Optimal Shape of Axisymmetric Solids Using NURBS and Automatic Differentiation

R. V. Linn, L. F. R. Espath, A. M. Awruch
Graduate Program in Civil Engineering (PPGEC), Federal University of Rio Grande do Sul (UFRGS)

Abstract

In structural mechanics, axisymmetric structures have a wide range of applications such as components of submarines and aircraft fuselages, rockets, metallic silos, pressure vessels, spherical domes, cooling towers, among others. As the stiffness of these structures depend on their geometry and material properties, shape optimization is necessary in order to obtain improvement in their behaviour for fixed materials. Axisymmetric shape, loads and boundary conditions of these structures also lead to two-dimensional simplification in the analysis of the problem. Thus, shape optimization of the three-dimensional solid geometry can be performed thought shape optimization of the cross section of the structure. Shape optimization deals with the modification of the structure using an optimization algorithm by the structural analysis and its sensitivity analysis. Therefore, different areas must be coupled together to establish a structural optimization system. Both the accuracy and efficiency of the optimization depends on all these areas. A shape optimization of axisymmetric solids of revolution using a geometry description by NURBS surfaces, with a Sequential Quadratic Programming algorithm (SQP) and Automatic Differentiation (AD) for the sensitivity analysis is proposed in this work. As a result, structures with improved structural performance are obtained for static linear elastic structural problems.

Keywords: shape optimization; automatic differentiation; NURBS; axisymmetric solids

1 Introduction

Structural optimization methods can be essentially divided in two aspects: the experience-based and the mathematical-based ones. A good experience has the advantage of being able to globally optimize the structural behavior using only the knowledge of the most dominant structural characteristics. It is a good methodology to optimize well understood mechanical systems where the number of parameters influencing their behavior are small. When the number of parameters describing the structural performance is high or when the problem is nonlinear, this sort of optimization relying on the designer experience for capturing the optimized structure by intuition becomes cumbersome.

Using a mathematical-based method, the complete influence of all structural characteristics can be properly captured. This mathematical-based method is well suitable for a computational approach using optimization algorithms. Once the material of a structure is fixed, it can be said that its structural performance is mainly governed by its geometry (for a given fixed set of boundary conditions). This central idea was certainly used by early designers. It can be somehow said that structural analysis and optimization techniques were conceptually very close to each other. The early structural analysis were performed using geometrical concepts instead of the force-equilibrium concepts employed nowadays [1]. Even that such geometrical-based calculus led to some structural mistakes (also because wrong hypotheses) and that the strength of materials was not well understood in the past (and so materials had not the high performance as in the present days) some improved designs were obtained. The arch-shaped bridge and the variable section domes crossing large spans are examples of experience-based shape optimized structures. Nowadays, the development of modern computers and numerical techniques for structural analysis such as the Finite Element Method and the mathematical programming techniques for optimization has established advanced computational studies on structural optimization.

Many different structural models and approaches have been focused in structural optimization. Some works of Ramm et al. [2] and Bletzinger et al. [3] use a thick and thin shell element formulation,
respectively, with an elastic formulation. Shape optimization of shells to prevent buckling are analyzed by Khosravi et al. [4] and Aubert and Rousselet [5], and the same problem, taking into account imperfections of the structure, were studied by Reitinger and Ramm [6]. Shape optimization of axisymmetric structures was analyzed by Özakçar et al. [7] using an adaptive finite element procedure and by Csonka and Kozák [8] using a higher-order shear deformation theory. AD with NURBS was investigated by the authors of this paper (Espath et al. [9]) to optimize shape of shell structures, obtaining a good performance.

Evaluating of derivatives plays a central role in many scientific and engineering problems. Particularly, on optimization problems, evaluating of derivatives are of great importance once it is one of the most important information used by the optimization algorithm to find an optimum point. The need for obtaining derivatives in optimization problems is directly related to its formulation. Therefore, the choose of the differentiation method should combine both precision and efficiency to be well suitable on optimization procedures. There are many differentiation techniques, and they can be generally classified as: analytical derivation, where an analytical closed solution for the derivative is obtained, it has highly precision and its evaluation is very fast and computationally cheap. However, in most part of problems it is an impracticable technique due to the complexity and nonlinearity of the functionals involved; in the numerical derivation, a numerical method is employed to evaluate the derivatives, being the finite difference method the most popular, it is fast although the choose of the perturbation size is a drawback of the method and it can sometimes lead to inaccurate evaluations; Semi-analytical methods, which lie between both of them, being analytical until some point and numerical until another. The Automatic Differentiation method (AD) is a numerical-computational method for evaluating derivatives but with the advantage of having analytical precision, limited only by the truncation error of the machine. Evaluation of gradients of functionals by AD have generally a much lower computational cost compared to another numerical methods, being well suitable for optimization applications, where those evaluations take massive computation efforts. However, additional computational memory can be needed when using AD.

The optimization algorithm used in the present work is the Sequential Quadratic Programming (SQP). It is a robust algorithm for deterministic non-linear optimization with continuum variables being well suitable for the current shape optimization purpose, where the geometrical constrains adopted are usually highly nonlinear. Also, as the SQP algorithm uses a quasi-Newton approach, only the gradient information is needed instead of a complete Hessian matrix information required by a Newton approach. This feature is well suitable for the coupled use of AD, with faster and more accurate evaluation of derivatives.

