Generalized Bezier components and successive component refinement using moving morphable components

T. Shannon · T. T. Robinson · A. Murphy · C. G. Armstrong

Received: 26 November 2021 / Revised: 25 May 2022 / Accepted: 26 May 2022 / Published online: 28 June 2022 © The Author(s) 2022

Abstract
This paper demonstrates developments that introduce generalized Bezier components in the Moving Morphable Components (MMC) optimization framework. Methods of enhancing the parameterization of the components to provide the opportunity for a better optimum, than can be achieved using existing approaches, are also described. The use of control points and Bezier curves for representing structural components provides both additional flexibility in the shape and a parameterization that complies with extrude and swept feature-based templates available in commercial computer-aided design (CAD) packages. Methods of representing these structural components, calculating analytical derivatives, and numerical examples demonstrating their integration in the MMC framework, are presented for a series of author-derived and literature problems. A successive refinement technique demonstrates how the additional flexibility in the structural components enables progressive improvement in the objective function. For the examined problems, increasing the design variables per component (from 5 to 15) resulted in solutions with 6% to 36% reduction in compliance. This improvement was achieved without increasing the number of components in the design space.

Keywords Topology optimization · Moving morphable components · Topology description function · Bezier component representation · Successive refinement

1 Introduction
Integrating topology optimization techniques in industrial design processes is an ongoing and dynamic field of research. Broadly speaking, two distinct research directions for topology optimization are followed. These fall into the categories of “Microstructure-approaches” and “Macrostructure-approaches”, terms coined in the review by Eschenauer and Olhoff (2001).

Microstructure-approaches typically employ a fixed analysis mesh, on which the optimal material distribution is sought. The material distribution is modeled by size variables, which define constant material properties within each finite element. These are also referred to as density-based methods (Sigmund and Maute 2013). Density-based topology optimization has reached a high level of maturity and consequently is commonly utilized in commercial software packages (Rozvany 2007). The maturity and availability of these approaches have stimulated the development of post-processing procedures to generate equivalent CAD representations (Yin et al. 2020). These seek to create a parameterized CAD model, from the raster model output from the density-based optimization process, which is not a trivial task.

The alternative macrostructure-approach seeks to explore methods of embedding explicit geometric parameters in the optimisation process. Historically, the explicit use of geometric parameters imposed limitations on the optimization process. A Lagrangian based mesh was employed, which needed to be updated for each iteration of the optimization. Furthermore, only shape and size variables could be considered, and additional techniques had to be introduced for modeling new holes (Sigmund and Maute 2013). This was pioneered in the bubble-method developed by Eschenauer et al. (1994), where new holes were inserted and then perturbed using shape optimization techniques. More recently, these approaches have been developed through feature-based topology optimization, categorized as “feature-mapping...
methods” in the recent review by Wein et al. (2020). In these methods the explicit use of geometric parameters is retained, however, they also can simultaneously model changes in the design topology. This addresses the former restriction of macro-structure approaches only considering variations in shape and size. The use of high-level geometric parameters forms a critical link with CAD systems, providing an alternative research direction from the Microstructure-approach. Here, rather than seeking to fit a geometric parameterization to the results of a density-based process, geometric parameters are defined for components at the outset of the process and used as the optimization design variables.

The motivation of this work is to enhance parameterization of components within the MMC optimization framework, to provide additional flexibility in component shape for a better optimum, and a parameterization that complies with modeling capabilities in commercial CAD packages. Despite the postulation that feature-mapping methods form a strong link with CAD systems, Wein et al. (2020) point out that there is little evidence demonstrating this link in published work. While it is acknowledged there are several technical reasons for this, the authors suggest that a significant reason is due to how the geometric features are modeled. Thus, this paper aims to develop and demonstrate geometric parameterizations useful in topology optimization, which also conform with modeling strategies that would typically be used in CAD-based design. This will help to address the challenge of the seamless integration of the optimization design into a CAD environment. Modeling practices are highly dependent on the type of components being modeled. This work focuses on modeling components that can be classified as geometrically long and slender, with the role of providing stiffness.

The following section briefly reviews preceding geometric parameterizations employed in the literature and proposes a parameterization that offers benefits to the optimization and a form that is appropriate for CAD representation using extrude and swept feature-based templates. This is followed by a detailed description of the proposed parameterization, its implementation within an optimization procedure, and its demonstration and assessment on several test problems. The paper then concludes with a discussion of the outstanding challenges, key conclusions and proposed future work.

## 2 Literature review

Feature-mapping refers to the process of representing an explicit geometric component on a fixed analysis mesh. This process is also referred to as component representation. This fundamental attribute of the MMC optimization process presents the primary challenge for representing structural components in a way that is consistent with conventional CAD modeling practices. Before outlining the proposed developments, a brief review will be given of the recent progression in MMC techniques.

Before the use of explicit geometric components to model solid geometry, geometric primitives were used as regions to subtract from a solid design domain (Cheng et al. 2006; Mei et al. 2008). The geometric primitives were modeled using a topology description function (TDF). Guo et al. (2014) developed this to construct a TDF based on a form of super-ellipse (Norato 2018) that modeled structural components directly. Rectangular shaped components were modeled using the component mid-point, length, width, and angular orientation. This was further developed by Zhang et al. (2016), who introduced additional geometric parameters to explicitly model a variation in width as a function of the length of the component. Both linear and quadratic width variations were modeled using polynomial equations. Further flexibility was introduced by Guo et al. (2016), who introduced the concept of “curved skeletons”. These enabled modeling components that had a central spine defined through a mathematical curve, for example, a non-parametric polynomial or trigonometric function. The width of the component could be uniform, or also variable along the length of the component spine.

An alternative method for modeling components utilizes what is known as a “closed B-spline” (Zhang et al. 2017a, b). To prevent self-intersection and irregular geometries, a closed loop is formed from the B-spline curve. Multiple closed loops may overlap, causing the interior control points to be deactivated and the formation of a new smooth curve using the exterior control points. The B-splines are used to model the boundary between solid and void regions. The optimization is progressed by perturbing the B-spline’s control points, Zhang et al. (2017a, b) and Du et al. (2019). While this method establishes a clearer link with CAD modeling capabilities, with most CAD systems supporting a form of B-spline, it will be recognized that this also does not follow a conventional modeling approach for structures such as trusses, frames and stiffened plates utilizing long, slender components. Additionally, the final representation of structural components presents challenges in using this parameterization for subsequent design changes.

