Structural and Multidisciplinary Optimization
https://doi.org/10.1007/s00158-021-03009-8

EDUCATIONAL PAPER

LayOpt: an educational web-app for truss layout optimization

Helen E. Fairclough[1] · Linwei He[1] · Thomas J. Pritchard[2] · Matthew Gilbert[1]

Received: 24 March 2021 / Revised: 25 June 2021 / Accepted: 28 June 2021

Abstract

A new interactive truss layout optimization web-app has been developed for educational use. This has been designed to be used on a range of devices, from mobile phones to desktop PCs. Truss designs are first generated via numerical layout optimization and then rationalized via geometry optimization. It is then shown that these designs can be simplified using a computationally inexpensive process that allows the user to control the trade-off between complexity and structural volume. The process involves the use of smooth Heaviside representations of member existence variables, with nodal slack forces employed that allow unstable intermediate truss structures. Full details of the web-app are provided in this contribution, from underlying formulation to cloud computing implementation. A range of numerical examples are used to demonstrate the efficacy of the web-app, and to show how it can potentially be used in educational and practical engineering settings.

Keywords Truss layout optimization · Web-app · Heaviside simplification · Topology optimization · Geometry optimization

1 Introduction

Interactive educational tools have been invaluable in raising awareness of the power of continuum topology optimization methods, from the TopOpt web-app launched in 2000 (Tcherniak and Sigmund 2001) to apps designed to run on mobile phones or tablets (Aage 2013; Nguyen et al. 2020). These tools are able to demonstrate to interested users the power of topology optimization, by allowing them to solve simple, generally two-dimensional, user-defined design problems. These tools, together with short educational scripts written in MATLAB and other high level languages (e.g. Sigmund 2001; Andreassen et al. 2011; Wei et al. 2018), have contributed to the development of a vibrant research and user community in the field.

Whereas those active in the continuum topology optimization field have benefited from the availability of interactive educational tools for the last two decades, there have to date been no similar interactive educational tools for truss layout optimization (also known as 'truss topology optimization'), though a number of short educational scripts have been made available (e.g. Sokół 2011; Zegard and Paulino 2014; He et al. 2019b). Also, a truss optimization method employing a 'growth' heuristic has been made available as a downloadable software program for PCs (Martinez et al. 2007). Truss layout optimization methods can identify structurally efficient arrangements of discrete structural elements forming a structure, and are particularly well suited for problems where the proportion of the available design domain occupied by structure is small, as is common for design problems encountered by structural engineers working in the construction industry.

However, until recently truss layout optimization methods have not found favour in industry. This is partly because the solutions obtained using basic numerical layout optimization methods will generally appear impractical to engineers working in practice, comprising numerous closely spaced elements that would be difficult to fabricate using traditional methods. Nevertheless, details of projects where optimal layouts have been used to guide and inspire designers working in practice have recently been outlined by Graczykowski and Lewiński (2020) and Zegard et al. (2020).

New digital fabrication techniques are now expanding the range of structures that can be fabricated and the current climate emergency is making increased material efficiency a far higher priority. Also, means of rationalizing

Responsible Editor: Emilio Carlos Nelli Silva

✉ Linwei He
linwei.he@sheffield.ac.uk

[1] Department of Civil and Structural Engineering, University of Sheffield, Mappin Street, Sheffield, S1 3JD, UK

[2] LimitState Ltd, Sheffield, UK

Published online: 07 August 2021

the raw solutions obtained via truss layout optimization have been developed in recent years (He and Gilbert 2015). Nevertheless, solutions will still often appear over-complex to practitioners and in the present contribution a new computationally efficient means of allowing the user to manage the trade-off between complexity and volume is presented.

To demonstrate the efficacy of truss layout optimization when used in conjunction with the aforementioned post-processing methods, these are here incorporated in a new interactive truss layout optimization web-app, LayOpt, publicly accessible via https://www.layopt.com. The web-app is designed to allow users to interactively solve a wide range of two-dimensional truss layout optimization problems, on a range of devices.

Specifically, the use of web-technologies and complementary serverless cloud computing techniques mean that computations can be carried out in the cloud, ensuring scalability (thousands of simultaneous connections are possible), and obviating the need for a powerful client, thereby allowing a wide range of devices to be used to access the web-app, including desktop computers, tablets and mobile phones, as shown in Fig. 1. The use of industry standard protocols means that the web-app will work on almost all web browsers released since 2011, and many released earlier than that too.

The paper is organized as follows: Section 2 presents details of the underlying numerical methods, including the new simplification process; Section 3 presents details of the cloud computing-based software architecture used by the web-app; Section 4 presents a wide range of example problems that demonstrate the range of applicability of the web-app; finally conclusions are drawn in Section 5.

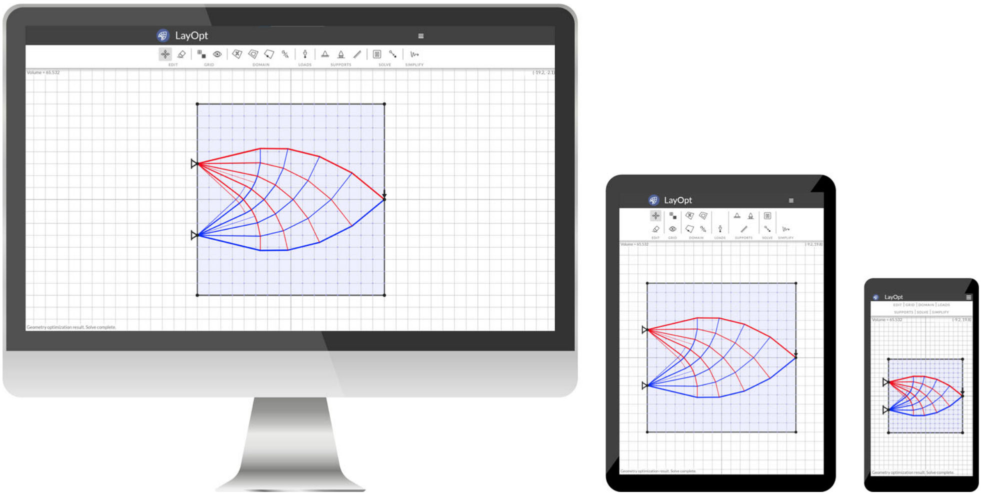
2 Numerical methods

2.1 Truss layout optimization

Truss layout optimization (after Dorn et al. 1964) provides a method of identifying minimum volume structures for a given design domain and set of loads and boundary conditions, e.g. see the example problem shown in Fig. 2a. This contains only a single point load, but any number of loads can be handled, applied either simultaneously or in separate load-cases.

Here, a rigid-plastic material model is assumed. Many common engineering materials exhibit a plastic response as failure is approached, and, by utilizing this behaviour, greater material savings can be realized. The use of a plastic formulation also obviates the need to include elastic compatibility constraints in the formulation. For single load-case problems a statically determinate layout can be found that will also be optimal when an elastic material is involved. However, for multiple load-case problems the optimal elastic and plastic solutions will diverge, with the volumes computed using the formulation described herein representing lower bounds on the corresponding elastic solutions.

In the truss layout optimization procedure, the design domain is first discretized using a series of nodes that are then connected by potential truss members to form a ‘ground structure’. For a fully general solution, each node should be connected to every other node; however, this becomes computationally challenging when large numbers of nodes are involved. To address this an adaptive ‘member adding’ strategy can be used (Gilbert and Tyas 2003; Pritchard et al. 2005; He et al. 2019b), which is guaranteed



(a) (b) (c)

Fig. 1 LayOpt web-app user interface as viewed on (a) a desktop computer; (b) a tablet; and (c) a mobile phone

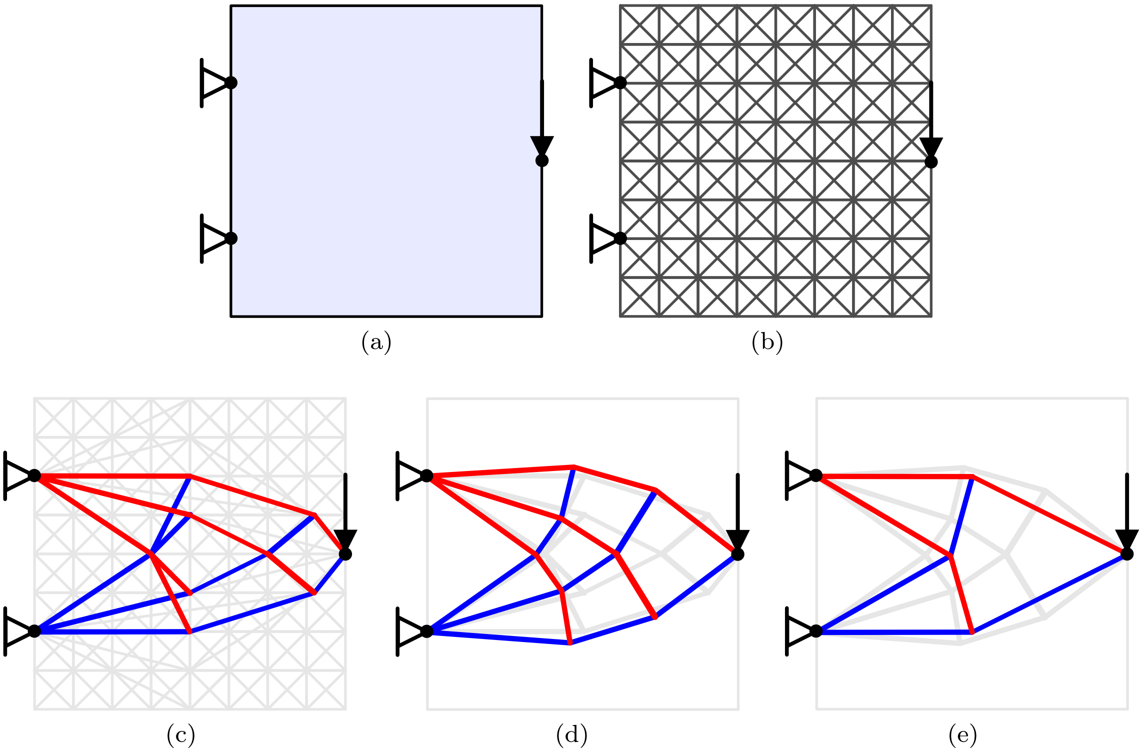


Fig. 2 Stages of the LayOpt optimization process: (a) specification of design domain, loading and supports; (b) discretization of problem using minimally connected ground structure with unit nodal spacing; (c) layout optimization solution (after member adding, volume = 57.60, assuming unit applied load and unit stress limits in tension and compression); (d) geometry optimization of solution (c) (volume after 5 iterations = 56.69) and (e) simplification of solution (d) (2% volume increase limit, volume = 57.71)

to obtain the same solution as would be obtained had all members been connected from the outset.

The iterative member adding strategy begins with a reduced ground structure, such as the one shown in Fig. 2b. Then at each iteration the following optimization problem (after He et al. 2019b) is assembled and solved, based on the current ground structure:

$$\begin{aligned}
&\underset{\mathbf{a},\mathbf{q}^{(k)}}{\text{minimize}} && V = \mathbf{l}^{\mathrm{T}}\mathbf{a} && \text{(1a)}\\
&\text{subject to} && \mathbf{B}\mathbf{q}^{(k)} = \mathbf{f}^{(k)} && \text{(1b)}\\
& && -\sigma^{-}\mathbf{a} \leq \mathbf{q}^{(k)} \leq \sigma^{+}\mathbf{a} && \text{(1c)}\\
& && \mathbf{a} \geq \mathbf{0}, && \text{(1d)}
\end{aligned}$$

where V is the total volume of members, $\mathbf{l} = [l_1, l_2, ..., l_m]^{\mathrm{T}}$ is a vector of element lengths. $\mathbf{a} = [a_1, a_2, ..., a_m]^{\mathrm{T}}$ is the vector of variables representing cross section areas of each element, $\mathbf{B}$ is a matrix of direction cosines, $\mathbf{q}^{(k)}$ is a vector of variables representing the axial force in each element in load-case k, and $f^{(k)} = [f_1^{k,x}, f_1^{k,y}, f_2^{k,x}, ..., f_n^{k,y}]^{\mathrm{T}}$ is the vector of externally applied forces. σ^{+} and σ^{-} are the permitted stresses in tension and compression respectively. Once the problem is solved, solutions for both the primal (1) and dual problems (see He et al. 2019b for details) are extracted. The element forces and areas obtained from the primal problem provide an interim solution, the minimum volume structure possible using the current ground structure, which can be presented to the user to indicate progress. The solution of the dual problem furnishes the virtual displacements of each node. These can be used to calculate virtual strain values for each potential member, regardless of whether or not this is present in the current ground structure. If the virtual strain of a potential member would violate the dual constraint value, then that potential member is considered as a candidate for admission to the ground structure in the next iteration. Figure 2c shows the current ground structure at the final iteration (in grey), as well as the identified optimal structure (where tensile members are shown in red and compressive members in blue; this convention is used throughout this paper).

