Optimization-driven conceptual design of truss structures in a parametric modelling environment

Linwei He[a], Qingpeng Li[b,c], Matthew Gilbert[a,*], Paul Shepherd[c], Catherine Rankine[d], Thomas Pritchard[e], Vincenzo Reale[d]

[a]*University of Sheffield, Sheffield S1 3JD, UK*
[b]*Nanjing University, Nanjing 210093, P.R. China*
[c]*University of Bath, Bath BA2 7AY, UK*
[d]*Arup, London W1T 4BQ, UK*
[e]*LimitState Ltd. Sheffield, S1 4DP, UK*

Abstract

Structural optimization methods can be extremely powerful when used at the initial, conceptual, design stage of a building or bridge structure, potentially identifying materially efficient forms that are beyond the imagination of a human designer. This is particularly important at present, given the pressing need to reduce the carbon footprint associated with the built environment in the face of the current climate emergency. In this contribution, a computationally efficient global-local optimization framework is proposed, in which a linear programming-based truss layout optimization step is employed to generate initial (near-)optimal designs, with a non-linear optimization step then used to generate designs that take account of real-world complexity. To facilitate rapid exploration of design concepts, the proposed global-local optimization framework has been made available in the Peregrine plug-in for the popular Rhino-Grasshopper parametric modelling environment. The efficacy of the approach is demonstrated through its application to a range of case study problems.

Keywords: Structural optimization, Layout optimization, Topology optimization, Structural design, Parametric design

1. Introduction

In recent years a wide range of digital tools have been applied to the design of building and bridge structures. As part of the workflow, optimization techniques can be employed to help designers identify materially efficient forms, that potentially consume less material and have a lower associated embodied carbon footprint. Structural optimization methods can have a particularly significant impact when used at the initial, conceptual, design stage, as they are capable of generating materially efficient forms that are often beyond the imagination of a human designer. These materially efficient forms are often geometrically complex, potentially increasing their cost due to fabrication challenges and the use of bespoke rather than standardized elements. However, the need for the world to quickly move towards net-zero carbon emissions in the light of the current climate emergency is creating a resurgence in interest in material efficiency. Also, manufacturing techniques have been developing rapidly in recent years, reducing, or sometimes even eliminating, the overhead associated with complexity, with CNC systems and automated steel welding equipment becoming commonplace, and with technologies such as additive manufacturing under development for civil and structural engineering applications. This is increasingly providing engineers with the capability to physically realise complex but highly materially-efficient structural forms.

In the structural design field, meta-heuristic optimization methods (e.g. genetic algorithms) have to date proved popular, since these only require problem evaluations (i.e., structural performance indicators), rather than derivations

[*]Corresponding author
Email address: `m.gilbert@sheffield.ac.uk` (Matthew Gilbert)

(i.e., mathematical information on the relationship between structural performance indicators and problem variables, such as gradients). This means that a wide range of problems can be tackled without difficulty. A large variety of meta-heuristics methods have been explored, including genetic algorithms [1], simulated annealing [2], ant colony optimization algorithms [3], the 'big bang-big crunch' algorithm [4], and firefly algorithms [5], with some methods incorporated in digital, commercially available, design tools (e.g., the Galapagos solver incorporated in the Rhino-Grasshopper parametric modelling software, which has proved very popular).

However, when using meta-heuristic methods, the lack of underpinning mathematics typically leads to computationally expensive processes and highly locally optimal solutions being identified; as Wolpert [6] suggests, there is 'no free lunch' when it comes to optimization methods. This is particularly true when meta-heuristic methods are applied at the conceptual design stage, arising from the typically large design freedom available. To address this, a designer may need to significantly constrain the problem, in some cases leaving only a handful of active design variables. However, this limits the range of solutions that can be generated and, although attempts have been made to generate families of designs for use at the conceptual design stage [7], the extent of the design space explored is still usually limited. Secondly, meta-heuristic methods usually rely on randomized factors to theoretically enable globally optimal solutions to be identified, if sufficient computational time is available. However, in practice available computational time is finite and solutions are often highly sub-optimal, with the gap between these solutions and the true mathematically optimal solution remaining unknown. Thirdly, when using meta-heuristic optimization tools, penalty methods are typically used to approximately enforce constraints. However, choosing suitable values for the penalty factors involved can be challenging, with large values rendering the original design objective (e.g., to minimize the volume of material consumed) comparatively less significant, and small values leading to large constraint violations. As an alternative, gradient-based optimization methods can be employed, which do not suffer from the aforementioned shortcomings, and which are potentially capable of solving problems with large design freedom; usage of these methods is therefore the focus of this paper.