Recent work by Zhu et al. (2021), has introduced wide Bezier components in the MMC framework. Previous use

---

1 This work concentrates on density-based and feature-mapping methods, hence these terms will be used subsequently in favor of microstructure and macrostructure-approaches, of which the aforementioned are subsets.
of wide Bezier components for structural optimization was presented by Wang and Yang (2009). This work, however, used a non-differentiable component representation, and so, a genetic algorithm was used for perturbing the design. The use of genetic algorithms is still deemed to be a computationally expensive method for solving structural design problems with many optimization variables. The implementation of Zhu et al. uses a differentiable method of representing the morphable components based on a signed distance function. Newton’s method was used to find the shortest distance from a finite element mesh node to the Bezier curve. The Bezier curve modeled the spine of the structural component.

In summary, there is a gap in the current literature concerning the optimization of problems requiring long, slender components and parameterizations which match typical CAD system representations. Therefore, herein a representation using a Bezier parameterization is described. In this work, it is recognized that the shortest distance to the curve from any mesh node will be a perpendicular vector from the Bezier spine. This means that the component can be constructed without iterative methods for second-order Bezier components, thereby maintaining computational efficiency. For higher-order curves, there may be multiple perpendicular vectors from any given mesh node to the curve. This can be solved by finding the solutions of higher-order polynomial equations, for which an iterative solver is required. Bezier curves will be used to model the component spines in one formulation, and the bounds of the component in the width direction in a second formulation. These methods will be combined to form a general formulation that can represent Bezier defined spines and width parameters simultaneously, retaining a differentiable form.

### 3 Methodology

This section outlines the methods utilized to construct the aforementioned Bezier defined components. In Sects. 3.1 and 3.2, Bezier spine components will be considered, followed by Bezier width components, Sect. 3.3. Building on these, a generalized formula will be suggested, Sect. 3.4. The overall MMC framework and optimization strategy follow the well-documented approach of Guo et al. (2014), and Zhang et al. (2016). More emphasis will be placed on the differentiability of the modeling strategies, as required for the sensitivity analysis, Sect. 3.5.

#### 3.1 Geometry projection techniques for representation of structural Bezier controlled components

The original TDF proposed by Guo et al. (2014) for a 2D problem is

\[
\phi(x; m) = 1 - \left( \frac{\cos(\theta)(x - x_0) + \sin(\theta)(y - y_0)}{L/2} \right)^m - \left( \frac{-\sin(\theta)(x - x_0) + \cos(\theta)(y - y_0)}{w/2} \right)^m, \tag{1}
\]

where \((x, y)\) is the mesh node coordinates, \((x_0, y_0)\) specifies the component midpoint, \(L\) the length, \(w\) the width, and \(\theta\) the angular orientation. A penalty parameter, \(m\), is used to increase the sharpness of the component boundary, set at 6. It was used in conjunction with a level set function

\[
\begin{align*}
\phi(x) &> 0, & x &\in \Omega, \\
\phi(x) & = 0, & x &\in \partial \Omega, \\
\phi(x) &< 0, & x &\in D \setminus (\Omega \cup \partial \Omega)
\end{align*}
\tag{2}
\]

to determine the region of the global design space, \(D\), occupied by a structural component, \(\Omega\). \(\partial \Omega\) indicates the boundary of the component. A maximum operator is used to combine multiple components. For \(N\) structural components, \(\phi^s(x) = \max(\phi_1(x), \ldots, \phi_N(x))\), where \(s\) denotes the aggregated representation of all components. The final structure is represented as

\[
\begin{align*}
\phi^s(x) &> 0, & x &\in \Omega^s, \\
\phi^s(x) & = 0, & x &\in \partial \Omega^s, \\
\phi^s(x) &< 0, & x &\in D \setminus (\Omega^s \cup \partial \Omega^s).
\end{align*}
\tag{3}
\]

The level set formulation is represented graphically for a single component in Fig. 1.

The TDF of (1) is equivalent to modeling linear components of fixed width, \(w\), between two control points, \((p_0, q_0)\) and \((p_1, q_1)\). A corresponding formulation for the TDF is

\[
\phi(x; m) = 1 - \left( \frac{(p_1 - p_0)}{L} \left( x - \frac{p_0 + p_1}{2} \right) + \left( q_1 - q_0 \right) \left( y - \frac{q_0 + q_1}{2} \right) \right)^m - \left( \frac{(q_1 - q_0)}{w} \left( x - \frac{p_0 + p_1}{2} \right) + \frac{(p_1 - p_0)}{L} \left( y - \frac{q_0 + q_1}{2} \right) \right)^m,
\tag{4}
\]

where

\[
L = \left( (p_1 - p_0)^2 + (q_1 - q_0)^2 \right)^{1/2}. \tag{5}
\]

#### 3.2 Bezier spine components

The TDF of Eq. 4 can be modified to represent components of fixed width, \(w\), that follow a Bezier spine curve (Fig. 2).
Bezier curves are parametric with the form $C(t) = (x(t), y(t))$, expressed as a function of the parameter $t \in [0, 1]$. Each mesh node can be projected onto the Bezier curve at a point $(x_r, y_r)$, by a vector that is perpendicular to the curve, $\mathbf{n}(t)$. This was achieved by solving the dot product of the projection vector with the tangent vector to the curve, $\mathbf{C}'(t)$. When perpendicular, the dot product will equal zero. The length of the perpendicular projection vector and the corresponding value of the intersection point, $t^*$, was used to construct a representation of the component as

$$
\phi(\mathbf{n}(t^*)(x), t^*(x); m_1, m_2) = \max \left(1 - \left(\frac{||\mathbf{n}(t^*(x))||}{n_2}\right)^{m_1} - \left(1 - t^*(x) + t^*(x)^2\right)^{m_2}\right),
$$

(6)

where $m_1$ and $m_2$ are independent penalty parameters.

The max operator is required as any given mesh node may be perpendicularly projected onto the Bezier curve at more than one intersection point. This selects the intersection point that is closest to the center of the component. While being valid for a Bezier curve of degree $d$, the derivation will be demonstrated on a curve of degree two, with three control points. The general expression for a Bezier curve is given as

$$
C(t) = \sum_{i=0}^{d} b_i B_{i,d}(t),
$$

(7)

where $b_i$ is the $i$th control point, and $B_{i,d}(t)$ is a polynomial of the form

$$
B_{i,d}(t) = \frac{d!}{(d-i)!} (1-t)^{d-i} t^i.
$$

(8)

For a curve of degree two, this equates to

$$
x(t) = (1-t)^2 p_0 + 2(1-t)t p_1 + t^2 p_2
$$

(9)

$$
y(t) = (1-t)^2 q_0 + 2(1-t)t q_1 + t^2 q_2
$$

(10)

where the three control points are, $b_0(p_0, q_0), b_1(p_1, q_1)$, and $b_2(p_2, q_2)$.