Figure 3 shows the layout optimization process, which ends with a filtering and validation stage.

2.2 Filtering and validation

Problem (1) generally has optimal solutions where the majority of the variables **a** and **q** are zero, i.e. most elements in the ground structure are not required in the final optimal structure. However, the numerical methods used to solve this optimization problem usually provide solutions where

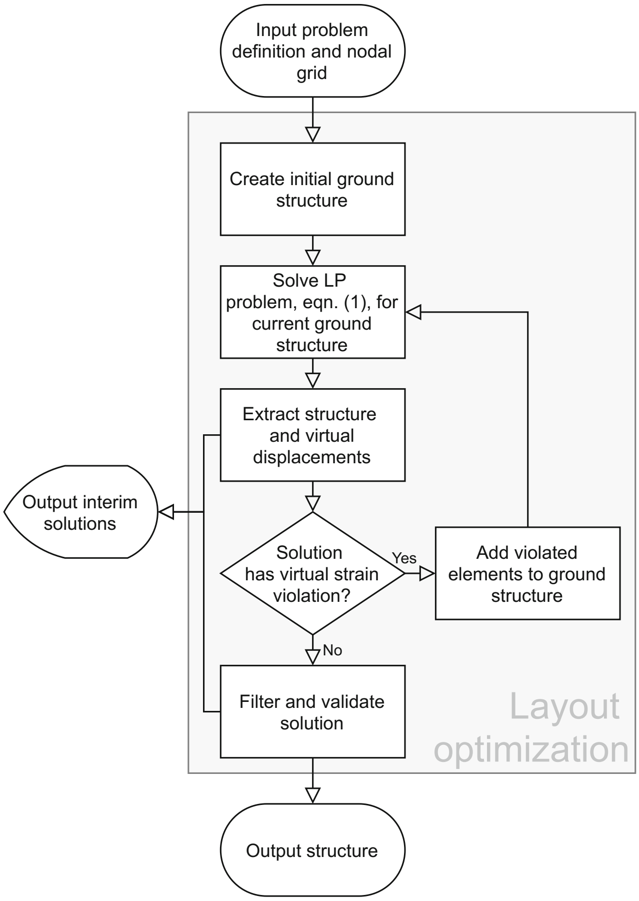

Input problem definition and nodal grid
Create initial ground structure
Solve LP problem, eqn. (1), for current ground structure
Extract structure and virtual displacements
Output interim solutions
Solution has virtual strain violation?
Yes
Add violated elements to ground structure
No
Filter and validate solution
Layout optimization
Output structure

Fig. 3 Layout optimization procedure

these variables are close to, but not exactly, zero. Thus, in order to use the solutions obtained in the subsequent geometry optimization stage, a filtering and validation step is required.

Unlike the filter algorithm proposed, e.g., by Ramos and Paulino (2016), where elements are removed during the optimization process, thus potentially influencing the form of the optimal solution, here filtering is only utilized after the optimal solution has been found. The resulting filtered solution is obtained by removing elements with an area below a certain filter value (e.g., the grey lines in Fig. 2d show the members that remain after filtering). Then a validation problem is set up and solved to check that the filtered solution is capable of supporting the required loads. If the validation fails, then the process is repeated using a different (lower) filter value. It is convenient to set this filter value relative to $a_{\max}$, the largest area in the solution; an initial filter value of $0.01a_{\max}$ is chosen by default. On subsequent attempts this reduces by an order of magnitude each time, to $0.001a_{\max}$, then $0.0001a_{\max}$ etc. Thus, relatively simple filtered structures are first tested, and only if these are found to be structurally unviable are structures with fewer filtered members tested.

The validation of the filtered problems involves solving an optimization problem that is similar to problem (1):

$$\begin{aligned}
&\underset{\mathbf{a},\mathbf{q}^{(k)},\mathbf{f}_{\mathrm{s}}^{(k)},\tilde{\mathbf{f}}_{\mathrm{s}}}{\text{minimize}} && \tilde{V} = \mathbf{l}^{\mathrm{T}}\mathbf{a} + \mu_{\mathrm{s}} \sum_{j=1}^{n} (\tilde{f}_{\mathrm{s},j}^{x} + \tilde{f}_{\mathrm{s},j}^{y}) && \text{(2a)}\\
&\text{subject to} && \mathbf{B}\mathbf{q}^{(k)} = \mathbf{f}^{(k)} + \mathbf{f}_{\mathrm{s}}^{(k)} && \text{(2b)}\\
& && -\sigma^{-}\mathbf{a} \le \mathbf{q}^{(k)} \le \sigma^{+}\mathbf{a} && \text{(2c)}\\
& && -\tilde{\mathbf{f}}_{\mathrm{s}} \le \mathbf{f}_{\mathrm{s}}^{(k)} \le \tilde{\mathbf{f}}_{\mathrm{s}} && \text{(2d)}\\
& && \mathbf{a} \ge \mathbf{0}, && \text{(2e)}
\end{aligned}$$

where, $\mathbf{f}_{\mathrm{s}}^{(k)} = [f_{\mathrm{s},1}^{k,x}, f_{\mathrm{s},1}^{k,y}, f_{\mathrm{s},2}^{k,x}, ..., f_{\mathrm{s,n}}^{k,y}]^{\mathrm{T}}$ are slack force variables for each node in load-case k, and $\tilde{\mathbf{f}}_{\mathrm{s}} = [\tilde{f}_{\mathrm{s},1}^{x}, \tilde{f}_{\mathrm{s},1}^{y}, \tilde{f}_{\mathrm{s},2}^{x}, ..., \tilde{f}_{\mathrm{s,n}}^{y}]^{\mathrm{T}}$ are the maximum absolute values of the slack forces across all load-cases, $\tilde{V}$ is the penalized structural volume, obtained using the sum of slack forces $\tilde{\mathbf{f}}_{\mathrm{s}}$, and μ_s is a large multiplier value, taken as $20V_0$, where V_0 is the volume after layout optimization. In a successful validation, the objective value from (2) will be very close to the objective value from (1), and the slack forces will not be used. However, if key elements have been removed by the filter, then the slack forces may be the only way constraint (2b) can be satisfied, leading to a substantial increase in the objective function. Alternatively, other load paths may be able to transmit the load previously carried by the removed key elements, but these will by definition generally be sub-optimal, and therefore will also lead to an increase in the objective value. A small tolerance of 1% increase in volume is permitted in this stage. Due to this validation requirement, the filtered structures may still contain some thin, but structurally important, members; Fig. 5a shows an example of such a structure.

2.3 Geometry optimization

Layout optimization can be used to identify the globally minimum volume structure comprising joints that lie on the original nodal grid. However, lower volume solutions can usually be obtained if the joints are allowed to migrate to other positions. Thus, a geometry optimization step can be used to improve the layout optimization solution by adding joint positions as optimization variables (see He and Gilbert 2015).

The geometry optimization formulation is very similar to (2) except that the geometrical coefficients, such as element lengths and direction cosines, are now calculated in the optimization process from new nodal position variables. Thus, objective function (2a) can now be written as:

$$\underset{\mathbf{a},\mathbf{q}^{(k)},\mathbf{f}_{\mathrm{s}}^{(k)},\tilde{\mathbf{f}}_{\mathrm{s}},\mathbf{x},\mathbf{y}}{\text{minimize}} \quad \tilde{V} = \mathbf{l}^{\mathrm{T}}\mathbf{a} + \mu_{\mathrm{s}} \sum_{j=1}^{n} (\tilde{f}_{\mathrm{s},j}^{x} + \tilde{f}_{\mathrm{s},j}^{y}). \qquad (3)$$

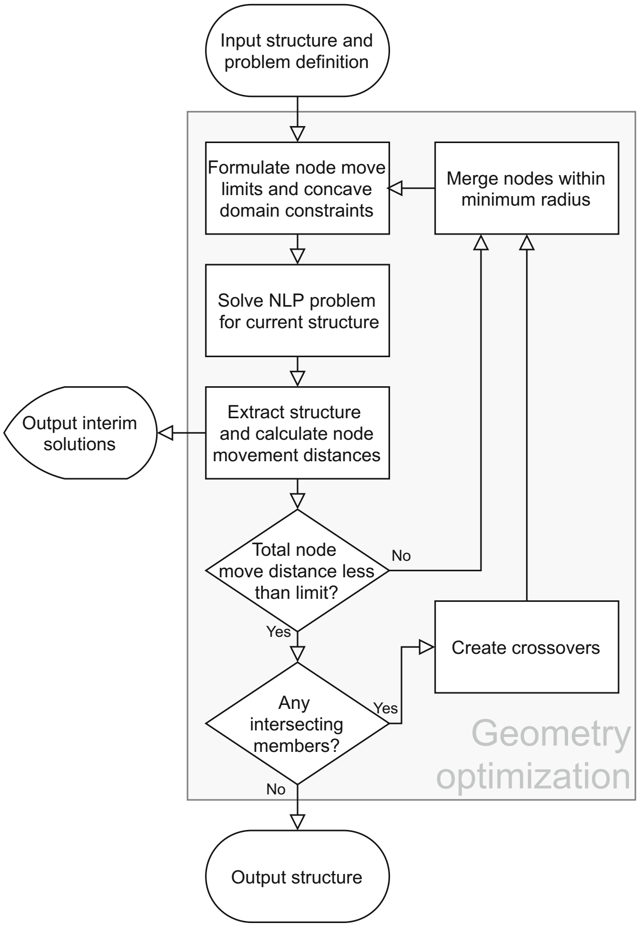

Input structure and problem definition
Formulate node move limits and concave domain constraints
Merge nodes within minimum radius
Solve NLP problem for current structure
Output interim solutions
Extract structure and calculate node movement distances
Total node move distance less than limit?
No
Yes
Create crossovers
Any intersecting members?
Yes
No
Geometry optimization
Output structure

Fig. 4 Geometry optimization procedure

where $\mathbf{x} = [x_1, x_2, ..., x_n]^{\mathrm{T}}$ and $\mathbf{y} = [y_1, y_2, ..., y_n]^{\mathrm{T}}$ are vectors containing x and y nodal positions, respectively. Note that the problem is no longer convex, and a globally optimum solution cannot be guaranteed. However, the use of a layout optimization solution as a starting point tends to ensure that high quality solutions can be obtained.

Once the nodes are permitted to move it is important to ensure that the extents of the design domain are respected. For a convex domain, this simply involves adding constraints to ensure that each node does not move outside the domain. However, when concave domains are involved this becomes more complex. Here, the approach described by He et al. (2019a) is used.

As the problem is now non-linear and non-convex, numerical instabilities may be encountered. To counter this, limits are placed on the distance moved by each node. This greatly improves stability, but also necessitates an iterative process so that nodes can gradually migrate towards their optimal location. This iterative process is stopped once the sum of the distances moved by all nodes is deemed sufficiently small.

The resulting structure will often contain elements that intersect, especially in the case of 2D problems. However, in practice any intersection point would usually be viewed as a joint, whose location therefore needs to be optimized. Thus, new nodes are created at intersection points, and the geometry optimization process is performed on the resulting new structure. The overall algorithm for this procedure is shown in Fig. 4.