At the conceptual design stage it is important to feed in as many options into the process as possible. Thus for structural design problems, topology or layout optimization methods are particularly useful. Considering topology optimization, a range of software tools are already available to designers, whether as standalone tools (e.g., Altair OptiStruct [8]), plug-ins (e.g., Ameba [9], Millipede [10]), or educational tools (e.g., the interactive web- [11] or app-based [12] TopOpt tools developed by Sigmund and co-workers). However, topology optimization tools, which involve identifying an optimal continuum form, are not particularly well-suited to identifying the kind of skeletal spatial structures generally used to form buildings or bridges. Although it is possible to derive truss layouts from topology optimization solutions (e.g. [13–15]), numerical layout optimization employing a 'ground structure' provides a powerful and highly efficient alternative for such structures, being capable of rapidly identifying (near-) optimal structural forms (e.g., [16–22]). For example with truss layout optimization, an optimal structure formed from discrete truss elements is sought, and linear programming (LP) can be used to solve basic problems. This leads to a very efficient optimization process, capable of identifying globally optimal solutions for the given numerical discretization. This brings two benefits: (i) the highly efficient nature of the optimization process means that a wide range of solutions can be obtained rapidly at the conceptual design stage [23]; (ii) the global optimality of the solution means that this can be used as a benchmark, against which other designs can be judged, as outlined by Fairclough et al.[24]. However, a framework that systematically exploits these benefits and is suitable for use by practitioners at the conceptual design stage is still lacking. To address this, a global-local framework is proposed, and is incorporated in the Rhino-Grasshopper parametric modelling software to provide a powerful conceptual design tool for use in industry, adding to the range of structural optimization tools already developed for this platform (e.g., [25, 26]).

To ensure the framework is useful, one issue to overcome is the fact that the raw solutions generated by the truss layout optimization process are often complex in form, and hence difficult to interpret and/or to directly fabricate in practice. One means of addressing this is to directly incorporate practicality constraints in the optimization problem formulation from the outset. However, this will generally lead to complex problem formulations that are computationally expensive to solve. Alternatively, the process can be divided into different stages, with layout optimization used in the first stage and practical issues taken into account in a follow-up, potentially computationally inexpensive, post-processing stage. The solution obtained in the first stage then also provides a benchmark against which trade-offs associated with various practical constraints can be evaluated. This multi-stage process forms a 'global-local' optimization framework, where globally optimal solutions obtained in the first stage are complemented by more practical, though only locally optimal, solutions obtained in the second stage. This is considered in more detail in this paper.

Note that in the present work priority has been given to computational efficiency, so that the resulting tool can be used to rapidly generate a range of designs for use at the conceptual design stage. To achieve this, the first stage of the global-local framework employs LP, which is computationally inexpensive. However, this also means that geometrical stability and/or Euler buckling are only considered in the second stage. For problems where stability considerations are important, semidefinite programming (SDP) or other methods could instead be used in the first stage with a view to obtaining improved second stage solutions, taking advantage of recently developed methods for treating global and/or local stability in truss layout optimization (e.g., [27–29]), albeit at the expense of increased computational effort.

The paper is organised as follows. In section two, the basic layout optimization procedure for trusses is briefly outlined, with the proposed global-local design framework then presented along with the associated workflow. In section three, usage of the framework within the Rhino/Grasshopper parametric modelling tool is briefly outlined. In section four, a number of case-study problems are described, with a range of design solutions presented to demonstrate the efficacy of the framework developed. Finally, in section five, key conclusions are drawn.