Shape optimization means optimization of the geometry. The characteristic variables of the optimization process are, therefore, geometrical parameter that define the geometry of the structure. The number of variables can be dramatically reduced if Computed Aided Geometry Design (CAGD) concepts are used. According to these concepts, free-form geometries can be described through a set of few points called control points. These control points are used as optimization parameters in the present work. Thus, a smooth and precise geometry description is looked for in the shape optimization context. A Non Uniform Rational B-Splines (NURBS) parametrization promotes an easy shape modification through the manipulation of the control points. Furthermore, it can precisely represent complex geometries with an efficient mathematical implementation. Such description is standard to describe and to model curves and surfaces in CAGD. A two-dimensional NURBS description is used for shape description and modification in the present work. More details of NURBS can be found in [14].

The structural analysis is performed using a standard Constant Stress Triangle (CST) finite element for static elastic linear axisymmetric structural problems. Avoiding large mesh distortion and self-interpenetration during the geometry modification, a mesh update scheme is used.

2 Theoretical Aspects

As stated before, many fields have to be coupled together in order to create an optimization system: the description of the geometry, the sensitivity analysis, the optimization algorithm and the structural analysis. The optimization of the geometry is performed by modification of the control points from a NURBS parameterization of the structure. The sensitivity analysis is done using the Automatic
Differentiation (AD) method. The optimization algorithm used here is the Sequential Quadratic Programming (SQP) and the structural analysis is performed using a CST finite element for axisymmetric problems.

2.1 Automatic Differentiation (AD)

The first-order sensitivity analysis is performed using Automatic Differentiation (AD) in the present work. This method is based on the graph theory. Computationally, all evaluations can be described as a trace (sequence) of calculus. This trace contains all the sequence of elemental evaluations performed by the computer in order to achieve the final result (the functional evaluation, for example). This means that to calculate a given functional \( F(x) \), where \( x \in \mathbb{R}^n \), it is possible to rewrite it by breaking the original functional into a functional evaluated by parts:

\[
F(x) = \bigcirc_{i=1}^l v_i \vDash v_{1-n} \circ \ldots \circ v_l
\]  

where these \( v_i \) parts are intermediate calculations, each of them evaluated by means of one or more previous intermediate calculations, making a sequence of operations to evaluate the functional. We also call each of these \( v_i \) steps of evaluations as variables. The initial steps of evaluation are called initial variables, with \( i \) ranging from 1 to \( n \) and the final variable is the functional itself, with \( i \) defined as \( l \). We can group all the variables \( v_i \) in a set \( V \):

\[
V = \{v_{1-n}, \ldots, v_0, v_1, v_2, \ldots, v_{l-1}, v_l\}
\]

Each of these \( v_i \) variables are the vertices of a graph (Fig. 1). The vertices \( v_i \) with \( i \geq 1 \) are obtained by applying an elementar function \( \phi_i \) to a given set or arguments \( v_j \), with \( i > j \). This can be stated as:

\[
v_i = \phi_i(v_j) \quad j \prec i
\]

where the symbol \( j \prec i \) means that a given variable \( v_i \) is directly depended on a set of given variables \( v_j \), with \( i > j \). An elementar function \( \phi_i \) is any function associating one or two variables \( v_j \) to a new variable \( v_i \), for example, the sum, the division and the multiplication of two variables and the exponential and the secant of one variable. It is obvious that any complicated functional can be expressed as a sequence of elementary operations over the independent variables. More generally this means that a functional evaluation can be interpreted as a graph \( G \) containing vertices \( V \) and edges \( E \). The vertices are the variables \( v_i \) whose relations are the edges, given by \( \phi_i(v_j)_j \preceq i \), where \( i > j \). In a more systemic notation it can be stated as (see Fig. 1):

\[
G(V,E): \mathbb{R}^n \rightarrow \mathbb{R}^1
\]

Once a graph (and also an entire algorithm which evaluates a given functional) is stated as above, it can be derived by two distinct ways. The first methodology is simply the application of the chain rule with respect to a given variable \( t \):

\[
\frac{\partial F(x)}{\partial t} = \frac{\partial F(x)}{\partial x} \frac{\partial x}{\partial t}
\]

where the term \( \frac{\partial F(x)}{\partial x} \) is the Jacobian matrix of \( F(x) \). This is called forward mode or tangent mode of differentiation in AD ([10]). The forward mode implies that for each variable of the graph also its corresponding derivative with respect to \( t \) must be evaluated. This way, the total size of this process
is exactly twice as the single evaluation of the functional. Applying the chain rule (5) on the graph (4) and taking into account (1) the following derivatives arise:

\[
\frac{\partial v_i}{\partial t} = \begin{cases} 
\frac{\partial x_i}{\partial t} & i = 1 - n \ldots 0 \\
\sum_{j \prec i} \frac{\partial \phi_i(v_j)}{\partial v_j} \frac{\partial v_j}{\partial t} & i = 1 \ldots l - 1 \\
\frac{\partial F(x)}{\partial v_i} & i = l
\end{cases}
\]

The derivative of the elementary function \( \frac{\partial \phi_i}{\partial v_i} \) is also called tangent function associated to the elementar function \( \phi_i \) and it is obtained by simple differentiation of the original function. For example, the elementar function associated with the multiplication of two variables \( \phi = a \cdot b \) has the tangent function \( \frac{\partial \phi}{\partial v} = \frac{\partial a}{\partial v} \cdot b + a \cdot \frac{\partial b}{\partial v} \) associated.