A further complication is that nodes may migrate towards each other. Thus any nodes that become too close to each other are merged between iterations.

The overall effect of the geometry optimization process is to rationalize the structures obtained via layout optimization, often removing many thin members from the structure, e.g. as is evident in Fig. 5b.

2.4 Structural simplification via Heaviside projection

Even if the filtering and geometry optimization procedures detailed in the previous sections are applied, the solutions obtained will often still be quite complex in form. This can make them appear impractical to designers, hampering uptake. Here a simplification method is proposed that allows the total number of members in a given structure to be reduced, thereby reducing overall structural complexity (Fig. 6). In the interests of computational efficiency this is

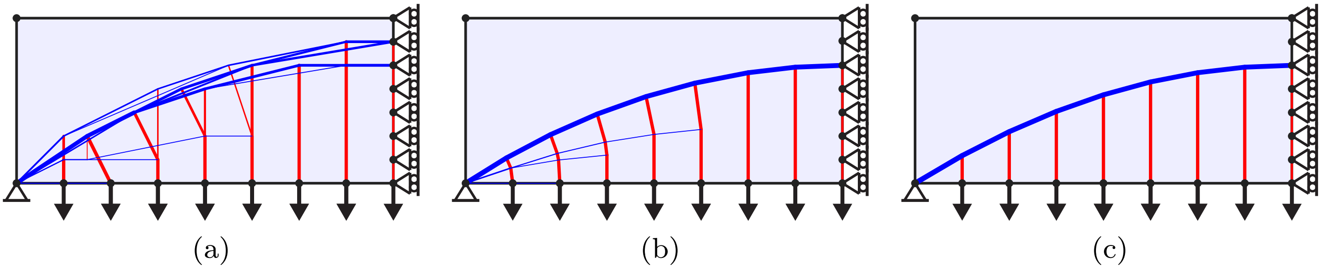


Fig. 5 LayOpt stages: (a) layout optimization solution (post-filtering volume = 66.485); (b) geometry optimization solution (volume = 64.639); (c) simplified solution (20% volume increase limit, volume = 65.123). Stress limit in compression = 6 units, limit in tension = 1 unit. Point loads have unit magnitude except the right-most point load, which has a magnitude of 0.5. Domain has dimensions of 16 × 7 units, with nodes positioned 1 unit apart

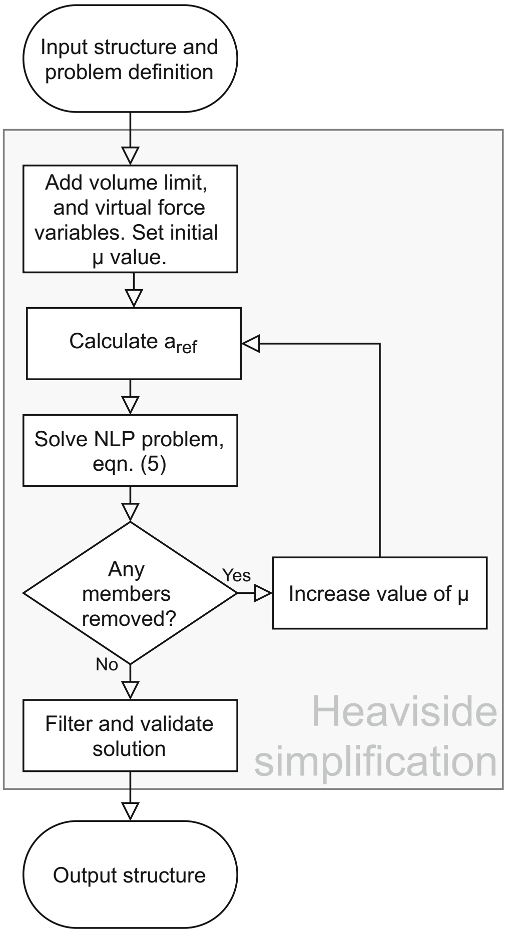

Input structure and problem definition
Add volume limit, and virtual force variables. Set initial μ value.
Calculate a_ref
Solve NLP problem, eqn. (5)
Any members removed?
Yes
Increase value of μ
No
Filter and validate solution
Heaviside simplification
Output structure

Fig. 6 Automatic simplification procedure

achieved by handling integer member 'existence' variables x via a smooth Heaviside projection $H(x)$ (Guest et al. 2004). Here the selected projection function is:

$$H(x) = \coth(\mu)\tanh(\mu x), \tag{4}$$

where, μ is a predefined projection factor that determines the accuracy of the approximation (see Fig. 7). Using this Heaviside projection, the 'existence' of a given member i can now be expressed by its cross-sectional area a_i. Let a_{ref} denote a reference member area such that the following is satisfied:

$$H(a_i/a_{\text{ref}}) \approx \begin{cases} 0, & \text{if } a_i \ll a_{\text{ref}} \\ 1, & \text{otherwise.} \end{cases} \tag{5}$$

Since reducing complexity is likely to lead to higher volume structures, it is convenient to manage the trade-off between complexity and volume via the following optimization problem (after He et al. 2018):

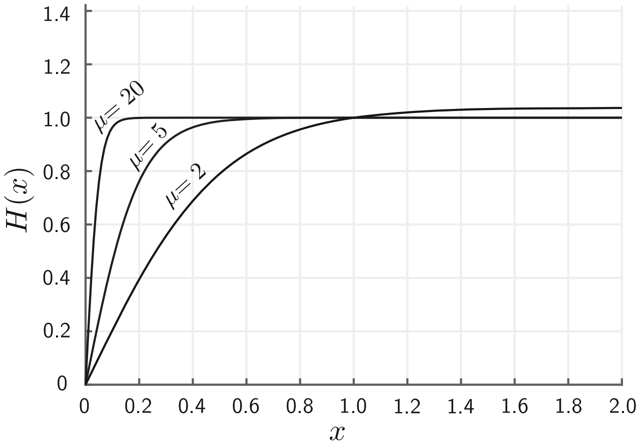


Fig. 7 Form of Heaviside function (5) employed by the simplification procedure

$$\begin{aligned}
\underset{\mathbf{a},\mathbf{q}^{(k)},\mathbf{f}_s^{(k)},\tilde{\mathbf{f}}_s,\mathbf{x},\mathbf{y}}{\text{minimize}} \quad & \Phi_{\text{M}} = \sum_{i=1}^{m} H(a_i/a_{\text{ref}}) && \text{(6a)}\\
\text{subject to} \quad & \mathbf{B}\mathbf{q}^{(k)} = \mathbf{f}^{(k)} + \mathbf{f}_s^{(k)} && \text{(6b)}\\
& -\sigma^{-}\mathbf{a} \le \mathbf{q}^{(k)} \le \sigma^{+}\mathbf{a} && \text{(6c)}\\
& -\tilde{\mathbf{f}}_s \le \mathbf{f}_s^{(k)} \le \tilde{\mathbf{f}}_s && \text{(6d)}\\
& \tilde{V} \le (1+\epsilon)V_{\text{ref}} && \text{(6e)}\\
& \mathbf{a} \ge \mathbf{0}, && \text{(6f)}
\end{aligned}$$

where the optimization objective is to minimize the total number of members Φ_{M}. Also, constraint (6e) defines the acceptable trade-off in structural volume, where ϵ is a specified allowable volume increase ratio and V_{ref} is a reference structural volume (taken as the volume prior to the start of the simplification step), and where a_{ref} is here taken as $a_{\max}$.

Problem (6) is solved using various values of the projection factor μ, progressively increasing this from 2, 5, 10 up to 20. Initially a small value of μ is used, which provides rich gradient information that can be used to minimize the objective function in the optimization; larger values improve the speed of convergence. In each case the goal is to remove members with small cross-sectional areas. The process terminates if no member can be removed or if μ has reached 20 (where the Heaviside projection closely resembles the integer values). Since this is a heuristic process, there is no guarantee that the simplification process will always be successful, and the solutions obtained will generally be only locally optimal. On the other hand, useful solutions can often be obtained in practice, and the speed of the process renders it suitable for use in the interactive web-app described herein; a sample simplified design is shown in Fig. 5c.

The process described is effective at reducing the total number of members in a structure. However, since the volume of the resulting structures need only satisfy the volume increase limit (6e), there is no requirement or expectation that the volume will be a minimum, even among structures with the same layout. Therefore, it is useful to undertake a further geometry optimization step (Section 2.3) after the simplification, which should be preceded by a filtering and validation step (Section 2.2). This does, however, mean that the resulting optimized structure may often have a markedly lower volume than implied by the specified volume increase ratio. This is demonstrated in Figs. 8 and 9, which show how the specified allowable volume increase ratio ϵ influences the layout of the generated structures.

Note that although the problem shown in Fig. 8 is symmetrical, several of the simplified solutions are asymmetric. Previous research has demonstrated that the optimal solution may be asymmetric when various practical constraints are considered, such as when discrete bar areas are stipulated (Stolpe 2016), or when limits are placed on the number of joints (Fairclough and Gilbert 2020). Thus, although the simplification method used here does not guarantee that the solutions will be globally optimal, asymmetric solutions are to be expected.

If symmetrical solutions are desired, for aesthetic or other reasons, this may be enforced using additional constraints, as outlined in Fairclough and Gilbert (2020). However, this may then result in solutions with a larger volume and/or a higher level of complexity.

3 Software architecture

This section describes in more detail the main features of the web-app and how these have been implemented. Readers who are more interested in the results that can be obtained using the web-app can proceed directly to Section 4.

The software comprises three main elements. Firstly, a series of client-side Javascript/HTML5 files which form the user interface, and allow the user to construct the desired problem. Secondly, a cloud-hosted server, which serves both the static user interface pages and dynamic pages containing solutions to the user-specified problems. Finally, the actual solving of the optimization problems occurs using a serverless compute platform. The interaction between these components is detailed in Fig. 10.

3.1 User interface

The user interface centres around a HTML5 canvas element (Fulton and Fulton 2013) which displays the problem setup and the solutions. The canvas is also the main input area, where the user can add and remove features from the problem. When the user interacts using a mouse input, a visual change in size assists the user in editing the desired point. When a touch input is detected, larger invisible touch targets are added to points and symbols for ease of selection.

The main canvas is accompanied by a ribbon bar containing various tools to allow the user to add elements (e.g. loads and supports) to the problem, as well as to move, rotate and delete problem elements. Implementation of the tools is simplified by use of the paper.js library (Lehni and Puckey 2011) which provides tools to assist drawing on the canvas and interacting with it.

The menu bars are laid out using responsive design principles (Marcotte 2011) to allow use on any size of screen, with the layout automatically collapsing to multiple rows, or drop-down menus, at small screen sizes, as shown in Fig. 1. Thus, the web-app is usable on a range of platforms from desktop computers to smartphones, and should remain compatible with new devices and platforms as long as they support a standard web browser.

All user interface actions are handled entirely client-side to improve responsiveness and to provide a smoother user experience. This permits features such as the snapping of points to the nodal grid.