The projection vector of a mesh node, $(x_A, y_A)$, onto the Bezier spine is expressed as

$$
\mathbf{n}(t^*) = (n_x(t^*), n_y(t^*)) = (x_i(t^*) - x_A, y_i(t^*) - y_A)
$$

(11)

where $t^*$ is the intersection point, and

$$
n_x(t^*) = (p_0 - 2p_1 + p_2) t^2 + (-2p_0 + 2p_1) t^1 + p_0 - x_A
$$

(12)

$$
n_y(t^*) = (q_0 - 2q_1 + q_2) t^2 + (-2q_0 + 2q_1) t^1 + q_0 - y_A
$$

(13)

The tangent vector at the intersection point is expressed as $\mathbf{C}'(t^*) = (x'(t^*), y'(t^*))$, where

$$
x'(t^*) = 2(p_0 - 2p_1 + p_2) t^1 + (-2p_0 + 2p_1)
$$

(14)
Solving gives the intersection points on the Bezier spine. For a Bezier spine of degree \(d\), this generates a polynomial of degree \(2d - 1\). For a Bezier curve of degree two, this results in a cubic polynomial which can be solved using Cardano’s cubic formula (Kurosh 1984). This ensures that the process of geometry projection is entirely algebraic which is an important development from methods using Bezier parameterizations, available in the literature, discussed in Sect. 2. For Bezier curves with a degree greater than two, numerical methods are required to find the roots of the polynomial.

Solutions for \(t^*\) can be substituted into (6) to form the representation of the Bezier component on a fixed mesh. A regularized Heaviside function is used to smooth the boundary between solid and void domains, and also control the lower material limit. Several variations are discussed by Wein et al. (2020), and the polynomial form used in this work is

\[
\phi(x; m_1) = \begin{cases} 
1, & \text{if } \phi > \epsilon, \\
\frac{3(1-\alpha)}{4} \left( \frac{\phi}{\epsilon} - \frac{3\phi}{\epsilon \alpha} \right) + \frac{1+\alpha}{2}, & \text{if } -\epsilon \leq \phi \leq \epsilon, \\
\alpha, & \text{otherwise.}
\end{cases}
\]

where \(\epsilon\) determines the width of the regularization region, set at 0.5, and \(\alpha\) is a lower material limit to ensure the stability of the finite element solution, set at 1e-2 in this work. The significance of applying the regularized Heaviside function to the TDF function values for defining the upper and lower function limits and smoothing the transition region is depicted in Figs. 4 and 5.

The influence of the independent penalty parameters in (6), \(m_1\) and \(m_2\), can be visualized by considering a degree two Bezier curve with control points \(b_0(-500, 0), b_1(0, 0),\)
and \( b_2(500, 0) \), and a width, \( w \), of 100 mm (Fig. 3). The \( \phi(x; m_1, m_2) \) and \( H(\phi; \varepsilon) \) values for one-dimensional cross-section “A–A” is plotted for a range of \( m_1 \) values in Fig. 4a and b, respectively. Note the influence of \( m_2 \) at this section is negligible, and the value of \( \epsilon \) has remained constant at 0.5. Similarly, \( \phi(x; m_1, m_2) \) and \( H(\phi; \varepsilon) \) is plotted for a range of \( m_2 \) values at section “B–B”, where the influence of \( m_1 \) is negligible (Fig. 5a and b, respectively). Again, the value of \( \epsilon \) has remained constant at 0.5. It will be observed from the horizontal axis in both plots that a greater \( m_2 \) penalty is required to ensure the physical width of the transition region at the ends of the component are comparable to the physical width along the component edges. Consequently, \( m_1 \) is set at 4 in this work, and \( m_2 \) is set at 50. The independent penalty parameters have only been used for component formulations that implement Bezier spines.

The regularized Heaviside function combined with a trapezoidal rule with multiple integration points determines the material fraction, \( v_i^e \), for each analysis element as

\[
v_i^e = \left( \frac{4\sum_{j=1}^{NG} \left( \sum_{i=1}^{4} H(\phi_{ji}; \varepsilon) \right)}{4NG} \right),
\]

where \( i \) is the \( i \)th node of the \( j \)th geometry element, and the \( e \)th analysis element consists of \( NG \) geometry elements. In this work 5 integration points have been used which corresponds to 25 geometry elements within each analysis element. The Young’s modulus is calculated as

\[
E_i^e = E_0(\varepsilon)^P,
\]

where the penalty \( P \) is set at 2 in this work, and \( E_0 \) is the Young’s modulus of solid material. The total material volume is found as the summation of \( v_i^e \) over all the elements, \( NE \), in the design domain. The material fraction distribution for a degree two Bezier curve with control points \( b_0(500, 200), b_1(1000, 1000), \) and \( b_2(2000, 800) \), and a width, \( w \), of 100 mm is depicted in Fig. 6.

### 3.3 Bezier width curves

Here, a formulation is described which utilizes Bezier curves as guide curves to control the width of a component as a function of its length, while assuming a linear spine between two control points. This enables curvature in the outer profile, as depicted in Fig. 7, providing a variable width. Design variables, \( d_i \), are linked to the Bezier control points, \( b_i \). These link the first and last control points to the start and end control points of the component spine. Multipliers are applied to the unit normal vector from the component spine, using parameters \( d_1 \) and \( d_2 \), respectively. The middle control point, \( b_i \), is defined by a multiplier on the unit normal vector.
of the spine and the distance from the start control point, using parameters $d_3$ and $d_4$, respectively.

The updated TDF

$$\phi(x, w(t); m_3) = 1 - \left( \frac{\frac{d_3-w}{L}}{2} \right) \left( x - \frac{d_3}{2} - \frac{d_4}{2} \right) + \left( \frac{\frac{d_4-w}{L}}{2} \right) \left( y - \frac{d_3}{2} - \frac{d_4}{2} \right)^m$$

(20)

is similar to (4) as only two control points are used to model the component spine, however, it enables variation of the width, $w$, along the length of the component. The penalty parameter, $m_3$, is set at 4 in this work. Each mesh node, $(x, y)$, will have a unique width as a function of the parameter, $t$, depicted in Fig. 8.