2. Conceptual design framework

2.1. Layout optimization

The standard layout optimization process for truss structures [16–18] involves a number of steps, as shown in Fig. 1. Firstly the design domain, load and support conditions are specified, Fig. 1(a). Secondly, the design domain is discretized using nodes, Fig. 1(b). Thirdly, these nodes are interconnected with potential members to create a 'ground structure', Fig. 1(c), which contains a very large number of potential structural layouts, comprising combinations of all the potential members present (up to $\frac{n(n-1)}{2}$, where n is the number of nodes). Here The goal is then to find the optimal structural layout, Fig. 1(d). When it is required to minimize the volume of material consumed, the problem can be formulated mathematically as follows:

$$\underset{\mathbf{a},\mathbf{q}^{(w)}}{\text{minimize}} \quad V = \mathbf{l}^{\mathrm{T}}\mathbf{a} \tag{1a}$$

subject to

$$\left.\begin{aligned} \mathbf{B}\mathbf{q}^{(w)} &= \mathbf{f}^{(w)} \\ \mathbf{q}^{(w)} &\geq -\sigma^{-}\mathbf{a} \\ \mathbf{q}^{(w)} &\leq \sigma^{+}\mathbf{a} \end{aligned}\right\} \text{for } w = 1, 2, ..., p \tag{1b}$$

$$\mathbf{a} \geq \mathbf{0}, \tag{1c}$$

where V is the structural volume, $\mathbf{l} = [l_1, l_2, ..., l_m]^{\mathrm{T}}$ is a vector of member lengths, and $\mathbf{a} = [a_1, a_2, ..., a_m]^{\mathrm{T}}$ is a vector containing member cross-sectional areas, with m denoting the number of members. Also, $\mathbf{B}$ is a $3n \times m$ equilibrium matrix comprising direction cosines (for three-dimensional problems), with n denoting the number of nodes, and $\mathbf{q}^{(w)}$ and $\mathbf{f}^{(w)}$ are vectors containing the internal member forces and the external forces, respectively, in the w^{th} of p load cases. Finally, σ^{+} and σ^{-} are respectively the limiting tensile and compressive stresses sustainable by the material employed.

Note that (1) is a plastic formulation, which is computationally efficient and which implicitly takes advantage of the plastic response exhibited by many common engineering materials, enabling greater material savings to be realized when multiple load cases are involved. (For single load-case problems a statically determinate layout can be found, which will also be optimal when an elastic material is involved; for multiple load-case problems the optimal elastic and plastic solutions will diverge, with the volumes computed using the formulation described here being lower bounds on the corresponding elastic solutions.)

With respect to optimization variables, which are member areas $\mathbf{a}$ and internal forces $\mathbf{q}$, problem (1) is a linear programming (LP) problem, which can be solved very efficiently using modern LP solvers (e.g. see [30]). This forms the starting point of a fully automated and systematic numerical approach that can handle problems with significant design freedom, thus providing great potential when used at the conceptual design stage.

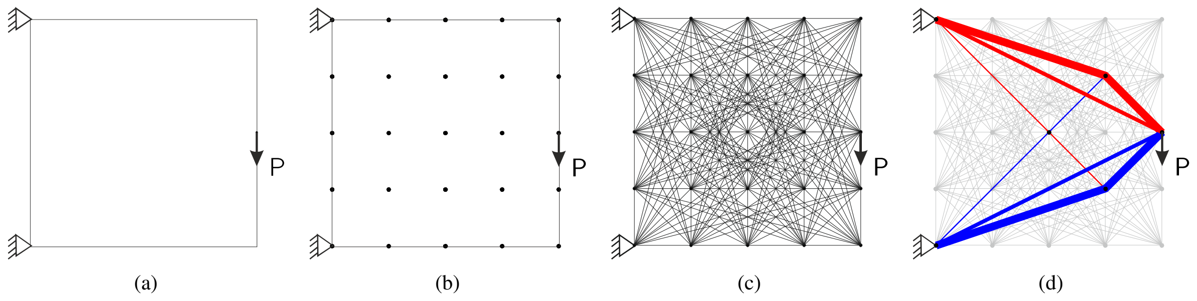


Figure 1: Truss layout optimization steps: (a) specify design domain, loads and supports; (b) discretize domain using nodes; (c) interconnect nodes with potential truss members; (d) use optimization to identify the optimal structural layout.