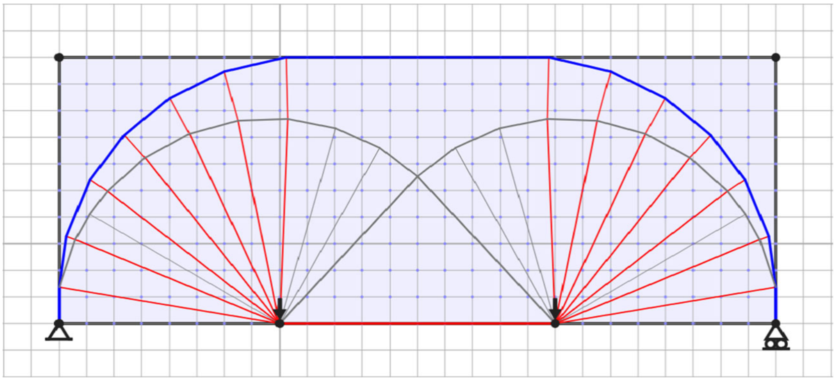


Fig. 8 Simple two load-case problem: problem definition and initial unsimplified solution (left and right hand point loads active only in load-cases 1 and 2 respectively; here and henceforth grey lines represent members that are required to carry both tensile and compressive forces, dependant upon load-case)

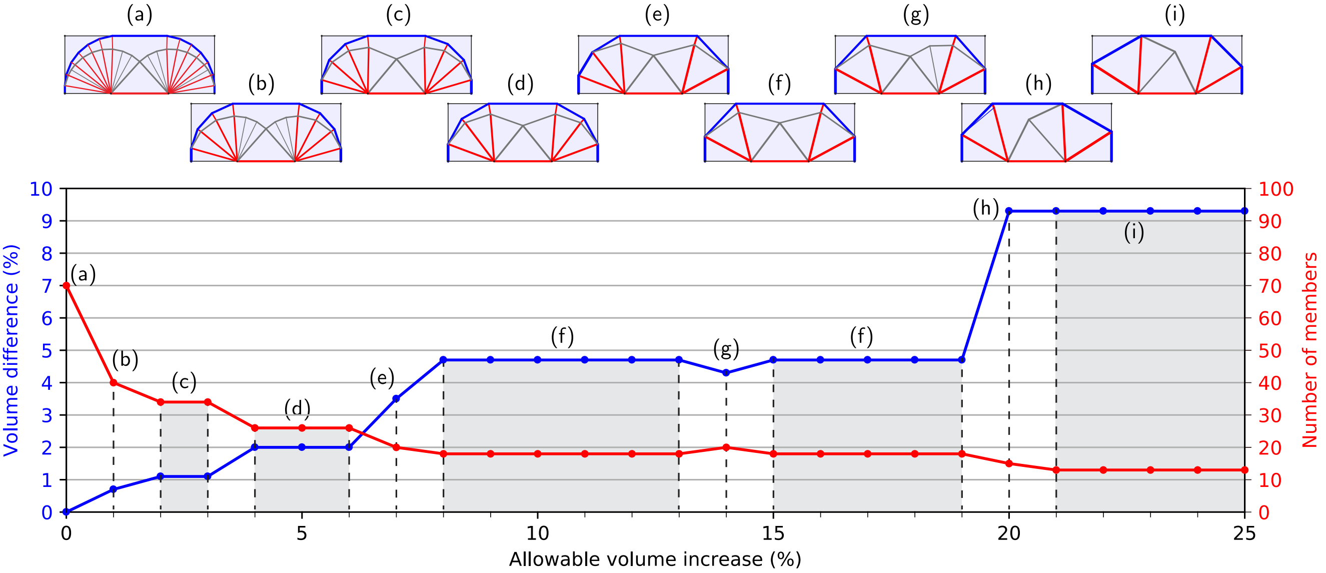


Fig. 9 Simple two load-case problem: effect of varying the allowable volume increase limit. Shaded regions indicate when the same solution is obtained across a range of allowable volume increase values

Solutions are displayed on screen with members coloured according to the forces they carry: members that carry only tensile forces are coloured red, members carrying only compressive forces are coloured blue, and members carrying both tensile and compressive forces in different load-cases are coloured grey. The thickness of the line representing each member is proportional to the square root of its area, i.e. all members are assumed to be constructed from the same cross-sectional shape that is then scaled by a fixed factor to the desired size. The volume of the current and benchmark solutions are also displayed on the user interface; this corresponds to the expression in (1a).

The client-side scripts also handle export of the problem to the user's computer. This can be in an XML format, which can be re-loaded by the app at a later date. Alternatively, downloads in a range of image formats are available. Static images showing the final structure may be downloaded in raster (.png) or vector (.svg) formats. An animation, which also shows all of the interim solutions generated during the member adding and geometry optimization phases, is also available (.gif). For all of the image download types, the view can be customized by adding or removing details such as the nodal grid or the design domain.

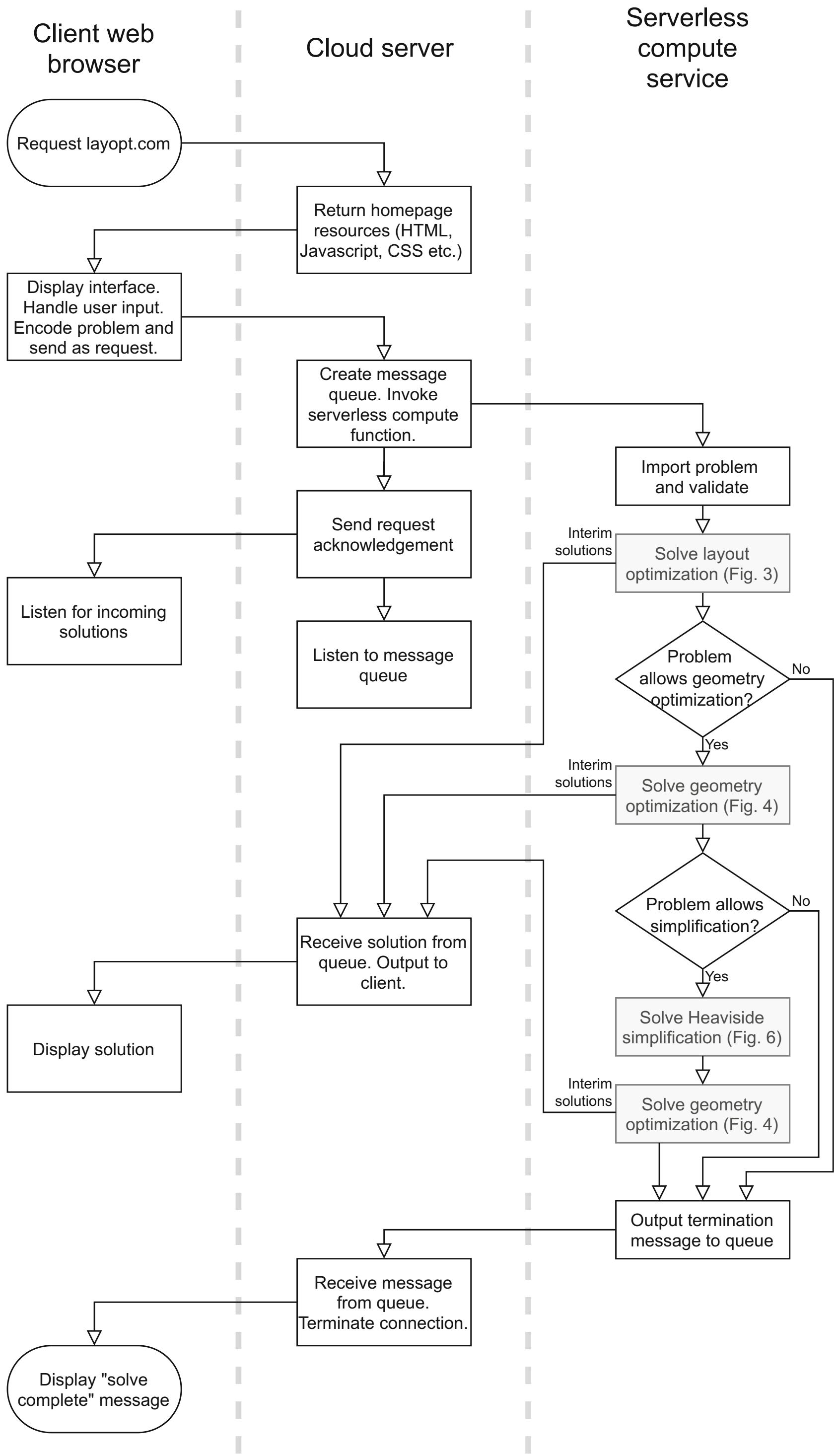
3.2 Cloud Server

To keep the user fully informed with progress towards a solution, and to provide increased educational value, the web-app displays solutions from intermediate iterations in the member adding (see Section 2.1) and geometry optimization (see Section 2.3) procedures. These are displayed in real time, as they are calculated, and can also be exported as an animation. Each problem that the user sends is handled with one AJAX (Asynchronous Javascript And XML) request-response pair (Brinzarea-Iamandi et al. 2009). This avoids the need to keep track of individual sessions on the server, as each problem request contains all the input data required to solve it. This is known as a RESTful, or stateless architecture (Richardson and Amundsen 2013).

To provide all intermediate solutions as they become available, the response is chunked. This is implemented on the server by using a PHP script (Brinzarea-Iamandi et al. 2009) to process messages from the serverless function, allowing flushing of the output buffers at appropriate points. This is the major hurdle in a fully serverless implementation.

The server interface also provides the ability to save and load solutions from the global database. The current problem is saved to the global database when 'Get sharable link' is clicked in the menu, and the generated ID for that problem is returned to the user. The ID of the problem is then added as a query string to the URL to produce a link that will allow anyone to access that problem. When the app is opened in this way, all functionality is maintained, so the new user can modify the opened problem. To save their modifications, they must generate a new sharable link, and cannot overwrite the existing saved file. This feature allows easy sharing of solutions on social media, with co-workers or in a classroom setting.

Fig. 10 Overall program flow and interaction of software components. Grey components refer to the procedures outlined in Figs. 3, 4 and 6


Client web browser
Cloud server
Serverless compute service
Request layopt.com
Return homepage resources (HTML, Javascript, CSS etc.)
Display interface. Handle user input. Encode problem and send as request.
Create message queue. Invoke serverless compute function.
Import problem and validate
Send request acknowledgement
Interim solutions
Solve layout optimization (Fig. 3)
Listen for incoming solutions
Listen to message queue
Problem allows geometry optimization?
No
Yes
Interim solutions
Solve geometry optimization (Fig. 4)
Problem allows simplification?
No
Receive solution from queue. Output to client.
Yes
Solve Heaviside simplification (Fig. 6)
Display solution
Interim solutions
Solve geometry optimization (Fig. 4)
Output termination message to queue
Receive message from queue. Terminate connection.
Display "solve complete" message

3.3 Serverless processing

To allow for maximum scalability, the main processing of the problems is implemented using serverless technology, namely the Lambda service offered by Amazon Web Services (AWS). This allows functions to be run without pre-provisioning the computational resources (Chapin and Roberts 2020). Whilst there is a small initial time overhead, this is comparatively small for all but the smallest truss optimization problems. Using this serverless approach, it is possible to avoid the substantial extra time that would be needed to dynamically provision additional computational resources to handle peaks in usage.

Note that there are limits on both source code size and duration of execution with most serverless platforms; however, these are generally sufficient for problems which would be solved in a while-you-wait scenario such as a web-app (at the time of writing, the execution time limit for the AWS Lambda service is 15 min).

This function has been programmed in the C++ language, using the Lambda runtime API. This allows the use of efficient third party solvers, namely MOSEK (MOSEK ApS 2019) for the solution of the linear layout optimization problem, and IPOPT (Wächter and Biegler 2006) for the non-linear geometry optimization and simplification problems. The use of C++ also allowed reuse of the codebase previously employed by He et al. (2019a, 2021a, 2021b), and also means that future versions of LayOpt can potentially readily take advantage of additional enhancements in that codebase. Note that the user interfaces described in He et al. (2019a), He et al. (2021a), and He et al. (2021b) are more powerful and provide access to more advanced features (e.g. solution of 3D design problems and CAD geometry design domain import), but have a considerably steeper learning curve for users than the LayOpt web-app. Links to these more advanced software tools can be found on the 'About' page of the LayOpt web-app.

4 Examples

4.1 Simple examples

A number of simple example problems that illustrate key features of the LayOpt web-app are first considered. These also allow a range of structural insights to be drawn.

4.1.1 Variation of support conditions

The first example, shown in Fig. 11, demonstrates the influence of boundary conditions on the identified optimal structures.

It is evident that as the level of support restraint is increased, lower volumes can be achieved. It is also evident that significantly simpler structures can be obtained using the Heaviside projection method, and that these structures require relatively little more material than the corresponding unsimplified designs.

