genmymodel.com/
- https://diagrams.visual-paradigm.com/

Appendix A

Jacobian matrix for Q4 and Q9 elements

Components of the Jacobian matrix:

$$J_{11} = \frac{\partial x}{\partial \xi} = \frac{\partial}{\partial \xi}\sum_{i=1}^{n} N_i x_i, \qquad J_{12} = \frac{\partial y}{\partial \xi} = \frac{\partial}{\partial \xi}\sum_{i=1}^{n} N_i y_i,$$
$$J_{21} = \frac{\partial x}{\partial \eta} = \frac{\partial}{\partial \eta}\sum_{i=1}^{n} N_i x_i, \qquad J_{22} = \frac{\partial y}{\partial \eta} = \frac{\partial}{\partial \eta}\sum_{i=1}^{n} N_i y_i.$$

Derivatives of the shape functions according to ξ and η (the shape function numbering relates to Figure 3.13):

	i	$\frac{\partial N_i}{\partial \xi}$	$\frac{\partial N_i}{\partial \eta}$
Q4	1	$\frac{1}{4}\left(\eta - 1\right)$	$\frac{1}{4}\left(\xi - 1\right)$
	2	$\frac{1}{4}\left(-\eta + 1\right)$	$\frac{1}{4}\left(-\xi - 1\right)$
	3	$\frac{1}{4}\left(\eta + 1\right)$	$\frac{1}{4}\left(\xi + 1\right)$
	4	$\frac{1}{4}\left(-\eta - 1\right)$	$\frac{1}{4}\left(-\xi + 1\right)$
Q9	1	$\frac{1}{4}\left(2\xi\eta^2 - 2\xi\eta - \eta^2 + \eta\right)$	$\frac{1}{4}\left(2\xi^2\eta - \xi^2 - 2\xi\eta + \xi\right)$
	2	$\frac{1}{4}\left(2\xi\eta^2 - 2\xi\eta + \eta^2 - \eta\right)$	$\frac{1}{4}\left(2\xi^2\eta - \xi^2 + 2\xi\eta - \xi\right)$
	3	$\frac{1}{4}\left(2\xi\eta^2 + 2\xi\eta + \eta^2 + \eta\right)$	$\frac{1}{4}\left(2\xi^2\eta + \xi^2 + 2\xi\eta + \xi\right)$
	4	$\frac{1}{4}\left(2\xi\eta^2 + 2\xi\eta - \eta^2 - \eta\right)$	$\frac{1}{4}\left(2\xi^2\eta + \xi^2 - 2\xi\eta - \xi\right)$
	5	$-\frac{1}{2}\left(2\xi\eta^2 - 2\xi\eta\right)$	$-\frac{1}{2}\left(2\xi^2\eta - \xi^2 - 2\eta + 1\right)$
	6	$-\frac{1}{2}\left(2\xi\eta^2 - 2\xi + \eta^2 - 1\right)$	$-\frac{1}{2}\left(2\xi^2\eta + 2\xi\eta\right)$
	7	$-\frac{1}{2}\left(2\xi\eta^2 + 2\xi\eta\right)$	$-\frac{1}{2}\left(2\xi^2\eta + \xi^2 - 2\eta - 1\right)$
	8	$-\frac{1}{2}\left(2\xi\eta^2 - 2\xi - \eta^2 + 1\right)$	$-\frac{1}{2}\left(2\xi^2\eta - 2\xi\eta\right)$
	9	$2\xi\eta^2 - 2\xi$	$2\xi^2\eta - 2\eta$

Components of the Jacobian matrix for Q4 elements:

$$J_{11} = \frac{1}{4}\Big[(\eta-1)x_1 + (-\eta+1)x_2 + (\eta+1)x_3 + (-\eta-1)x_4\Big]$$
$$J_{12} = \frac{1}{4}\Big[(\eta-1)y_1 + (-\eta+1)y_2 + (\eta+1)y_3 + (-\eta-1)y_4\Big]$$
$$J_{21} = \frac{1}{4}\Big[(\xi-1)x_1 + (-\xi-1)x_2 + (\xi+1)x_3 + (-\xi+1)x_4\Big]$$
$$J_{22} = \frac{1}{4}\Big[(\xi-1)y_1 + (-\xi-1)y_2 + (\xi+1)y_3 + (-\xi+1)y_4\Big]$$