The second methodology to propagate derivatives is called reverse mode of propagation in AD. In this mode not only a directional derivative but the complete gradient of the functional is evaluated. It can be stated:

\[
\nabla^T F(x) = \frac{\partial F(x)}{\partial x}
\]

The total size of this procedure is no more then twice the original functional evaluation because there is a dependence on the particular graph structure. To apply the gradient on the graph, the propagation of evaluations over the variables \( v_i \) must be performed in a reverse way, from the functional to the independent variables \( x \):

\[
\frac{\partial F(x)}{\partial v_i} = \begin{cases} 
1 & i = l \\
\sum_{j \succ i} \frac{\partial \phi_i(v_j)}{\partial v_j} \frac{\partial v_j}{\partial F(x)} \frac{\partial F(x)}{\partial v_i} & i = l - 1 \ldots 1 \\
\nabla^T F(x) & i = 0 \ldots 1 - n
\end{cases}
\]

In the statement above, the need of evaluation of the contribution of all the successors \( j \succ i \) of a given variable \( v_i \) can be computationally inappropriate because of the need of information not directly available from the elementary functions \( \phi_i \) and its arguments. This way, it it necessary to obtain for each variable \( v_j \) a list of all elementary functions depending of the variables \( v_j \). This approach is also called non-incremental reverse mode. However, the most practical for computational applications is the incremental reverse mode, witch is identically to the non-incremental mode except that the intermediate variables are evaluated as follows:

\[
\frac{\partial F(x)}{\partial v_i} = \frac{\partial F(x)}{\partial v_i} + \frac{\partial \phi_i(v_j)}{\partial v_j} \frac{\partial F(x)}{\partial v_i} \quad \text{for} \quad j \prec i, i = l - 1 \ldots 1
\]
The computational complexity of a evaluation procedure can be measured in terms of the number of computational flops involved and it is directly related with the computational cost (physical time required by the machine to execute the evaluation). The complexity involved in forward mode is [11]:

\[ \frac{\text{flops}\left( \frac{\partial F(x)}{\partial t} \right)}{\text{flops}\left( F(x) \right)} \leq 1 + 3n \] (8)

where \( \text{flops}\left( F(x) \right) \) is the number of flops needed to evaluate the original functional \( F(x) \) and \( \text{flops}\left( \frac{\partial F(x)}{\partial t} \right) \) is the number of flops needed to evaluate the forward propagation of derivatives in AD. For the reverse mode the complexity is:

\[ \frac{\text{flops}\left\{ \nabla^T F(x) \right\}}{\text{flops}\left\{ F(x) \right\}} \leq 5 \] (9)

where \( \text{flops}\left\{ \nabla^T F(x) \right\} \) is the number of flops needed to evaluate the gradient of a scalar functional \( F(x) \) in the reverse mode of AD. It is explicit in the statement above that it holds to be independent of any length of \( x \). This independence with respect to the number of independent variables is high efficient in optimization problems where the number of variables can be very large and the computational costs involved are very high. Although the reverse complexity was stated in the present work only to scalar-valuable functionals, it can be also extended to vectorial-valuable functionals \( F(x) : \mathbb{R}^n \rightarrow \mathbb{R}^m \):

\[ \frac{\text{flops}\left\{ \nabla^T F(x) \right\}}{\text{flops}\left\{ F(x) \right\}} \leq 1 + 4m \] (10)

being the scalar case \( m = 1 \) a particular case of the general complexity relation. It is evident that when the number of required first-order derivatives of the functional is larger compared with the total number of independent variables the reverse mode is preferable instead of the forward mode due to less computational cost involved.

Although the reverse mode is very computationally cheaper compared with other numerical procedures to evaluate derivatives, it has a drawback: the computational memory needed. This occurs due to the requirement to store all the computational graph in the memory and also its derivatives. Special treatment can be performed to optimize the memory usage in this process, with allocation and deallocation of variables and taking into account the Jacobian matrix sparsity [12].

In the present work, automatic differentiation tool TAPENADE AD [13], developed by the INRIA in France, is employed to evaluate the derivatives with reverse automatic differentiation.

### 2.2 Non-Uniform Rational B-Spline (NURBS)

In order to represent a complex surface, a parametric representation is used. NURBS parameterization is well suitable for shape optimization in any physical problem involving curves, surfaces and solids. For a bidimensional representation of structures, a plane surface representation in the form

\[ S(\xi, \eta) = (x(\xi, \eta), y(\xi, \eta)) \quad (\xi, \eta) \in [0, 1] \times [0, 1] \]

is looked for.