The width is calculated by firstly determining the value of $t_{spn}$ where the mesh node projects onto the component spine using (16) and the procedure outlined in Sect. 3.2. A unit normal vector at this point is projected onto the Bezier guide curve to determine the guide curve intersection point, $t_{bez}$. The distance between the intersection points $B(t_{bez})$ and $C(t_{spn})$ is equal to the half width of the component, as given by

$$w(t) = \|B(t_{bez}) - C(t_{spn})\|.$$  

(21)

Calculating the intersection point on the Bezier guide curve involves finding the solutions of a polynomial equation, hence more than one value for the component width may be found, Fig. 9. The correct solution is selected as that which has the smallest corresponding normalized value, $t_{bzn}$, given by

$$t_{bzn} = (t_{bez} - 0.5)^2.$$  

(22)

The single intersection point, $t_{bez}$, used in (21) is selected from the vector $t_{bez}$ at the index determined by the index of smallest value in the corresponding normalized vector, $\min(t_{bzn})$.

As with the Bezier spine components, Eqs. 17 and 18 can be used to determine the material fraction assigned to each finite element. This is depicted in Fig. 10 for a guide curve with control points $b_0(1000, 700)$, $b_1(1100, 600)$, and $b_2(2000, 600)$. The component spine is a line between points $(1000, 500)$ and $(2000, 500)$.
3.4 Generalized Bezier components

The Bezier spine curve and the Bezier representation of width can be united in a single representation. A Bezier curve is used to model the component spine as in Sect. 3.2, and an additional dimension is included in the control points to model the in-plane width $w$, i.e. the Bezier curve is $(x, y, w) = b(t)$. Alternatively, the component spine and component boundary could be modeled by two independent Bezier curves, however, this would require additional computational effort in constructing the geometry representation as intersection points must be calculated for both Bezier curves. The chosen approach introduces a minor restriction on the design freedom, as the width parameters are constrained to the Bezier spine control point positions. However, this minor restriction provides computational savings in constructing the components, as intersection points are only calculated for the spine Bezier curve. The process for constructing the component representation is similar to that in Sect. 3.2, however, the width parameter, $w$, of Eq. 6 is now a function of the parameter $t$. This is a trivial calculation using the general Bezier curve formula (7), after having found the intersection point, $t^*$, on the Bezier spine curve. The TDF can be updated as

![Variable width defined by Bezier guide curve](image1)

![Bezier intersection points](image2)

![Material fraction for Bezier guide curve component](image3)


\[
\phi(n(t^*(x)), w(t^*(x)), t^*(x); m_1, m_2) = \max\left(1 - \left(\frac{n(t^*(x))}{w(t^*(x))}\right)^{m_1} - \left(1 - t^*(x) + t^*(x)^2\right)^{m_2}\right).
\]

(23)

As multiple mesh nodes may be perpendicularly projected onto the Bezier spine curve, the max function is required to select the mesh node that is closest to the center of the component.

The modeling flexibility this enables is shown in Fig. 11 for a degree three Bezier curve. The corresponding control points are: \(b_0(500, 200, 300)\), \(b_1(1000, 1000, -100)\), \(b_2(2500, -200, 400)\), and \(b_3(2500, 800, 50)\). This demonstrates how the small increment in parameters provides the potential for a significantly more complex component compared with the previous representations.

In summary, the theory outlined develops methods for representing structural components defined using explicit control points. The \(i^{th}\) structural component is parameterized by a vector of design variables, \(D^i\), that can be used directly as optimization variables in the MMC framework. \(D^i = (b_0, \ldots, b_d)\), where \(d\) is the degree of the component, and \(b_i\) is a vector of variables for the \(i^{th}\) control point, \((x, y, w)\). For \(N\) structural components, the vector of design variables is given as, \(D = (D^1, \ldots, D^N)\).

### 3.5 Optimization framework and sensitivity calculation

The MATLAB implementation of the MMC framework as provided by Zhang et al. (2016) is used as the basis for this work. Readers are referred to the original work for more details on the particulars of the process. While Zhang et al.’s implementation has an integrated FEA routine, in this work the commercial FEA package MSC Nastran was used. This was used to calculate the design response of interest, and formed part of the calculation of the geometric sensitivities. All other aspects of the framework have been implemented in MATLAB. The Method of Moving Asymptotes (MMA), (Svanberg 1987), was used to solve the optimisation problem. The convergence criterion is defined as a limit on the change in successive values of the objective function, set as 0.005% in this work. This tolerance must be achieved on two successive iterations to confirm convergence. Additionally, the volume constraint must be respected.

The sensitivity of compliance, \(C\), with respect to (wrt) each geometric parameter, \(a\) of component \(r\), can be found using a chain rule formula

\[
\frac{\partial C}{\partial a} = \sum_{e=1}^{NE} \frac{\partial C}{\partial E^e} \left(4NG \sum_{j=1}^{NG} \left(\sum_{i=1}^{4} P(H(\phi_{j,s}^e))\right)^{-1} \min\left(\frac{\partial H(\phi_{j,s}^e)}{\partial \phi_{j,s}^e}, \frac{\partial H(\phi_{j,s}^e)}{\partial \phi_{j,s}^e}\right)\right),
\]

(24)

where \(NE\) is the number of finite elements, and

\[
\frac{\partial H(\phi; e)}{\partial \phi} = \begin{cases} 
3(1-a)/4 & \text{if } \phi > \epsilon, \\
\frac{1 - \phi^2}{\epsilon^3} & \text{if } -\epsilon \leq \phi \leq \epsilon, \\
0 & \text{otherwise}.
\end{cases}
\]

(25)

The sensitivity of compliance wrt a change in material fraction, \(\partial C/\partial E^e\), is calculated using the SOL200 optimization module in MSC Nastran. The derivative of the TDF wrt \(a, \partial \phi/\partial a\), can be determined analytically for all the formulations given in Eqs. 6, 20, and 23. To demonstrate this, \(\partial \phi/\partial \phi_0\) is derived in Appendix 1 for Bezier spine components (6). Equivalent derivatives can be found for all design variables.
4 Numerical examples

To demonstrate the proposed MMC framework, three examples are provided. The first is a concentrated torque problem. This is followed by problems from published work, including a bridge design problem, previously presented by Zhang et al. (2016) and Guo et al. (2016), and a hanging load problem, previously considered by Zhu et al. (2021).

4.1 Torque beam

In this example, a torque is applied to the midpoint of the right edge of a square 1000 × 1000 mm design domain, discretized with an 80 × 80 FE mesh. The left edge of the design domain has a built-in constraint (Fig. 12). The objective of the optimization is to minimize the compliance subject to an optimization constraint on the design volume, which is set as 0.2.

4.1.1 Initial design

Initially, two linear morphable structural components are placed within the design domain. Each component is defined with two control points and a single width variable, \((p_0, q_0, p_1, q_1, w)\), and the representation constructed using (4). The initial design is shown in Fig. 12 with the geometric parameters for an individual component. The converged design is shown in Fig. 13 with a compliance of 26.78 mJ after 327 iterations. This provided the base model for introducing additional design variables, which are presented subsequently in Sects. 4.1.2–4.1.4.