It is assumed that Q9 element is "properly" formed, i.e. nodes 5, 6, 7, 8 are located exactly in the middle of corresponding side and node 9 lies in the intersection of straight lines, which connect the opposite midpoints. Therefore, coordinates of these nodes can be expressed as:

$$P_5[x_5, y_5] = \left[\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2}\right], \qquad P_6[x_6, y_6] = \left[\frac{x_2+x_3}{2}, \frac{y_2+y_3}{2}\right],$$
$$P_7[x_7, y_7] = \left[\frac{x_3+x_4}{2}, \frac{y_3+y_4}{2}\right], \qquad P_8[x_8, y_8] = \left[\frac{x_4+x_1}{2}, \frac{y_4+y_1}{2}\right],$$
$$P_9[x_9, y_9] = \left[\frac{x_1+x_2+x_3+x_4}{4}, \frac{y_1+y_2+y_3+y_4}{4}\right].$$

Components of the Jacobian matrix for Q9 elements:

$$J_{11} = \frac{x_1}{4}\left(2\xi\eta^2 - 2\xi\eta - \eta^2 + \eta\right) + \frac{x_2}{4}\left(2\xi\eta^2 - 2\xi\eta + \eta^2 - \eta\right) + \frac{x_3}{4}\left(2\xi\eta^2 + 2\xi\eta + \eta^2 + \eta\right) +$$
$$\frac{x_4}{4}\left(2\xi\eta^2 + 2\xi\eta - \eta^2 - \eta\right) - \frac{x_1+x_2}{4}\left(2\xi\eta^2 - 2\xi\eta\right) - \frac{x_2+x_3}{4}\left(2\xi\eta^2 - 2\xi + \eta^2 - 1\right) -$$
$$\frac{x_3+x_4}{4}\left(2\xi\eta^2 + 2\xi\eta\right) - \frac{x_4+x_1}{4}\left(2\xi\eta^2 - 2\xi - \eta^2 + 1\right) + \frac{x_1+x_2+x_3+x_4}{4}\left(2\xi\eta^2 - 2\xi\right),$$

$$\frac{x_1}{4}\left(2\xi\eta^2 - 2\xi\eta - \eta^2 + \eta - 2\xi\eta^2 + 2\xi\eta - 2\xi\eta^2 + \eta^2 + 2\xi - 1 + 2\xi\eta^2 - 2\xi\right) = \frac{x_1}{4}(\eta-1),$$
$$\frac{x_2}{4}\left(2\xi\eta^2 - 2\xi\eta + \eta^2 - \eta - 2\xi\eta^2 + 2\xi\eta - 2\xi\eta^2 - \eta^2 + 2\xi + 1 + 2\xi\eta^2 - 2\xi\right) = \frac{x_2}{4}(-\eta+1),$$
$$\frac{x_3}{4}\left(2\xi\eta^2 + 2\xi\eta + \eta^2 + \eta - 2\xi\eta^2 - 2\xi\eta - 2\xi\eta^2 - \eta^2 + 2\xi + 1 + 2\xi\eta^2 - 2\xi\right) = \frac{x_3}{4}(\eta+1),$$
$$\frac{x_4}{4}\left(2\xi\eta^2 + 2\xi\eta - \eta^2 - \eta - 2\xi\eta^2 - 2\xi\eta - 2\xi\eta^2 + \eta^2 + 2\xi - 1 + 2\xi\eta^2 - 2\xi\right) = \frac{x_4}{4}(-\eta-1),$$

$$J_{11} = \frac{1}{4}\Big[(\eta-1)x_1 + (-\eta+1)x_2 + (\eta+1)x_3 + (-\eta-1)x_4\Big]$$