2.2. *Global-local optimization framework*

The two stages of the proposed global-local design optimization framework can be illustrated via the use of the example shown in Fig. 2. Since the idealized layout optimization problem (1) is convex (see dashed line in Fig. 2(a)), the global optimality of the solution obtained for a given 'ground structure' is mathematically guaranteed (i.e., the design consuming the least amount of material is found). However, being mathematically optimal does not mean a given generated design is necessarily practically feasible or desirable. In fact, in addition to often being over-complex in form, the linear nature of the standard LP layout optimization problem formulation (1) means that important practical considerations cannot be directly addressed. For example, Euler buckling cannot be addressed directly since this would require the inclusion of non-linear constraints. To address this, rather than attempting to navigate the search space directly (e.g., Fig. 2(b)), here an incremental approach involving two optimization stages is proposed. In the first stage, the standard layout optimization problem with relaxed (linearized) constraints is solved, allowing rapid exploration of a large solution space. When a fine nodal discretization is used, the solution obtained can be considered for practical purposes to be equivalent to the globally optimal solution for the problem. In the second stage, the (im-)practicality of the Stage 1 solution(s) is addressed by solving one or more potentially more complex optimization problems with reduced design freedom in order to generate a more practical design (Fig. 2(c)).

This forms the basis of a global-local optimization design framework, which has the following benefits:

1. The identified globally optimal solution to the relaxed problem provides an invaluable benchmark that can be used to evaluate the quality of other design solutions (since the gap to the globally optimal solution is known).

2. When the Stage 1 solution lies close to good local optima, as it often will, these can then be identified relatively quickly.

Each of the two stages in the process are now considered in more detail.

2.3. *Stage 1: layout optimization*

Numerical layout optimization provides a reliable and computationally inexpensive means of obtaining solutions to the idealized (or 'relaxed') problem. Usually each node should be connected to every other node in the design domain to ensure the available design space is extensively explored. If this is not done, then less efficient designs may be identified (e.g., see Fig. 3). Additionally, a reasonably fine nodal grid should be used to generate (near-) optimal solutions in order to provide a suitably accurate benchmark solution. It has been shown that solutions will converge on the theoretical minimum by increasing nodal density in the 'ground structure'; e.g., see Darwich et al 31, Bolbotowski et al 32.

To solve problems involving fine nodal grids, interior-point LP solvers generally outperform solvers that use the traditional simplex method [30], and an adaptive 'member adding' solution scheme [17, 21] can be used. Initially only a small proportion of the potential members in the fully connected 'ground structure' are included in (1); then a small number of members are successively added to improve the solution, until the optimal solution is found. This means that, rather than solving a single large-scale LP problem, a series of relatively small-scale LP problems are instead