4.1.2 Bezier spine solution

The morphable structural components of the converged solution of Fig. 13 were reparametrized to include an additional control point, \((p_2, q_2)\). For transitioning from two to three control points, the degree elevation formula,

\[
Q_i = \frac{i}{d+1} P_{i-1} + \left(1 - \frac{i}{d+1}\right) P_i, \quad 1 \leq i \leq d
\]  

(26)

was used to determine the coordinates of the additional control point. This ensures consistency in the shape of the component, avoiding unnecessary fluctuations in the structural response of the design problem by random shape variations. With three control points, the components are constructed using (6). The optimization with the reparametrized components converges to the design shown in Fig. 14, with a compliance of 23.03 mJ after 115 iterations. Figure 14 also shows the geometric parameters defining a single component.

4.1.3 Bezier width solution

As with Sect. 4.1.2, the solution of Fig. 13 can also be reparametrized to enable curvature of the component boundaries using Bezier guide curves. This is facilitated by (20), which provides an additional three design variables over the original parameterization. These additional variables were initialized to match the widths of the optimized components in Fig. 13. The variation achieved in the width variation along the length of the components is shown for the converged model in Fig. 15, with a compliance of 22.98 mJ after 341 iterations.
4.1.4 Successive refinement using generalized Bezier formulation

This section makes use of the generalized Bezier formulation (23), to demonstrate how structural components can be successively refined to improve the structural response. Taking the solution of Fig. 13 as a baseline, the first adaptation involved increasing the degree of the Bezier curve to two, Fig. 16. After convergence, an additional control point was added using (26). The degree three Bezier component design (Fig. 17) was then solved before adding a final control point, again using (26). The design problem was then solved using the degree four Bezier components (Fig. 18). The progression in the detail for the degree two, three, and four, Bezier components is depicted in Figs. 16, 17 and 18, respectively. The compliance of each model was 17.74 mJ, 17.17 mJ, and 17.14 mJ, respectively. A summary of the results for all formulations is presented in Table 1. Note, the number of iterations is the iterations for the given model and does not include the iterations from the previous stages in the successive refinement process. The total time reflects the cumulative time from previous stages.

4.2 Bridge design

A distributed load is applied to a fixed solid region at the top edge of a rectangular 3000×1000 mm design domain, discretized with a 120×40 FE mesh. Constraints in both the x and y directions are defined on the bottom two corners (Fig. 19). The objective of the optimization is to minimize the compliance subject to an optimization constraint on the design volume, which is set as 0.4.
model for introducing additional design variables, which are presented subsequently in Sects. 4.2.2–4.2.4.

4.2.2 Bezier spine solution

It is possible to parameterize the initial design space (Fig. 19) with three control points using the Bezier spine formulation (6). Figure 21 shows the solution when an additional control point was added midway between the start and end control points shown in Fig. 19. While the solution converged, the solution contains invalid self-intersecting geometry, highlighted in Fig. 21. This issue should be managed in the optimisation process, as attempting to reconstruct this geometry in a CAD system would cause errors.

Boundary curve self-intersections can be prevented by enforcing a constraint that the component width is less than the radius of curvature of the spine curve. Figure 22 shows an enlarged view of the self-intersecting region, with the region violating the constraint depicted between the two black lines. An alternative approach has been implemented in this work, using a successive refinement approach, where a lower degree solution is reparametrized to include an additional control point. This avoids the computation of the additional constraint but still proved effective in managing this issue in the constant width examples studied in this paper.

The morphable structural components of the converged solution of Fig. 20 were reparametrized to include an additional control point, using (26) and a Bezier spine formulation (6). The optimization with the reparametrized components converges to the design shown in Fig. 23, with a compliance of 1947.1 mJ after 87 iterations. Figure 23 also shows the geometric parameters defining a single component.

4.2.3 Bezier width solution

As with Sect. 4.2.2, the solution of Fig. 20 can also be reparametrized to enable curvature of the component boundaries using Bezier guide curves. These additional variables were initialized to match the widths of the optimized components.

| Model     | Description                               | Design variables per component | Iterations for this model | Compliance (mJ) | Total time (mins) |
|-----------|-------------------------------------------|--------------------------------|----------------------------|-----------------|-------------------|
| Figure 13 | Linear spine, constant width              | 5                              | 327                        | 26.78           | 101.6             |
| Figure 14 | Degree 2 spine, constant width            | 7                              | 115                        | 23.03           | 151.4             |
| Figure 15 | Linear spine, degree 2 Bezier width       | 8                              | 341                        | 22.98           | 384.1             |
| Figure 16 | Generalized degree 2 Bezier               | 9                              | 275                        | 17.74           | 292.4             |
| Figure 17 | Generalized degree 3 Bezier               | 12                             | 224                        | 17.17           | 571.6             |
| Figure 18 | Generalized degree 4 Bezier               | 15                             | 49                         | 17.14           | 637.1             |
in Fig. 20. The variation achieved in the width variation along the length of the components is shown for the converged model in Fig. 24, with a compliance of 1931.8 mJ after 255 iterations.

### 4.2.4 Successive refinement using generalized Bezier formulation

This section makes use of the generalized Bezier formulation (23) with a successive refinement approach. Taking the solution of Fig. 20 as a baseline, the first adaption involved increasing the degree of the Bezier curve to two, Fig. 25.
After convergence, an additional control point was added using (26). The degree three Bezier component design (Fig. 26) was then solved before adding a final control point, again using (26). The design problem was then solved using the degree four Bezier components (Fig. 27). The progression in the detail for the degree two, three, and four, Bezier components is depicted in Fig. 25, 26 and 27, respectively. The compliance of each model was 1897.5 mJ, 1871.6 mJ, and 1871.2 mJ, respectively. A summary of the results for all formulations is presented in Table 2. Note, as in the previous example the number of iterations does not include the iterations from the previous stages in the successive refinement process. The total time reflects the cumulative time from previous stages.