The NURBS surface in homogeneous coordinates is defined as

\[ S^w(\xi, \eta) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(\xi) N_{j,q}(\eta) P_{i,j}^w \] (11)

where \((p, q), (n, m), (N_{i,p}(\xi), N_{j,q}(\eta))\) are the degrees, the numbers of basis functions, and the basis functions in \((\xi, \eta)\) directions, respectively. \( P_{i,j}^w = (w_{i,j}x_{i,j}, w_{i,j}y_{i,j}, w_{i,j}) \) are the control points in homogeneous coordinates. Computational algorithms in homogeneous coordinates are very efficient.
The basis functions in recursive form are defined as

\[ N_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1} \\ 0 & \text{otherwise} \end{cases} \]

\[ N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi) \]

over the following knot vectors, in \((\xi, \eta)\) directions

\[ \Xi = \{0, \ldots, 0, \xi_{p+1}, \ldots, \xi_{r-p-1}, 1, \ldots, 1\}, \quad \text{with} \quad r = n + p + 1 \]

\[ \mathcal{H} = \{0, \ldots, 0, \eta_{q+1}, \ldots, \eta_{s-q-1}, 1, \ldots, 1\}, \quad \text{with} \quad s = m + q + 1 \]

In the shape modification, when a control point, a weight and/or a knot position is modified the parametric mesh remains the same; however, the mesh in then Euclidean space assumes a new position according with the new geometry. This mapping may be not the best choice, because the mesh distortion is not controlled, although this problem is somehow minimized with a convenient initial mesh. An additional update scheme to control mesh distortion is also used in the present work and it is explained in the section 3.

The geometry description used in the present work uses a weight \(w = 1\) witch leads the general NURBS description to a particular Bézier description case. In the optimization process, the external control points (lying on the boundary) \(P_{i,j}\) of the geometry are used as optimization variables and the internal ones are defined as dependent of the external control points (this dependence is also the mesh distortion control scheme).

A complete overview of NURBS and Bézier theories can be found in Piegl and Tiller [14].

### 2.3 Numerical Optimization

In the context of the numerical shape optimization, the SQP method is used. The optimization problem is stated as

\[
\min_{\mathbf{x} \in \mathbb{R}^n} \mathcal{F}(\mathbf{x}) \quad \text{subject to} \quad \left\{ \begin{array}{l}
\mathcal{C}_i(\mathbf{x}) = 0, \quad i \in \mathcal{E} \\
\mathcal{C}_i(\mathbf{x}) \geq 0, \quad i \in \mathcal{I}
\end{array} \right.
\]

where \(\mathcal{F}, \mathcal{C}_i\) are defined in \(\mathbb{R}^n\); and \(\mathcal{E}\) and \(\mathcal{I}\) are two finite index sets, \(\mathbf{x}\) are the independent variables, \(\mathcal{F}\) is the objective function, \(\mathcal{C}_i, i \in \mathcal{E}\) are equality constraints and \(\mathcal{C}_i, i \in \mathcal{I}\) are inequality constraints.

The Lagrangian is defined as

\[
\mathcal{L}(\mathbf{x}, \lambda) = \mathcal{F}(\mathbf{x}) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i \mathcal{C}_i(\mathbf{x})
\]

where \(\lambda_i\) are the Lagrange multipliers. If the linear independence constraint qualification holds, then the optimum \((\mathbf{x}^*, \lambda^*)\) must satisfy the Karush-Kuhn-Tucker (KKT) condition, i.e.,

\[
\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}^*, \lambda^*) = 0,
\]

\[
\mathcal{C}_i(\mathbf{x}^*) = 0, \quad \forall i \in \mathcal{E},
\]

\[
\mathcal{C}_i(\mathbf{x}^*) \geq 0, \quad \forall i \in \mathcal{I},
\]

\[
\lambda_i^* \geq 0, \quad \forall i \in \mathcal{I},
\]

\[
\lambda_i^* \mathcal{C}_i(\mathbf{x}^*) = 0, \quad \forall i \in \mathcal{E} \cup \mathcal{I}
\]
The proof and the complete theory related is shown by Nocedal and Wright [15]. The KKT conditions, Eqs. [14] to [18], must be satisfied for a given tolerance range. The numerical approach for the optimization problem using a SQP algorithm starts from a initial point \( x_0 \), and then the SQP algorithm generates a sequence of iterations \( \{x_k\}_{k=0}^{\infty} \) that ends when any progress can be made or when this point is the approximate solution with sufficient accuracy. The SQP algorithm approximates the objective function \( F(x_k) \) by a quadratic form and the restrictions \( C_i \) (of both equality and inequality) by a linear form. Therefore, applying the Newton method to the Lagrangian function with these approximations for the objective function and restrictions, the SQP method arrives to:

\[
\min_{\delta_k} \frac{1}{2} \delta_k^T \nabla_{xx} L(x_k, \lambda_k) \delta_k + \nabla F(x_k)^T \delta_k
\]

subjected to

\[
C_i(x_k) + C_i(x_k)^T \delta_k = 0, \quad i \in E
\]

and

\[
C_i(x_k) + C_i(x_k)^T \delta_k \geq 0, \quad i \in I
\]

(19)

where \( \nabla_{xx} L(x_k, \lambda_k) \) is a positive defined approximation of the Hessian matrix. An algorithm for solving quadratic problems is applied on the problem defined above in order to obtain the search direction \( \delta_k = x_k - x_{k-1} \), being \( \delta_k \) the solution of the quadratic problem and a linear search method is applied in the present work to find a new point \( x_{k+1} \). In the linear search strategy the algorithm chooses a direction \( \delta_k \) and searches through this direction a new position in which the functional has a lower value, starting from the current interaction \( x_k \). The distance over the direction \( \delta_k \) can be approximated by the solution of a unidimensional minimization:

\[
\min_{\alpha_k > 0} f(x_k + \alpha_k \delta_k)
\]

where \( \alpha_k \) is the length of distance through \( \delta_k \). An exact minimization is not needed in this step. While the optimal conditions are not satisfied, new points and new directions with new length are evaluated on each step and the process is repeated. Every new point \( x_{k+1} \) is updated by:

\[
x_{k+1} = x_k + \delta_k
\]