$$J_{12} = \frac{y_1}{4}\left(2\xi\eta^2 - 2\xi\eta - \eta^2 + \eta\right) + \frac{y_2}{4}\left(2\xi\eta^2 - 2\xi\eta + \eta^2 - \eta\right) + \frac{y_3}{4}\left(2\xi\eta^2 + 2\xi\eta + \eta^2 + \eta\right) +$$
$$\frac{y_4}{4}\left(2\xi\eta^2 + 2\xi\eta - \eta^2 - \eta\right) - \frac{y_1+y_2}{4}\left(2\xi\eta^2 - 2\xi\eta\right) - \frac{y_2+y_3}{4}\left(2\xi\eta^2 - 2\xi + \eta^2 - 1\right) -$$
$$\frac{y_3+y_4}{4}\left(2\xi\eta^2 + 2\xi\eta\right) - \frac{y_4+y_1}{4}\left(2\xi\eta^2 - 2\xi - \eta^2 + 1\right) + \frac{y_1+y_2+y_3+y_4}{4}\left(2\xi\eta^2 - 2\xi\right),$$

$$\frac{y_1}{4}\left(2\xi\eta^2 - 2\xi\eta - \eta^2 + \eta - 2\xi\eta^2 + 2\xi\eta - 2\xi\eta^2 + \eta^2 + 2\xi - 1 + 2\xi\eta^2 - 2\xi\right) = \frac{y_1}{4}(\eta - 1),$$
$$\frac{y_2}{4}\left(2\xi\eta^2 - 2\xi\eta + \eta^2 - \eta - 2\xi\eta^2 + 2\xi\eta - 2\xi\eta^2 - \eta^2 + 2\xi + 1 + 2\xi\eta^2 - 2\xi\right) = \frac{y_2}{4}(-\eta + 1),$$
$$\frac{y_3}{4}\left(2\xi\eta^2 + 2\xi\eta + \eta^2 + \eta - 2\xi\eta^2 - 2\xi\eta - 2\xi\eta^2 - \eta^2 + 2\xi + 1 + 2\xi\eta^2 - 2\xi\right) = \frac{y_3}{4}(\eta + 1),$$
$$\frac{y_4}{4}\left(2\xi\eta^2 + 2\xi\eta - \eta^2 - \eta - 2\xi\eta^2 - 2\xi\eta - 2\xi\eta^2 + \eta^2 + 2\xi - 1 + 2\xi\eta^2 - 2\xi\right) = \frac{y_4}{4}(-\eta - 1),$$

$$J_{12} = \frac{1}{4}\Big[(\eta - 1)y_1 + (-\eta + 1)y_2 + (\eta + 1)y_3 + (-\eta - 1)y_4\Big]$$

$$J_{21} = \frac{x_1}{4}\left(2\xi^2\eta - \xi^2 - 2\xi\eta + \xi\right) + \frac{x_2}{4}\left(2\xi^2\eta - \xi^2 + 2\xi\eta - \xi\right) + \frac{x_3}{4}\left(2\xi^2\eta + \xi^2 + 2\xi\eta + \xi\right) +$$
$$\frac{x_4}{4}\left(2\xi^2\eta + \xi^2 - 2\xi\eta - \xi\right) - \frac{x_1 + x_2}{4}\left(2\xi^2\eta - \xi^2 - 2\eta + 1\right) - \frac{x_2 + x_3}{4}\left(2\xi^2\eta + 2\xi\eta\right) -$$
$$\frac{x_3 + x_4}{4}\left(2\xi^2\eta + \xi^2 - 2\eta - 1\right) - \frac{x_4 + x_1}{4}\left(2\xi^2\eta - 2\xi\eta\right) + \frac{x_1 + x_2 + x_3 + x_4}{4}\left(2\xi^2\eta - 2\eta\right),$$

