Department of Construction Sciences
Solid Mechanics

ISRN LUTFD2/TFHF-21/5243-SE(1-57)

Isogeometric analysis as a tool for large deformation shape optimization

Master's Dissertation by

Max Kurki

Supervisors:
Prof. Mathias Wallin, Div. of Solid Mechanics

Examiner:
Assoc. Prof. Håkan Hallberg, Div of Solid Mechanics

Abstract

The framework for isogeometric analysis (IGA) is presented, especially the construction and use of non-uniform rational B-splines (NURBS) to represent geometry, and a method utilizing IGA for structural optimization both for linear elastic and non-linear elastic deformations is implemented. Topology optimization of a linear elastic cantilever is performed, showing that IGA is inefficient for topology optimization. Shape optimization of the same cantilever, as well as two non-linear hyperelastic brackets, is performed producing promising results. IGA is found to be efficient in representing geometries using fewer parameters than classic finite elements, allowing for shape optimization with considerably fewer design variables. A pseudo-contact formulation is implemented, and one of the brackets is optimized using this formulation, the results of which indicate that IGA shows promise for use with contact problems as well. Some downsides to IGA are also found, resulting from the more complex geometry representation that NURBS entail.

Acknowledgements

First of all I would like to thank my supervisor professor Mathias Wallin for the continued support and encouragement during this project! He has provided a much appreciated optimistic outlook throughout the past few months, and he was able to ask exactly the right question to make me think about and further understand my own work!

Also a thank you to M.Sc Anna Dalklint and M.Sc Filip Sjövall for joining in on our meetings and providing some well needed perspectives during discussions!

Finally a thank you to my best friend and bartender extraordinaire Alex Haase for meaningful distractions from working once in a while, and making sure I had some amount of social interaction during this time of isolation!

Contents

1 Introduction

Structural optimization is the method by which the optimal size, shape, and topology of a structure is found, given certain constraints in order to maximize desirable traits. This is generally done through the use of finite element analysis (FEA), commonly with triangular or rectangular elements. An alternative to theses polygonal elements was suggested by Hughes et. al. in 2005 [1]. Their idea was to represent the geometry using isogeometric elements based on non-uniform rational B-splines (NURBS), calling this concept isogeometric analysis (IGA). A method for shape optimization using IGA was developed by Wall et. al in 2008, but only considering small deformations [2].

The idea of using NURBS for representing complex geometries was suggested by Versprille already in 1975 [3], built upon by Tiller in 1983 [4], and the theory was more or less finalized by Piegl and Tiller in 1987 [5]. The benefit of using NURBS is its ability to represent most geometries exactly, including circles and other conic sections, thus eliminating the approximations necessary when constructing geometries from polygons. A further motivation for the choice of using NURBS is that CAD software already uses it for constructing geometries, and IGA would allow one to perform optimization using output from CAD software with no changes to the geometry, and the optimized structure could just as easily be transferred back into the CAD program.

An application where there is a need for exact geometry representation is for example when modeling wave propagation, as can be seen in an 2018 article by Alberdi et. al. dealing with phononic crystals [6]. In order to accurately model the wave propagation the boundaries of the periodic structure in the crystal needs to be very smooth, and IGA offers the ability to model this boundary very accurately while using substantially fewer degrees of freedom than classic FEA.

This thesis aims to investigate what benefits and drawbacks IGA has compared to classic FEA, with the main focus being on its usefulness for large deformation shape optimization. Is it possible to achieve accurate results while keeping system matrices small? What changes in computational cost are there when using IGA over classic FEA? Does the use of NURBS for geometry representation present any new difficulties?

The thesis will open with a fairly detailed description of NURBS, highlighting certain strengths and weaknesses, in order to give a good basis for understanding the effects the geometry representation has on the optimization process. A continuum description of the governing equations will then be presented, both for linear and non-linear elasticity. The NURBS based finite element formulation will be presented, once again both for the case of linear and non-linear elasticity. To motivate the focus on shape optimization, a short section where topology optimization of a cantilever is performed is included where some drawbacks to IGA become apparent. As proof of concept a simple shape optimization of a linear elastic cantilever is first done, before moving on to the main focus of the thesis which is the shape optimization of large deformation structures. This is done for two different geometries: a square bracket and a semi-circular bracket. Finally, to demonstrate the strength of the geometry representation shape optimization of the same square bracket will be performed using a pseudo-contact model.

2 Geometry representation using NURBS

Non-uniform rational B-splines (NURBS) are, as the names implies, based on basis splines (B-splines) which are polynomial functions that can be used to represent curves, surfaces, and solids in physical space. Unlike with the classic FEM, geometries can be represented exactly with NURBS and for most geometries NURBS allows for substantially fewer elements to be used. Before moving on to NURBS, an introduction of B-splines is necessary. A table of the nomenclature used in section 2 can be found in appendix A.

2.1 B-splines

In one dimensional parameter space, B-splines are defined on a *knot vector*, which is a set of non-decreasing coordinates, e.g. $\Xi = [\xi_1, \xi_2, \dots \xi_{n+p+1}]$. Here ξ_i is the i^{th} knot, i is the index, n is the number of basis functions, and p is the polynomial order of the basis functions. The knot vector determines the shape of the basis functions, as well as the number of elements that the represented curve (usually referred to as a *patch*) is partitioned into. If the knots are all equally spaced then the knot vector is called *uniform.*

The basis functions are defined recursively, starting with polynomials of degree zero. The lowest order basis functions are thus constant functions that are non-zero only over an interval $[\xi_i, \xi_{i+1})$, which is referred to as a *knot span*, according to

$$N_{i,0}(\xi) = \begin{cases} 1 & \xi_i \le \xi < \xi_{i+1} \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

These are then used to create basis functions of higher order according to

$$N_{i,p}(\xi) = \frac{(\xi - \xi_i)N_{i,p-1}(\xi)}{\xi_{i+p} - \xi_i} + \frac{(\xi_{i+p+1} - \xi)N_{i+1,p-1}(\xi)}{\xi_{i+p+1} - \xi_{i+1}}, \tag{2.2}$$

which is called the *Cox-de Boor formula* [7][8]. Using (2.2) directly becomes cumbersome as the polynomial order increases, and more efficient algorithms, which are still based on the Cox-De Boor formula, are available for computing the basis functions. These algorithms will not be covered here but the ones that will be used have been adapted from algorithms in *The NURBS book* [9].

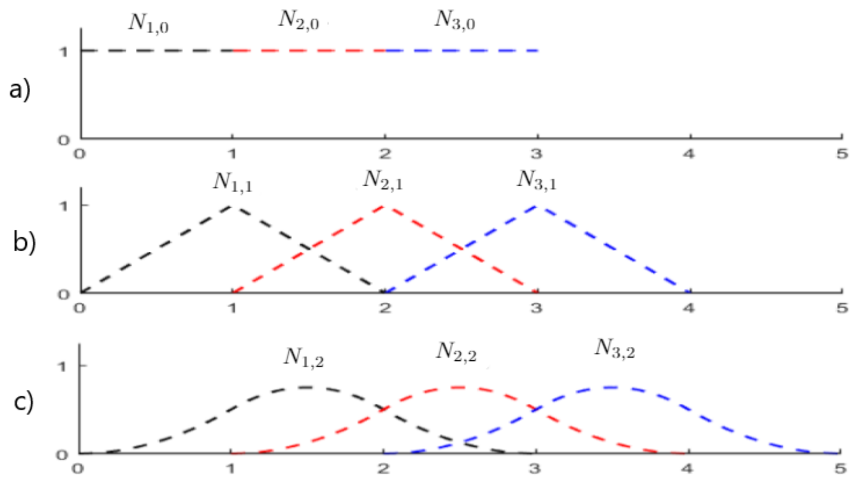

Figure 2.1: B-spline basis functions of order a) 0, b) 1, c) 2, defined on a uniform knot vector $\Xi = [0, 1, 2, 3, 4, \dots]$.

Figure 2.1 shows the first three basis functions of order 0-2 defined on a uniform knot vector $\Xi = [0, 1, 2, 3, 4, \dots]$. A few observations should be made in this figure. First, it is clear that increasing the order of the basis function by one also increases the number of knot spans in which it has support by one. This can also be realized by looking at (2.2) since each $N_{i,p}(\xi)$ will be non-zero on the intervals where $N_{i,p-1}(\xi)$ and $N_{i+1,p-1}(\xi)$ are non-zero. Secondly, each individual basis function is pointwise non-negative everywhere. Thirdly, the basis is a partition of unity, that is, the sum of all base functions at each point in the interval $[\xi_{p+1}, \xi_{n+1})$ is equal to one. The final observation is that each basis function of order p has $p-1$ continuous derivatives across knots, which means that the derivatives will be continuous across element boundaries when used for analysis.

It is possible, and often useful, to define a non-uniform knot vector which will govern the properties of the basis functions, and especially interesting are knot vectors where certain knots appear more than once. An example of this is shown in figure 2.2 where the quadratic basis functions have been calculated for the knot vector $\Xi = [0, 0, 0, 1, 2, 3, 3, 4, 5, 5, 5]$. A knot vector where the first and last knot is repeated $p+1$ times is called an *open knot vector*. An observation here is that while the quadratic basis functions are generally C^1-continuous, at $\xi = 3$ where we have the repeated knot the curve is C^0-continuous. It is a general rule when repeating knots that the basis functions at that point will be C^{p-m_i}-continuous, where m_i is the multiplicity of the knot. A result of the open knot vector is that the basis functions at the first and last knot are C^{-1}-continuous, i.e. discontinuous, which creates a natural boundary for the patch.

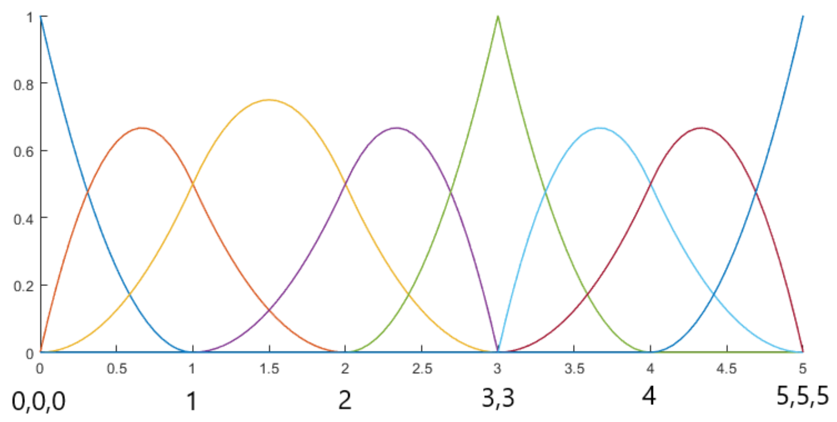

Figure 2.2: Quadratic B-spline basis functions defined on the knot vector $\Xi = [0, 0, 0, 1, 2, 3, 3, 4, 5, 5, 5]$.

Now that the basis functions have been properly introduced, it is possible to use them to construct various geometries, starting with a simple curve. A B-spline curve $\boldsymbol{C}(\xi)$ is given by

$$\boldsymbol{C}(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi)\boldsymbol{B}_i \tag{2.3}$$

where $\boldsymbol{B}_i \in \mathbb{R}^d, i = 1, 2 \ldots, n$ are the *control points* for the curve. An example of a curve is shown in figure 2.3 using the basis functions from figure 2.2.

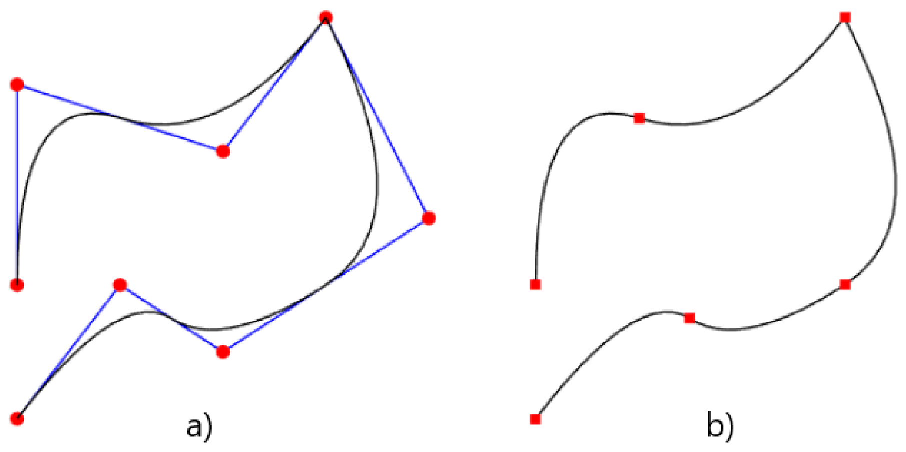

Figure 2.3: A B-spline curve (in black) constructed using quadratic base functions. a) Red dots indicate control points. b) Red squares indicate knots which segment the curve into elements.

The control points are given in vector form as

$$\boldsymbol{B}_i = \big[(1,1) \quad (2,3) \quad (3,2) \quad (5,4) \quad (4,7) \quad (3,5) \quad (1,6) \quad (1,3)\big]\,.$$

In a) the curve is shown with the control points, and a result of the repeated knot at $\xi = 3$ is visible here, namely that the curve becomes interpolatory at this point, assuming the exact value of the corresponding control point. This is also true at the beginning and end of the curve, as a result of the open knot vector. In b) the knots have been marked instead, and the resulting segmentation produces elements analogous to classic finite elements. The smoothness of the curve depends entirely on the basis functions, and the benefit of repeating a knot and having C^0-continuity is shown here, in that sharp corners can be achieved.

Once the B-spline curve is constructed it is straightforward to create B-spline surfaces, and by extension also solids which will however not be covered here. A B-spline surface is defined in much the same way as a B-spline curve, but using two knot vectors to create a two-dimensional parameter space. Control points are now connected in a *control net* $\boldsymbol{B}_{ij}, i = 1, 2, \ldots, n, j = 1, 2, \ldots, m$, and the two knot vectors are given by $\Xi = [\xi_1, \xi_2, \ldots, \xi_{n+p+1}]$ and $H = [\eta_1, \eta_2, \ldots, \eta_{m+q+1}]$ with polynomial degrees p and q. This then gives the B-spline surface as

$$\boldsymbol{S}(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta) \boldsymbol{B}_{ij}. \tag{2.4}$$

An example using the knot vectors $\Xi = [0, 0, 0, 0.5, 0.5, 1, 1, 1]$ and $H = [0, 0, 0, 1, 1, 1]$ is shown in figure 2.4: a) The control net, b) The patch in physical space and the two resulting elements, c) The patch in parameter space, with the basis functions from the two knot vectors. The control net is given in matrix form as

$$\boldsymbol{B}_{ij} = \begin{bmatrix} (0,0) & (-1,0) & (-2,0) \\ (0,1) & (-1,1) & (-2,1) \\ (0.5,1.5) & (-1,3) & (-2,4) \\ (1,2) & (1,3) & (1,4) \\ (2,2) & (2,3) & (2,4) \end{bmatrix}$$

Like before, the repeated knot at $\xi = 0.5$, together with the open knot vector in the η-direction, makes the surface interpolatory at all corresponding control points, which are the two control points on the diagonal line in figure 2.4a) that lie on the edge of the design.

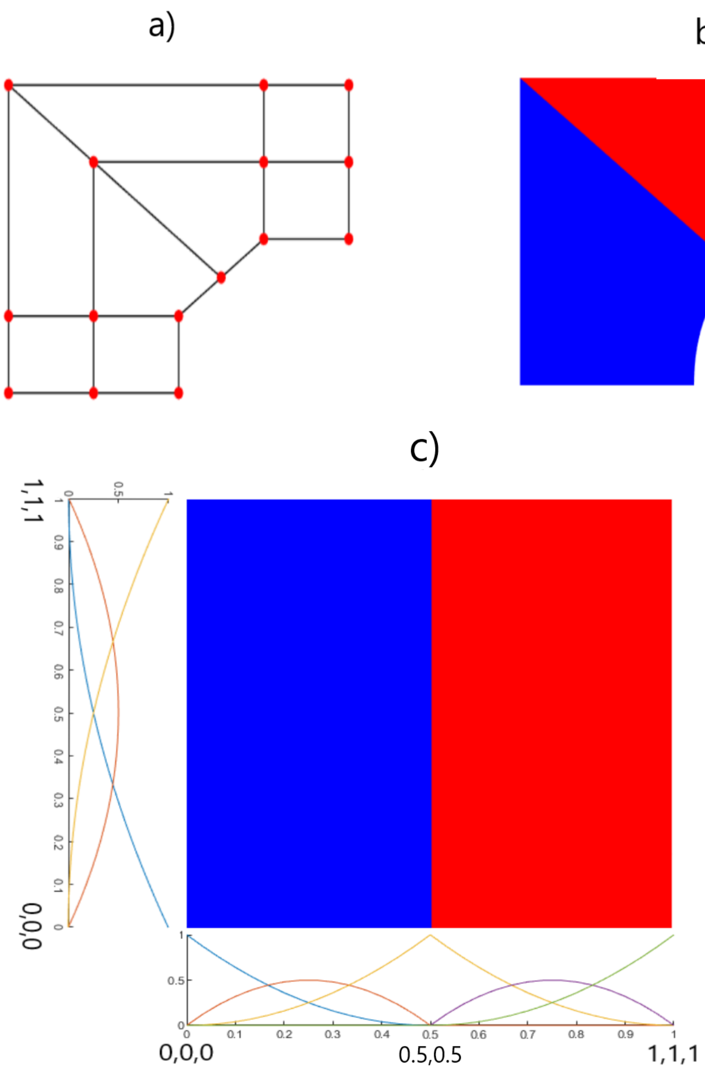

Figure 2.4: A B-spline surface constructed using quadratic base functions. a) The control net. b) The patch in physical space and the two resulting elements. c) The patch in parameter space, with the basis functions from the two knot vectors.