(20)

While the optimum is not reached, the Hessian matrix is updated by the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method. The BFGS method is an algorithm which generates an approximation for the Hessian based on the gradients of the current and previous iteration. This mean that BFGS is a quasi-Newton method, i.e., the Hessian is not evaluated exactly, but approximated by gradients evaluations. These gradients evaluations are performed using AD.

### 2.4 Structural Analysis

The structural analysis is performed using the finite element method for an axisymmetric problems. The Constant Stress Triangle (CST) for linear elastic static analysis, commonly related in the literature, is employed.

### 3 Shape Modifications and Shape Optimization

The coupling of the optimization procedure to shape optimization is presented in this section. The shape modification is performed by moving control points \( P_{i,j} \) belonging to the NURBS description of the geometry. As the problem is axisymmetric and only the cross section is modified, only the control points on the boundary \( \Gamma \) need to be chosen as optimization variables and the internal control points within the domain \( \Omega \) are updated in order to prevent inter-penetration of the control points and also to avoid large mesh distortions. Thus:

\[
x = P_{i,j} \in \Gamma
\]

(21)
The boundary nodes which are optimization variables are allowed to move in just one given direction. An upper and lower bound are imposed over this node movement as an optimization restriction. Also, a minimal distance is imposed as optimization restriction to avoid interpenetration of the nodes. When the new position of these optimization variables nodes are obtained by the optimization algorithm, intermediate points are then updated by linearly interpolation over the distance between the boundary nodes along the moving direction. This process improves the mesh update and is represented in Fig. 2 and Fig. 3. In these figures, the red nodes represent nodes which are optimization variables and the blue ones the other points. Fig. 2 represents the initial condition of the mesh and geometry. After, the two middle optimization nodes (red) on the top and bottom boundaries are moved in one direction. As a result, intermediate points (blue) between points representing the variables (red) are also updated, holding an equal distance along the moved direction. In Fig. 3 it can be seen that this procedure moves the mesh following the change of the geometry.

The chosen objective function to be minimized $F$ is the internal strain energy in linear elastic problems, which is given by

$$F = \int_{\Omega} \sigma : \varepsilon \, d\Omega = U^T K U$$

(22)

where $\sigma$ is the stress tensor, $\varepsilon$ is the strain tensor, $U$ is the displacement vector and $K$ is the stiffness matrix. The optimization problem is performed using relative objective functions values, i.e., referred to an initial value.

This minimization of $F$ is then constrained by geometrical or mechanical functions, such as constant volume restriction, limiting upper and lower bound of the points and the minimum distance between boundary nodes in the given optimized direction. These restrictions may be written as follows:
\[ C = 1 - \frac{V}{V_{ini}} = 0 \]  \hspace{1cm} (24)
\[ C = x_{min} \geq x \geq x_{max} \]  \hspace{1cm} (25)
\[ C = (x_i^a - x_i^b) \in \Gamma - tol \geq 0 \]  \hspace{1cm} (26)

where \( V \) is the current volume, \( V_{ini} \) is the initial volume, \( x_{min} \) is the lower bound, \( x_{max} \) is the upper bound, \( a \) and \( b \) indicate optimization nodes on different boundaries and \( tol \) is the minimum distance adopted. These functions are evaluated by FEM, and the gradients by AD. The complete algorithm of shape optimization can be stated as follows:

1. Perform a structural analysis on initial geometry by FEM,
2. Evaluate the required gradients by reverse AD,
3. Verifies KKT conditions. If convergence is not obtained, find a new search direction and perform unidimensional minimization along this direction in order to obtain a new position \( x_{n+1} \) for the optimization points \( P_{i,j} \),
4. Update internal points of the geometry by linear interpolation of the distance between opposite boundaries in a given direction,
5. Perform a structural analysis on the new geometry by FEM.

The algorithm loops over the second and fifth steps if the convergence is not achieved. When convergence is achieved, the algorithm just performs the final structural analysis (step 5).

\section{4 Numerical Applications}

In order to demonstrate and validate the shape optimization algorithm, some examples are presented in this section. The material properties for all the examples are: Young’s elastic modulus \( E = 3.10 \cdot 10^{10} \), Poisson coefficient \( \nu = 0.2 \) and specific mass \( \gamma = 2.5 \cdot 10^4 \). The values used are in the SI. The objective function to be minimized in all examples is the strain energy and the equality constrain of constant volume is imposed in all examples. Other constrains adopted for each example are properly specified in the description of each case.