$$\frac{x_1}{4}\left(2\xi^2\eta - \xi^2 - 2\xi\eta + \xi - 2\xi^2\eta + \xi^2 + 2\eta - 1 - 2\xi^2\eta + 2\xi\eta + 2\xi^2\eta - 2\eta\right) = \frac{x_1}{4}(\xi - 1),$$
$$\frac{x_2}{4}\left(2\xi^2\eta - \xi^2 + 2\xi\eta - \xi - 2\xi^2\eta + \xi^2 + 2\eta - 1 - 2\xi^2\eta - 2\xi\eta + 2\xi^2\eta - 2\eta\right) = \frac{x_2}{4}(-\xi - 1),$$
$$\frac{x_3}{4}\left(2\xi^2\eta + \xi^2 + 2\xi\eta + \xi - 2\xi^2\eta - \xi^2 + 2\eta + 1 - 2\xi^2\eta - 2\xi\eta + 2\xi^2\eta - 2\eta\right) = \frac{x_3}{4}(\xi + 1),$$
$$\frac{x_4}{4}\left(2\xi^2\eta + \xi^2 - 2\xi\eta - \xi - 2\xi^2\eta - \xi^2 + 2\eta + 1 - 2\xi^2\eta + 2\xi\eta + 2\xi^2\eta - 2\eta\right) = \frac{x_4}{4}(-\xi + 1),$$

$$J_{21} = \frac{1}{4}\Big[(\xi - 1)x_1 + (-\xi - 1)x_2 + (\xi + 1)x_3 + (-\xi + 1)x_4\Big]$$

$$J_{22} = \frac{y_1}{4}\left(2\xi^2\eta - \xi^2 - 2\xi\eta + \xi\right) + \frac{y_2}{4}\left(2\xi^2\eta - \xi^2 + 2\xi\eta - \xi\right) + \frac{y_3}{4}\left(2\xi^2\eta + \xi^2 + 2\xi\eta + \xi\right) +$$
$$\frac{y_4}{4}\left(2\xi^2\eta + \xi^2 - 2\xi\eta - \xi\right) - \frac{y_1 + y_2}{4}\left(2\xi^2\eta - \xi^2 - 2\eta + 1\right) - \frac{y_2 + y_3}{4}\left(2\xi^2\eta + 2\xi\eta\right) -$$
$$\frac{y_3 + y_4}{4}\left(2\xi^2\eta + \xi^2 - 2\eta - 1\right) - \frac{y_4 + y_1}{4}\left(2\xi^2\eta - 2\xi\eta\right) + \frac{y_1 + y_2 + y_3 + y_4}{4}\left(2\xi^2\eta - 2\eta\right),$$

$$\frac{y_1}{4}\left(2\xi^2\eta - \xi^2 - 2\xi\eta + \xi - 2\xi^2\eta + \xi^2 + 2\eta - 1 - 2\xi^2\eta + 2\xi\eta + 2\xi^2\eta - 2\eta\right) = \frac{x_1}{4}(\xi - 1),$$
$$\frac{y_2}{4}\left(2\xi^2\eta - \xi^2 + 2\xi\eta - \xi - 2\xi^2\eta + \xi^2 + 2\eta - 1 - 2\xi^2\eta - 2\xi\eta + 2\xi^2\eta - 2\eta\right) = \frac{x_2}{4}(-\xi - 1),$$
$$\frac{y_3}{4}\left(2\xi^2\eta + \xi^2 + 2\xi\eta + \xi - 2\xi^2\eta - \xi^2 + 2\eta + 1 - 2\xi^2\eta - 2\xi\eta + 2\xi^2\eta - 2\eta\right) = \frac{x_3}{4}(\xi + 1),$$
$$\frac{y_4}{4}\left(2\xi^2\eta + \xi^2 - 2\xi\eta - \xi - 2\xi^2\eta - \xi^2 + 2\eta + 1 - 2\xi^2\eta + 2\xi\eta + 2\xi^2\eta - 2\eta\right) = \frac{x_4}{4}(-\xi + 1),$$