2.2 Knot insertion

One of the defining tools when working with NURBS is so called *knot insertion*. This simply means adding knots to the knot vector, and calculating new basis functions and control points in such a way that the geometry remains unchanged. This allows the solution space to be refined as desired without having to rebuild the whole geometry from scratch each time. From the definition of the knot vector we have that the number of knots, N, can be written as

$$N = n + p + 1 \iff n = N - p - 1 \tag{2.5}$$

which shows that adding a knot also adds one new basis function. Since the number of control points is equal to the number of basis functions, a new control point must be inserted. The new basis functions are simply calculated as before, using the Cox-de Boor formula, but the new control points must be calculated from the old control points so as to not disrupt the geometry. This is done similarly to the Cox-de Boor formula according to

$$\bar{\boldsymbol{B}} = \boldsymbol{T}^p \boldsymbol{B} \tag{2.6}$$

where $\bar{\boldsymbol{B}}$ are the updated control points, and the matrix $\boldsymbol{T}^p$ is calculated according to

$$T_{ij}^0 = \begin{cases} 1 & \xi_j \leq \bar{\xi}_i < \xi_{j+1} \\ 0 & \text{otherwise.} \end{cases} \tag{2.7}$$

and

$$T_{ij}^{k+1} = \frac{\bar{\xi}_{i+k} - \xi_j}{\xi_{j+k} - \xi_j} T_{ij}^k + \frac{\xi_{j+k+1} - \bar{\xi}_{i+k}}{\xi_{j+k+1} - \xi_{j+1}} T_{ij+1}^k \tag{2.8}$$

where $\bar{\xi}_i$ are the knots in the now expanded knot vector, and ξ_j are the knots in the old knot vector. This is known as Boehm's algorithm [10]. This algorithm only allows for insertion of one knot at a time, but can of course be repeated for as many knots as one wants. An example of knot insertion is shown in figure 2.5, where the curve from figure 2.3 is unchanged after inserting one additional knot at $\xi = 2.5$.

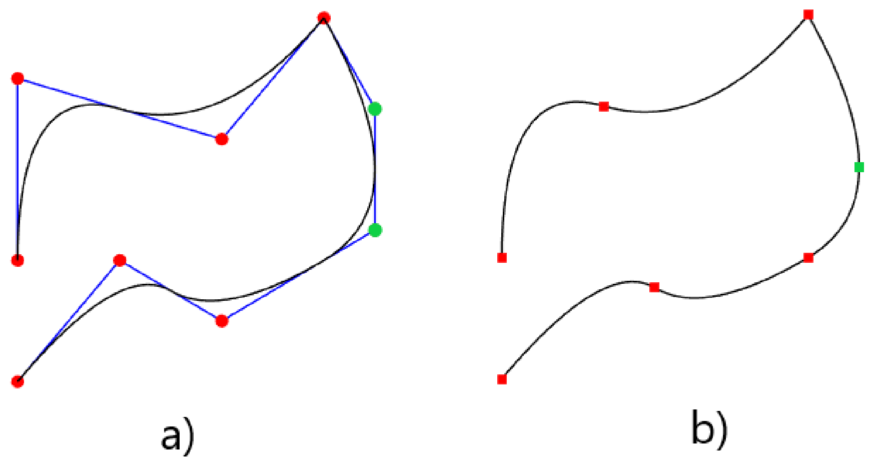

Figure 2.5: The same geometry as shown in figure 2.3, with one additional knot inserted at $\xi = 2.5$. a) The affected control points are shown in green. b) The inserted knot is shown in green.

The new control points are

$$\boldsymbol{B}_i = \big[(1,1) \quad (2,3) \quad (3,2) \quad (4.5,3.5) \quad (4.5,5.5) \quad (3,5) \quad (1,6) \quad (1,3)\big]\,.$$

and we see that most of the control points remain unchanged. This is of course because the inserted knot only affects the basis functions which have support in that point, while the rest remain the same. It can also be seen that the insertion of a new unique knot results in the addition of a new element. A two dimensional example is shown in figure 2.6, which uses the geometry from figure 2.4, with an additional knot inserted at $\xi = 0.25$.

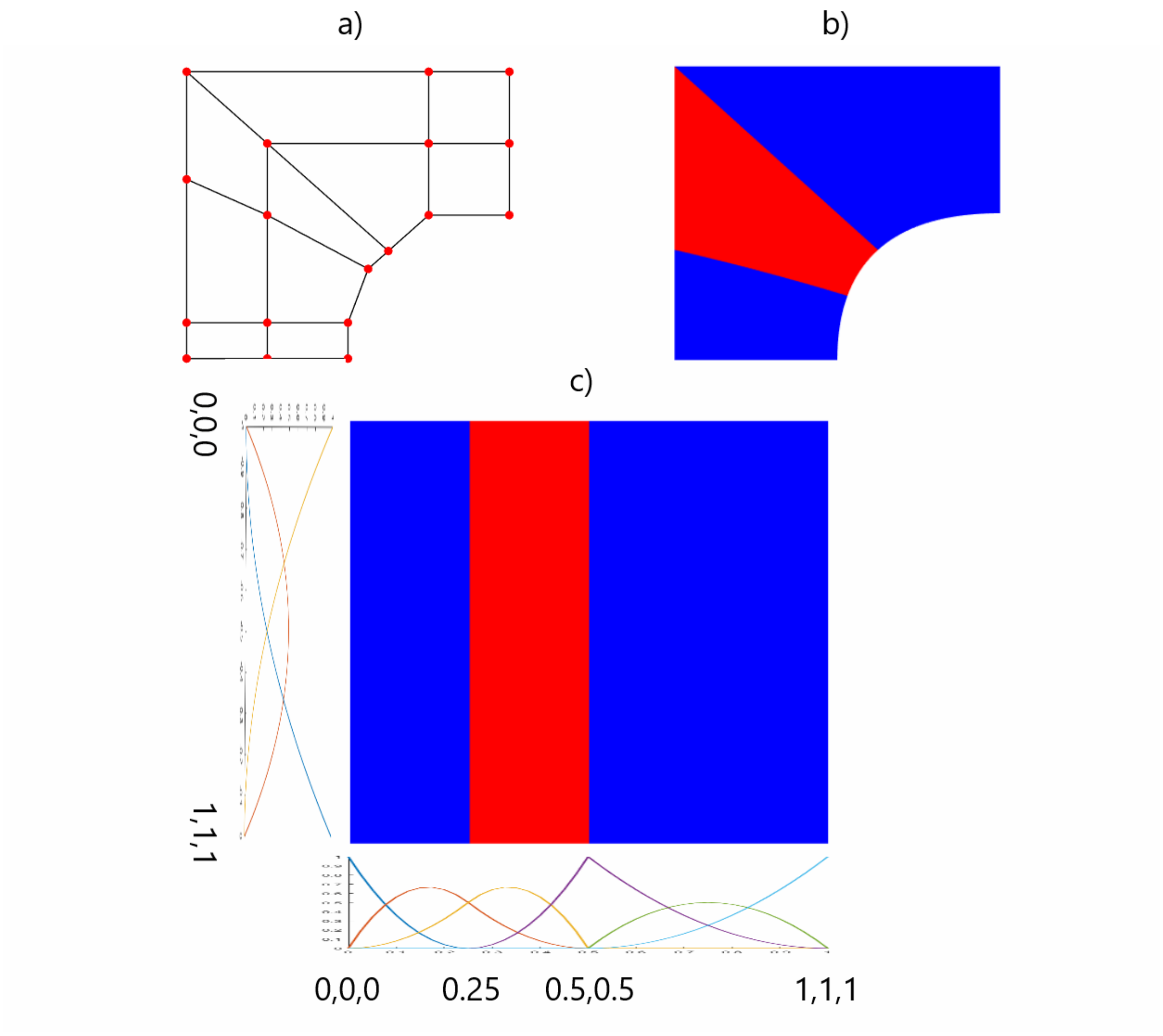

Figure 2.6: The same geometry as shown in figure 2.4, with one additional knot inserted at $\xi = 0.25$. a) The new control net. b) The patch in physical space and the three resulting elements. c) The patch in parameter space, with the basis functions from the two knot vectors.

2.3 NURBS

NURBS are an extension of the regular B-splines, which implies that B-splines are a special case of NURBS. The basis functions used are defined from B-spline basis functions, in conjunction with weighted control points. Similarly to (2.3) the parameterization of a curve is given by

$$\boldsymbol{C}(\xi) = \sum_{i=1}^{n} \frac{N_{i,p}(\xi) w_i}{W(\xi)} \boldsymbol{B}_i = \sum_{i=1}^{n} R_i^p(\xi) \boldsymbol{B}_i \tag{2.9}$$

where R_i^p is the *NURBS basis function*, w_i is the weight of the ith control point, and $W(\xi)$ is the *weighting function*, defined as

$$W(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi) w_i. \tag{2.10}$$

From a geometric point of view, (2.9) is equivalent to performing a perspective projection of a B-spline curve in $\mathbb{R}^{d+1}$ to $\mathbb{R}^d$. An example of this, resulting in a two dimensional NURBS curve, is shown in figure 2.7.

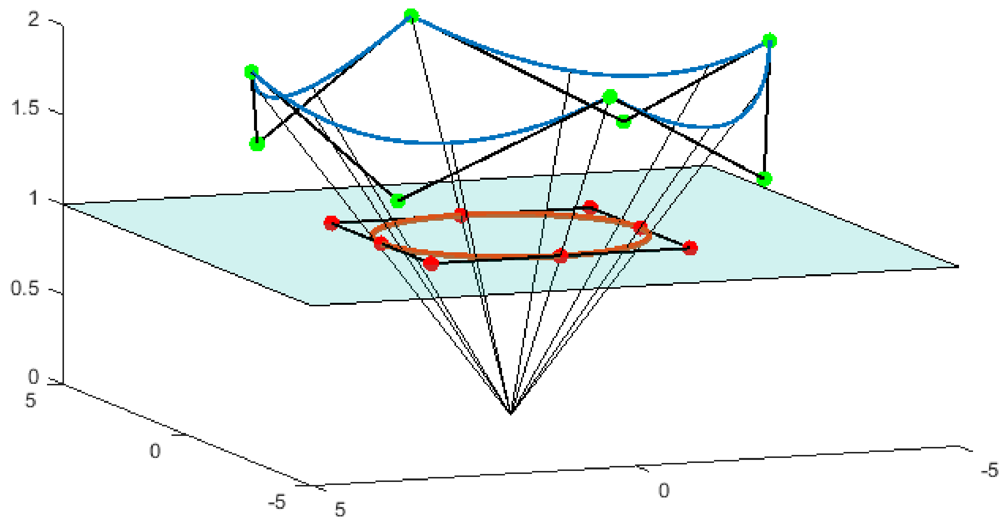

Figure 2.7: A B-spline curve (blue) with corresponding control points (green), and the projective transformation of the curve (orange) with corresponding control points (red).

The control point coordinates (marked in red in figure 2.7) are given by

$$\boldsymbol{B}_i = \begin{bmatrix}(2,0) & (2,2) & (0,2) & (-2,2) & (-2,0) & (-2,2) & (0,-2) & (2,-2) & (2,0)\end{bmatrix}$$

and the associated weights by

$$w_i = \begin{bmatrix}1 & \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} & 1\end{bmatrix}.$$

A B-spline curve (blue) is constructed using nine control points (in green). The first and last control point coincide so as to create a closed curve. Both control points and curve should then be projected onto the plane $z = 1$, which is done along a line from each point towards the origin. This is accomplished by dividing all coordinates of a point with its height, that is $(x, y, z) \to (\frac{x}{z}, \frac{y}{z}, 1)$. The projected curve (orange) is here a circle, and the projected control points (red) form a square. From (2.9) it is now possible to connect this geometrical interpretation to the expression of the NURBS curve. $\boldsymbol{B}_i$ contains the x- and y-coordinates of the projected control points, and the weights w_i are set equal to the height of the B-spline control points. The product $w_i\boldsymbol{B}_i$ then gives the x- and y-coordinates of the green B-spline control points. Thus the parameterization of the x- and y-coordinates of the B-spline curve is given by

$$(x(\xi), y(\xi)) = \sum_{i=1}^{n} N_{i,p}(\xi) w_i \boldsymbol{B}_i. \tag{2.11}$$

and the height by

$$z(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi) w_i. \tag{2.12}$$

Hence the projection of the B-spline curve onto the plane $z = 1$ is

$$\left(\frac{x(\xi)}{z(\xi)}, \frac{y(\xi)}{z(\xi)}\right) = \frac{\sum_{i=1}^{n} N_{i,p}(\xi) w_i \boldsymbol{B}_i}{\sum_{i=1}^{n} N_{i,p}(\xi) w_i} = \sum_{i=1}^{n} \frac{N_{i,p}(\xi) w_i}{W(\xi)} \boldsymbol{B}_i \tag{2.13}$$

and we have obtained the expression for the NURBS curve. The extension of this to surfaces is equally straightforward as in the case of B-splines and gives the expression

$$\boldsymbol{S}(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{N_{i,p}(\xi) M_{j,q}(\eta) w_{ij}}{W(\xi, \eta)} \boldsymbol{B}_{ij} = \sum_{i=1}^{n} \sum_{j=1}^{m} R_{ij}^{pq}(\xi, \eta) \boldsymbol{B}_{ij} \tag{2.14}$$

where the new weighting function is given by

$$W(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta) w_{ij}. \tag{2.15}$$

A major benefit of NURBS over the standard non-rational B-splines is that circles, ellipses, and all other conic sections can be exactly represented. This is because B-splines consist of linear combinations of piecewise polynomial functions, which can only be used to approximate circles. They can however exactly represent parabolas, and with the following definition [11]

> A conic section in two space is the perspective projection of a parabola in Euclidian three space into a plane

clearly NURBS are capable of representing all conic sections as a NURBS curve is simply a perspective projection of a B-spline curve. The above definition clearly also implies that spheres, ellipsoids, and other three dimensional extensions of conic sections can be represented through rotation of the two dimensional objects.

There are, however, some drawbacks to the use of perspective projection to create these shapes. Figure 2.8 shows a NURBS surface in the shape of a ring, where the parameterization in the tangential direction is done using quadratic NURBS, and in the radial direction using linear NURBS.

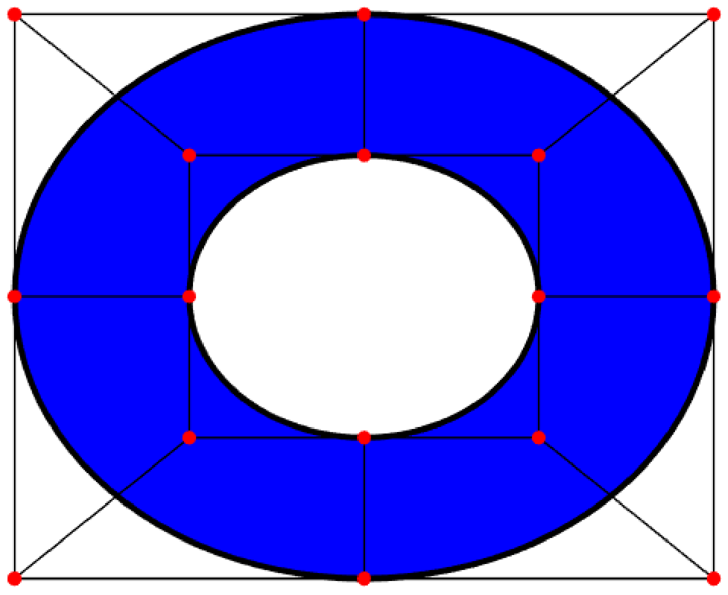

Figure 2.8: A NURBS surface in the shape of a ring, and the control grid used.

The outer ring, and the outermost control points, are the same ones shown in figure 2.7. The knot vector used in the tangential direction is $\Xi = [0,0,0,1,1,2,2,3,3,4,4,4]$, where the repeated internal knots are necessary to produce the four separate 90° arcs of the circle. As a result of these repeated knots the B-spline curve is only C^0-continuous at these points, and the NURBS curve inherits this property as well. This reduced continuity is obvious when looking at the B-spline curve, but after the projection to create the NURBS curve there is no indication of it. For the case of a circle there is no reasonable way to avoid C^0-continuity in some knots, and generally the continuity of any NURBS object is restricted by the shape of its associated B-spline object [12].

3 Continuum description

3.1 Linear elasticity

Before any kind of finite element analysis can be performed, a continuum description of the boundary value problem must be obtained. Starting by introducing the infinitesimal strain tensor as

$$\epsilon_{ij} = u_{(i,j)} = \frac{u_{i,j} + u_{j,i}}{2} \tag{3.1}$$

where u_i is the displacement in the ith direction, and $u_{i,j}$ is the partial derivative of u_i with respect to the jth direction, and applying Hooke's law, gives the stress tensor as

$$\sigma_{ij} = c_{ijkl}\epsilon_{kl} \tag{3.2}$$

where c_{ijkl} is the constitutive tensor, which in this case is the tensor for homogeneous elastic materials. The strong form of the boundary value problem is then given by

$$\begin{aligned} \sigma_{ij,j} + f_i &= 0 \quad \text{in} \quad \Omega && \text{(3.3a)} \\ u_i &= g_i \quad \text{on} \quad \Gamma_D && \text{(3.3b)} \\ \sigma_{ij}n_j &= h_i \quad \text{on} \quad \Gamma_N && \text{(3.3c)} \end{aligned}$$

where f_i is the body force, g_i is the prescribed displacement on the boundary Γ_D, h_i is the traction prescribed on the boundary Γ_N, and n_j represents the normal of the same boundary. The weak form of the problem is obtained by multiplying (3.3a) with a weight function w_i and integrating by parts, resulting in

$$\int_\Omega w_{(i,j)}c_{ijkl}u_{(k,l)}d\Omega = \int_\Omega w_i f_i d\Omega + \int_{\Gamma_N} w_i h_i d\Gamma \tag{3.4}$$

where (3.1) and (3.2) has been used in the first integral, and (3.3c) has been used for the boundary integral.