\subsection{4.1 Example 1}

The initial geometry of this example is a disk shape. An initial height of \( h = 1 \) and radius \( R = 5 \) are used. The disk is simply supported on its external point and an uniform distributed load of \( q = 750 \cdot 10^3 \) is applied on the upper boundary (see Fig. 4). No body forces are considered in this example.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{fig4.png}
\caption{Initial geometry and boundary conditions for example 1.}
\end{figure}
The finite element mesh has 2121 nodes and 4000 elements as shown in Fig. 5. The NURBS parametrization is done with 16 points, where 8 points were adopted as optimization points (4 in the upper boundary and 4 in the bottom). Fig. 6 shows the adopted optimization nodes with red color and the other nodes with blue color. The basis function are defined over the knot vectors

$$\Xi = \{0, 0, 0, 0, 1, 1, 1, 1\}$$  \hspace{1cm} (27)

$$\mathcal{H} = \{0, 0, 0, 0, 1, 1, 1, 1\}$$  \hspace{1cm} (28)

The optimization points are allowed to move in the vertical direction bounded by an upper limit of 25 and lower limit of -25. Also, vertical distance between upper and lower boundary nodes are limited to be great or equal to 0.15 (to avoid interpenetration of one point in other one). The initial coordinates of the 16 points are shown in table 1.

Table 1: Initial coordinates of control points for example 1

<table>
<thead>
<tr>
<th>P</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0,0)</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>(1,0)</td>
<td>1.67</td>
<td>0.00</td>
</tr>
<tr>
<td>(2,0)</td>
<td>3.33</td>
<td>0.00</td>
</tr>
<tr>
<td>(3,0)</td>
<td>5.00</td>
<td>0.00</td>
</tr>
<tr>
<td>(0,1)</td>
<td>0.00</td>
<td>0.33</td>
</tr>
<tr>
<td>(1,1)</td>
<td>1.67</td>
<td>0.33</td>
</tr>
<tr>
<td>(2,1)</td>
<td>3.33</td>
<td>0.33</td>
</tr>
<tr>
<td>(3,1)</td>
<td>5.00</td>
<td>0.33</td>
</tr>
<tr>
<td>(0,2)</td>
<td>0.00</td>
<td>0.67</td>
</tr>
<tr>
<td>(1,2)</td>
<td>1.67</td>
<td>0.67</td>
</tr>
<tr>
<td>(2,2)</td>
<td>3.33</td>
<td>0.67</td>
</tr>
<tr>
<td>(3,2)</td>
<td>5.00</td>
<td>0.67</td>
</tr>
<tr>
<td>(0,3)</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>(1,3)</td>
<td>1.67</td>
<td>1.00</td>
</tr>
<tr>
<td>(2,3)</td>
<td>3.33</td>
<td>1.00</td>
</tr>
<tr>
<td>(3,3)</td>
<td>5.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>

The imposed constrain of constant volume forces the geometry to search the best mass distribution in order to minimize the strain energy with fixed boundary conditions and load, avoiding a simple mass
increase in order to obtain a lower strain energy. Fig. 7 shows the evolution of the geometry along the iterations until an optimization shape is obtained.

Figure 7: Sequence of shapes corresponding to different number of iteration for example 1.

At the 28th iteration, the optimum condition is reached. The final relative objective function obtained $F_{opt}$ is $F_{28}/F_{ini} = 0.114$. A comparison between the displacements and the von Mises stresses in initial and final optimized geometry is shown in Fig. 8.
The decrease of the relative objective function in terms of the iterations is presented in Fig. 9. In this same figure is also presented the final geometry. The final coordinates of the 16 points are shown in table 2.

Table 2: Final coordinates of control points for example 1

<table>
<thead>
<tr>
<th>P</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0,0)</td>
<td>0.00</td>
<td>-14.98</td>
</tr>
<tr>
<td>(1,0)</td>
<td>1.67</td>
<td>-14.45</td>
</tr>
<tr>
<td>(2,0)</td>
<td>3.33</td>
<td>-13.02</td>
</tr>
<tr>
<td>(3,0)</td>
<td>5.00</td>
<td>-9.26</td>
</tr>
<tr>
<td>(0,1)</td>
<td>0.00</td>
<td>-14.93</td>
</tr>
<tr>
<td>(1,1)</td>
<td>1.67</td>
<td>-14.40</td>
</tr>
<tr>
<td>(2,1)</td>
<td>3.33</td>
<td>-12.97</td>
</tr>
<tr>
<td>(3,1)</td>
<td>5.00</td>
<td>-8.50</td>
</tr>
<tr>
<td>(0,2)</td>
<td>0.00</td>
<td>-14.88</td>
</tr>
<tr>
<td>(1,2)</td>
<td>1.67</td>
<td>-14.35</td>
</tr>
<tr>
<td>(2,2)</td>
<td>3.33</td>
<td>-12.92</td>
</tr>
<tr>
<td>(3,2)</td>
<td>5.00</td>
<td>-7.74</td>
</tr>
<tr>
<td>(0,3)</td>
<td>0.00</td>
<td>-14.83</td>
</tr>
<tr>
<td>(1,3)</td>
<td>1.67</td>
<td>-14.30</td>
</tr>
<tr>
<td>(2,3)</td>
<td>3.33</td>
<td>-12.87</td>
</tr>
<tr>
<td>(3,3)</td>
<td>5.00</td>
<td>-6.98</td>
</tr>
</tbody>
</table>
It can be noticed from Fig. 9 that the initial iterations imply in the greatest reduction of the objective function. The final iterations do not reduce the objective function so much, but they play a role in fixing the curvature, as can be observed in Fig. 7. From the comparison between displacement and von Mises stresses it can be noticed the significant reduction of both in the optimized geometry. Along the optimization iterations, the original high flexural and shear stresses are progressively reduced, while the membrane stresses are increased. The final geometry configuration tends to approach a parabolic arc shape. It can be noticed that using the optimized shape with an applied load in the inverse direction, i.e. oriented from the bottom to the external surface, the structure will work primary with compression stresses. This is an important result and has been extensively used for many engineering applications (such as the arc bridge). This type of geometry is a well-known established result of minimization of strain energy. For the one-dimensional case, this minimization leads to a catenary shape configurations. Analogous results for shell structures are presented by Ramm and Wall [16]. An experimental work on this area was performed by Isler [17], where the form was found by hanging models and a membrane configuration under compression was also obtained.