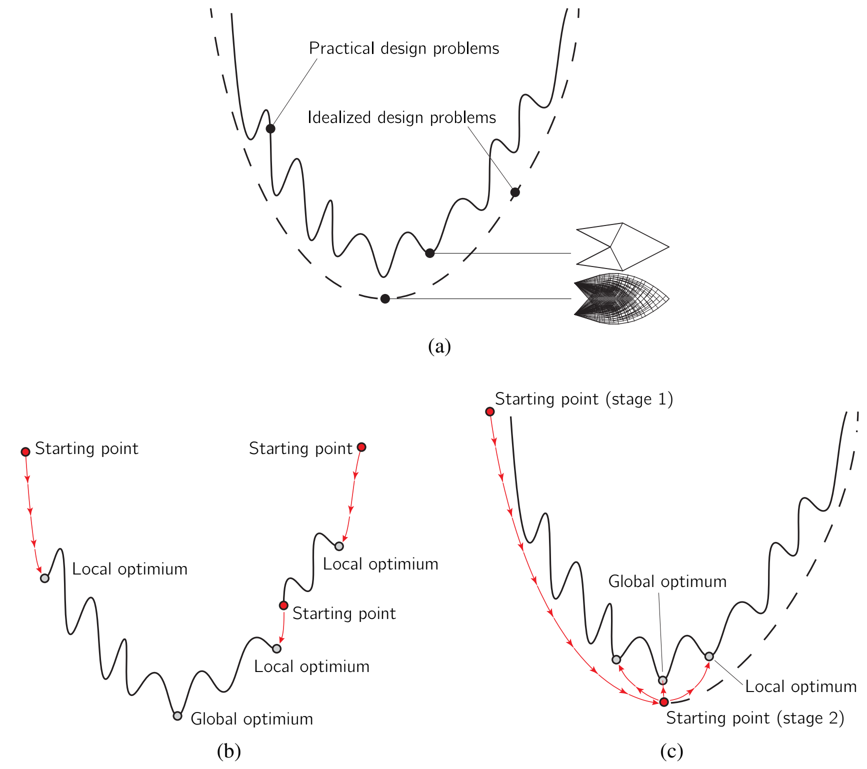


Figure 2: Sample design problem: (a) solution space, showing idealized LP layout optimization and more practical design solutions; (b) navigating solution space with practicality constraints included directly; (c) navigating solution space using a two-stage process, as part of the proposed global-local optimization framework.

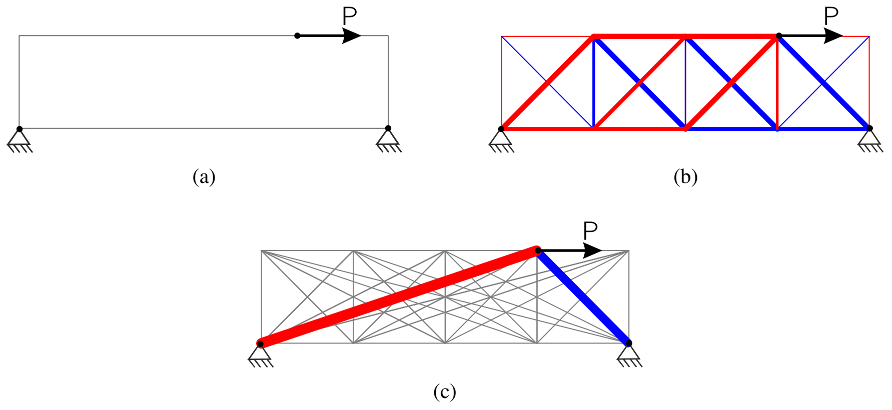


Figure 3: Simple example to illustrate the importance of including sufficient design freedom in the problem setup: (a) problem specification (design domain, supports and applied load); (b) optimum design obtained using a ground structure involving a coarse nodal grid and only adjacent connectivity; (c) optimum design obtained using a ground structure with full connectivity, which consumes 33% less material than design (b).

solved in sequence, reducing total CPU cost and memory requirements; see Fig. 4. Note also that although efficient parallel computing schemes for solving LP problems are not yet widely available, the process of identifying which members to add to a given small-scale problem in the adaptive 'member adding' scheme is readily parallelizable. The peak number of members involved when using this approach is normally significantly smaller than when the full problem is solved directly (e.g., 24,151 compared with 1,086,938 members in the half wheel example considered in Fig. 4). The approach is also mathematically rigorous, with the solution guaranteed to converge to the same solution as is obtained when the full problem is solved. An educational Python script implementing the adaptive 'member adding' solution scheme is available for interested readers [21].

Figure 5 shows that for the half wheel example considered above a numerical solution within 1% of the theoretical optimal [33] can be obtained quickly (in 1s), though significantly more time is required to reduce the error down to 0.1% (requires 470s). However, for use at the conceptual design stage, the former solution may be of sufficient accuracy to provide a reasonable benchmark, and, as the corresponding layout obtained is comparatively simple, it is also likely to provide a more useful starting point for the second stage of the proposed two stage framework. (For example, the presence of fewer slender members reduces the likelihood of Euler buckling issues in Stage 2, although for problems where local Euler buckling effects are significant, an improved starting point for Stage 2 could potentially be obtained simply by adjusting the limiting compressive stress as part of an iterative LP layout optimization process, similar to that undertaken in [34] to solve a cantilever truss problem originally studied by Achtziger [35].)