3.2 Non-linear hyperelasticity

For simplicity a Saint Venant–Kirchhoff model is used for the constitutive relation. Thus a linear constitutive relation is still be used between the strain and stress measurements. These are in this case the Green strain, and the second Piola-Kirchoff stress. For clarity the tensors used here are shown in their matrix representation rather than using index notation. The Green strain is defined as

$$\boldsymbol{E} = \frac{1}{2}\left(\boldsymbol{F}^T\boldsymbol{F} - \boldsymbol{I}\right) \tag{3.5}$$

where $\boldsymbol{F}$ is the deformation tensor

$$\boldsymbol{F} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} \frac{\partial u_x}{\partial x} & \frac{\partial u_x}{\partial y} \\ \frac{\partial u_y}{\partial x} & \frac{\partial u_y}{\partial y} \end{bmatrix} \tag{3.6}$$

.

The Green strain can be written in Voigt notation as

$$\boldsymbol{E} = \begin{bmatrix} E_{xx} & E_{xy} \\ E_{xy} & E_{yy} \end{bmatrix} \implies \underline{\boldsymbol{E}} = \begin{bmatrix} E_{xx} \\ E_{yy} \\ 2E_{xy} \end{bmatrix} \tag{3.7}$$

which allows the constitutive relation to be written as

$$\underline{\boldsymbol{S}} = \boldsymbol{D}\underline{\boldsymbol{E}} \tag{3.8}$$

where $\boldsymbol{D}$ is the same constitutive tensor as in the linear elastic case, and the second Piola-Kirchoff stress is also in Voigt notation. At some points it will be advantageous to write the stress as a square matrix, reversing the process in (3.7), in which case the stress tensor will be denoted $\boldsymbol{S}$, without the bar underneath.

The equilibrium equations are derived by starting again from (3.3a), which can be expressed in matrix notation as

$$div(\boldsymbol{F} \cdot \boldsymbol{S}) + \boldsymbol{b} = 0. \tag{3.9}$$

Multiplying this by a virtual displacement field $\delta\boldsymbol{u}$, and integrating over the entire domain gives the virtual work as

$$\int_\Omega \delta\boldsymbol{u} \cdot div(\boldsymbol{F} \cdot \boldsymbol{S}) d\Omega + \int_\Omega \delta\boldsymbol{u} \cdot \boldsymbol{b} d\Omega = 0 \tag{3.10}$$

which can be rewritten using the Green-Gauss theorem as

$$\int_{\Omega} \delta \boldsymbol{E} : \boldsymbol{S} d\Omega - \int_{\Gamma} \delta \boldsymbol{u} \cdot \boldsymbol{t} d\Gamma - \int_{\Omega} \delta \boldsymbol{u} \cdot \boldsymbol{b} d\Omega = 0 \tag{3.11}$$

where $\boldsymbol{t}$ are the traction forces acting on the boundary Γ. This can be split into the external and internal virtual work as

$$V_{int} = \int_{\Omega} \delta \boldsymbol{E} : \boldsymbol{S} d\Omega, \quad V_{ext} = \int_{\Gamma} \delta \boldsymbol{u} \cdot \boldsymbol{t} d\Gamma + \int_{\Omega} \rho \delta \boldsymbol{u} \cdot \boldsymbol{b} d\Omega. \tag{3.12}$$

Using (3.7) the variation of the Green strain becomes

$$\delta \underline{\boldsymbol{E}} = \begin{bmatrix} \frac{\partial \delta u_x}{\partial x} \\ \frac{\partial \delta u_y}{\partial y} \\ \frac{\partial \delta u_x}{\partial y} + \frac{\partial \delta u_y}{\partial x} \end{bmatrix} + \begin{bmatrix} \frac{\partial u_x}{\partial x} \frac{\partial \delta u_x}{\partial x} + \frac{\partial u_y}{\partial x} \frac{\partial \delta u_y}{\partial x} \\ \frac{\partial u_x}{\partial y} \frac{\partial \delta u_x}{\partial y} + \frac{\partial u_y}{\partial y} \frac{\partial \delta u_y}{\partial y} \\ \frac{\partial \delta u_x}{\partial x} \frac{\partial u_x}{\partial y} + \frac{\partial u_x}{\partial x} \frac{\partial \delta u_x}{\partial y} + \frac{\partial \delta u_y}{\partial x} \frac{\partial u_y}{\partial y} + \frac{\partial u_y}{\partial x} \frac{\partial \delta u_y}{\partial y} \end{bmatrix} \tag{3.13}$$

4 Finite element formulation

Deriving the finite element formulation for IGA is very similar to classic FEA and should be straight forward for anyone who has previous experience with it. The meshing is done slightly differently however. All meshes that will be generated are initially constructed using the minimum number of knots and control points required to obtain the desired shape, and knot insertion is then used to refine the mesh to the desired resolution. This makes meshing very simple and allows one to modify the mesh as required.

In constructing the finite element formulation four connectivity matrices (the INC-, IEN-, ID-, and LM-matrix) are used to keep track of global and local shape function numbers, degrees of freedom, element numbers etc. The system used here is the one developed by Hughes, and an explanation of how these matrices are constructed can be found in appendix B [12].

4.1 Linear elasticity

A finite dimensional approximation of the displacement u_i is introduced according to

$$u_i \approx \sum_{A=1}^{A_{max}} R_A d_{iA} = \boldsymbol{R} \boldsymbol{d}_i \tag{4.1}$$

where the summation is performed over all global shape function numbers, R_A is the shape function with global shape function number A, d_{iA} is the corresponding control points displacement in the ith spatial direction, and $\boldsymbol{R}$ and $\boldsymbol{d}_i$ are the corresponding matrix representations. This can be seen as a slightly modified version of (2.14), where the indexation has been changed. In accordance with Galerkin's method, the weight function w_i is approximated in an analogous way according to

$$w_i \approx \sum_{A=1}^{A_{max}} R_A c_{iA} = \boldsymbol{R}\boldsymbol{c}_i \tag{4.2}$$

where c_{iA} are arbitrary weighting coefficients, and $\boldsymbol{c}_i$ is once again the corresponding matrix representation. For the constitutive relation between stress and strain Voigt notation is again adopted according to

$$\boldsymbol{\sigma} = \boldsymbol{D}\boldsymbol{\epsilon}(\boldsymbol{u}) \tag{4.3}$$

where $\boldsymbol{D}$ is the constitutive matrix, which assuming plane stress is given by

$$\boldsymbol{D} = \frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix} \tag{4.4}$$

where E is the Young's modulus, and ν is the Poisson's ratio, and

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{12} \end{bmatrix}, \quad \boldsymbol{\epsilon}(\boldsymbol{u}) = \begin{bmatrix} u_{1,1} \\ u_{2,2} \\ u_{1,2} + u_{2,1} \end{bmatrix}, \quad \boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}. \tag{4.5}$$

This allows rewriting the left hand side of (3.4) in matrix form as

$$\int_\Omega \boldsymbol{\epsilon}(\boldsymbol{w})^T \boldsymbol{D}\boldsymbol{\epsilon}(\boldsymbol{u}) d\Omega = \boldsymbol{c}^T \int_\Omega \boldsymbol{B}^T\boldsymbol{D}\boldsymbol{B} d\Omega \boldsymbol{d} = \boldsymbol{c}^T\boldsymbol{K}\boldsymbol{d} \tag{4.6}$$

where the matrix $\boldsymbol{B}$ has been introduced according to

$$\boldsymbol{B} = \begin{bmatrix} \frac{\partial R_1}{\partial x} & 0 & \frac{\partial R_2}{\partial x} & 0 & \cdots \\ 0 & \frac{\partial R_1}{\partial y} & 0 & \frac{\partial R_2}{\partial y} & \cdots \\ \frac{\partial R_1}{\partial y} & \frac{\partial R_1}{\partial x} & \frac{\partial R_2}{\partial y} & \frac{\partial R_2}{\partial x} & \cdots \end{bmatrix}. \tag{4.7}$$

The matrix formulation of the right hand side of (3.4) is given by

$$\boldsymbol{c}^T \int_\Omega \boldsymbol{R}^T f_i d\Omega + \boldsymbol{c}^T \int_{\Gamma_N} \boldsymbol{R}^T h_i d\Gamma_N = \boldsymbol{c}^T\boldsymbol{F}. \tag{4.8}$$

This gives the matrix equation

$$\boldsymbol{c}^T \boldsymbol{K}\boldsymbol{d} = \boldsymbol{c}^T \boldsymbol{F} \iff \boldsymbol{K}\boldsymbol{d} = \boldsymbol{F} \tag{4.9}$$

where the equivalence holds since the entries of $\boldsymbol{c}^T$ are arbitrary. The stiffness matrix $\boldsymbol{K}$ is calculated element wise by Gaussian quadrature according to

$$\boldsymbol{k}^e = \int_{\Omega^e} t\boldsymbol{B}^T \boldsymbol{D}\boldsymbol{B}|\boldsymbol{J}|d\Omega \tag{4.10}$$

where t is the element thickness, $|\boldsymbol{J}|$ is the determinant of the mapping from parent element coordinates to physical coordinates, and $\boldsymbol{B}$ is now reduced to contain only the partial derivatives of the local shape functions. Since the shape functions are functions of the parametric coordinates and we need the derivatives with respect to the physical coordinates we need to apply the chain rule, for example

$$\frac{dR_1}{d\boldsymbol{x}} = \frac{dR_1}{d\boldsymbol{\xi}}\frac{d\boldsymbol{\xi}}{d\boldsymbol{x}} \tag{4.11}$$

where $\boldsymbol{x} = (x, y)$ and $\boldsymbol{\xi} = (\xi, \eta)$. The derivatives of the shape functions with respect to the parametric coordinates are given simply by applying the quotient rule to the expression for the NURBS basis functions as (compare with (2.14))

$$\frac{\partial R_A}{\partial \xi} = \frac{\frac{dN_A}{d\xi}M_A W - \frac{\partial W}{\partial \xi}N_A M_A}{W^2} w_A \tag{4.12}$$

and similarly for $\frac{\partial R_A}{\partial \eta}$. The notation here has been changed from R_{ij} to R_A, and the connection between these two notations is given by the INC matrix. The derivatives of the weighting function is given by

$$\frac{\partial W}{\partial \xi} = \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{N_i}{d\xi} M_j w_{ij} \tag{4.13}$$

where we keep the i, j- notation for summation purposes, and similarly for $\frac{\partial W}{\partial \eta}$. The derivatives of the B-spline basis functions N and M can be calculated using (2.2), but much like when generating the basis functions themselves there are more efficient algorithms for calculating the derivatives which are used. Several such algorithms exist, but the one used here is once again based on the method from *The NURBS book* [9]. The gradient of the mapping from physical coordinates to parametric coordinates is obtained by calculating the inverse first. The mapping from parametric coordinates to physical coordinates is given by (2.14), and taking the derivative of this gives

$$\frac{d\boldsymbol{x}}{d\boldsymbol{\xi}} = \sum_{A=1}^{A_{max}} \frac{dR_A}{d\boldsymbol{\xi}}\boldsymbol{B}_A. \tag{4.14}$$

This is then inverted to obtain $\frac{d\boldsymbol{\xi}}{d\boldsymbol{x}}$.

Once the element stiffness matrix has been calculated, the LM matrix can be used to remove the components that correspond to a degree of freedom with prescribed zero displacement. The force vector is calculated element wise as

$$\boldsymbol{f}^e = \int_{\Omega_e} t\boldsymbol{R}f_e|\boldsymbol{J}|d\Omega + \int_{\partial\Omega_e} t\boldsymbol{R}t_e|\boldsymbol{J}|d\Omega \tag{4.15}$$

where $\boldsymbol{R}$ contains the local shape functions for the element. The first integral represents body forces f_e acting on the element, and the second integral represents surface forces from the traction t_e acting on the boundary of the element. In this thesis body forces will not be considered, and in practice the traction forces are applied as point forces and the surface integral rarely needs to be evaluated.

Since the degrees of freedom are the x- and y-positions of the control points, both external forces and boundary conditions are applied at control points, the position of which do not necessarily correspond exactly to the physical boundary. Taking the ring from figure 2.8 and plotting it so that the individual elements are visible gives figure 4.1.

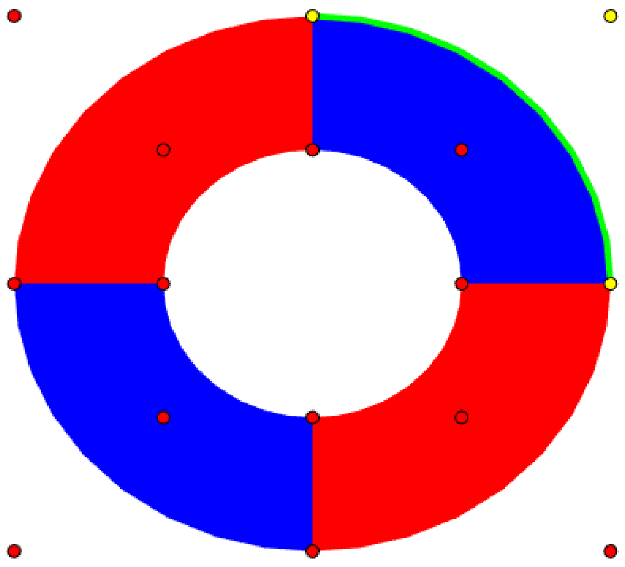

Figure 4.1: A NURBS surface in the shape of a ring, with the resulting elements. The boundary marked in green corresponds to the control points marked in yellow..

The boundary marked in green corresponds to the control points marked in yellow, and in order to prescribe zero displacement along this boundary the six degrees of freedom corresponding to the three control points must be prescribed zero displacement. If a displacement is to be prescribed that is non-zero then the corresponding displacement of the control points must be found first, and this will then be applied while solving the system of equations. This can make prescribing boundary conditions when using IGA slightly more complicated than when using classic FEA, but adding more knots to the knot vector, thus increasing the resolution of the mesh, can make this easier while sacrificing some of the simplicity of a coarser mesh.

The assembly of the global stiffness matrix and force vector is done by extracting the indexation of the local components from the ID or LM matrix and inserting them into the correct position in the global matrices.

As mentioned before all integration is done by Gaussian quadrature, which is performed in a parent element that is identical for all elements. The parent element is a square with side length two and its center at the origin, shown in figure 4.2.

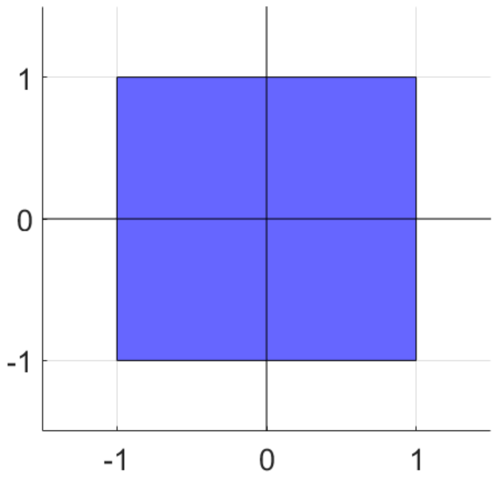

Figure 4.2: The parent element in which Gaussian quadrature is performed.

When performing integration, the Gauss points are provided and the shape functions are evaluated by calculating their value in the corresponding parametric coordinates. Going from the parent element coordinates to parametric coordinates amounts to a linear mapping as the parent element is a square, and all elements in parameter space are rectangular. This mapping, in one dimension, can be written as

$$\xi = \frac{\xi_i + \xi_{i+1}}{2} + \frac{\xi_{i+1} - \xi_i}{2}\bar{\xi} \tag{4.16}$$

where ξ_i and ξ_{i+1} are the two adjacent knot values that define the element, and $\bar{\xi}$ is the parent element coordinate. After the integration has been performed in the parent element the Jacobian determinant of the mapping from parent element coordinates to physical coordinates must be calculated. This is given by

$$|\boldsymbol{J}| = \det\left(\frac{d\boldsymbol{x}}{d\bar{\boldsymbol{\xi}}}\right) = \det\left(\frac{d\boldsymbol{x}}{d\boldsymbol{\xi}}\frac{d\boldsymbol{\xi}}{d\bar{\boldsymbol{\xi}}}\right) \tag{4.17}$$

where we now have the parent element coordinates $\bar{\boldsymbol{\xi}} = (\bar{\xi}, \bar{\eta})$, and the two gradients need to be calculated. The gradient of the mapping from parent element coordinates to parametric coordinates is acquired simply from (4.16), and the gradient of the mapping from parameter space to physical space has already been calculated in (4.14).