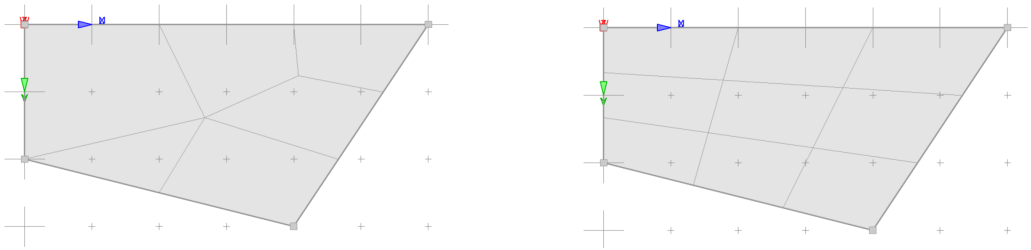
$$J_{22} = \frac{1}{4}\Big[(\xi - 1)y_1 + (-\xi - 1)y_2 + (\xi + 1)y_3 + (-\xi + 1)y_4\Big]$$

Components of the Jacobian matrix are calculated identically for Q4 and Q9 element if the Q9 element is "properly" formed. If additional nodes of the Q9 element do not fulfill this assumption, calculation of the Jacobian matrix would not be simplified and real coordinates would have to be accounted for.

Appendix B

Input and Output files of MeshEvaluator

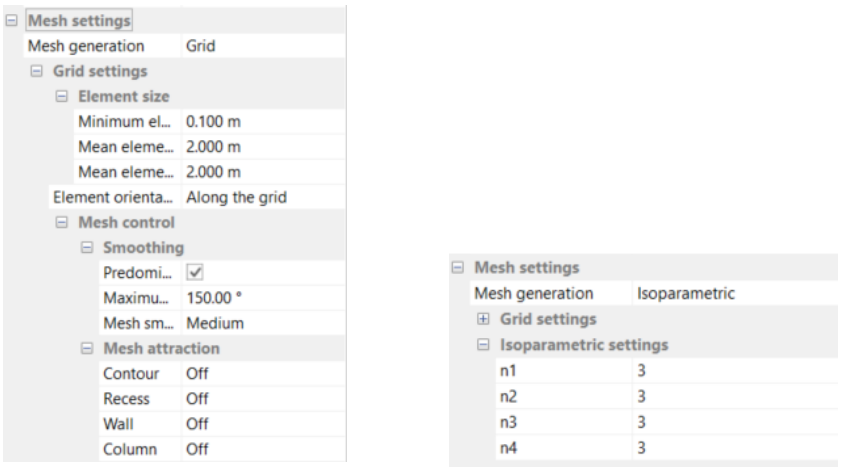
By defining the geometry of a planar structure in *RIB iTWO structure fem SLAB*, its mesh is automatically generated. Two types of mesh generation can be chosen - "grid" and "isoparametric" (examples in Figure B.1).



(a) "Grid" mesh **(b)** "Isoparametric" mesh

Figure B.1: Types of meshes generated by *RIB iTWO structure fem SLAB*

Mesh settings can be altered by the user (Figure B.2).



(a) Grid mesh **(b)** Isoparametric mesh

Figure B.2: Mesh settings

When the user adds material properties, restrains and loading scenario (Figure B.3), an analysis can be executed.

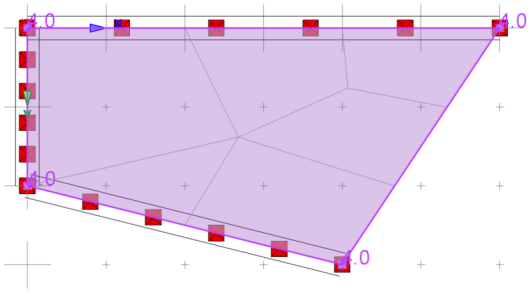


Figure B.3: Solved planar problem

After the calculation is completed, a text file, which contains the geometrical information about the mesh, can be extracted (Figure B.4).