2.4. *Stage 2: improving practicality*

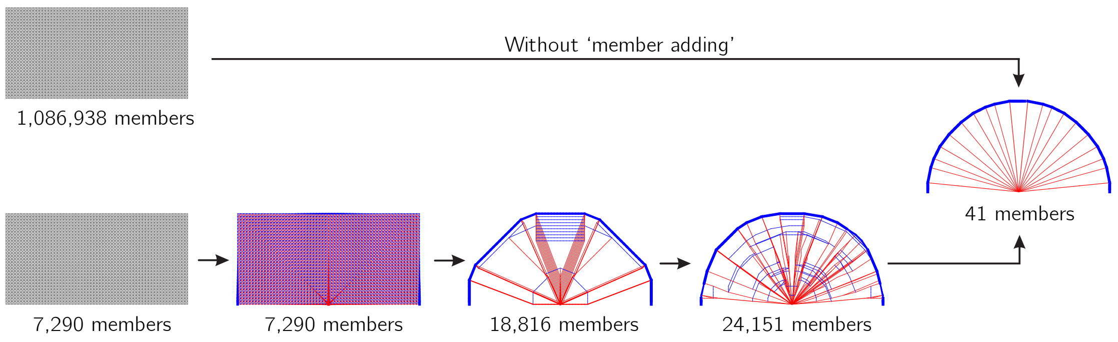
As indicated in Fig. 2, the starting point for Stage 2 of the proposed global-local optimization framework is a truss layout generated via layout optimization during Stage 1. In the interests of practicality, and to ensure Stage 2 computations take a reasonable time, it is usually prudent to employ a truss generated using a relatively coarse nodal grid as the starting point, such that the number of members present in the optimized layout is not excessive.

2.4.1. *Rationalization*

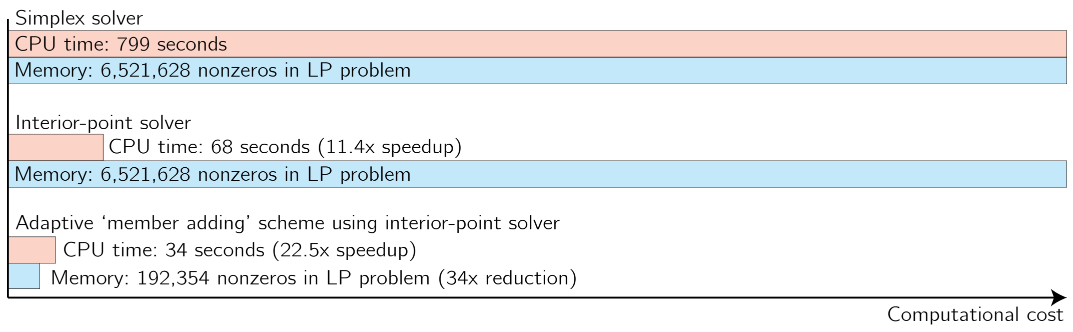
In layout optimization the nodal positions are fixed, so the forms of the layouts generated are influenced by the positions of the nodes. However, geometry optimization, which involves adjusting the nodal positions, can be used to improve and rationalize the solutions [38]. This means that the optimization objective function becomes:

$$\underset{\mathbf{a},\mathbf{q}^{(w)},\mathbf{x},\mathbf{y},\mathbf{z}}{\text{minimize}}\ V = \mathbf{l}^{\mathrm{T}}\mathbf{a}, \tag{2}$$

where, $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ are vectors containing x, y, and z nodal positions, respectively. With these new design variables, it becomes a nonlinear optimization problem (NLP) (e.g., member length l is a nonlinear function of the nodal positions),



(a)



(b)

Figure 4: Effect of using an interior-point solver and adaptive 'member adding' solution scheme on CPU time and memory usage for a half wheel truss layout optimization problem comprising 60 × 30 nodal divisions and approx. one million members in the 'ground structure': (a) initial ground structures and intermediate and final solutions; (b) comparative CPU times and memory usage. (Python script presented in [21] used to solve (1), with the Gurobi simplex [36] and Mosek interior-point [37] solvers used on a laptop equipped with an Intel i7-7700HQ CPU. The problem comprises roller supports at the bottom corners of the domain and a downward vertical point load applied midway between these supports. The same volume was obtained in all cases, 0.3% above the exact analytical solution.)