4.2 Non-linear hyperelasticity

Introducing the NURBS-approximation as before as well as using Galerkin's method for approximating the virtual displacements

$$\boldsymbol{u} \approx \boldsymbol{R}\boldsymbol{d}, \quad \delta\boldsymbol{u} \approx \boldsymbol{R}\delta\boldsymbol{d} \tag{4.18}$$

allows us to rewrite the Green strain and it's variation as

$$\underline{\boldsymbol{E}} = \left(\boldsymbol{B}_o^l + \frac{1}{2}\boldsymbol{A}(\boldsymbol{d})\boldsymbol{H}_o\right)\boldsymbol{d} \implies \delta\underline{\boldsymbol{E}} = \left(\boldsymbol{B}_o^l + \boldsymbol{A}(\boldsymbol{d})\boldsymbol{H}_o\right)\delta\boldsymbol{d} = \boldsymbol{B}_o\delta\boldsymbol{d} \tag{4.19}$$

with $\boldsymbol{B}_o^l$ from 4.7, $\boldsymbol{H}_o$ as

$$\boldsymbol{H}_o = \begin{bmatrix} \frac{\partial R_1}{\partial x} & 0 & \frac{\partial R_2}{\partial x} & 0 & \dots \\ \frac{\partial R_1}{\partial y} & 0 & \frac{\partial R_2}{\partial y} & 0 & \dots \\ 0 & \frac{\partial R_1}{\partial x} & 0 & \frac{\partial R_2}{\partial x} & \dots \\ 0 & \frac{\partial R_1}{\partial y} & 0 & \frac{\partial R_2}{\partial y} & \dots \end{bmatrix}. \tag{4.20}$$

and

$$\boldsymbol{A}(\boldsymbol{d}) = \begin{bmatrix} \frac{\partial d_x}{\partial x} & 0 & \frac{\partial d_y}{\partial x} & 0 \\ 0 & \frac{\partial d_x}{\partial y} & 0 & \frac{\partial d_y}{\partial y} \\ \frac{\partial d_x}{\partial y} & \frac{\partial d_x}{\partial x} & \frac{\partial d_y}{\partial y} & \frac{\partial d_y}{\partial x} \end{bmatrix} \tag{4.21}$$

where the components are calculated from

$$\boldsymbol{H}_o\boldsymbol{d} = \begin{bmatrix} \frac{\partial d_x}{\partial x} & \frac{\partial d_x}{\partial y} & \frac{\partial d_y}{\partial x} & \frac{\partial d_y}{\partial y} \end{bmatrix}^T. \tag{4.22}$$

The equilibrium equations can then be rewritten in a discrete form as

$$\delta \boldsymbol{d}^T \left(\int_\Omega \boldsymbol{B}_o^T \underline{\boldsymbol{S}} d\Omega - \int_\Gamma \boldsymbol{R}^T t d\Gamma - \int_\Omega \boldsymbol{R}^T b d\Omega \right) = 0 \tag{4.23}$$

or with

$$\boldsymbol{F}_{int} = \int_\Omega \boldsymbol{B}_o^T \underline{\boldsymbol{S}} d\Omega, \quad \boldsymbol{F}_{ext} = \int_\Gamma \boldsymbol{R}^T t d\Gamma + \int_\Omega \boldsymbol{R}^T b d\Omega \tag{4.24}$$

as

$$\boldsymbol{F}_{ext} - \boldsymbol{F}_{int} = \boldsymbol{r} \tag{4.25}$$

where the residual $\boldsymbol{r}$ has also been introduced. At equilibrium we require the residual to be equal to zero, and to this end a Newton-Rhapson method is used. Considering a Taylor expansion of the residual, dropping terms of higher than linear order, gives

$$\boldsymbol{r}(\boldsymbol{d} + \Delta \boldsymbol{d}) = \boldsymbol{r}(\boldsymbol{d}) + \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{d}}(\boldsymbol{d}) \Delta \boldsymbol{d}. \tag{4.26}$$

Since the external forces are constant, the derivative required can be calculated from (3.12) as

$$\frac{\partial V_{int}}{\partial \boldsymbol{d}} = \int_\Omega \frac{\partial \delta \boldsymbol{E}}{\partial \boldsymbol{d}} : \boldsymbol{S} d\Omega + \int_\Omega \delta \boldsymbol{E} : \frac{\partial \boldsymbol{S}}{\partial \boldsymbol{d}} d\Omega = \int_\Omega \left(\frac{\partial^2 \boldsymbol{E}}{\partial \boldsymbol{d}^2} \delta \boldsymbol{d} \right) : \boldsymbol{S} d\Omega + \int_\Omega \left(\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{d}} \delta \boldsymbol{d} \right) : \boldsymbol{D} : \left(\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{d}} \right) d\Omega \tag{4.27}$$

and then

$$\frac{\partial V_{int}}{\partial \boldsymbol{d}} = \delta \boldsymbol{d}^T \frac{\partial \boldsymbol{F}_{int}}{\partial \boldsymbol{d}} = \delta \boldsymbol{d}^T \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{d}} = \delta \boldsymbol{d}^T \left(\int_\Omega \left(\frac{\partial^2 \boldsymbol{E}}{\partial \boldsymbol{d}^2} \right)^T : \boldsymbol{S} d\Omega + \int_\Omega \left(\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{d}} \right)^T : \boldsymbol{D} : \left(\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{d}} \right) d\Omega \right) \tag{4.28}$$

at which point the residual equation can be solved iteratively through

$$\boldsymbol{K}_t^i \Delta \boldsymbol{d}^i = -\boldsymbol{r}(\boldsymbol{d}^i), \quad \boldsymbol{d}^{i+1} = \boldsymbol{d}^i + \Delta \boldsymbol{d}^i \tag{4.29}$$

where $\boldsymbol{K}_t$ is the tangential stiffness matrix found from (4.28) in matrix form as

$$\boldsymbol{K}_t = \int_\Omega \boldsymbol{B}_o^T \boldsymbol{D} \boldsymbol{B}_o d\Omega + \int_\Omega \boldsymbol{H}_o^T \boldsymbol{R}_o \boldsymbol{H}_o d\Omega \tag{4.30}$$

with the matrix $\boldsymbol{R}_o$ defined as

$$\boldsymbol{R}_o = \begin{bmatrix} \boldsymbol{S} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{S} \end{bmatrix} \tag{4.31}$$

5 Topology optimization

The focus of this thesis is on shape optimization, but for the purpose of illustrating some strengths and weaknesses of IGA a simple topology optimization will be performed of a linear elastic cantilever. The geometry is shown in figure 5.1. 385 elements were used for two different choices of the order of the basis functions: $p = q = 2$ and $p = q = 4$. The mesh corresponding to $p = q = 2$ is shown in figure. 5.2.

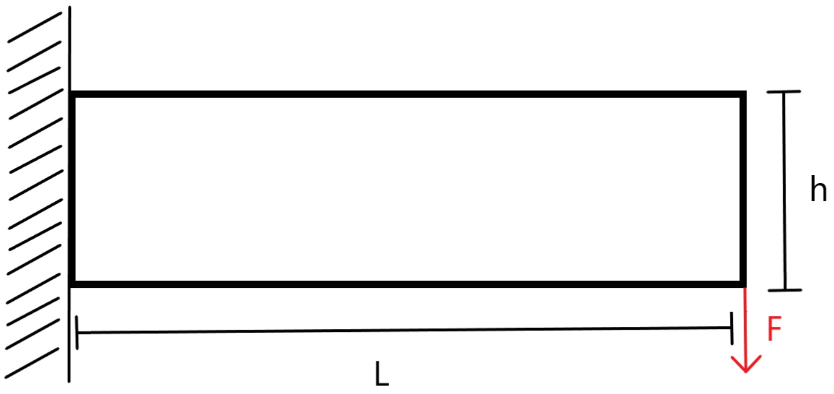

Figure 5.1: The geometry of the cantilever with an applied force.

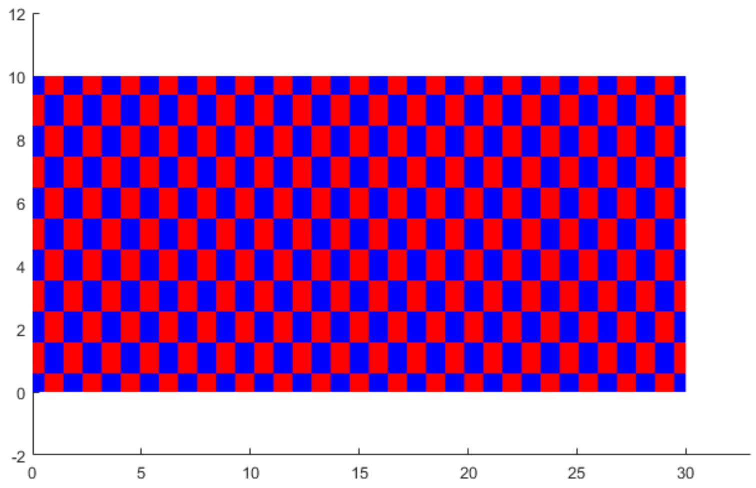

Figure 5.2: The mesh used for optimizing the cantilever with quadratic basis functions.

The objective is to minimize the compliance of the cantilever under a volume constraint. The problem formulation is as follows:

$$\mathbb{P}_1 \begin{cases} \underset{\boldsymbol{\rho}}{min} \quad c = \boldsymbol{F}^T \boldsymbol{d}(\boldsymbol{\rho}) \\ \text{s.t.} \begin{cases} \boldsymbol{K}\boldsymbol{d} = \boldsymbol{F} \\ g_1 = \sum\limits_{e=1}^{nelm} t_e A_e - V_{max} \leq 0 \\ \epsilon \leq \rho_i \leq 1, \quad i = 1...k \end{cases} \end{cases} \tag{5.1}$$

where c is the compliance, ρ_i are the design variables, $nelm$ is the number of elements, t_e and A_e are element thickness and area respectively, V_{max} is the maximum allowed volume, $\epsilon > 0$ is a minimum allowed value for the design variables to avoid singularities, and k is the number of design variables. The design variables will be z-values assigned to each control point used to define the geometry.

The stiffness matrix is modified with a penalization factor applied to the thickness [13] according to

$$\boldsymbol{k}^e = \int_{\Omega^e} t_e^3 \boldsymbol{B}^T \boldsymbol{D} \boldsymbol{B} |\boldsymbol{J}| d\Omega \tag{5.2}$$

When performing the optimization the thickness is assumed to be constant in each element and is calculated as an average according to

$$t_e = \frac{\bar{V}_e}{4} = \int_{\bar{\Omega}_e} \frac{\sum_{a=1}^{a_{max}} R_a \rho_a}{4} d\bar{\Omega} \tag{5.3}$$

where $\bar{V}_e$ is the element volume calculated in the parent element which is then divided by the parent element area of four. The summation inside the integral is done for the elements local shape function numbers a.

5.1 Sensitivities

The sensitivity of the stiffness matrix with respect to the design variables is then calculated element wise according to

$$\frac{\partial \boldsymbol{k}_e}{\partial \rho_i} = \frac{\partial \boldsymbol{k}_e}{\partial t_e} \frac{\partial t_e}{\partial \rho_i} = \int_{\Omega^e} 3 t_e^2 \boldsymbol{B}^T \boldsymbol{D} \boldsymbol{B} |\boldsymbol{J}| d\Omega \int_{\bar{\Omega}_e} \frac{R_i}{4} d\bar{\Omega} \tag{5.4}$$

where it should be noted that while both integrals are evaluated in the parent element, the Jacobian of the mapping to physical coordinates is only applied on the first integral as the thickness is calculated directly in the parent element. From (5.3) we also obtain an expression for the element volume which is used to calculate the structures total volume for the constraint function as

$$V_{tot} = \sum_{e=1}^{nelm} t_e A_e = \sum_{e=1}^{nelm} J_e \bar{V}_e \tag{5.5}$$

where J_e is the determinant of the Jacobian of the mapping from the parent element coordinates to physical coordinates averaged over the element, and this gives the sensitivity of the constraint function as

$$\frac{\partial g_1}{\partial \rho_i} = \sum_{e=1}^{nelm} J_e \frac{\partial \bar{V}_e}{\partial \rho_i} = \sum_{e=1}^{nelm} J_e \int_{\bar{\Omega}_e} R_i d\bar{\Omega}. \tag{5.6}$$

6 Shape optimization

6.1 Linear elasticity

As one of the main benefits of IGA is its ability to accurately represent geometries using much fewer elements than classic FEA, it seems that shape optimization is a very natural application for it. A simple example consisting of the shape optimization of the same cantilever as before will be used as a proof of concept before moving on to more complex problems. The problem formulation is the same as given for the topology optimization in (5.1), except the design variables are no longer the z-coordinates of the control points, but rather parameters controlling the y-coordinates, and the thickness is constant, $t = 1$, in the design domain. Only the top row of control points are directly affected by the optimization process, the bottom row being completely fixed, and the middle row is connected to the top row so that the y-coordinate values are half of that of the control point directly above it. This means that there are only six design variables, one parameter for each of the y-coordinates of the control points in the top row.

$$\mathbb{P}_2 \begin{cases} \underset{\boldsymbol{\alpha}}{min} \quad c = \boldsymbol{F}^T \boldsymbol{d}(\boldsymbol{\alpha}) \\ \text{s.t.} \begin{cases} \boldsymbol{K}\boldsymbol{d} = \boldsymbol{F} \\ g_1 = \sum\limits_{e=1}^{nelm} V_e - V_{max} \leq 0 \\ \epsilon \leq \alpha_i \leq 1, \quad i = 1...k \end{cases} \end{cases} \tag{6.1}$$

The y-coordinates of the control points are parameterized on the form

$$y = c + \alpha L, \quad 0 \leq \alpha \leq 1 \tag{6.2}$$

where c is some minimum constant value, α is the design variable, and L is the maximum allowed movement of the control point. The mesh used is shown in figure 6.1 a), along with an illustration of the parameterization in b). Quadratic basis functions are used in both directions.

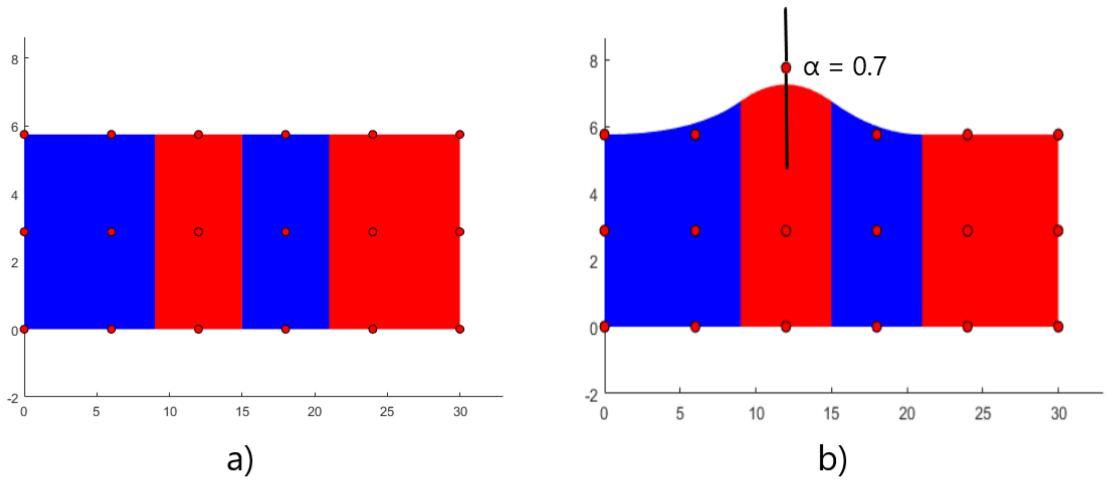

Figure 6.1: a) The element mesh and control points for shape optimization of the cantilever. b) Showing one degree of freedom with the parameterization of the y-coordinate of the control point.

6.1.1 Sensitivities

The sensitivities of the objective function and constraint function with respect to the design variables must be calculated, which is once again done element wise. Starting from (4.10) we have

$$\frac{\partial \boldsymbol{k}^e}{\partial \alpha_i} = \int_{\Omega^e} \frac{\partial \boldsymbol{B}^T}{\partial \alpha_i} \boldsymbol{D}\boldsymbol{B}|\boldsymbol{J}| + \boldsymbol{B}^T\boldsymbol{D}\frac{\partial \boldsymbol{B}}{\partial \alpha_i}|\boldsymbol{J}| + \boldsymbol{B}^T\boldsymbol{D}\boldsymbol{B}^T\frac{\partial |\boldsymbol{J}|}{\partial \alpha_i} d\Omega \tag{6.3}$$

and from (5.5)

$$\frac{\partial V_{tot}}{\partial \alpha_i} = \sum_{e=1}^{nelm} \frac{\partial J_e}{\partial \alpha_i}\bar{V}_e = 4\sum_{e=1}^{nelm} \frac{\partial J_e}{\partial \alpha_i} \tag{6.4}$$

where the last equality comes from the fact that the parent element volume is now constant $\bar{V}_e = 4$ as the thickness is constant $t = 1$. The next step is to derive analytical expressions for $\frac{\partial \boldsymbol{B}}{\partial \alpha_i}$ and $\frac{\partial |\boldsymbol{J}|}{\partial \alpha_i}$ which will be done by the method developed by Haslinger and Mäkinen [14]. From (4.17) we see that the Jacobian used is a product of two other Jacobians, only the first of which is affected by the control points since the mapping from parent element space to parametric space remains unchanged, therefore

$$|\boldsymbol{J}| = \left|\frac{d\boldsymbol{x}}{d\boldsymbol{\xi}}\right|\left|\frac{d\boldsymbol{\xi}}{d\bar{\boldsymbol{\xi}}}\right| = |\boldsymbol{J}_1||\boldsymbol{J}_2| \implies \frac{\partial |\boldsymbol{J}|}{\partial \alpha_i} = \frac{\partial |\boldsymbol{J}_1|}{\partial \alpha_i}|\boldsymbol{J}_2| = |\boldsymbol{J}_1|'|\boldsymbol{J}_2| \tag{6.5}$$

where for convenience the partial derivative $\frac{\partial}{\partial \alpha_i}$ is denoted by a prime, as it will be from here on out. Now, define two matrices as follows

$$\boldsymbol{G} = \begin{bmatrix} \frac{\partial R_1}{\partial x} & \frac{\partial R_2}{\partial x} & \cdots & \frac{\partial R_n}{\partial x} \\ \frac{\partial R_1}{\partial y} & \frac{\partial R_2}{\partial y} & \cdots & \frac{\partial R_n}{\partial y} \end{bmatrix}$$

$$\hat{\boldsymbol{G}} = \begin{bmatrix} \frac{\partial R_1}{\partial \xi} & \frac{\partial R_2}{\partial \xi} & \cdots & \frac{\partial R_n}{\partial \xi} \\ \frac{\partial R_1}{\partial \eta} & \frac{\partial R_2}{\partial \eta} & \cdots & \frac{\partial R_n}{\partial \eta} \end{bmatrix}$$

where n is the number of local basis functions in each element. The chain rule gives

$$\hat{\boldsymbol{G}} = \boldsymbol{J}_1 \boldsymbol{G} \tag{6.6}$$

and rewriting the control point matrix for the element as follows

$$\boldsymbol{X} = \begin{bmatrix} B_{1x} & B_{1y} \\ B_{2x} & B_{2y} \\ \vdots & \vdots \\ B_{nx} & B_{ny} \end{bmatrix}$$

where the index is once again local basis functions numbers, and the naming of $\boldsymbol{X}$ instead of $\boldsymbol{B}_{ij}$ is so as to avoid confusion with the B-matrix, gives

$$\boldsymbol{J}_1 = \hat{\boldsymbol{G}} \boldsymbol{X}. \tag{6.7}$$

Clearly, $\hat{\boldsymbol{G}}$ is not dependent on the control point coordinates as it only contains functions of the parameters ξ and η, and therefore

$$\boldsymbol{0} = \hat{\boldsymbol{G}}' = \boldsymbol{J}_1' \boldsymbol{G} + \boldsymbol{J}_1 \boldsymbol{G}' \tag{6.8}$$

and (6.7) gives

$$\boldsymbol{J}_1' = \hat{\boldsymbol{G}} \boldsymbol{X}' \tag{6.9}$$

Putting (6.6), (6.8), and (6.9) together gives

$$\boldsymbol{G}' = -\boldsymbol{J}_1^{-1} \boldsymbol{J}_1' \boldsymbol{G} = -\boldsymbol{J}_1^{-1} \hat{\boldsymbol{G}} \boldsymbol{X}' \boldsymbol{G} = -\boldsymbol{G} \boldsymbol{X}' \boldsymbol{G}. \tag{6.10}$$