In terms of the specific structures obtained, the structure shown in Fig. 11b may not seem intuitive at first sight. However, it can perhaps be better understood by considering a beam with the same loading and support conditions. The bending moment diagram for such a beam would have sagging moments over the majority of its length, but with a hogging moment in the vicinity of the fixed support. This is reflected in the solution obtained by the presence of two separate structures in Fig. 11b; the left hand structure has a typical cantilever form, with members in tension along

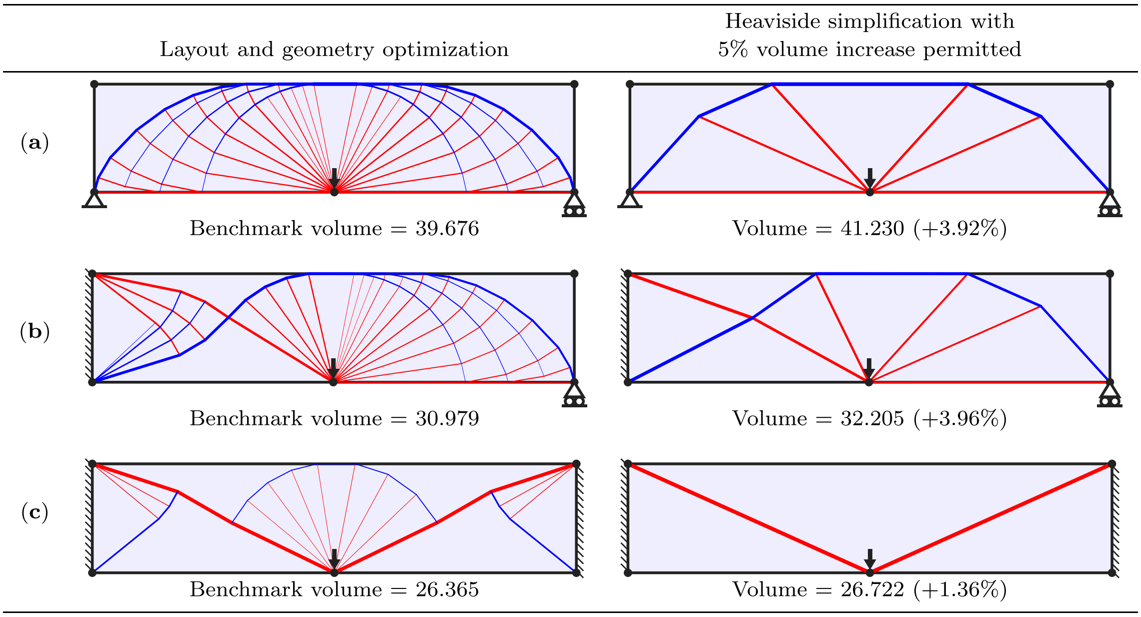

Layout and geometry optimization
Heaviside simplification with
5% volume increase permitted
(a)
Benchmark volume = 39.676
Volume = 41.230 (+3.92%)
(b)
Benchmark volume = 30.979
Volume = 32.205 (+3.96%)
(c)
Benchmark volume = 26.365
Volume = 26.722 (+1.36%)

Fig. 11 Midspan point load problem: influence of varying support conditions (a), (b) and (c) on form of optimized structures (20 × 5 unit design domain; nodal grid spacing = 0.5; unit applied load and limiting stresses)

the top edge, i.e. resisting a hogging moment, whilst the right hand structure resembles the spanning structure in Fig. 11a, with compressive elements on the top edge, i.e resisting a sagging moment. These two structures meet at a single point, coinciding with the point of zero moment in the bending moment diagram.

A similar effect can be observed in Fig. 11c, where the principal tensile member is partially supported by compressive members emanating from the supports and by a wheel structure at midspan. Again, this approximately reflects the bending moment diagram of the corresponding beam. However, the depth of the beam here is such that these additional elements provide only a slight volume saving, and the Heaviside simplification produces a structure consisting of only two tensile members.

4.1.2 Variation of simplification tolerance

The second example involves a classical cantilever problem, shown in Fig. 12. The first row shows the solution after layout and geometry optimization, which resembles the solution given by Chan (1960). However, the presence of ribs of members that are incomplete introduces irregularity to the structure. These incomplete ribs offer only minimal advantage in terms of volume, and can therefore be removed by specifying a small permitted volume increase in the Heaviside simplification phase. In this example a 0.1% permitted volume increase removes all incomplete ribs from the solution.

When larger volume increases are permitted in the simplification stage, more significant changes to the solutions obtained are evident. The objective of the non-linear optimization problem solved in the simplification stage is to reduce the number of members; however, the simplified solutions also demonstrate improvements in a number of other measures that would affect the real-world difficulty of fabricating a design, such as the number of joints and the presence of small inclined angles between adjacent members.

As previously noted, asymmetric solutions are often obtained, even when considering symmetrical or antisymmetrical problems, as seen in Fig. 12. When an asymmetric solution is obtained, it can be observed to frequently consist of parts of two neighbouring symmetrical solutions. For example in Fig. 12, the 2% permitted volume increase solution has characteristics of both the 1% and 3% solutions. Whilst this may not seem immediately intuitive, it means that a wider range of solutions can be identified. Additionally, the speed of the proposed simplification approach means that it is feasible to try a wide range of permitted volume increases in order to obtain the most visually satisfying result.

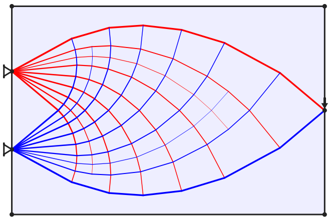
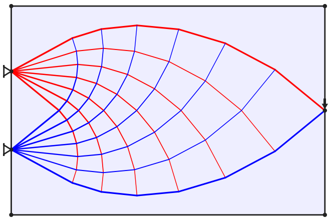
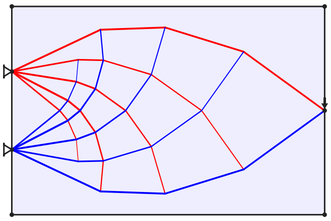
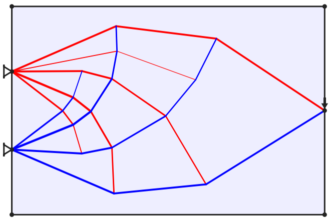
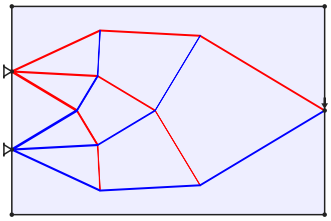
Permitted Vol. Increase	Volume	Solution
0%	119.2	
0.1%	119.34 +0.08%	
1%	120.3 +0.91%	
2%	121.5 +1.91%	
3%	122.4 +2.63%	

Fig. 12 Cantilever problem: influence of permitted volume increase on form of optimized structures generated using Heaviside simplification procedure (24 × 16 unit domain; unit applied load and limiting stresses)

4.1.3 Problems with non-convex design domains

Figure 13 shows variants of the problem presented in Fig. 12 though now with a hole placed in the centre of the design domain, rendering this non-convex. When the design domain is non-convex, it is necessary to remove elements from the ground structure if they protrude outside the domain. In the layout optimization stage this can be easily calculated a priori, whilst in the geometry optimization and

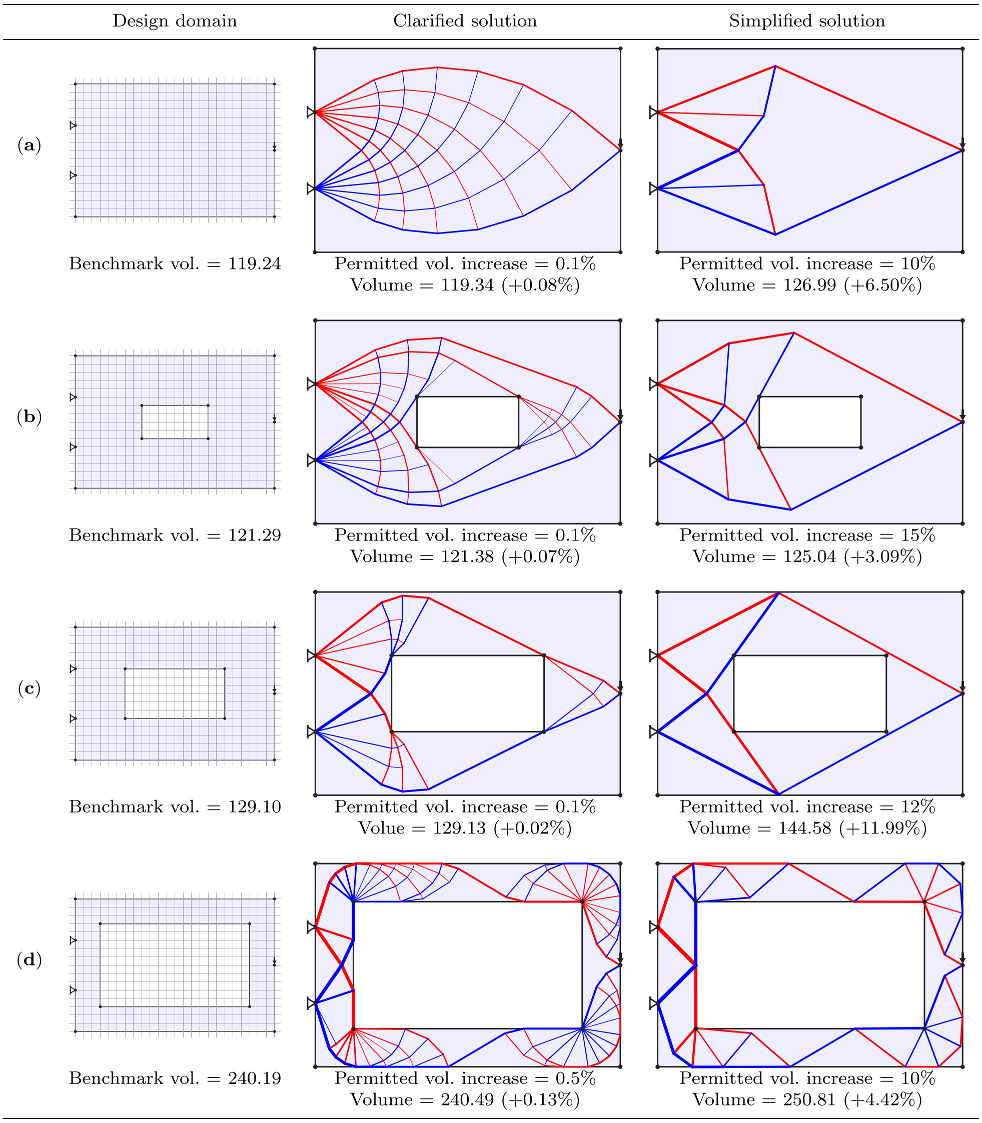


Fig. 13 Cantilever problem: influence of hole size on form of optimized structures (unit applied load and limiting stresses; unit nodal grid spacing with dimensions of design domain shown in first column)

simplification stages, additional constraints must be added, using the method described by He et al. (2019a).

Figure 13 shows solutions for a range of hole sizes. The first solutions shown for each problem have a small permitted volume increase (0.1–0.5%) during the simplification stage, chosen to eliminate incomplete ribs and to provide a clearer layout. The second solution for each problem includes a more significant level of simplification, corresponding to a permitted volume increase of 10–15%. Again the exact value for each problem has been chosen to produce visually satisfying results, for example by favouring symmetrical structures or minimizing the number of incomplete ribs.

4.2 Educational examples

In this section examples are presented that demonstrate a range of principles of optimal structures. These could provide suitable starting points for using the LayOpt web-app as a teaching aid on a taught module, with the concepts involved illustrated in an engaging and interactive way.

4.2.1 Statically determinate and indeterminate problems

In Michell’s seminal 1904 paper on optimal structures (Michell 1904), the half-wheel structure problem shown in the first column of Fig. 14 is considered. Michell gives the volume of this structure as

$$\text{Volume} = Fa\frac{\pi}{2}\left(\frac{1}{\sigma_C}+\frac{1}{\sigma_T}\right) \tag{7}$$

where F is the magnitude of the force (here taken as 1), a is the distance from a support to the load (here taken as 8), and σ_T and σ_C are the limiting tension and compression stresses respectively. Thus, for the first structure shown in Fig. 14b, the minimum volume would be $8\pi \approx 25.13$. The numerical techniques embedded in the LayOpt web-app provide upper bound solutions, which closely approximate the corresponding analytical solutions when high nodal resolutions are employed. An interesting exercise for students could include observing the volume as the 'grid refinement' slider is changed, and potentially extrapolating the results to infer the solution with infinitely many nodes, using the approach described by Darwich et al. (2010).