subs

```
VERSION 15    02.12.2019  11:07:04
EXTERN-VERSION 19.8 23102019
MAX-NODE-NUMBER 11
MAX-ELEMENT-NUMBER 6
ND-COOR
    1   0.000000    0.000000    0.000000
    2   2.000000    0.000000    0.000000
    3   2.679178    1.383832    0.000000
    4   0.000000    2.000000    0.000000
    5   4.000000    0.000000    0.000000
    6   4.070559    0.761692    0.000000
    7   6.000000    0.000000    0.000000
    8   5.329928    1.005108    0.000000
    9   2.000000    2.500000    0.000000
   10   4.664964    2.002554    0.000000
   11   4.000000    3.000000    0.000000
ND-END-COOR
EL-TOP
P04Q  1  4      1      2      3      4
P04Q  2  4      2      5      6      3
P04Q  3  4      5      7      8      6
P03T  4  3      4      3      9
P04Q  5  4      3     10     11      9
P04Q  6  4      8     10      3      6
EL-END-TOP
EL-CORSYS
 1 x-y 0  0 0 0  1 0 0  0 1 0
EL-END-CORSYS
EL-DESIGN
  1 TO 6  - 0
EL-END-DESIGN
```

Figure B.4: Generated input text file

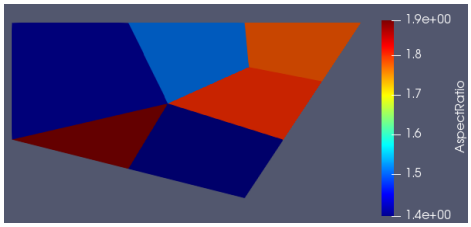
The text file is used as an input file to the mesh quality evaluating tool MeshEvaluator created as a part of the thesis. The geometry of the mesh is required in order to calculate mesh quality metrics. After the calculation is done, the output XML file is created by the MeshEvaluator (Figure B.5). It provides the geometry of the mesh as well as the calculated metrics.

```
<?xml version="1.0"?>
<VTKFile type="UnstructuredGrid">
    <UnstructuredGrid>
        <Piece NumberOfPoints="11" NumberOfCells="6">
            <PointData>
                <DataArray Name="PointNumbers" NumberOfComponents="1" type="Float32" format="ascii">
0
1
2
3
4
5
6
7
8
9
10
</DataArray>
            </PointData>
            <CellData>
                <DataArray Name="CellNumbers" NumberOfComponents="1" type="Float32" format="ascii">
0
1
2
3
4
5
</DataArray>
                <DataArray Name="ElementArea" NumberOfComponents="1" type="Float32" format="ascii">
4.06301
1.93568
1.47615
1.28596
2.48203
1.75717
</DataArray>
                <DataArray Name="AspectRatio" NumberOfComponents="1" type="Float32" format="ascii">
1.40241
1.50609
1.78018
1.88756
1.39251
1.80085
</DataArray>
                <DataArray Name="Skewness" NumberOfComponents="1" type="Float32" format="ascii">
0.290462
0.290462
0.374334
0.550198
0.218376
0.61077
</DataArray>
                <DataArray Name="JacobianRatio" NumberOfComponents="1" type="Float32" format="ascii">
0.689093
0.602368
0.654416
1
0.930547
0.654761
</DataArray>
```