The components of $\boldsymbol{X}'$ are simply calculated as the L-values from (6.2), and once $\boldsymbol{G}'$ is calculated it is straightforward to extract the individual terms and assemble them into their correct positions to obtain $\boldsymbol{B}'$. For the derivative of the Jacobian a result from linear algebra is used:

$$|\boldsymbol{J}_1|' = |\boldsymbol{J}_1| tr \left(\boldsymbol{J}_1^{-1} \boldsymbol{J}_1' \right) \tag{6.11}$$

where tr denotes the trace of the matrix, and using (6.9) and (6.6) gives

$$|\boldsymbol{J}_1|' = |\boldsymbol{J}_1| tr\left(\boldsymbol{J}_1^{-1}\hat{\boldsymbol{G}}\boldsymbol{X}'\right) = |\boldsymbol{J}_1| tr\left(\boldsymbol{J}_1^{-1}\boldsymbol{J}_1\boldsymbol{G}\boldsymbol{X}'\right) = |\boldsymbol{J}_1| tr\left(\boldsymbol{G}\boldsymbol{X}'\right). \tag{6.12}$$

6.2 Non-linear hyperelasticity

The problem formulation is unchanged from the linear case, only the equilibrium equation has changed.

$$\mathbb{P}_3 \begin{cases} \underset{\boldsymbol{\alpha}}{min} \quad c = \boldsymbol{F}^T\boldsymbol{d}(\boldsymbol{\alpha}) \\ \text{s.t.} \begin{cases} \boldsymbol{r} = 0 \\ g_1 = \sum_{e=1}^{nelm} V_e - V_{max} \leq 0 \\ \epsilon \leq \alpha_i \leq 1, \quad i = 1...k \end{cases} \end{cases} \tag{6.13}$$

Two different geometries will be considered: a rectangular bracket, as well as a semi-circular bracket. Both structures are optimized using four different choices of basis function orders of $p = 2, 3, 4, 5$, where p is the order of basis functions running along the length of the structure (the tangential direction for the semi-circular bracket). The basis functions running along the width of the structure (the radial direction for the semi-circular bracket) is fixed to $q = 2$. The geometries and meshes for $p = q = 2$ for the rectangular and semi-circular brackets are shown in figures 6.2 and 6.3 respectively. The control point grid is kept identical for all choices of basis function orders so as to keep the same number of design variables. It should be noted that this increase in order is balanced out by a decrease in number of elements, in accordance with (2.5).

For the square bracket the force is applied downwards at the very center of the top boundary. As for the cantilever only the y-coordinates of the control points at the top boundary are directly affected by the optimization. The middle row of control points are updated so as to be exactly halfway between the corresponding control points at the top and bottom row.

For the semi-circular bracket the force is applied upwards at the very top of the bracket. Both the x- and y-coordinates of the outer row of control points are directly affected by the optimization, and as before the middle row of control points are updated so as to be exactly halfway between the corresponding control points at the inner and outer row.

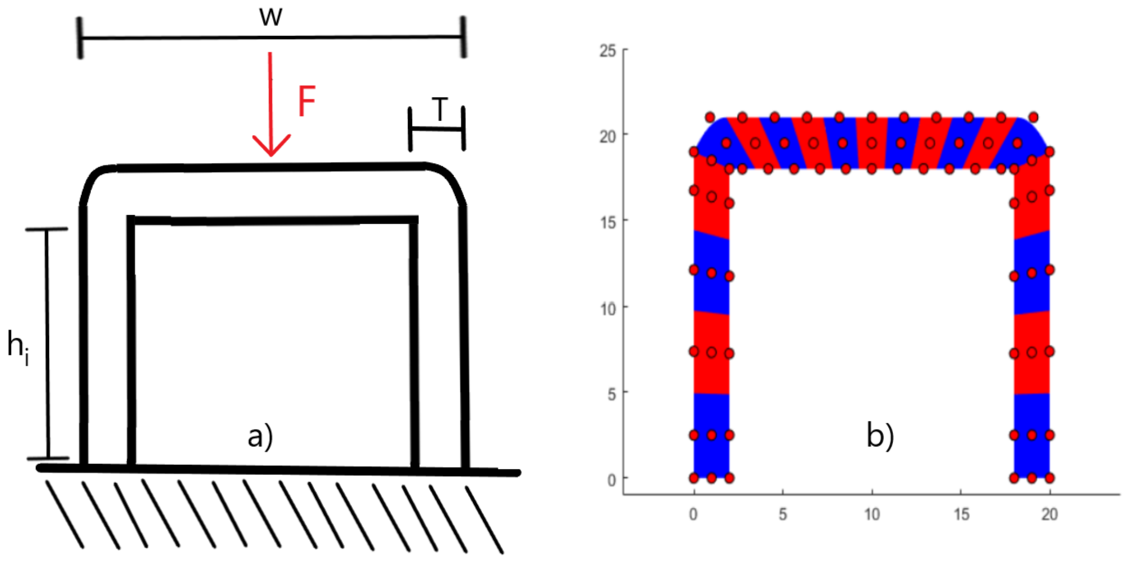

Figure 6.2: a) The geometry for the square bracket b) The mesh when $p = q = 2$.

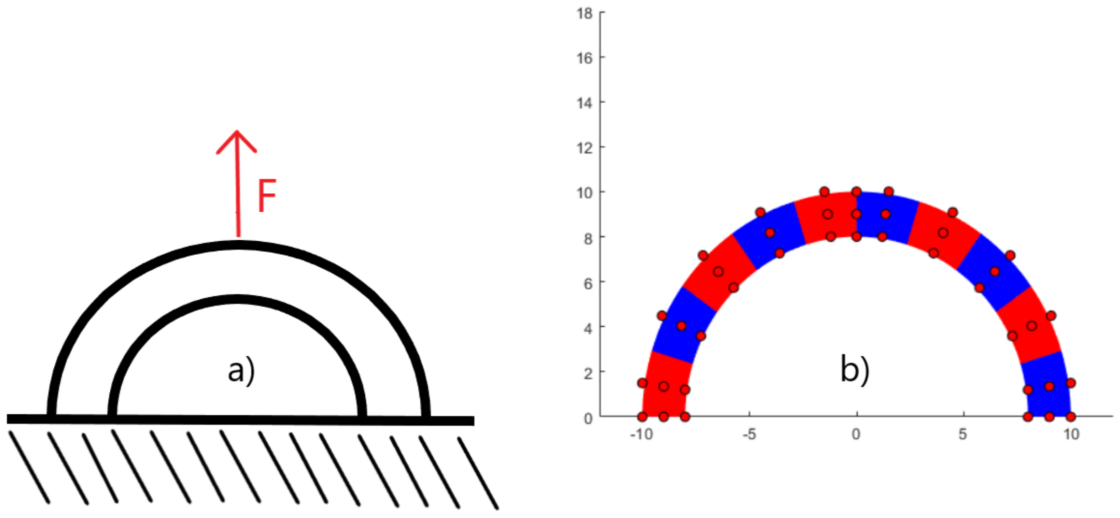

Figure 6.3: a) The geometry for the semi-circular bracket b) The mesh when $p = q = 2$.

6.2.1 Sensitivities

The compliance in the deformed configuration is chosen as objective function, and to calculate the sensitivity of the objective function a modification is made as

$$g_0 = \boldsymbol{F}^T \boldsymbol{d} = \boldsymbol{F}^T \boldsymbol{d} + \boldsymbol{\lambda}^T \boldsymbol{r} \tag{6.14}$$

where the addition of the Lagrangian multipliers $\boldsymbol{\lambda}$ multiplied by the residual does not change the value of the objective function since at equilibrium $\boldsymbol{r} = 0$. Then, assuming design independent loads,

$$\frac{dg_0}{d\alpha_i} = \boldsymbol{F}^T \frac{\partial \boldsymbol{d}}{\partial \alpha_i} + \boldsymbol{\lambda}^T \left(\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{d}} \frac{d\boldsymbol{d}}{d\alpha_i} + \frac{\partial \boldsymbol{r}}{\partial \alpha_i} \right) \tag{6.15}$$

and introducing the adjoint equation

$$\left(\boldsymbol{F}^T - \boldsymbol{\lambda}^T \boldsymbol{K}_t \right) \frac{d\boldsymbol{d}}{d\alpha_i} = 0 \iff \boldsymbol{K}_t \boldsymbol{\lambda} = \boldsymbol{F} \tag{6.16}$$

the solution of which is inserted into (6.15) resulting in

$$\frac{dg_0}{d\alpha_i} = \boldsymbol{\lambda}^T \frac{\partial \boldsymbol{r}}{\partial \alpha_i} \tag{6.17}$$

The sensitivity is then calculated by differentiation of (4.25)

$$\frac{\partial \boldsymbol{r}}{\partial \alpha_i} = -\frac{\partial \boldsymbol{F}_{int}}{\partial \alpha_i} = -\frac{\partial}{\partial \alpha_i} \left(\int_\Omega \boldsymbol{B}_o^T \underline{\boldsymbol{S}} d\Omega \right) \tag{6.18}$$

which is done element wise in practice, so we need to calculate

$$-\frac{\partial \boldsymbol{F}_{int}^e}{\partial \alpha_i} = -\frac{\partial}{\partial \alpha_i} \left(\int_{\Omega^e} \boldsymbol{B}_o^T \underline{\boldsymbol{S}} |\boldsymbol{J}| d\Omega \right) = -\int_{\Omega^e} \frac{\partial \boldsymbol{B}_o^T}{\partial \alpha_i} \underline{\boldsymbol{S}} |\boldsymbol{J}| + \boldsymbol{B}_o^T \frac{\partial \underline{\boldsymbol{S}}}{\partial \alpha_i} |\boldsymbol{J}| + \boldsymbol{B}_o^T \underline{\boldsymbol{S}} \frac{\partial |\boldsymbol{J}|}{\partial \alpha_i} d\Omega. \tag{6.19}$$

The partial derivative of the jacobian has already been calculated in (6.12) so there are only two new terms to consider. From (4.19) we get

$$\frac{\partial \boldsymbol{B}_o}{\partial \alpha_i} = \frac{\partial \boldsymbol{B}_o^l}{\partial \alpha_i} + \frac{\partial \boldsymbol{A}(\boldsymbol{d})}{\partial \alpha_i} \boldsymbol{H}_o + \boldsymbol{A}(\boldsymbol{d}) \frac{\partial \boldsymbol{H}_o}{\partial \alpha_i} \tag{6.20}$$

with the terms in $\frac{\partial \boldsymbol{B}_o^l}{\partial \alpha_i}$ and $\frac{\partial \boldsymbol{H}_o}{\partial \alpha_i}$ being calculated in the same way as in the linear case from (6.10) , and the terms in $\frac{\partial \boldsymbol{A}(\boldsymbol{d})}{\partial \alpha_i}$ being calculated from (4.22) as

$$\frac{\partial \boldsymbol{H}_o}{\partial \alpha_i} \boldsymbol{d} = \begin{bmatrix} \frac{\partial^2 d_x}{\partial x \partial \alpha_i} & \frac{\partial^2 d_x}{\partial y \partial \alpha_i} & \frac{\partial^2 d_y}{\partial x \partial \alpha_i} & \frac{\partial^2 d_y}{\partial y \partial \alpha_i} \end{bmatrix}^T . \tag{6.21}$$

From (3.8) and (4.19) we have

$$\underline{\boldsymbol{S}} = \boldsymbol{D} \underline{\boldsymbol{E}} = \boldsymbol{D} \left(\boldsymbol{B}_o^l + \frac{1}{2} \boldsymbol{A}(\boldsymbol{d}) \boldsymbol{H}_o \right) \boldsymbol{d} \tag{6.22}$$

which gives

$$\frac{\partial \underline{\boldsymbol{S}}}{\partial \alpha_i} = \boldsymbol{D} \left(\frac{\partial \boldsymbol{B}_o^l}{\partial \alpha_i} + \frac{1}{2} \left(\frac{\partial \boldsymbol{A}(\boldsymbol{d})}{\partial \alpha_i} \boldsymbol{H}_o + \boldsymbol{A}(\boldsymbol{d}) \frac{\partial \boldsymbol{H}_o}{\partial \alpha_i} \right) \right) \boldsymbol{d} \tag{6.23}$$

where all the individual terms have already been derived. The sensitivity of the objective function can then be calculated, and the sensitivity of the volume constraint is unchanged from the linear case.

7 Pseudo-contact formulation

A true formulation involving calculating contact forces between two IGA-objects is outside the scope of this thesis, but a simple formulation using pseudo-springs will be implemented to demonstrate how using IGA can be beneficial when dealing with contact problems. The method is an adaptation of the method developed for FEA by Kang et. al. [15]. It should be noted that this does not constitute a proper contact formulation, but rather tries to emulate one for demonstration purposes.

The geometry used is the same bracket as in figure 6.2, with basis function orders $p = q = 2$, but instead of applying a fixed force at one of the control points, a cylinder of radius 4 m is placed above the structure with its center at $(10, 27)$ and moved downwards until its center is at $(10, 17)$ as shown in figure 7.1.

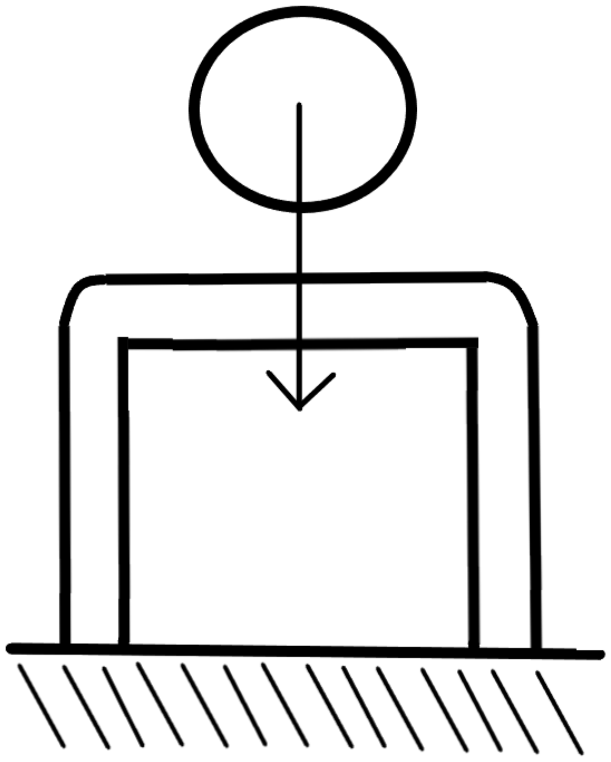

Figure 7.1: The square bracket with the cylinder to be moved downwards to displace the bracket.

The cylinder is not modeled with any kind of FEA or IGA, but is rather a completely analytical object and will therefore not contribute to the stiffness matrix. The top boundary of the bracket is connected to the center of the cylinder through pseudo-springs, each one of which will exert a force in its axial direction when the length of the spring becomes smaller than the radius of the cylinder. The contact points on the bracket are defined for each of the elements along the top boundary in the parent element coordinates

$$(\bar{\xi}, \bar{\eta}) = (-0.8, 1), \quad (-0.4, 1), \quad (0, 1), \quad (0.4, 1), \quad (0.8, 1) \tag{7.1}$$

which corresponds to five equidistant points in each element along the top boundary in the undeformed structure. The location of these points in physical space is then given by (2.14) together with (4.16), and since the location of the center of the cylinder is controlled, the length of each pseudo-spring can easily be calculated. The force of each spring is given by a smoothed step function

$$N = \frac{F_s}{1 + e^{5+m(l-r)}} \tag{7.2}$$

which will quickly go from 0 to F_s as the length of the spring l becomes smaller than the radius of the cylinder r. The parameter m determines how steep the step should be, and should be chosen as high as possible while maintaining stability of the solution. The parameter F_s should be chosen so that penetration of the bracket into the cylinder is small. Choosing m and F_s when dealing with large deformations becomes slightly difficult, as the response of the structure depends on the current deformation. In order to maintain accuracy in this case, smaller values of both m and F_s can be used, while allowing the contact force to be applied several times. That is, the contact force is applied in small steps, solving for equilibrium at each step, and repeating until the structure is no longer in contact with the cylinder. This makes solving very slow, but allows for more accurate results.

Once the spring force has been calculated, it is split into its x- and y- components and divided over the two control points adjacent to the point of contact. This division is done so that the force at each of the control points is inversely proportional to the distance from the control point to the point of contact.

8 Results

The optimization was performed in MATLAB using the method of moving asymptotes (MMA) [16]. The program was created entirely from scratch, except the actual MMA-solver which is open source code. Convergence is achieved when the norm of the difference in compliance from one iteration to the next falls below the tolerance of $\tau_{opt} = 10^{-3}$. In the Newton-Rhapson iterations for the non-linear cases convergence is achieved when the norm of the residual is below the tolerance of $\tau_{nr} = 10^{-2}$. For the linear elastic results the Young's modulus E is set to 210 GPa, and for the non-linear hyperelastic results to 21 kPa. The choice of Young's modulus here is simply to scale forces and displacements to appropriate sizes, and does not affect the results qualitatively. The Poisson's ratio ν is set to 0.3 for all cases. The applied force F is 10 kN for the cantilever in both cases, 1.6 kN for the square bracket, and 5 kN for the semi-circular bracket. Body forces are neglected in all cases. These parameters are gathered in table 8.1.

Table 8.1: Material parameters and applied force for the different structures.