An issue with invalid geometry was encountered when solving the bridge problem with the generalized Bezier formulation. Figure 28 shows a case of invalid geometry demonstrating a two-fold issue. Firstly, the left support member is detached from the fixed region. Although this is displayed...
as a gap in the geometry, the finite element size is larger than the gap; hence, in the analysis there is continuous material between the two components. A refined finite element mesh would be required to resolve this issue. The second issue is due to the form of the left boundary of the same supporting component. A portion of the boundary curve is projected through itself due to the curvature in the component spine, combined with a large change in the component width. This results in geometry that appears to violate the boundary curve, such as point A in Fig. 29, which is calculated as
being outside the component relative to \( t_{0.84} \) and inside the component relative to \( t_{0.98} \). This scenario can be identified when the normal vector to the component boundary curve becomes perpendicular to the corresponding normal projection vector from the component spine curve in the interval \( t \in (0, 1) \), identified by Point B in Fig. 30. In fact, at this point the same constraint as in Fig. 22 is violated, i.e. the component width is less than the radius of curvature of the spine curve. The constraint was managed in this example by enforcing a lower limit of 980 mm on the \( q_2 \) parameter highlighted in Fig. 28. This was applied in the solutions of Figs. 25, 26 and 27.

### 4.3 Hanging load design

A vertical point load is applied at the midpoint of the bottom edge of a rectangular 3000 × 1000 mm design domain, discretized with a 120 × 40 FE mesh. Constraints in both

---

**Table 2 Bridge optimization results**

| Model     | Description                      | Design variables per component | Iterations for this model | Compliance (mJ) | Total time (mins) |
|-----------|----------------------------------|--------------------------------|---------------------------|-----------------|-------------------|
| Figure 20 | Linear spine, constant width     | 5                              | 247                       | 1991.5          | 57.6              |
| Figure 23 | Degree 2 spine, constant width   | 7                              | 87                        | 1947.1          | 96.5              |
| Figure 24 | Linear spine, degree 2 Bezier width | 8                              | 255                       | 1931.8          | 265.6             |
| Figure 25 | Generalized degree 2 Bezier      | 9                              | 185                       | 1897.5          | 214.1             |
| Figure 26 | Generalized degree 3 Bezier      | 12                             | 131                       | 1871.6          | 413.7             |
| Figure 27 | Generalized degree 4 Bezier      | 15                             | 11                        | 1871.2          | 432.0             |

---

Fig. 28 Invalid geometry due to left boundary curve

Fig. 29 Mesh node classification
the $x$ and $y$ directions are defined on the bottom two corners (Fig. 31). The objective of the optimization is to minimize

the compliance subject to an optimization constraint on the design volume, which is set as 0.4.

4.3.1 Initial design

Initially, four linear morphable structural components are placed within the design domain. Each component is defined with two control points and a single width variable, $(p_0, q_0, p_1, q_1, w)$, and the representation constructed using (4). This initial design is shown in Fig. 31 with the geometric parameters for an individual component. The converged design is shown in Fig. 32, with a compliance of 27.33 mJ after 425 iterations. This provided the base model for introducing additional design variables, which are presented subsequently in Sects. 4.3.2—4.3.4.

4.3.2 Bezier spine solution

The morphable structural components of the converged solution of Fig. 32 were reparametrized to include an additional control point using (26) and a Bezier spine formulation (6). The optimization with the reparametrized components converges to the design shown in Fig. 33, with a compliance
of 26.05 mJ after 62 iterations. Figure 33 also shows the geometric parameters defining a single component.

4.3.3 Bezier width solution

As with Sect. 4.2.2, the solution of Fig. 32 can also be reparametrized to enable curvature of the component boundaries using Bezier guide curves. These additional variables were initialized to match the widths of the optimized components in Fig. 32. The variation achieved in the width variation along the length of the components is shown for the converged model in Fig. 34 with a compliance of 26.05 mJ after 210 iterations.

4.3.4 Successive refinement using generalized Bezier formulation

This section makes use of the generalized Bezier formulation (23) with a successive refinement approach. Taking the solution of Fig. 32 as a baseline, the first adaption involved increasing the degree of the Bezier curve to two, Fig. 35. After convergence, an additional control point was added using (26). The degree three Bezier component design (Fig. 36) was then solved before adding a final control point, again using (26). The design problem was then solved using the degree four Bezier components (Fig. 37). The progression in the detail of the degree two, three, and four, Bezier components is depicted in Figs. 35, 36 and 37, respectively. The compliance of each model was 25.72 mJ, 25.20 mJ, and 24.91 mJ, respectively. A summary of the results for all formulations is presented in Table 3. As before the number of iterations does not include the iterations from the previous stages in the successive refinement process, while the total time reflects the cumulative time from previous stages.

Fig. 33 Hanging load solution with degree two Bezier spine, constant width components

Fig. 34 Hanging load solution with a linear spine, and degree two Bezier width components
5 Discussion

The progression through the numerical example models in Sect. 4 demonstrates how the addition of parameters can enable an improvement of the structural performance of the model. For each of the three examples, the same trend was observed, where the resulting compliance decreased as the model progressed through each of the refinements (Bezier spine, Bezier width and successive refinements, degree 2...
Bezier, degree 3 Bezier and degree 4 Bezier). The only exception was for the hanging load example, in which the compliance in the model with Bezier width parameterized was the same as in the model with the Bezier spine parameterized. While design improvements may also be achieved by increasing the number of components within the design space, this requires the subsequent merging of components to represent a clean geometry. This limitation has not been addressed in this work, and so only a small component count is used in these examples.

Comparing the minimized compliance for each formulation in the bridge design case enables the evaluation of the effectiveness of the additional parameters. As is expected, a better solution is achievable when additional parameters are used to define the structural components. However, diminishing returns are achieved as can be seen from Fig. 38. This is most evident in the transition from degree three to degree four generalized Bezier components, where a reduction in compliance of only 0.4 mJ is achieved in the bridge example. This indicates that alternative strategies must be employed, such as methods to change the design topology, if further reduction in the objective function is desired.

In terms of compliance improvement, the torque beam degree 4 Bezier gave a 36% improvement over the initial model, for the bridge a 6% reduction was achieved and for the hanging load a 9% improvement was achieved. Although the savings are small in real terms, the relative benefits of using these more flexible parameterization schemes are clearly illustrated through the range of problems examined and the consistent behavior witnessed. This has synergies with the concept of parametric effectiveness discussed by Robinson et al. (2013). This was followed by demonstrating methods of calculating geometric sensitivities (Agarwal et al. 2018), and parameter insertion techniques for improving CAD-based shape optimization (Agarwal et al. 2019).