4.2 Example 2

The initial geometry of this example is analogous to the example 1 but with a central hole. An initial height of $h = 1.5$, internal radius $R_i = 5$ and external radius $R_e = 10$ were adopted using the same boundary condition of example 1 (simple supported) and an uniform distributed load of $q = 268 \cdot 10^3$ is applied on the upper boundary (see Fig. 10). No body forces are considered in this example.

The finite element mesh has 952 nodes and 1742 elements as shown in Fig. 11. The NURBS parametrization is done with 42 points, where 14 points were adopted as optimization points (7 in the upper boundary and 7 in the bottom). Fig. 12 shows the adopted optimization nodes with red color and the others with blue color. The basis function are defined over the knot vectors:

$$\Xi = \{0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1\}$$  \hspace{1cm} (29)

$$\mathcal{H} = \{0, 0, 0, 0, 0, 0, 1, 11, 1, 1, 1\}$$  \hspace{1cm} (30)
The 14 optimization points are allowed to move in the vertical direction bounded by an upper limit of 7.5 and a lower limit of -7.5. Also, vertical distance between upper and lower boundary nodes are limited to be great or equal to 0.15. Fig. 13 shows the geometry optimization evolution along the iterations. At the 39th iteration, the optimum condition is reached. The final relative objective function obtained $F_{\text{opt}}$ is $F_{\text{opt}}/F_{\text{ini}} = 0.068$. Fig. 14 shows a comparison between the initial and the final optimized geometry for displacements and von Mises stresses.
Figure 13: Sequence of shapes corresponding to different number of iterations for example 2.
Figure 14: Comparison between displacements and von Mises stresses in the initial and the final optimized geometries for example 2.

The decrease of the relative objective function as a function of the iterations is presented in Fig. 15. In this figure is also presented the final geometry. The initial and final coordinates of the 16 optimization points are shown in table 3.

Figure 15: Relative objective function as function of the iterations for example 2.
Table 3: Initial and final coordinates of optimization control points for example 2

<table>
<thead>
<tr>
<th>P</th>
<th>( x )</th>
<th>( y_i )</th>
<th>( y_f )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0,0)</td>
<td>5.00</td>
<td>0.00</td>
<td>-7.50</td>
</tr>
<tr>
<td>(1,0)</td>
<td>5.83</td>
<td>0.00</td>
<td>-7.50</td>
</tr>
<tr>
<td>(2,0)</td>
<td>6.67</td>
<td>0.00</td>
<td>-7.05</td>
</tr>
<tr>
<td>(3,0)</td>
<td>7.50</td>
<td>0.00</td>
<td>-6.01</td>
</tr>
<tr>
<td>(4,0)</td>
<td>8.33</td>
<td>0.00</td>
<td>7.35</td>
</tr>
<tr>
<td>(5,0)</td>
<td>9.17</td>
<td>0.00</td>
<td>-7.50</td>
</tr>
<tr>
<td>(6,0)</td>
<td>10.00</td>
<td>0.00</td>
<td>-7.50</td>
</tr>
<tr>
<td>(0,5)</td>
<td>5.00</td>
<td>1.50</td>
<td>-7.35</td>
</tr>
<tr>
<td>(1,5)</td>
<td>5.83</td>
<td>1.50</td>
<td>-7.24</td>
</tr>
<tr>
<td>(2,5)</td>
<td>6.67</td>
<td>1.50</td>
<td>-6.90</td>
</tr>
<tr>
<td>(3,5)</td>
<td>7.50</td>
<td>1.50</td>
<td>-5.86</td>
</tr>
<tr>
<td>(4,5)</td>
<td>8.33</td>
<td>1.50</td>
<td>-7.50</td>
</tr>
<tr>
<td>(5,5)</td>
<td>9.17</td>
<td>1.50</td>
<td>-3.83</td>
</tr>
<tr>
<td>(6,5)</td>
<td>10.00</td>
<td>1.50</td>
<td>-3.13</td>
</tr>
</tbody>
</table>

In can be seen from Fig. 15 that in the first iteration a great reduction of the objective function is obtained and in the next ones reduction of the value of the objective function continue but with a relatively lower intensity. This is a consequence of the shape optimization evolution shown in Fig. 13, where initially a double curvature shape is formed with the mass being concentrated at the supports of the structure due to the stress concentration on this area. However, it can be seen that the upper and lower bounds fixed for the optimization points displacements are reached. This way, these restrictions do not allow curvature to increase and the optimal geometry obtained looks like two inverted parabolas.