Structure	E	ν	F	τ_{opt}	τ_{nr}
Cantilever	210 GPa	0.3	10 kN	10^{-3}	-
Square bracket	21 kPa	0.3	1.6 kN	10^{-3}	10^{-2}
semi-circular bracket	21 kPa	0.3	5 kN	10^{-3}	10^{-2}

8.1 Topology optimization of a linear elastic cantilever

The dimensions of the cantilever are set to $L = 30$ m, and $h = 10$ m. The maximum volume is set to 35 % of the completely filled domain, that is $V_{max} = 0.35*30*10\text{m}^3 = 105\text{m}^3$.

The optimized structure is shown in figure 8.1 for both choices of basis function orders, and the evolutionary histories of the objective and constraint functions in figure 8.2. The case $p = q = 2$ converged in 27 iterations to a minimum compliance of $c = 19.8$ Nm, and the case $p = q = 4$ converged in 25 iterations to a minimum compliance of $c = 22.6$ Nm.

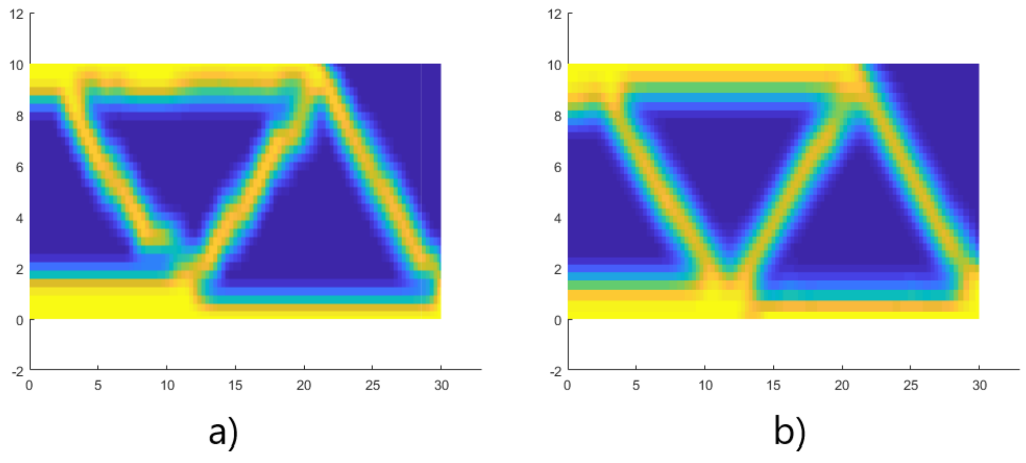

Figure 8.1: The optimized structure using 385 elements for two different choices of order of basis functions. a) $p = q = 2$ b) $p = q = 4$.

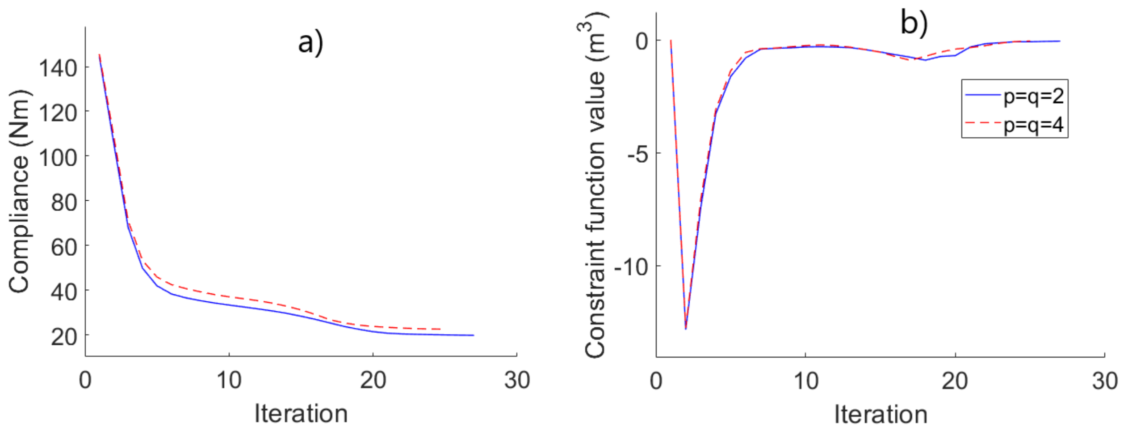

Figure 8.2: Evolutionary histories of a) The objective function b) The constraint function.

8.2 Shape optimization

8.2.1 Linear elastic cantilever

The parameters of (6.2) are set to $c = 1.5$ and $L = 8.5$, and initially all design variables are set to $\alpha_i = 0.5$. The maximum allowed volume V_{max} is set to 70 % of the design domain, i.e. $V_{max} = 0.7 * 30 * 10 = 210$ m^3.

The optimized structure is shown in figure 8.3 with the resulting control points, and the evolutionar histories of the objective and constraint functions in figure 8.4. The program converged in 26 iterations to a minimum compliance of $c = 9.09$ Nm. The results are as expected, and a smooth boundary has been obtained.

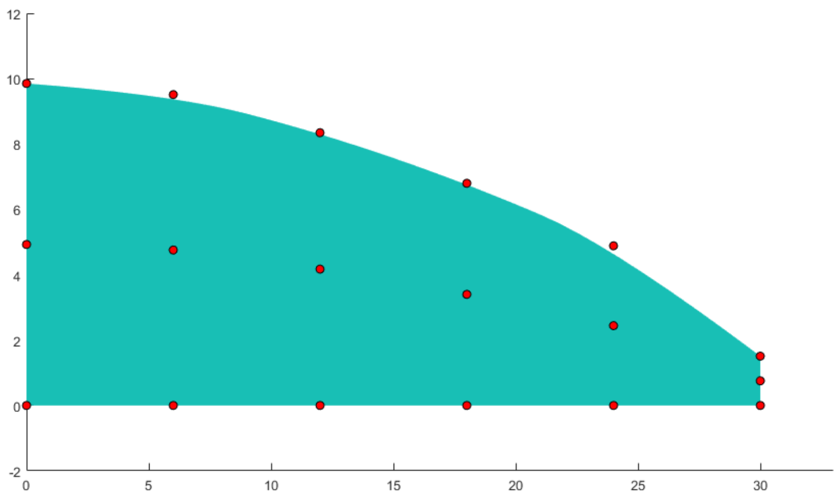

Figure 8.3: The optimized cantilever, with the resulting control points in red.

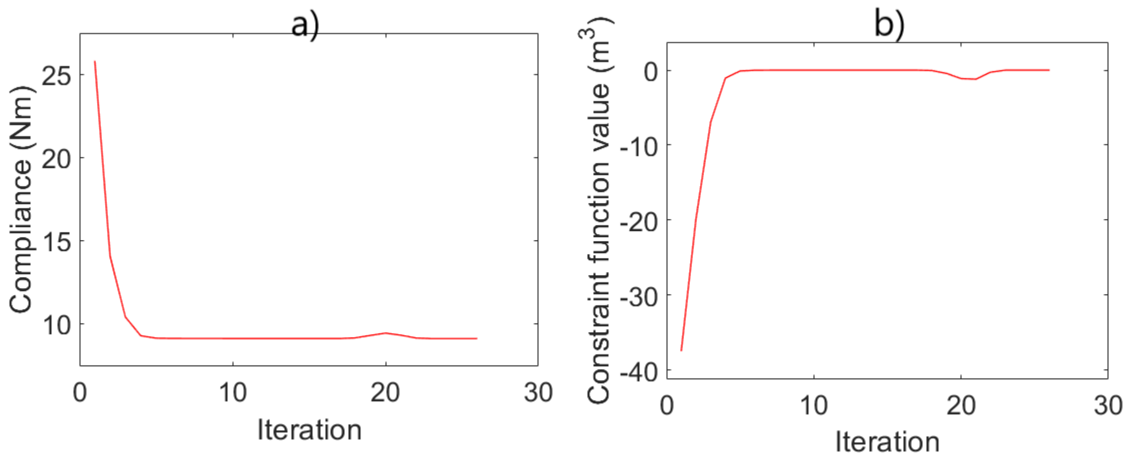

Figure 8.4: Evolutionary histories of a) The objective function b) The constraint function.

8.2.2 Non-linear hyperelastic square bracket

The inside height h_i is fixed at 18 m, the width w is 20 m, and the thickness of the 'legs' T is 2 m. The parameters of (6.2) are set to $c = 19$ m and $L = 4$ m, and initially all design variables are set to $\alpha_i = 0.5$. The maximum allowed volume V_{max} is set to the volume of the initial structure.

The optimized bracket is shown in figure 8.5 for the different choices of order of basis functions. The deformed initial and optimized structures are shown in figures 8.6 and 8.7 respectively, and the evolutionary histories of the objective and constraint functions are shown in figure 8.9 a) and b) respectively.

In figure 8.9 c) a compliance c_p calculated with an alternative method is presented. The compliance in the objective function is calculated as the scalar product of the applied force and the displacement of the control points, but since the control points do not correspond exactly to the position of the boundary, an alternative compliance can be calculated where the physical coordinates of the point of the applied force is calculated and the displacement of this point is multiplied by the applied force. This compliance is slightly more consistent for the different cases, but it should be noted that this is not the objective function that was minimized. To illustrate this the undeformed and deformed optimized structure when $p = 5$ is shown in figure 8.8, with the control point where the force is applied marked in yellow. Clearly the displacement of the control point is greater than that of the actual boundary, motivating why two different compliances can be calculated.

All results are also gathered in table 8.2.

Table 8.2: Iterations for convergence, minimized objective function value, and minimized compliance from physical coordinates for the different choices of basis functions

p	iterations	c (Nm)	c_p (Nm)
2	23	5536	5364
3	74	6266	5597
4	63	6326	5026
5	38	7715	5512

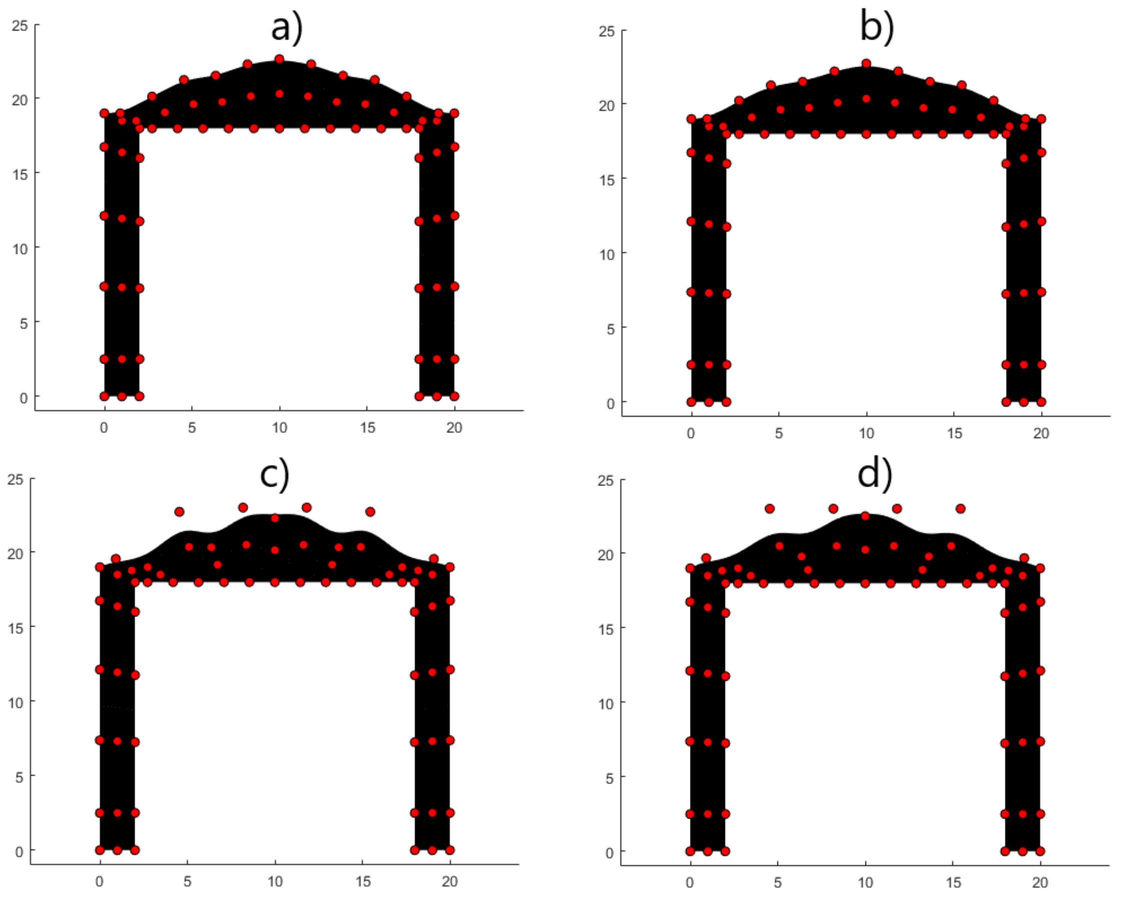

Figure 8.5: The optimized square bracket with resulting control points for a) $p = 2$, b) $p = 3$, c) $p = 4$, d) $p = 5$.

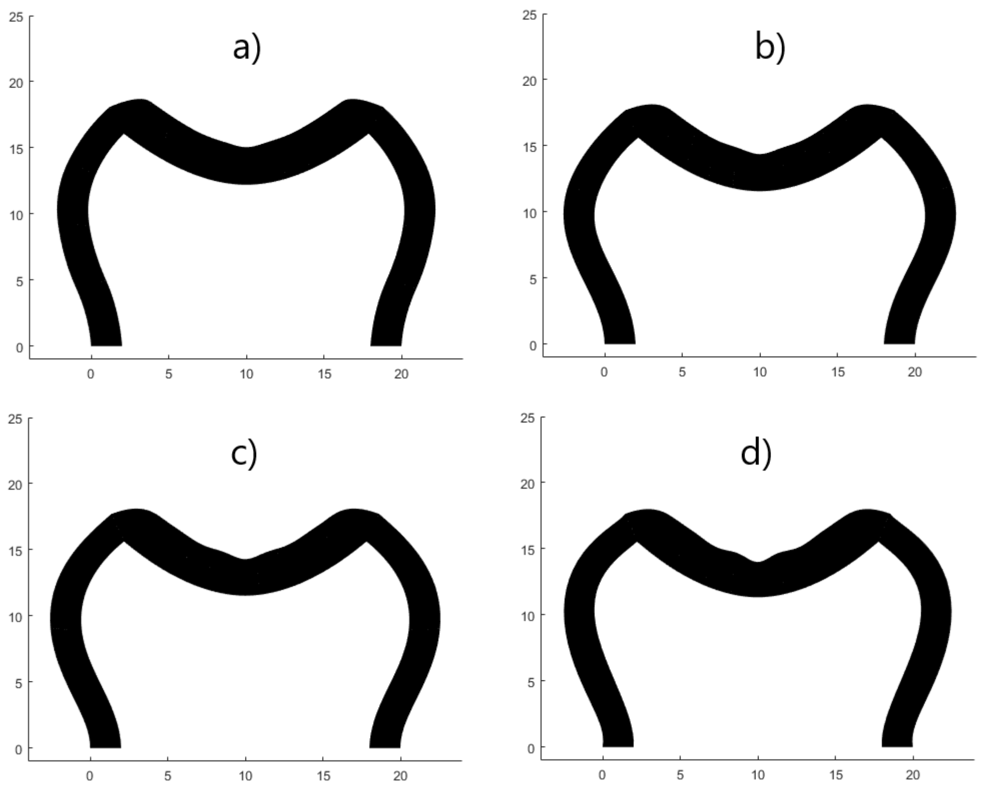

Figure 8.6: The deformed initial square bracket for a) $p = 2$, b) $p = 3$, c) $p = 4$, d) $p = 5$.

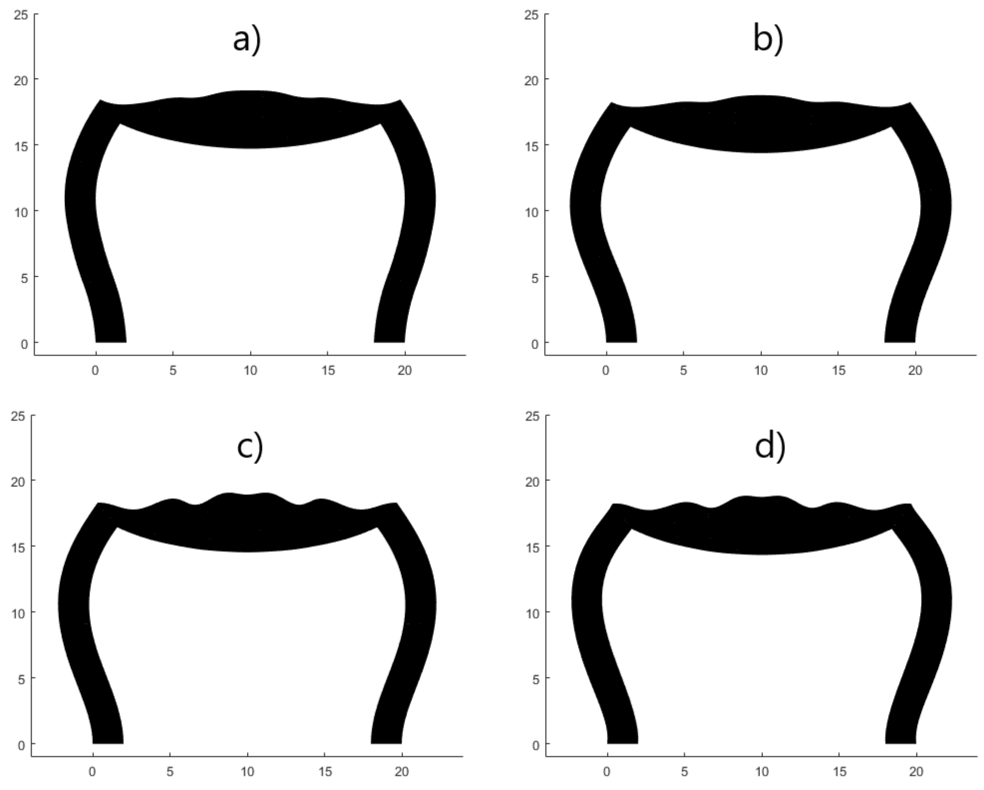

Figure 8.7: The deformed optimized square bracket for a) $p = 2$, b) $p = 3$, c) $p = 4$, d) $p = 5$.