The time taken per iteration is a combination of the time required to construct the geometric representation of the components, and the time required for the response analysis and optimisation solver. The latter components remain consistent for each design scenario, regardless of the component complexity. However, a significant increase in time is required for constructing higher-order generalized Bezier representations. For example, in the Bridge design, using linear components required ~ 14 s per iteration, while using degree four generalized Bezier components required ~ 100 s per iteration (on 3.4 GHz Intel Core i7 CPU with 16 GB RAM). Of these times, ~ 10 s was attributed to the FEA, and the remaining time was required by MATLAB to calculate the geometry representation and the MMA optimisation step. The increase in time for Bezier components is a direct

| Model   | Description                                | Design variables per component | Iterations for this model | Compliance (mJ) | Total time (mins) |
|---------|--------------------------------------------|--------------------------------|---------------------------|-----------------|-------------------|
| Figure 32 | Linear spine, constant width             | 5                              | 425                       | 27.33           | 133.7             |
| Figure 33 | Degree 2 spine, constant width             | 7                              | 62                        | 26.05           | 161.1             |
| Figure 34 | Linear spine, degree 2 Bezier width        | 8                              | 210                       | 26.05           | 377.6             |
| Figure 35 | Generalized degree 2 Bezier               | 9                              | 221                       | 25.72           | 333.8             |
| Figure 36 | Generalized degree 3 Bezier               | 12                             | 132                       | 25.20           | 538.1             |
| Figure 37 | Generalized degree 4 Bezier               | 15                             | 55                        | 24.91           | 638.6             |
implication of the root-finding algorithms required to determine where each mesh node projects onto the Bezier spine. No parallelization has been implemented.

The component representation schemes used in this work have followed the methodology presented by Guo et al. (2014), in which a penalty parameter \( m \) is used to increase the sharpness of the component boundary (1). For Bezier spine components it was found two penalty parameters were required to accurately represent the geometry of the component (6). This helped maintain a consistent width of the regularization region around the boundary of the components. While the chosen parameters were appropriate for the examples presented, it should be noted that other feature-mapping formulations do not require the specification of such penalty parameters and so may be more convenient for the end-user.

Throughout this work, several geometry issues have been highlighted, such as those depicted in Figs. 21 and 28. While the strategies employed have been demonstrated to be effective in managing these issues, future work should consider fully integrating geometric constraints within the optimization formulation. This will ensure the robustness of the framework.

Finally, it will be observed that non-differentiable max/min functions are employed in the component representation formulations (6, 22 and 23). While the non-differentiable nature of these functions has not been accounted for, it is suggested this is only a concern for situations where there is self-intersecting geometry. The max/min functions select the appropriate intersection points of a projected mesh node onto the Bezier curve. For mesh nodes on the boundary of the component, this will be a unique position within \( t \in [0, 1] \), unless component boundaries are self-intersecting. As previously discussed, methods of controlling self-intersecting components are also desirable from a component representation aspect. Maintaining appropriate geometric controls on the structural components will ensure stability in the optimization process.

6 Conclusions

This paper has presented an extension to the MMC framework that enables the modeling of structural components using a control point methodology. In particular,

- The use of Bezier curves for representing the components has been presented for various modeling scenarios, including Bezier spine curves, Bezier width curves, and a generalized Bezier formulation for both the spine and width.
- TDFs have been derived that provide an analytical method of representing the structural components on a fixed analysis mesh.
- The numerical examples demonstrate the application of the various methods of parameterization, and also demonstrate how a successive refinement approach can be implemented. This enables further reduction in the objective function through adding optimization variables in the form of additional control points.
- For the examined problems, transitioning from linear, constant width components to degree four generalized Bezier components facilitated between a 6 and 36% reduction in the objective function.

7 Future work

Future work will seek to address the challenges of maintaining non-intersecting geometries, focusing on techniques using the geometry projection approach due to the considerations outlined in Sect. 5. While this work has discussed successive refinement by increasing the design variables of each component, the same could be achieved by introducing additional structural components, another area of key interest for the MMC framework. While the proposed approach has sought to establish the link between the MMC framework and CAD systems, further investigation is required for appropriate methods to enable a seamless transition between the optimisation solution and its CAD representation (including assessing the need for geometry clean-up). Lastly, using the MSC Nastran solver provides scope for calculating the sensitivity to other quantities such as natural frequency and eigenvalue buckling.

Appendix 1: Analytical derivatives of a Bezier TDF

This appendix outlines the process for calculating the derivatives of the Bezier curve defined components, (6). Only the \( p_0 \) parameter is considered here, however, a similar process can be used for all the geometric parameters used to define the structural components.

Expressing the Bezier TDF (6) in terms of \( f \) and \( g \):

\[
\phi(p(t^*(x)), t^*(x)) = 1 - f^{m_1} - g^{m_2}
\]  

where

\[
f = 2r^{-1}\left[\left(n_1(t^*)\right)^2 + \left(n_2(t^*)\right)^2\right]^{0.5}
\]

\[
g = 1 - t^* + t^*^2
\]

The expressions for \( n_1(t^*) \) and \( n_2(t^*) \) have been defined previously in Eqs. 12 and 13, respectively.

Then:
\[
\frac{df}{dp_0} = -m_1 r^{(m_1-1)} \cdot \frac{d}{dp_0} - m_2 s^{(m_2-1)} \cdot \frac{d}{dp_0}
\]  
(30) \[
\frac{da}{dp_0} = 4(p_0 - 2p_1 + p_2)
\]  
(42)

where

\[
\frac{df}{dp_0} = r^{-1} \left[ (n_x(r^*)^2 + (n_y(r^*)^2 \right]^{-0.5}.
\]

\[
2(n_x(r^*) \cdot \frac{dn_x}{dp_0} + 2(n_y(r^*) \cdot \frac{dn_y}{dp_0})
\]

(31)

\[
\frac{dg}{dp_0} = (-1 + 2r^*) \frac{dr^*}{dp_0}
\]

(32)

\[
\frac{dn_x}{dp_0} = r^2 + 2r^*(p_0 - 2p_1 + p_2) \cdot \frac{dr^*}{dp_0} - 2r^* + (-2p_0 + 2p_1) \cdot \frac{dr^*}{dp_0} + 1
\]

(33)

\[
\frac{dn_y}{dp_0} = 2r^*(q_0 - 2q_1 + q_2) \cdot \frac{dr^*}{dp_0} + (-2q_0 + 2q_1) \cdot \frac{dr^*}{dp_0}
\]

(34)

The expression \(\frac{dr^*}{dp_0}\) refers to the derivative of the point of intersection, \(r^*, \text{ wrt } p_0\). The expression for \(r^*\) is given as:

\[
\alpha r^3 + \beta r^2 + \gamma r + \delta = 0
\]

(35)

where

\[
a = 2 \left[ (p_0 - 2p_1 + p_2)^2 + (q_0 - 2q_1 + q_2)^2 \right]
\]

(36)

\[
b = 3 \left( (p_0 - 2p_1 + p_2) + (q_0 - 2q_1 + q_2) \right)
\]