4.3 Example 3

The initial geometry of this example is a cylinder. An initial height of \( h = 10 \), internal radius \( R_i = 23.50 \) and external radius \( R_e = 25.00 \) are used. The adopted boundary conditions are that the vertical and the horizontal displacements of the bottom boundary are constrained, an uniform distributed vertical load of \( q_v = 50 \cdot 10^3 \) is applied on the upper boundary and an uniform distributed horizontal load of \( q_h = 25 \cdot 10^3 \) is applied also on the upper boundary in the right direction (see Fig. 16). Body forces (self-weight) are considered in this example.

![Figure 16: Initial geometry and boundary conditions for example 3.](image-url)
The finite element mesh has 606 nodes and 1000 elements as shown in Fig. 17. The NURBS parametrization is done with 36 points, where 10 are adopted as optimization points (5 in the left boundary and 5 in the right boundary). Fig. 18 shows the adopted optimization nodes with red color and the others with blue color. The basis function are defined over the knot vectors:

\[
\Xi = \{0, 0, 0, 0, 0, 1, 1, 1, 1, 1\} \\
\mathcal{H} = \{0, 0, 0, 0, 0, 1, 1, 1, 1, 1\}
\]

The 10 optimization points are allowed to move in the horizontal direction bounded by an upper limit equal to 75 and a lower limit equal to 0. Also, horizontal distance between left and right boundary
nodes are limited to be great or equal to 0.5. At the 52th iteration, the optimum condition is reached. The final relative objective function obtained $F_{opt}$ is $F_{52}/F_{ini} = 0.199$. Fig. 19 shows a comparison between the initial and the final optimized geometry for displacements and von Mises stresses.

The decrease of the relative objective function as a function of the iterations is presented in Fig. 20. In this figure is also presented the final geometry. The initial and final coordinates of the 16 optimization points are shown in table 4.

### Table 4: Initial and final coordinates of optimization control points for example 3

<table>
<thead>
<tr>
<th>P</th>
<th>$y$</th>
<th>$x_i$</th>
<th>$x_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0,0)</td>
<td>0.00</td>
<td>23.50</td>
<td>28.48</td>
</tr>
<tr>
<td>(1,0)</td>
<td>2.00</td>
<td>23.50</td>
<td>27.49</td>
</tr>
<tr>
<td>(2,0)</td>
<td>4.00</td>
<td>23.50</td>
<td>26.42</td>
</tr>
<tr>
<td>(3,0)</td>
<td>6.00</td>
<td>23.50</td>
<td>25.61</td>
</tr>
<tr>
<td>(4,0)</td>
<td>8.00</td>
<td>23.50</td>
<td>24.42</td>
</tr>
<tr>
<td>(5,0)</td>
<td>10.00</td>
<td>23.50</td>
<td>23.50</td>
</tr>
<tr>
<td>(0,5)</td>
<td>0.00</td>
<td>25.00</td>
<td>29.73</td>
</tr>
<tr>
<td>(1,5)</td>
<td>2.00</td>
<td>25.00</td>
<td>28.70</td>
</tr>
<tr>
<td>(2,5)</td>
<td>4.00</td>
<td>25.00</td>
<td>27.86</td>
</tr>
<tr>
<td>(3,5)</td>
<td>6.00</td>
<td>25.00</td>
<td>26.77</td>
</tr>
<tr>
<td>(4,5)</td>
<td>8.00</td>
<td>25.00</td>
<td>26.03</td>
</tr>
<tr>
<td>(5,5)</td>
<td>10.00</td>
<td>25.00</td>
<td>25.00</td>
</tr>
</tbody>
</table>

The final optimized geometry has a shape equivalent to a truncated cone with a hole and the thickness is almost constant, with a slight increase in the upper surface. The inclination of the structure becomes practically aligned with the resultant of applied load, generating, as a result, a main compression state. Also, it can be observed that the displacements on the optimized geometry are uniformly distributed among the height. The displacements and stresses decreases significantly, as it can be observed.

Figure 19: Comparison between displacements and von Mises stresses in initial and final optimized geometry for example 3.

### 5 Conclusions

The model developed to optimize axisymmetric solids showed to be very useful. The optimized shapes found are consistent, having direct physical interpretation and better structural performance. The shape manipulation using a NURBS description promotes an easy form to perform the changes in the geometry while preventing the use of many optimization variables. The methodology applied to move
the mesh, avoiding high mesh distortions and control points interpenetration has shown to be a good option instead of the use of remeshing techniques, which are much more expensive computationally and the imposition of this method is coupled with the optimization algorithm, once constraints are imposed to move the mesh. The sensitivity analysis performed by reverse automatic differentiation guarantees not only exact derivative computations but also computational efficiency, and, as commented earlier, gradient evaluation is one of the main well-known numerical difficulties of optimization problems. Indeed, the use of AD for solid optimization is a little explored area until now. The SQP is a robust numerical optimization approach, as it can be observed once the initial geometry is compared with the final one obtained, both being very distinct in many attributes. The optimization procedure provides here the possibility of a continuum variation of the thickness, which is harder to be obtained with the use of a finite element shell formulation. On the other hand, axisymmetric geometries and loads are a more limited problem. The use of this methodology for plane stress and plane strain cases is straightforward.

![Figure 20: Relative objective function as function of the iterations for example 3.](image)

References