The $\frac{1}{\sigma_C}+\frac{1}{\sigma_T}$ multiplier in (7) is common to all of the solutions presented by Michell, and originates from his use of Maxwell (1872)'s earlier theorem. It can be seen from the first column of Fig. 14 that this proportionality holds for the sub-optimal layout depicted. Furthermore, this implies that the layout of the optimal structure does not change with different stress ratios, since only member sizes are changed. This can also be observed in the first column of Fig. 14.

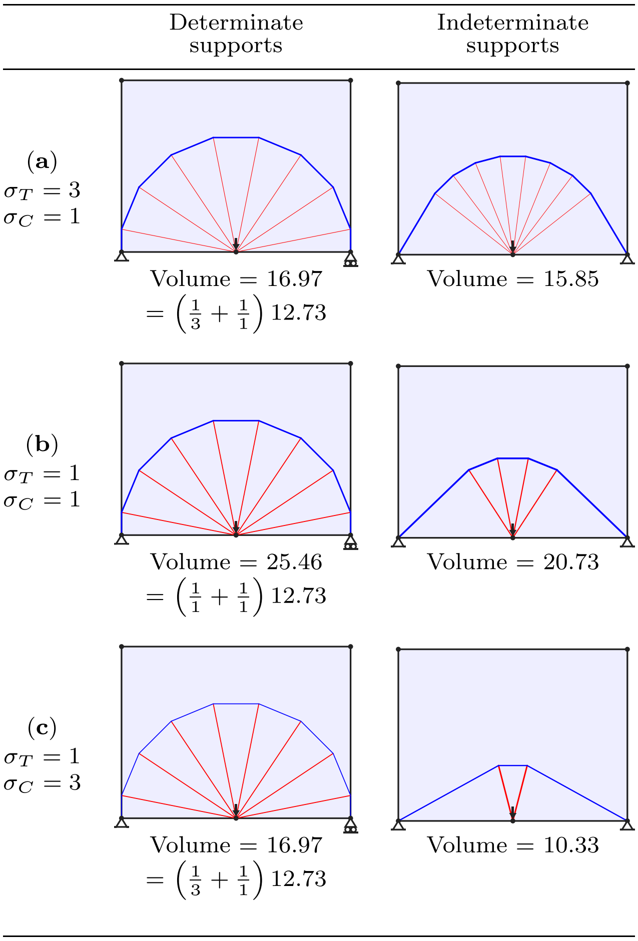


Fig. 14 Midspan point load problem: influence of changing tensile and compressive stress limits on form of optimized structures for statically determinate and indeterminate problems (unit applied load)

However, as outlined by Rozvany (1996), Michell and Maxwell's theories are only valid for support conditions which are statically determinate. For statically indeterminate problems, the criteria given by Hemp (1973) must be used instead.

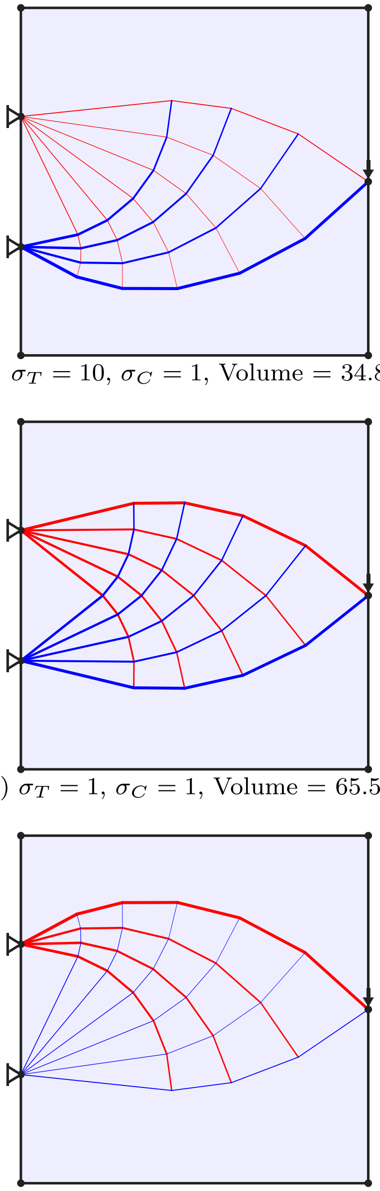
The second column of Fig. 14 shows a variant of the problem where both supports are fixed pins, rendering the problem statically indeterminate. It is evident that the layout of the optimal solution now changes as the limiting stresses are changed. These different layouts also generate different reaction forces (the direction of the resultant reaction force will coincide with that of the single bar connected to each support). Finally, it is evident that there is no longer a simple relationship between the volumes of the different solutions.

4.2.2 Near-optimal structures

Figure 15 shows solutions for a simple cantilever problem for a range of limiting tensile and compressive stress values. This problem has been widely studied in the literature since it was first studied by Chan (1960). In contrast to the circular/radial members present in Fig. 14, the optimal cantilever layouts involve tensile and compressive members that follow more complex curved trajectories, but which remain (near-)orthogonal to each other.

As the numerical methods of the LayOpt app cannot obtain true truss-like continua, the tension and compression members can generally only be near-orthogonal. Similar discretised structures have been considered previously, for example by Prager (1978) and Mazurek et al. (2011), where regular meshes comprising triangular and quadrilateral cells have been constructed. Similar regular meshes can often be identified using the LayOpt app, especially if a small level of simplification is used to remove incomplete and branching ribs, as has been done in Fig. 15 (see also Fig. 12).

In such a structure, quadrilateral cells are always cyclic (i.e. opposite angles sum to 180°). LayOpt's image export feature allows solutions to be exported to a graphics software package so this could potentially be checked as part of an exercise. When the limiting stresses in tension and compression are equal, two of the angles (the top and bottom in Fig. 15) in each cell are exactly equal to 90°, whatever the discretization level. This can be more easily seen in the structures with fewer members, e.g. see the 3% simplification solution shown in Fig. 12. When the allowable stresses are unequal, one of these angles will approach a right angle from above, and the other from


(**a**) $\sigma_T = 10$, $\sigma_C = 1$, Volume $= 34.890$

(**b**) $\sigma_T = 1$, $\sigma_C = 1$, Volume $= 65.532$

(**c**) $\sigma_T = 1$, $\sigma_C = 10$, Volume $= 34.890$

Fig. 15 Cantilever problem: influence of changing tensile and compressive stress limits on form of optimized structures (16 × 16 unit domain; supports 6 units apart; unit applied load; 0.1% volume increase permitted during simplification)

below. Of the remaining two angles, the one in the direction where the coordinate curves diverge (i.e. the right side in Figs. 15 and 12) approaches 90° from below, whilst the remaining angle approaches a right angle from above. LayOpt provides a useful interface for students to explore variants of the problems presented in Fig. 15, helping them to gain an intuitive understanding of the different behaviours possible.

As the problem in Fig. 15 is statically indeterminate, the layout changes with different stress limits. However, the (near-)orthogonality of tension and compression members is still maintained. The differences in layout may perhaps be most clearly seen in the triangle formed by the edge of the domain between the two supports, and the innermost elements of the two fan regions. Deriving the angles of this triangle may prove a useful exercise for students, following a Mohr's circle based approach similar to that presented by Rozvany and Gollub (1990). It is found that the angle at the top support (for the truss-like continuum solution) becomes $\alpha = \arctan\left(\sqrt{\frac{\sigma_C}{\sigma_T}}\right)$, with the angle at the bottom support being $\alpha - 90°$. The numerical results available from the LayOpt app will approach these values (from above) as the resolution is increased.

4.2.3 Multiple loads and multiple load-cases

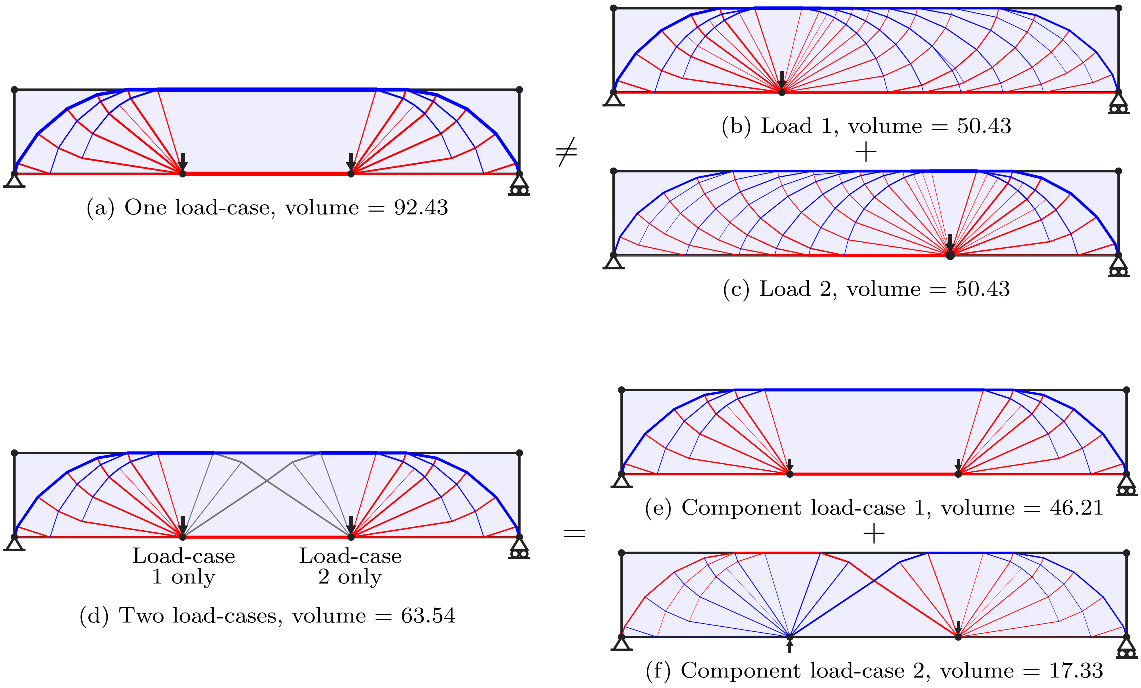
The simply supported spanning structure problems shown in Fig. 16 can be used to illustrate fundamental differences between single and multiple load-case problems. In an educational setting students can be asked to consider how they expect the loads applied to influence the optimal structural forms.

Initially the two loads can be considered entirely separately, with Fig. 16b and c showing the resulting optimal solutions.

If both loads will always be applied at the same time, then these can be considered together as part of a single load-case. This gives the optimal structure shown in Fig. 16a. Note that the volume of the structure shown in Fig. 16a is less than the sum of the volumes of the structures shown in Fig. 16b and c, indicating the benefits that can be accrued by considering the two loads together. However, it can be observed that if the structure shown in Fig. 16a were to be loaded with only one of the loads (or indeed with any unequal combination of these), then it would collapse as it is in a state of unstable equilibrium.

Conversely, suppose that the two loads shown in Fig. 16b and Fig. 16c can only be applied separately. In this case, they can be applied as part of two separate load-cases, giving rise to the optimal structure shown in Fig. 16d. This structure again has a significantly smaller volume than would be obtained by combining the structures shown in Fig. 16b and c.