(37)

\[
c = 4 \left( -p_0 + p_1 \right)^2 + \left( -2p_0 + 2p_1 + q_0 - 2q_1 + q_2 \right) \left( q_0 - x_A \right)
\]

\[+ 4 \left( -q_0 + q_1 \right)^2 + \left( q_0 - 2q_1 + q_2 \right) \left( q_0 - y_A \right)
\]

(38)

\[
d = (-2p_0 + 2p_1) \left( p_0 - x_A \right) + (-2q_0 + 2q_1) \left( q_0 - y_A \right)
\]

(39)

This can be differentiated implicitly:

\[
\frac{da}{dp_0} r^3 + 3ar^2 \cdot \frac{dr^*}{dp_0} + \frac{db}{dp_0} r^2 + 2br^*
\]

\[
\cdot \frac{dr^*}{dp_0} + \frac{dc}{dp_0} r^* + c \cdot \frac{dr^*}{dp_0} + \frac{dd}{dp_0} = 0
\]

(40)

\[\frac{dr^*}{dp_0} = -\frac{\left( \frac{da}{dp_0} r^3 + \frac{db}{dp_0} r^2 + \frac{dc}{dp_0} r^* + \frac{dd}{dp_0} \right)}{3ar^2 + 2br^* + c},
\]

where expressions for \(a, b, c, \) and \(d\), are given in Eqs. 36–39, respectively, and their derivatives are:

Acknowledgements The authors would express their appreciation for the support received from the Northern Ireland Department for the Economy. Also, for the fruitful discussions with experts at the recent WCSMO-14. The authors thank our collaborators in Rolls-Royce for their valuable comments throughout this work.

Declarations

Conflict of interest The authors declared that they have no conflict of interest.

Replication of results The main code for this work (MATLAB) was based heavily on that published by Zhang et al. (2016). With reference to that work, keys changes and additional functions that construct the Bezier components and their derivatives are available on the institutional repository (https://doi.org/10.17034/4b8eea86-1be7-46ed-b29f-68aa25249f69). The variables required for constructing the reported examples are also available.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Agarwal D, Robinson TT, Armstrong CG, Marques S, Vasilopoulos I, Meyel M (2018) Parametric design velocity computation for CAD-based design optimization using adjoint methods. Eng Comput 34:225–239. https://doi.org/10.1007/s00366-017-0534-x

Agarwal D, Robinson TT, Armstrong CG, Kapellos C (2019) Enhancing CAD-based shape optimisation by automatically updating the CAD model’s parameterisation. Struct Multidisc Optim 59:1639–1654. https://doi.org/10.1007/s00158-018-2152-7

Cheng G, Mei Y, Wang X (2006) A feature-based structural topology optimization method. In: Bendsoe MP, Olhoff N, Sigmund O (eds) IUTAM symposium on topological design optimization
of structures, machines and materials. Solid mechanics and its applications. Springer, Dordrecht, pp 505–514. https://doi.org/10.1007/1-4020-4752-5_48

Du B, Yao W, Zhao Y, Chen X (2019) A moving morphable voids aApproach for topology optimization with closed B-splines. J Mech Des 141(8):081401. https://doi.org/10.1115/1.4043369

Eschenauer HA, Kobelev VV, Schumacher A (1994) Bubble method for topology and shape optimization of structures. Struct Optim 8:42–51. https://doi.org/10.1007/BF01742933

Eschenauer HA, Olhoff N (2001) Topology optimization of continuum structures: a review. ASME Appl Mech Rev 54(4):331–390. https://doi.org/10.1115/1.1388075

Guo X, Zhang W, Zhang J, Yuan J (2016) Explicit structural topology optimization based on moving morphable components (MMC) with curved skeletons. Comput Methods Appl Mech Eng 310:711–748. https://doi.org/10.1016/j.cma.2016.07.018

Guo X, Zhang W, Zhong W (2014) Doing topology optimization explicitly and geometrically—a new moving morphable components based framework. J Appl Mech 81(8):081009. https://doi.org/10.1115/1.4027609

Kurosh AG (1984) Higher algebra, 4th edn. MIR, Moscow

Mei Y, Wang X, Cheng G (2008) A feature-based topological optimization for structure design. Adv Eng Softw 39(2):71–87. https://doi.org/10.1016/j.advensoft.2007.01.023

Norato JA (2018) Topology optimization with supershapes. Struct Multidisc Optim 58:415–434. https://doi.org/10.1007/s00158-018-2034-z

Robinson TT, Armstrong CG, Chua H (2013) Determining the parametric effectiveness of a CAD model. Eng Comput 29:111–126. https://doi.org/10.1007/s00366-011-0248-4

Rozvany GIN (2007) A critical review of established methods of structural topology optimization. Struct Multidisc Optim 37:217–237. https://doi.org/10.1007/s00158-007-0217-0

Sigmund O, Maute K (2013) Topology optimization approaches. Struct Multidisc Optim 48:1031–1055. https://doi.org/10.1007/s00158-013-0978-6

Svanberg K (1987) The method of moving asymptotes—a new method for structural optimization. Int J Numer Methods Eng 24(2):359–373. https://doi.org/10.1002/nme.1620240207

Wang NF, Yang YW (2009) Structural design optimization subjected to uncertainty using fat Bezier curve. Comput Methods Appl Mech Eng 199:210–219. https://doi.org/10.1016/j.cma.2009.10.004

Wein F, Dunning PD, Norato JA (2020) A review on feature-mapping methods for structural optimization. Struct Multidisc Optim 62:1597–1638. https://doi.org/10.1007/s00158-020-02649-6

Yin G, Xiao X, Cirak F (2020) Topologically robust CAD model generation for structural optimisation. Comput Methods Appl Mech Eng 369:113102. https://doi.org/10.1016/j.cma.2020.113102

Zhang W, Yuan J, Zhang J, Guo X (2016) A new topology optimization approach based on Moving Morphable Components (MMC) and the ersatz material model. Struct Multidisc Optim 53:1243–1260. https://doi.org/10.1007/s00158-015-1372-3

Zhang W, Yang W, Zhou J, Li D, Guo X (2017a) Structural topology optimization through explicit boundary evolution. J Appl Mech 84(1):011011. https://doi.org/10.1115/1.4034972

Zhang W, Zhao L, Gao T, Cai S (2017b) Topology optimization with closed B-splines and Boolean operations. Comput Methods Appl Mech Eng 315:652–670. https://doi.org/10.1016/j.cma.2016.11.015

Zhu B, Wang R, Wang N, Li H, Zhang X, Nishiwaki S (2021) Explicit structural topology optimization using moving wide Bezier components with constrained ends. Struct Multidisc Optim 64:53–70. https://doi.org/10.1007/s00158-021-02853-y

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.