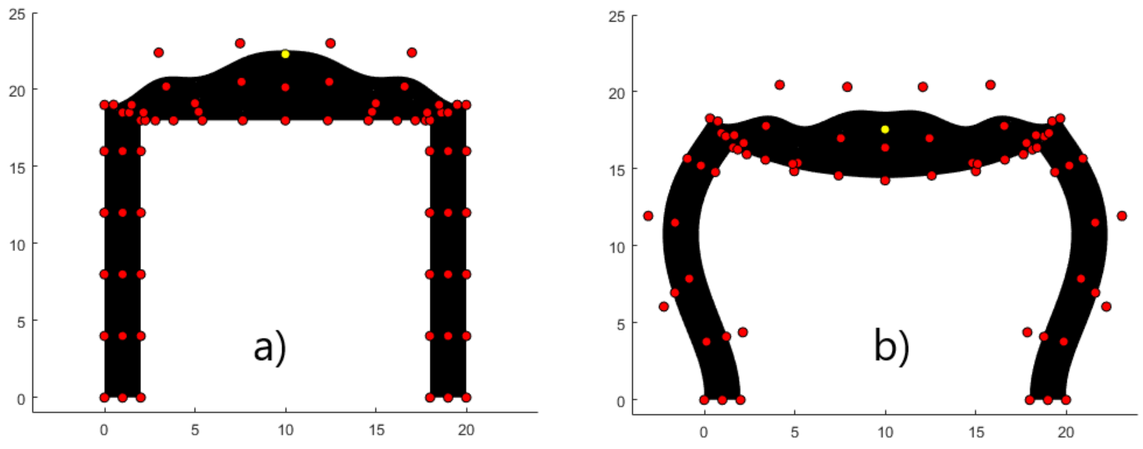

Figure 8.8: The undeformed, a), and deformed, b), optimized structure for $p = 5$, with control points. The control point marked in yellow is where the force is applied, and one can see that the displacement of the control point is greater than that of the physical boundary, motivating that a compliance can be calculated using the displacement of the control point or using the displacement of the corresponding point on the physical boundary.

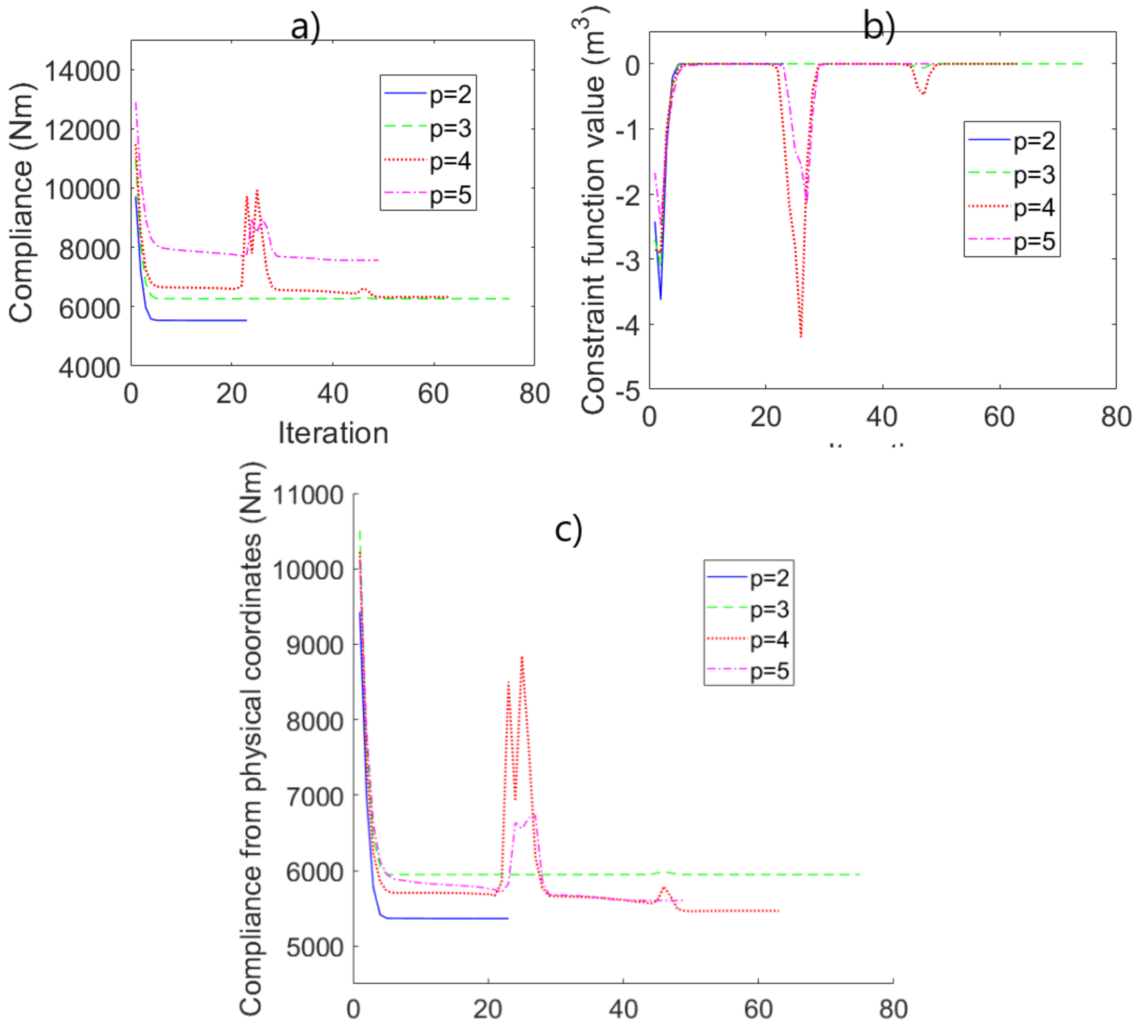

Figure 8.9: Evolutionary histories of a) The objective function, b) The constraint function, c) The compliance calculated from physical coordinates

8.2.3 Non-linear hyperelastic semi-circular bracket

The inner radius is fixed at $r_i = 8$ and the outer radius at the wall is fixed at $r_o = 10$. The freedom of movement is set to $L = 3$ m for all control points in both the x- and y-direction, and all design variables are initially set to $\alpha_i = 0.5$. The constants of the control points c are set so that the initial configuration produces a semi-circle of radius 10. The maximum allowed volume V_{max} is once again set to the volume of the initial structure.

The optimized bracket is shown in figure 8.10 for the different choices of order of basis functions. The deformed initial and optimized structures are shown in figures 8.11 and 8.12 respectively, and the evolutionary histories of the objective and constraint functions are shown in figure 8.13 a) and b) respectively.

Table 8.3: Iterations for convergence, and minimized objective function value for the different choices of basis functions.

p	iterations	c (Nm)
2	49	8577
3	15	8917
4	71	8830
5	29	10157

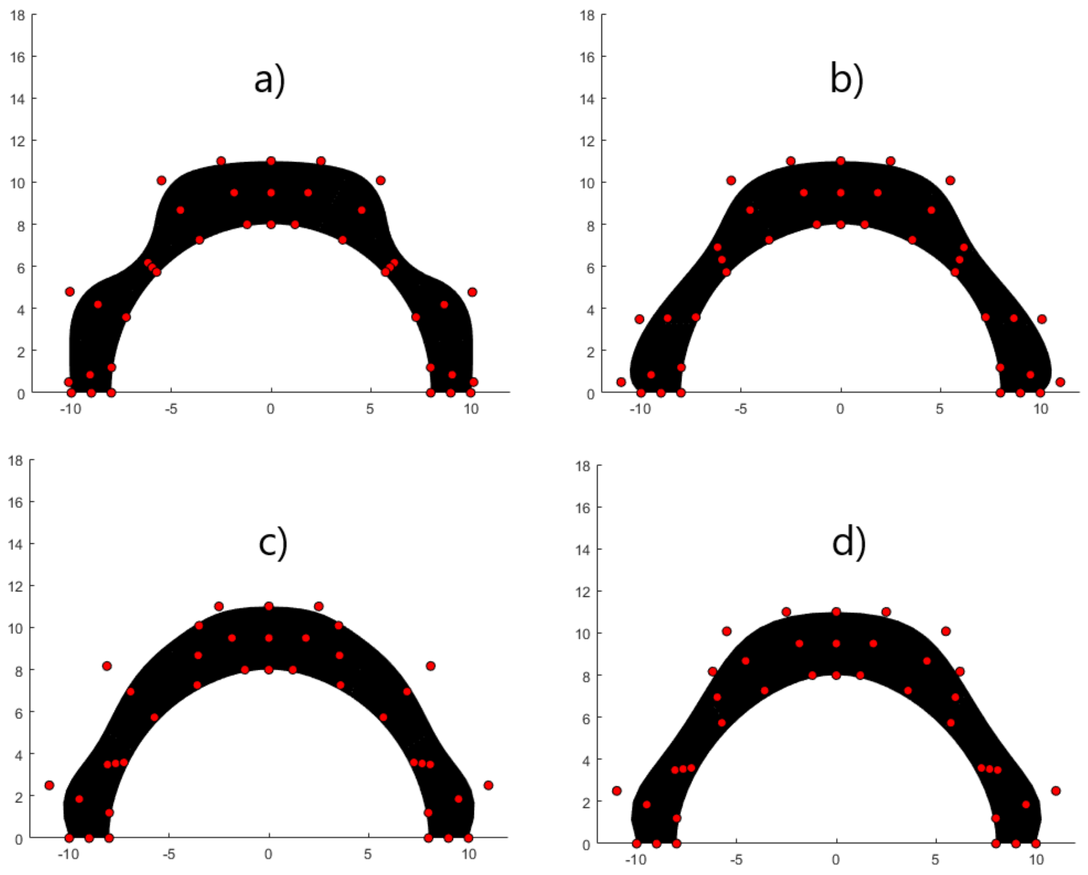

Figure 8.10: The optimized semi-circular bracket with resulting control points for a) $p = 2$,
b) $p = 3$, c) $p = 4$, d) $p = 5$.

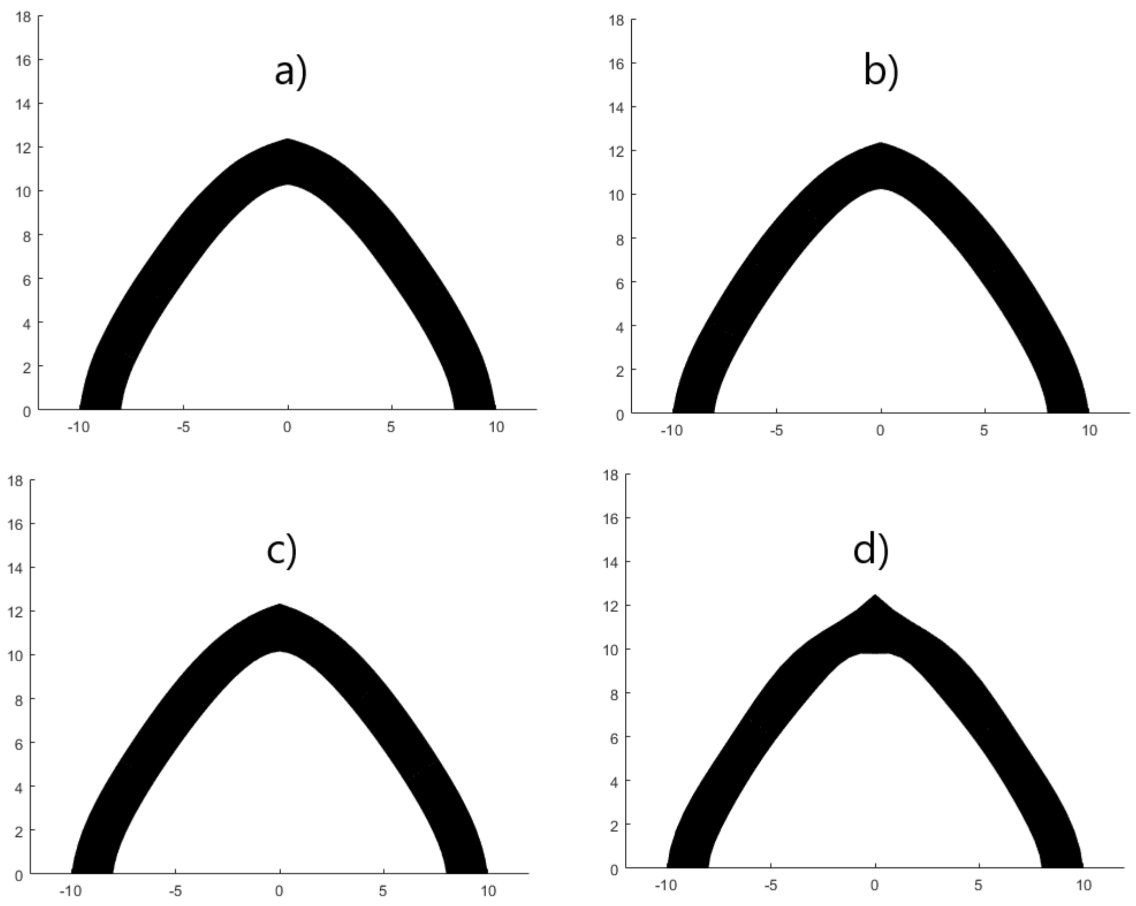

Figure 8.11: The deformed initial semi-circular bracket for a) $p = 2$, b) $p = 3$, c) $p = 4$,
d) $p = 5$.

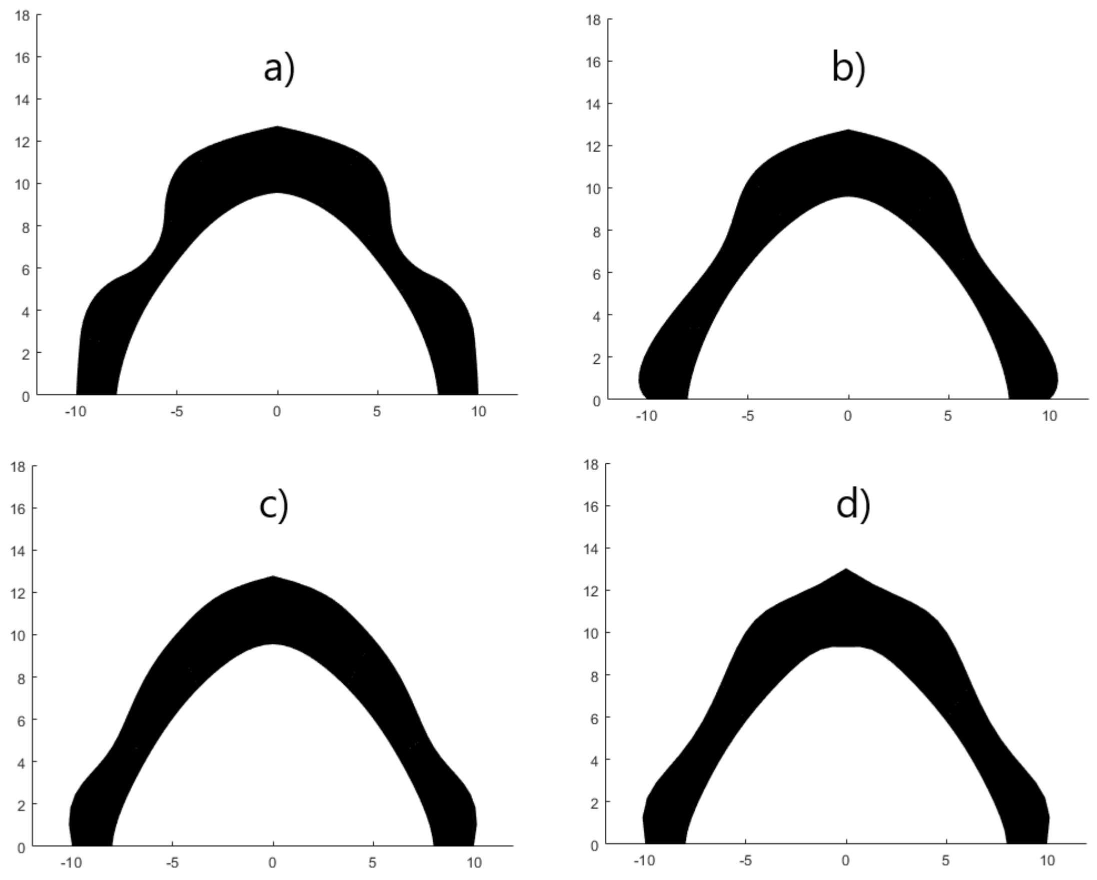

Figure 8.12: The deformed optimized semi-circular bracket for a) $p = 2$, b) $p = 3$, c) $p = 4$,
d) $p = 5$.

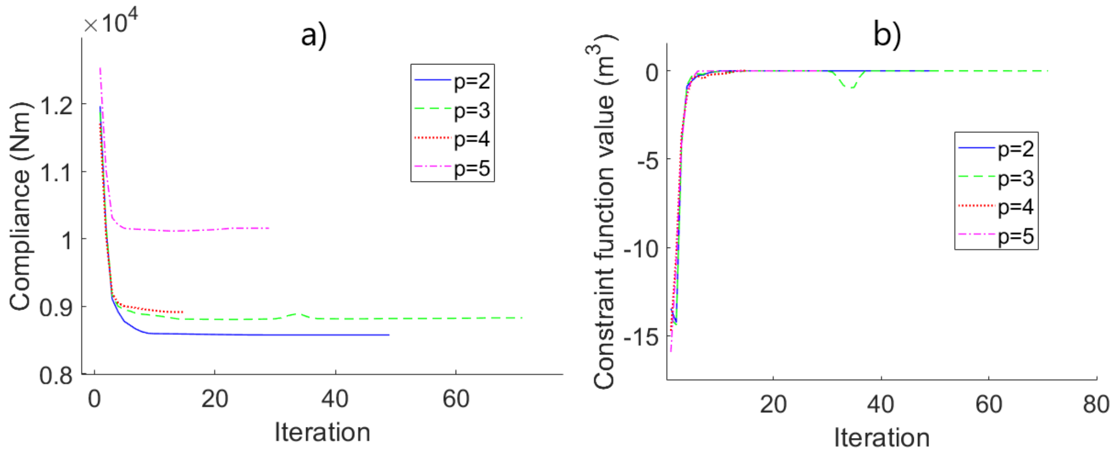

Figure 8.13: Evolutionary histories of a) The objective function, b) The constraint function.