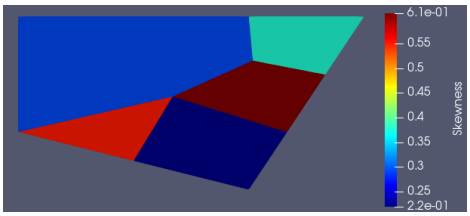
```
                <DataArray Name="Distortion" NumberOfComponents="1" type="Float32" format="ascii">
0.815932
0.751847
0.791115
1
0.964024
0.791366
</DataArray>
                <DataArray Name="NumberOfNodes" NumberOfComponents="1" type="Int32" format="ascii">
4
4
4
3
4
4
</DataArray>
                <DataArray Name="Configuration" NumberOfComponents="1" type="Int32" format="ascii">
1
1
1
2
1
1
</DataArray>
                <DataArray Name="MidpointDifference" NumberOfComponents="1" type="Float32" format="ascii">
0
0
0
0
0
0
</DataArray>
            </CellData>
            <Points>
                <DataArray Name="PointsCoordinates" NumberOfComponents="3" type="Float32" format="ascii">
0.000000 0.000000 0.000000
2.000000 0.000000 0.000000
2.679178 1.383832 0.000000
0.000000 2.000000 0.000000
4.000000 0.000000 0.000000
4.070559 0.761692 0.000000
6.000000 0.000000 0.000000
5.329928 1.005108 0.000000
2.000000 2.500000 0.000000
4.664964 2.002554 0.000000
4.000000 3.000000 0.000000
</DataArray>
            </Points>
            <Cells>
                <DataArray Name="connectivity" type="Float32" format="ascii">
0 1 2 3
1 4 5 2
4 6 7 5
3 2 8
2 9 10 8
7 9 2 5
</DataArray>
                <DataArray Name="offsets" type="Int32" format="ascii">
4
8
12
15
19
23
</DataArray>
                <DataArray Name="types" type="UInt8" format="ascii">
9
9
9
5
9
9
</DataArray>
            </Cells>
        </Piece>
    </UnstructuredGrid>
</VTKFile>
```

Figure B.5: XML output file

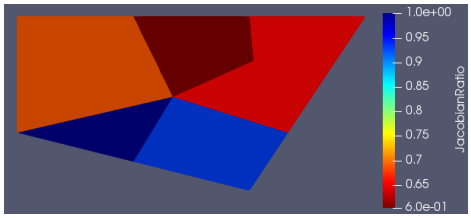
XML output can be uploaded into the graphical software *ParaView* in order to get a visualization of a solved problem. Calculated metrics can be displayed in color scale (Figure B.6).



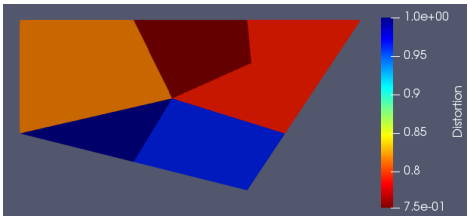
(a) Aspect ratio



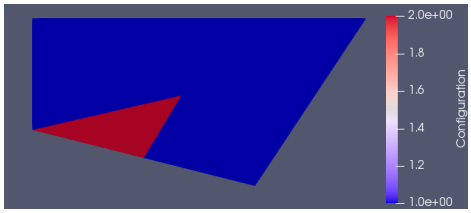
(b) Skewness



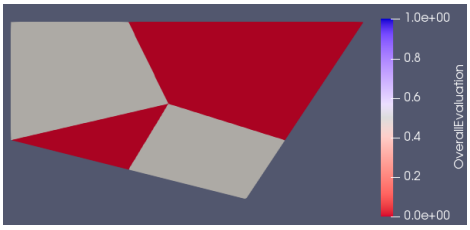
(c) Jacobian ratio



(d) Distortion



(e) Configuration



(f) Overall evaluation

Figure B.6: Graphical visualization in *ParaView*

Appendix C

Additional checks

C.1 Cantilever beam

An analysis was conducted in order to clarify increasing stresses at the upper and bottom edges of the support of the cantilever beam example (Figure 5.1). Principal stresses as well as normal stresses are observed to be increasing as the mesh gets denser and no converging trend is observed.