The form of the structure shown in Fig. 16d may be better understood by considering the superposition principle described by Nagtegaal and Prager (1973). For a problem involving a material with equal limiting tension and compression stress, and supporting external force vectors $\mathbf{f}^{(1)}$ and $\mathbf{f}^{(2)}$ in load-cases 1 and 2 respectively, the solution may be found by superimposing the optimized structures obtained by solving two single load-case problems. However, these single load-case prob-



(a) One load-case, volume = 92.43

(b) Load 1, volume = 50.43

(c) Load 2, volume = 50.43

(d) Two load-cases, volume = 63.54

(e) Component load-case 1, volume = 46.21

(f) Component load-case 2, volume = 17.33

Fig. 16 Two loads problem: (a) optimized structure when both loads are applied in a single load-case, (b) and (c) optimized structures to resist only one of the loads, (d) optimized structure when loads are applied in separate load-cases, (e) and (f) optimized structures for component load-cases of (d), using superposition principle (24 × 4 unit domain; unit loads applied at span/3 from supports; unit limiting stresses; nodal grid spacing = $\frac{2}{3}$ units)

lems are not simply the two load-cases of the original problem; instead they have external force vectors given by:

$$\bar{\mathbf{f}}^{(+)} = \frac{\mathbf{f}^{(1)} + \mathbf{f}^{(2)}}{2} \qquad \bar{\mathbf{f}}^{(-)} = \frac{\mathbf{f}^{(1)} - \mathbf{f}^{(2)}}{2} \tag{8}$$

These are referred to as component load-cases, or ‘sum’ and ‘difference’ load-cases respectively. Figure 16e and f show the component load-cases and their solutions for the two load-case problems considered earlier. Note that these combine, both in form and volume, to give the structure shown in Fig. 16d. The signs of the forces in the members in Fig. 16f are different in each of the two load-cases; thus, these members appear in grey in Fig. 16d.

Note that the form of the structure shown in Fig. 16d is also capable of handling any convex combination of the two loads. This can be easily seen from the component load-case solutions, by combining the forces of Fig. 16e with the forces of Fig. 16f multiplied by an appropriate factor between −1 and 1.

Finally, Fig. 17 shows the result of performing Heaviside simplification on the structure shown in Fig. 16d, using a 20% permitted volume increase. The structure obtained in this case resembles a Warren truss, a commonly adopted design in structural engineering practice. This suggests that the Warren truss provides a reasonably economical design for a spanning problem of this sort. Note that in general the superposition principle cannot be used during the simplification stage and the full multiple load-case problem must instead be solved.

4.3 Structural engineering examples

Finally, examples are presented that show how the LayOpt web-app can be applied to problems of the sort tackled

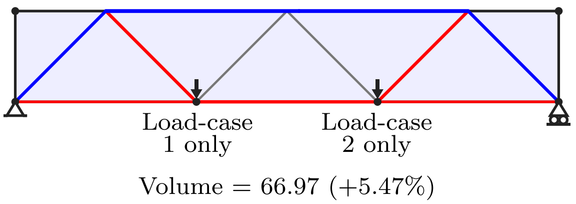


Fig. 17 Two loads problem: simplified form of structure shown in Fig. 16d, which resembles a Warren truss (20% permitted volume increase)

by structural engineers. Although the problems considered here are highly simplified, they suggest that the web-app could be useful when seeking structurally efficient layouts at the initial conceptual design stage. Alternatively, more complex problems can be tackled when the same underlying technology is embedded in parametric CAD software (He et al. 2021a, b).

4.3.1 Portal structure

Figure 18 shows a portal structure designed to enclose a large open space, e.g. for a warehouse or sports centre. The problem involves three load-cases, the first involving only vertical gravity loading (represented here by a midspan vertical load), the second also involving wind type loading (represented here by the addition of a left to right horizontal point load), and the third with the direction of the horizontal load reversed.

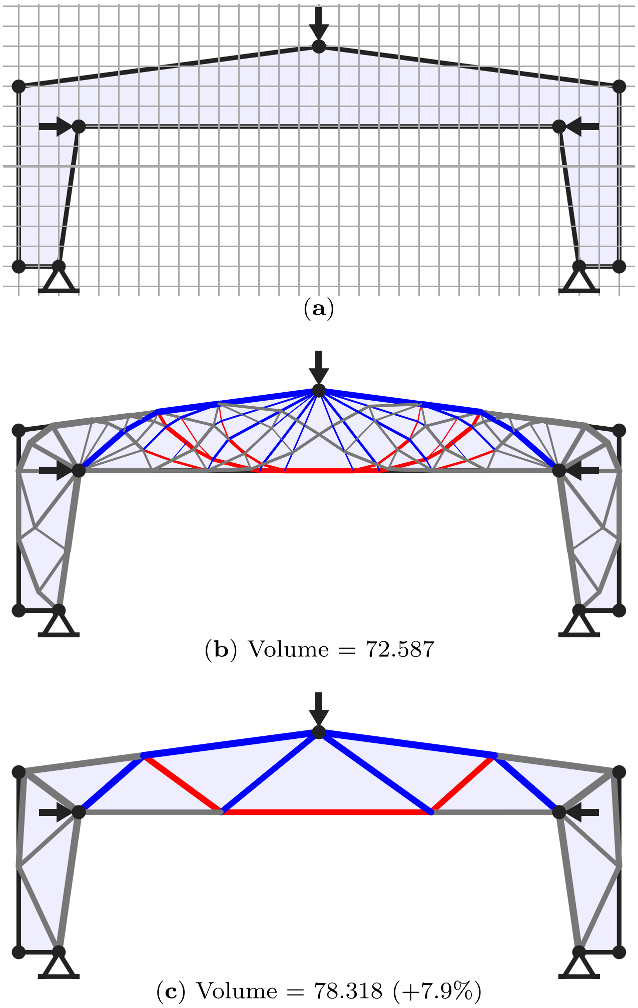


Fig. 18 Portal structure: (a) problem definition, showing design domain and force locations (nodal grid spacing = 0.5 units), (b) optimized structure generated via layout and geometry optimization, (c) simplified structure obtained with a 20% permitted volume increase (three load-cases: vert. only, vert. + left horiz., vert. + right horiz.; all loads have unit magnitude; grey lines represent members carrying both tensile and compressive forces in different load-cases.)

The structure obtained following layout and geometry optimization, Fig. 18b, is clearly quite complex. However, Fig. 18c shows that the Heaviside simplification method can be used to provide a much simpler alternative, with a volume only 7.9% higher in this case.

4.3.2 Multi-storey building

The example shown in Fig. 19 shows a simplified representation of a multi-storey building structure. Initially, the design domain is restricted to a rectangular region, with forces applied in three load-cases, to points lying on the outer envelope of the building as shown in Fig. 19a. As with the portal frame example, the first load-case involves only vertical gravity loading (represented here by vertical loads applied at each storey), the second also involves wind type loading (represented here by the addition of left to right horizontal point loads applied at each storey), and the third load-case is the same but with the directions of the horizontal loads reversed.

Figure 19b shows the outcome of performing layout optimization (only) — a standard cross braced design. However, when geometry optimization is also undertaken, the structure shown in Fig. 19c is obtained. This closely resembles the alternative optimized bracing form identified in previous studies (e.g. Stromberg et al. 2012).

If the design domain is expanded then a slightly lower volume structure involving the use of outrigger members lying outside the envelope of the main building can be obtained (Fig. 19d, e); in this case the load locations and magnitudes have not been changed and the solution shown is after both layout and geometry optimization steps have been performed.

4.3.3 Bridge structure

Figure 20 shows an example of a simplified bridge type problem. The problem is statically indeterminate and therefore, as previously discussed, the optimal form will vary based on the specified limiting stress in tension and compression. Solutions for a range of limiting stress values are shown in Fig. 20b–f. These solutions display similarities with the results obtained by Pichugin et al. (2015), though in that work a problem involving an infinite number of spans was considered.

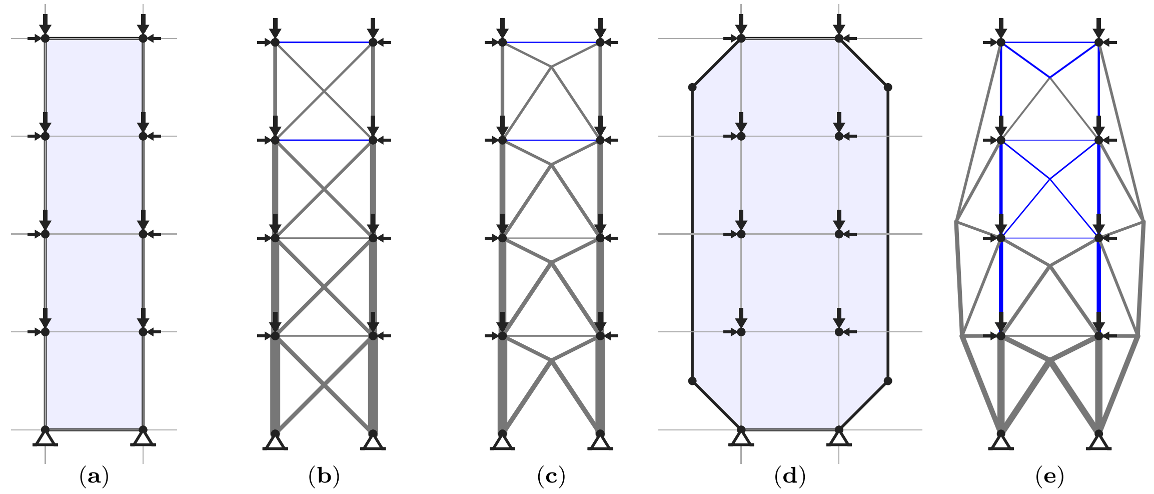


Fig. 19 Multi-storey building: (a) design domain for (b) and (c), (b) layout optimization solution (vol. = 184.0), (**c**) solution after geometry optimization (vol. = 179.0), (d) design domain for (e), (e) new layout and geometry optimization solution (vol. = 164.5) (16 unit high domain, nodal grid spacing = 4 units; vert. and horiz. loads = 1 and 0.5 respectively; three load-cases: vert. only, vert. + left horiz., vert. + right horiz)

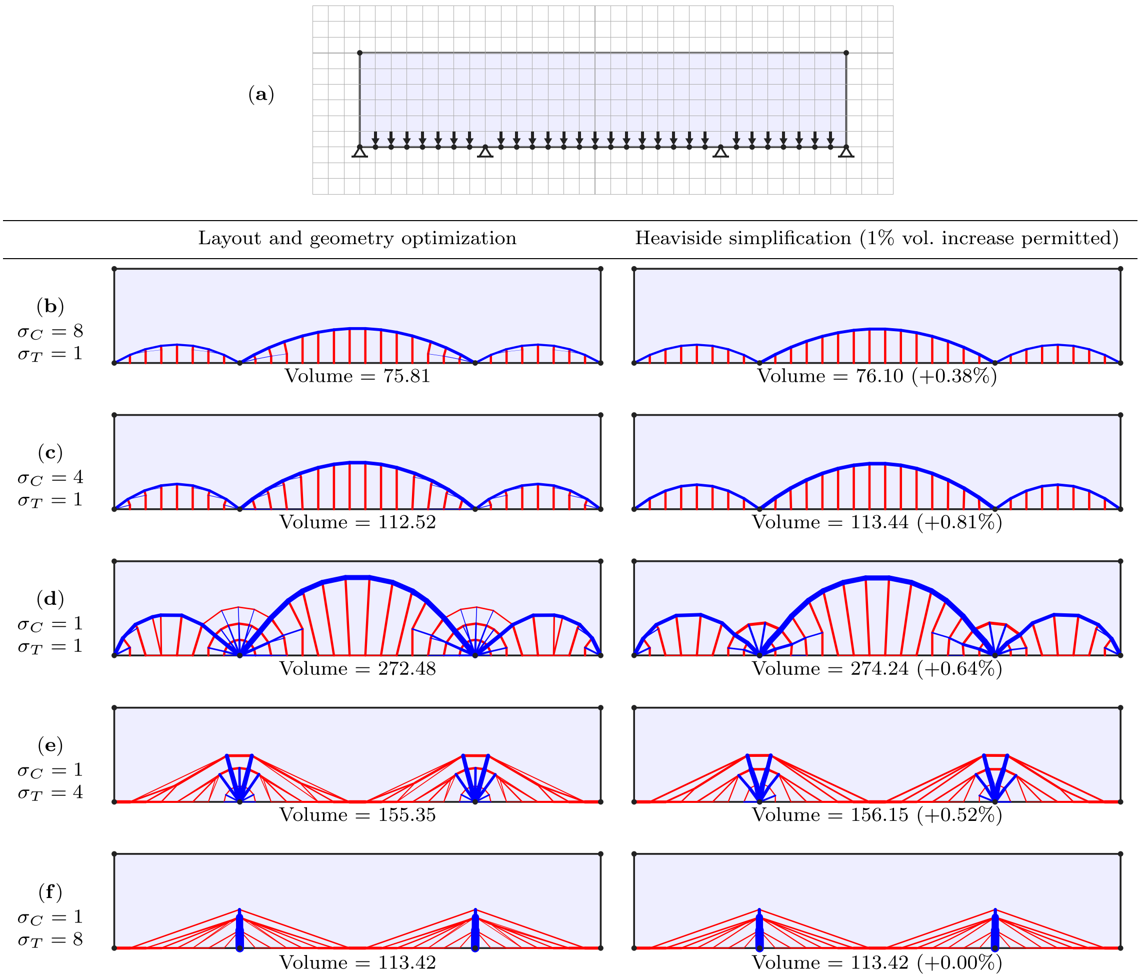

(a)
Layout and geometry optimization
Heaviside simplification (1% vol. increase permitted)
(b) $\sigma_C = 8$ $\sigma_T = 1$
Volume = 75.81
Volume = 76.10 (+0.38%)
(c) $\sigma_C = 4$ $\sigma_T = 1$
Volume = 112.52
Volume = 113.44 (+0.81%)
(d) $\sigma_C = 1$ $\sigma_T = 1$
Volume = 272.48
Volume = 274.24 (+0.64%)
(e) $\sigma_C = 1$ $\sigma_T = 4$
Volume = 155.35
Volume = 156.15 (+0.52%)
(f) $\sigma_C = 1$ $\sigma_T = 8$
Volume = 113.42
Volume = 113.42 (+0.00%)