8.2.4 Non-linear hyperelastic square bracket with contact

The geometry and volume constraint is exactly the same as before, and $p = q = 2$. The center of the cylinder is moved down from $(10, 27)$ to $(10, 17)$ in 0.4 m increments. Figure 8.14 shows the deformed initial structure, the optimized structure, the deformed optimized structure, and the evolutionary history of the objective function. The compliance is still chosen as objective function, but because the displacement of the circle is fixed the compliance is maximized rather than minimized since this corresponds to a greater force being required to obtain the same displacement.

The program converged after 27 iterations, and the maximized compliance was $c = 23855$ Nm.

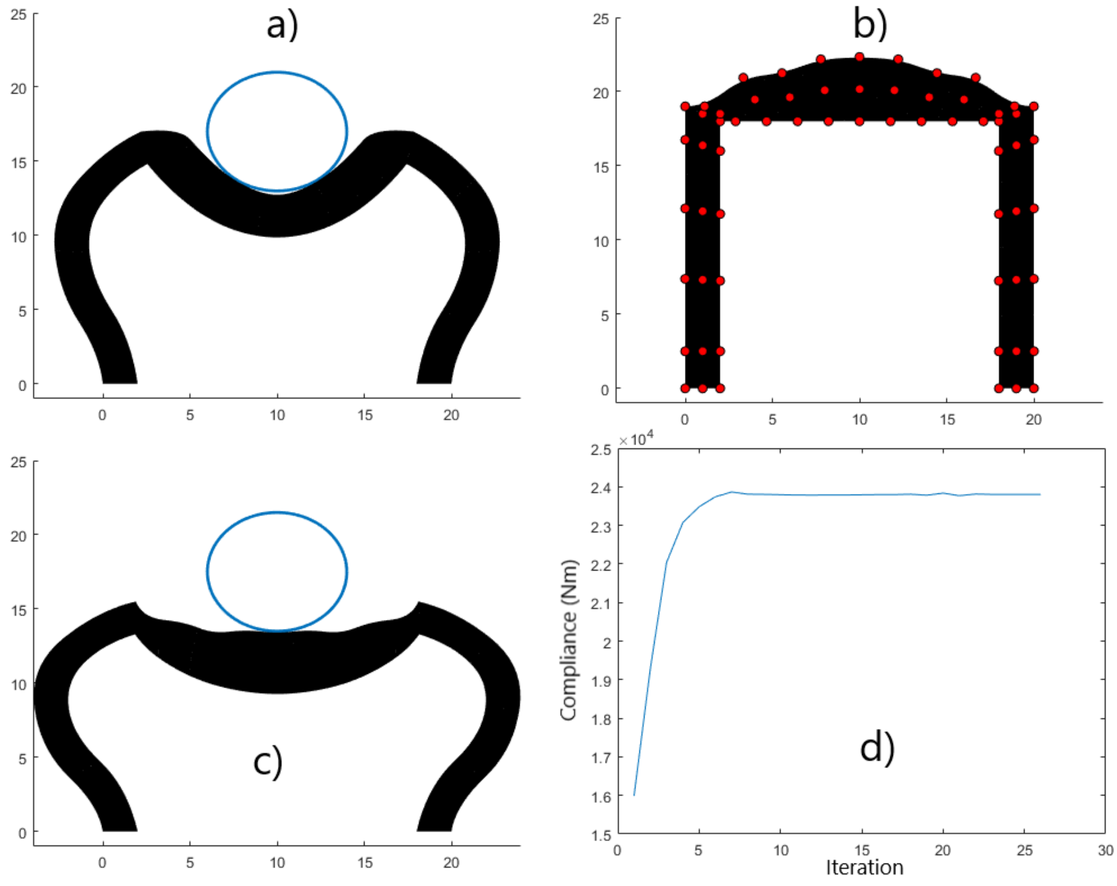

Figure 8.14: a) The deformed initial bracket, b) The optimized bracket, c) The deformed optimized bracket, d) The evolutionary history of the objective function.

9 Discussions and conclusions

The cantilevers topology was optimized successfully, reaching a design that is a familiar results in structural optimization. The design achieved using higher order basis functions provides a smoother optimized structure, which is to be expected as this results in more basis functions having support in each element which works to smooth out the density distribution. This does however slow down the optimization as the basis functions are calculated recursively from lower order basis functions. No filtering techniques need to be applied, as the continuous density representation works as a filter in and of itself and provides natural designs as long as the resolution is high enough.

To speed up the process it would be possible to pre-calculate and store the values of the basis functions in all evaluation points, but this requires quite a bit of storage and the storage needed grows with the order of the basis functions as well. The requirement of a resolution which is comparable to that of classic FEA becomes a considerable drawback when using IGA as calculating the stiffness matrix, constraint function values, and sensitivities, are all more demanding processes.

The shape optimization of the same cantilever was also carried out successfully and while this is a very simple problem it demonstrates the strength of being able to represent smooth geometries with very few degrees of freedom.

The results are equally promising when considering large deformations, as both the square and semi-circular bracket converge nicely to fairly consistent designs. The exception being the sudden sharp increases in compliance and decreases in constraint function value that happen for $p = 4$ and $p = 5$ when optimizing the square bracket. These are however due to numerical errors in the optimization process, combined with the fact that this particular geometry is very sensitive to asymmetric loading. At certain points during the optimization, numerical inaccuracies would result in slight asymmetries appearing in the structure. These would increase, while still remaining small, until the response of the structure became asymmetric to the point that it would bend and snap. This seems to occur more easily with higher order basis functions, as this seems to generally lead to the structures being more compliant as can also be seen in the increase of the minimized compliance as the basis functions order increases. The program was however able to correct this and would always quickly return to a stable position.

The semi-circular bracket shows none of these issues, most likely due to the force being tensile rather than compressive as this will not cause any kind of snapping. The deformed structures, both initial and optimized, when $p = 5$ do however show a different problem: the very sharp point right where the force is applied. This highlights exactly the issue of continuity when using NURBS, as mentioned at the end of section 2.3. The boundary is actually C^0-continuous at the point where the force is applied, but this is not apparent when looking at the initial geometry. This sharp point is actually visible for all choices of basis order function, but it becomes most prominent when the basis order increases as this, once again, seems to result in the structures being more compliant. This problem could be solved by setting up the geometry in a slightly different way, so that the force is not applied right at the point of discontinuity, but this does require the discontinuous point to be moved somewhere else which could also cause issues. A more realistic solution is probably to increase the resolution of the mesh, which should serve to smooth out the boundary.

The number of iterations required for convergence is seemingly very unpredictable, as it varies significantly for the different choices of basis function order. It is also not consistent when comparing the results from the two different brackets, as for the square bracket $p = 2$ is by far the fastest and $p = 4$ is the second slowest, and for the semi-circular bracket this is reversed. It should once again be noted that using higher order basis functions does significantly slow down the solving due to the increased computational cost of calculating basis function values. Unlike in the topology optimization, this could realistically be avoided by pre-calculating and storing shape function values, as the system is generally very small. For the square bracket the dimensions of the stiffness matrix is only 126x126! To achieve similar results with the classic FEM would require substantially larger systems.

The square bracket with the simplified contact formulation results in an optimized structure that is quite similar to that of figure 8.5a) as one would expect. The top of the structure is slightly flatter as a result of the contact forces being a distributed force rather than a point force. While the model used is overly simplistic, it still highlights the ability of the bracket to follow the shape of an object in contact due to the nature of the geometry representation. Furthermore it shows that the use of IGA allows one to evaluate contacts in any point along the boundary very easily, which means that a more sophisticated method could handle contacts over a continuous surface and does not necessarily even require having predetermined points for contact evaluation.

Overall the results are promising, though IGA certainly presents its own problems that need to be tackled. Two main benefits moving forward will most likely be the ability to exactly represent geometries for applications where high levels of precision is needed, as well as the ability to keep system matrices small which could lead to substantially decreased computational costs.

Appendix A

Nomenclature for NURBS and B-splines used.

A	Global shape function number
a	Local shape function number
$\boldsymbol{B}_{ij}$	Control points
n, m	Number of basis function is the ξ- and η-directions
n_i, n_j	Index space coordinates
$N_{i,p}(\xi)$, $M_{j,q}(\eta)$	B-spline basis functions
m_i	Multiplicity of the ith knot
p, q	Polynomial degree of basis functions in ξ- and η-directions
$R_{ij}^{pq}(\xi, \eta)$	NURBS basis functions
$\boldsymbol{T}^p$	Boehm's algorithm matrix for calculating control points
w_{ij}	Control point weights
W	Weighting function
x, y, z	Physical coordinates
Ξ, H	Knot vectors
ξ_i, η_j	Knots
ξ, η	Parametric coordinates
$\bar{\xi}$, $\bar{\eta}$	Parent element coordinates

Appendix B

A system for numbering shape functions and degrees of freedom is introduced, adopting the convention that is used in [12]. Defining two knot vectors $\Xi = [0, 0, 0, 0.5, 1, 1, 1]$ and $H = [0, 0, 0, 1, 1, 1]$, the parameter space representation of these are shown in figure 9.1a), and the *index space* representation in b). In the index space representation all knots are at equal distance from each other, and no consideration is taken for repeated knot values. Using basis functions of order two in both directions result in the two-dimensional shape functions having support in a 3x3 square in the index space, two examples of this are shown in figure 9.1b). A number system for the shape functions is introduced as: the global shape function number A is 1 for the shape function that has its lower left corner at index space coordinates (ξ_1, η_1), and then increases by one for each step to the right. Once each shape function in the row has been numbered, the numbering is continued by moving back to the leftmost part of the grid and moving up one row. This means that in figure 9.1b) the shape function marked in blue has $A = 1$, and the one marked in red has $A = 11$. Combining all global shape function numbers with the corresponding index space coordinates of the lower left corner of the shape function results in the INC matrix, which is shown for the current example in table 9.1.

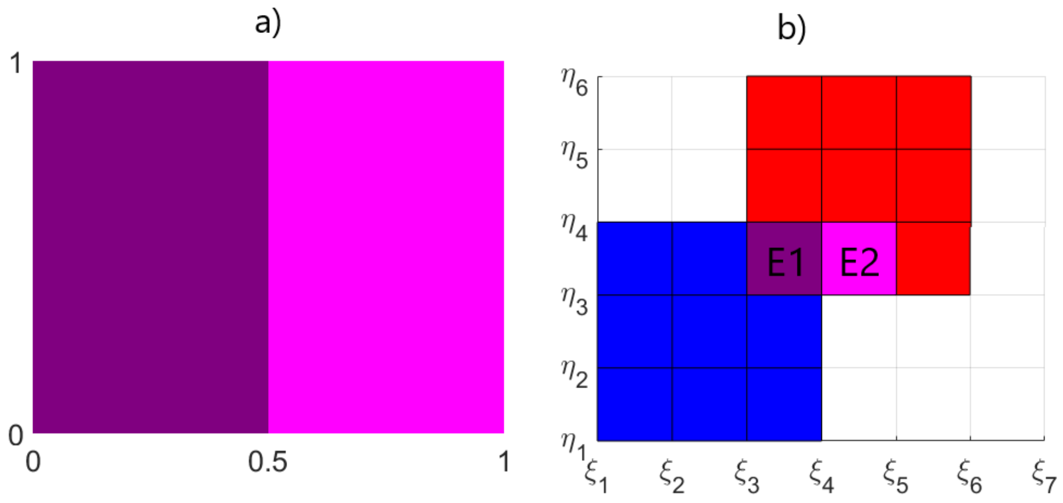

Figure 9.1: The two resulting elements from knot vectors $\Xi = [0, 0, 0, 0.5, 1, 1, 1]$ and $H = [0, 0, 0, 1, 1, 1]$. a) The parameter space representation. b) The index space representation, with two of the basis functions, $R_{1,1}$ (blue) and $R_{3,3}$ (red), and elements marked with E1 and E2.

Next, looking at the individual elements formed from the two knot vectors, which are marked with E1 and E2 in figure 9.1, we can define local shape function numbers a for each element. The numbering is such that a=1 for the shape function that has its lower left corner at the elements lower left corner, and then a increases by one for

A	1	2	3	4	5	6	7	8	9	10	11	12
n_i	1	2	3	4	1	2	3	4	1	2	3	4
n_j	1	1	1	1	2	2	2	2	3	3	3	3

Table 9.1: The INC matrix, that connects a global shape function number A to the corresponding coordinates in index space n_i and n_j.

a	1	2	3	4	5	6	7	8	9
A_1	11	10	9	7	6	5	3	2	1
A_2	12	11	10	8	7	6	4	3	2

Table 9.2: The IEN matrix, that connects the local shape function numbers a of an element i to its corresponding global shape function numbers A_i.

each step to the left, and now moving down a row once the end of the row is reached. Essentially the same as for the global shape function but in reverse. Connecting the local shape function numbers to the global ones for each element results in the IEN matrix, which is shown in table 9.2.

Since IGA does not use nodes in the same way as classic FEA, the degrees of freedom are not determined by a node number, but rather each shape function takes the place of a node in the assembly of the system. For the two-dimensional example here this would mean that each shape function contributes two degrees of freedom, in the x- and y-direction respectively. A numbering of the degrees of freedom is necessary and this is done by simply letting the shape function with A=1 be associated with degrees of freedom 1 and 2, A=2 gives degrees of freedom 3 and 4, and so forth. However, certain degrees of freedom will be fixed by Dirichlet boundary conditions and do not need to be part of the solving process. These degrees of freedom should be numbered 0 and the remaining numbers should be truncated so as to not leave any gaps in the numbering. If for example the left boundary in figure 9.1 is fixed in both x- and y-direction, that means that the shape functions with support at that boundary should have their corresponding degree of freedom numbers set to 0. Connecting global shape function numbers with degrees of freedom results in the ID matrix, which is shown in table 9.3. It is important to notice here that the degrees of freedom are still with respect to physical coordinates, not parametric coordinates, and so one must be aware of the mapping from physical space to parameter space so that the right global shape functions are used.

The final connectivity matrix to be introduced is the LM matrix, which is really a combination of the ID and IEN matrices. The LM matrix simply connects local shape functions numbers for each element and a direction of freedom to the corresponding degree of freedom number. The LM matrix for the example is shown in table 9.4.

A	1	2	3	4	5	6	7	8	9	10	11	12
dof$_x$	0	1	3	5	0	7	9	11	0	13	15	17
dof$_y$	0	2	4	6	0	8	10	12	0	14	16	18

Table 9.3: The ID matrix, which connects a global shape function number A to its corresponding degrees of freedom dof$_x$ and dof$_y$ in the x- and y-direction respectively.

a$_i$	1_x	1_y	2_x	2_y	3_x	3_y	4_x	4_y	5_x	5_y	6_x	6_y	7_x	7_y	8_x	8_y	9_x	9_y
dof$_1$	15	16	13	14	0	0	9	10	7	8	0	0	3	4	1	2	0	0
dof$_2$	17	18	15	16	13	14	11	12	9	10	7	8	5	6	3	4	1	2

Table 9.4: The LM matrix, which connects the local shape function numbers a and a direction x or y of an element i to its corresponding degrees of freedom dof$_i$

References

[1] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement. *Comput. Methods Appl. Mech. Engrg.*, 194:4135–4195, 2005.

[2] Wolfgang A. Wall, Moritz A. Frenzel, and Christian Cyron. Isogeometric structural shape optimization. *Computer Methods in Applied Mechanics and Engineering*, 197(33):2976–2988, 2008.

[3] Kenneth James Versprille. *Computer-Aided Design Applications of the Rational b-Spline Approximation Form.* PhD thesis, USA, 1975. AAI7607690.

[4] W. Tiller. Rational b-splines for curve and surface representation. *IEEE Computer Graphics and Applications, Computer Graphics and Applications, IEEE, IEEE Comput. Grap. Appl*, 3(6):61 – 69, 1983.

[5] Leslie Piegl and Wayne Tiller. Curve and surface constructions using rational b-splines. *Computer-Aided Design*, 19(9):485 – 498, 1987.

[6] Ryan Alberdi, Guodong Zhang, and Kapil Khandelwal. An isogeometric approach for analysis of phononic crystals and elastic metamaterials with complex geometries. *Computational Mechanics*, 62(3):285 – 307, 2018.

[7] M. G. COX. The Numerical Evaluation of B-Splines*. *IMA Journal of Applied Mathematics*, 10(2):134–149, 10 1972.

[8] Carl de Boor. On calculating with b-splines. *Journal of Approximation Theory*, 6(1):50 – 62, 1972.

[9] L. Piegl and W. Tiller. *The NURBS book (2nd ed.).* Springer, 1997.

[10] Wolfgang Boehm. Inserting new knots into b-spline curves. *Computer-Aided Design*, 12(4):199 – 201, 1980.

[11] G. Farin. From conics to nurbs: A tutorial and survey. *IEEE Computer Graphics and Applications, Computer Graphics and Applications, IEEE, IEEE Comput. Grap. Appl*, 12(5):78 – 86, 1992.

[12] J. Austin Cottrell, T.J.R. Hughes, and Y. Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA.* John Wiley & Sons, 2009.

[13] M.P. Bendsøe. Optimal shape design as a material distribution problem. *Structural Optimization*, 1(4):193–202, 1989.

[14] J. Haslinger and R.A.E Mäkinen. *Introduction to shape Optimization-Theory, Approximation, and Computation.* Siam, Philadelphia, 2003.

[15] Yangjun Luo, Ming Li, and Zhan Kang. Topology optimization of hyperelastic structures with frictionless contact supports. *International Journal of Solids and Structures*, 81:373–382, 2016.

[16] K Svanberg. The method of moving asymptotes—a new method for structural optimization. *International journal for numerical methods in engineering*, 24(2):359–373, 1987.