In order to analyze that, two types of the boundary conditions are applied on the cantilever vertical edge:

(a) line support that restrains:

- displacements in x- and z-directions, rotation in y-direction

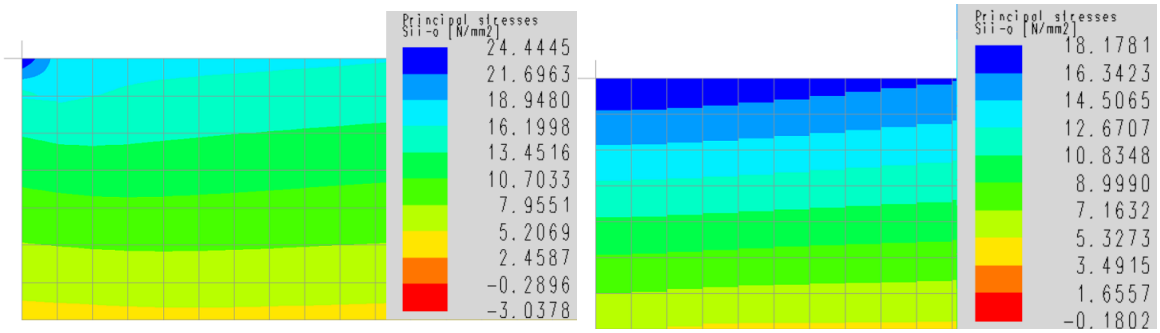
(b) line support that restrains:

- displacements in x-direction, rotation in y-direction and

point support in the middle that restrains:

- displacements in x- and z-directions and rotation in y-direction.

Figure C.1 depicts isolines of the principal stress in the upper corner. Dense regular mesh is used (2400 elements). Figure C.1a shows the stress peak that is created presumably due to the movement restrictions in all directions. The highest calculated value reaches 24.44 MPa. If even denser mesh is used, the stress peak becomes higher. On the other hand, Figure C.1b displays the principal stress obtained from the analysis, which restraines displacement in z-direction only at the center of the vertical edge. There is allowable vertical movement in the corners and therefore the principal stress is smaller and no peaks occur. In addition, the same value is obtained for normal stress, which complies with the assumption that only the bending stress acts at the edge of the cross-section. The shear stress is equal to zero at the top and the bottom surface. The calculated stress value is 18.18 MPa, which is reasonable in comparison with the analytical solution (18 MPa). Denser mesh provides similar solution, therefore convergence can be observed.



(a) Three directions fixed

(b) Two directions fixed

Figure C.1: Comparison of the different boundary conditions in the top corner

C.2 Two-way slab

Deviations in the results were observed while analyzing the two-way slab from Figure 5.8 in different programs and by considering the given analytical solution. The following paragraphs clarify their cause.

Both analytical solution and the tool *SlaFoR* are based on *Kirchhoff theory*. However, *RIB iTWO structure fem* is set to calculate according to *Mindlin theory*, which takes the shear deformation into account. Therefore the calculated displacement is larger.

The effect of the shear is demonstrated in Figure C.2 where the ratio between the displacement calculated by *RIB iTWO structure fem* and the analytical value is captured in dependence on the density of the mesh. Thinner slab has smaller shear effect and hence the calculated displacement gets closer to the analytical solution assuming Kirchhoff theory.

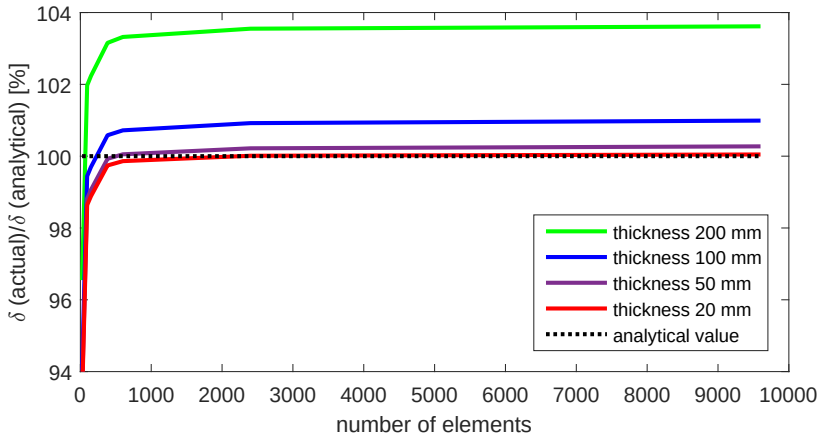


Figure C.2: Convergence check