Fig. 20 Bridge structure: (a) problem specification, showing design domain, loads and supports, (b–f) optimized structures for a range of limiting stress values; the first and second columns respectively show structures before and after Heaviside simplification with a permitted volume increase of 1% (nodal grid spacing = 1; all loads have unit magnitude and are applied in a single load-case)

It is evident that when the limiting compressive stress is larger than the limiting tensile stress, arch type solutions are produced. Conversely, when the limiting tensile stress is much higher, cable stayed forms are produced.

Even a small permitted volume increase at the simplification stage can make the optimized structures obtained clearer. For example in the case of the structures shown in Fig. 20e and f, regions of isotropic strain (so-called Maxwell regions or type S regions) are evident. In these regions, every possible all-tensile set of members is equally optimal, and thus the unsimplified solutions typically comprise multiple, redundant sets of members. However, specifying even an infinitesimal volume increase in the simplification stage allows redundant paths to be eliminated.

5 Conclusions

A new interactive truss layout optimization web-app has been developed for educational use. The web-app first uses layout and geometry optimization methods in sequence to rapidly find optimized truss layouts. However, since the resulting structures can be rather complex in form, a means of reducing their complexity has also been developed. This simplification step involves the use of smooth Heaviside representations of member existence variables, with the user able to manage the trade-off between complexity and structural volume.

In designing the web-app priority has been given to speed and ease of use. Thus no plugins need to be downloaded or installed to use the web-app and the web based interface developed is compatible with a wide range of computing devices (e.g. phones, tablets and desktop PCs). Also, to ensure scaleability and responsiveness, advantage has been taken of modern serverless cloud computing resources.

Finally, the utility of the web-app has been demonstrated via application to a wide range of educational and more practical example problems.

Appendix. Web links for example problems

Table 1 provides web links corresponding to the problems presented in this paper. For cases where several similar problems are presented in one figure, only a single link is given; the related problems may be obtained by altering the parameters as specified in the relevant figure. The links by default show the saved optimal layout; pausing and re-starting solving allows the solution process to be viewed.

Funding This work received financial support from the Engineering and Physical Research Council (EPSRC) under grant reference EP/N023471/1 and a follow up Knowledge Exchange grant that has enabled the web-app to be made available for long-term educational use.

Declarations

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

Replication of results Web links are provided in Table 1 in the Appendix for all the problems presented in the paper.

Table 1 Web links for the problems presented

Figure	Link	Figure	Link
2	www.layopt.com/truss/?prob=0qq2ghd	15	www.layopt.com/truss/?prob=0qq2fmf
5	www.layopt.com/truss/?prob=0qon7r8	16a	www.layopt.com/truss/?prob=0ql1143
8 and 9	www.layopt.com/truss/?prob=0qn6u0m	16b	www.layopt.com/truss/?prob=0qq2g32
11a	www.layopt.com/truss/?prob=0qola34	16c	www.layopt.com/truss/?prob=0qq2g0q
11b	www.layopt.com/truss/?prob=0qola6l	16d	www.layopt.com/truss/?prob=0ql113a
11c	www.layopt.com/truss/?prob=0qolaa4	16e	www.layopt.com/truss/?prob=0qomih1
12 and 13a	www.layopt.com/truss/?prob=0qokci9	16f	www.layopt.com/truss/?prob=0qomiga
13b	www.layopt.com/truss/?prob=0qokcdr	17	www.layopt.com/truss/?prob=0ql111l
13c	www.layopt.com/truss/?prob=0qokcb0	18	www.layopt.com/truss/?prob=0qn8djt
13d	www.layopt.com/truss/?prob=0qokc4b	19a–c	www.layopt.com/truss/?prob=0ql11oa
14, Determinate	www.layopt.com/truss/?prob=0qq2fjb	19d–e	www.layopt.com/truss/?prob=0ql161h
14, Indeterminate	www.layopt.com/truss/?prob=0qq2fk7	20	www.layopt.com/truss/?prob=0qmj9er

References

Aage N (2013) Nobel-Jørgensen M, Andreasen CS, Sigmund O. Interactive topology optimization on hand-held devices. Struct Multidisc Optim 47(1):1–6

Andreassen E, Clausen A, Schevenels M, Lazarov BS, Sigmund O (2011) Efficient topology optimization in MATLAB using 88 lines of code. Struct Multidisc Optim 43(1):1–16

Brinzarea-Iamandi B, Darie C, Hendrix A (2009) AJAX and PHP: Building Modern Web Applications, 2nd edn. Packt

Chan ASL (1960) The design of Michell optimum structures. College of Aeronautics Cranfield, Tech. rep.

Chapin J, Roberts M (2020) Programming AWS Lambda. O'Reilly

Darwich W, Gilbert M, Tyas A (2010) Optimum structure to carry a uniform load between pinned supports. Struct Multidisc Optim 42(1):33–42

Dorn WS, Gomory RE, Greenberg HJ (1964) Automatic design of optimal structures. J Mècanique 3:25–52

Fairclough H, Gilbert M (2020) Layout optimization of simplified trusses using mixed integer linear programming with runtime generation of constraints. Struct Multidisc Optim 61(5):1977–1999

Fulton S, Fulton J (2013) HTML5 Canvas: Native interactivity and animation for the Web, 2nd edn. O'Reilly

Gilbert M, Tyas A (2003) Layout optimization of large-scale pin-jointed frames. Eng Comput 20(8):1044–1064

Graczykowski C, Lewiński T (2020) Applications of Michell's theory in design of high-rise buildings, large-scale roofs and long-span bridges. Computer Assisted Methods in Engineering and Science 27(2-3):133–154

Guest JK, Prévost JH, Belytschko T (2004) Achieving minimum length scale in topology optimization using nodal design variables and projection functions. Int J Num Meth Eng 61(2):238–254

He L, Gilbert M (2015) Rationalization of trusses generated via layout optimization. Struct Multidisc Optim 52(4):677–694

He L, Gilbert M, Shepherd P, Ye J, Koronaki A, Fairclough H, Davison B, Tyas A, Gondzio J, Weldeyesus A (2018) A new conceptual design optimization tool for frame structures. In: Mueller C, Adriaenssens S (eds) Creativity in Structural Design: Proceedings of the IASS Symposium 2018, Boston, USA

He L, Gilbert M, Johnson T, Pritchard T (2019a) Conceptual design of AM components using layout and geometry optimization. Comput Math Appl 78(7):2308–2324

He L, Gilbert M, Song X (2019b) A Python script for adaptive layout optimization of trusses. Struct Multidisc Optim 60(2):835–847

He L, Li Q, Gilbert M, Shepherd P, Rankine C, Pritchard TJ, Reale V (2021a) Optimization-driven conceptual structural design in a parametric modelling environment. (Submitted)

He L, Pritchard T, Maggs J, Gilbert M, Lu H (2021b) Peregrine user manual (v5.0). https://www.limitstate.com/documentation-peregrine

Hemp WS (1973) Optimum structures. Clarendon Press, Oxford

Lehni J, Puckey J (2011) Paper.js. http://paperjs.org/, Accessed 2021-03-12

Marcotte E (2011) Responsive web design. A Book Apart

Martinez P, Marti P, Querin O (2007) Growth method for size, topology, and geometry optimization of truss structures. Struct Multidisc Optim 33(1):13–26

Maxwell JC (1872) On reciprocal figures, frames and diagrams of force. Trans Roy Soc Edinb 21(1)

Mazurek A, Baker WF, Tort C (2011) Geometrical aspects of optimum truss like structures. Struct Multidisc Optim 43(2):231–242

Michell AGM (1904) The limits of economy of material in frame-structures. Phil Mag 8(47):589–597

MOSEK ApS (2019) MOSEK Optimizer API for C 9.1.13. https://docs.mosek.com/9.1/capi/index.html

Nagtegaal J, Prager W (1973) Optimal layout of a truss for alternative loads. Int J of Mech Sci 15(7):583–592

Nguyen TT, Bærentzen JA, Sigmund O, Aage N (2020) Efficient hybrid topology and shape optimization combining implicit and explicit design representations. Struct Multidisc Optim 62(3):1061–1069

Pichugin AV, Tyas A, Gilbert M, He L (2015) Optimum structure for a uniform load over multiple spans. Struct Multidisc Optim 52(6):1041–1050

Prager W (1978) Optimal layout of trusses with finite numbers of joints. J Mech Phys Solids 26(4):241–250

Pritchard T, Gilbert M, Tyas A (2005) Plastic layout optimization of large-scale frameworks subject to multiple load cases, member self-weight and with joint length penalties. 6th World Congresses of Structural and Multidisciplinary Optimization, Rio de Janeiro, Brazil

Ramos AS, Paulino GH (2016) Filtering structures out of ground structures - a discrete filtering tool for structural design optimization. Struct Multidisc Optim 54:95–116

Richardson L, Amundsen M (2013) RESTful Web APIs. O'Reilly

Rozvany G, Gollub W (1990) Michell layouts for various combinations of line supports — I. Int J Mech Sci 32(12):1021–1043

Rozvany GIN (1996) Some shortcomings in Michell's truss theory. Struct Optim 12(4):244–250

Sigmund O (2001) A 99 line topology optimization code written in MATLAB. Struct Multidisc Optim 21(2):120–127

Sokół T (2011) A 99 line code for discretized Michell truss optimization written in Mathematica. Struct Multidisc Optim 43(2):181–190

Stolpe M (2016) Truss optimization with discrete design variables: A critical review. Struct Multidisc Optim 53(2):349–374

Stromberg LL, Beghini A, Baker WF, Paulino GH (2012) Topology optimization for braced frames: combining continuum and beam/column elements. Eng Struct 37:106–124

Tcherniak D, Sigmund O (2001) A web-based topology optimization program. Struct Multidisc Optim 22(3):179–187

Wächter A, Biegler LT (2006) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math Program 106(1):25–57

Wei P, Li Z, Li X, Wang MY (2018) An 88-line MATLAB code for the parameterized level set method based topology optimization using radial basis functions. Struct Multidisc Optim 58(2):831–849

Zegard T, Paulino GH (2014) GRAND — ground structure based topology optimization for arbitrary 2D domains using MATLAB. Struct Multidisc Optim 50(5):861–882

Zegard T, Hartz C, Mazurek A, Baker WF (2020) Advancing building engineering through structural and topology optimization. Struct Multidisc Optim 62:915–935

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