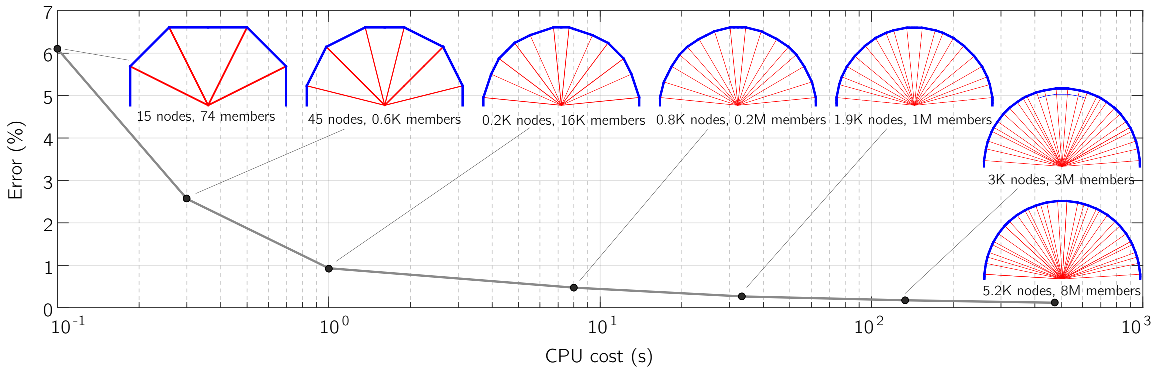


Figure 5: Influence of nodal density on accuracy for the half wheel example. (Python script presented in [21] used to obtain solutions.)

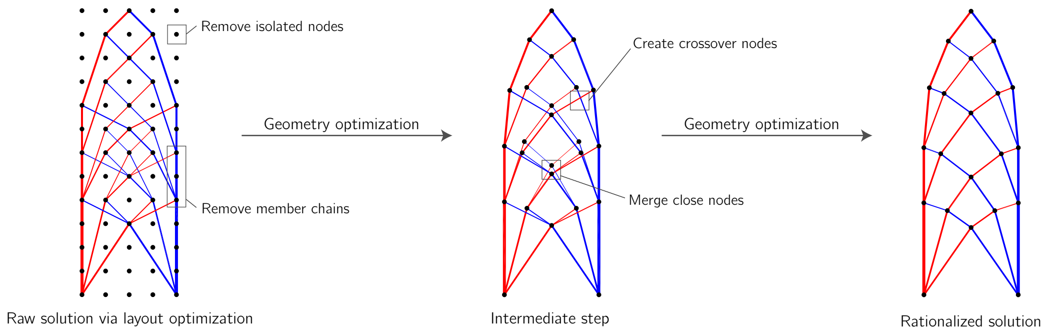


Figure 6: Use of geometry optimization to rationalize a solution generated via layout optimization, after [38].

which is now more challenging to solve. However, as this NLP problem is here solved as a post-processing step, the number of variables is significantly smaller than in Stage 1, ensuring the computational cost associated with the process is generally not excessive, and much lower than when both these stages are solved simultaneously (e.g., see [39, 40]). Prior to the process commencing, isolated nodes are removed and co-linear chains of members are converted to single members; then at each iteration in the geometry optimization process nodes in close proximity to one another are merged and nodes at crossover locations are created; see Fig. 6.

2.4.2. *Euler buckling*

The use of NLP optimization not only allows rationalization of structural forms, but also provides a means of addressing various practical constraints that are non-linear in form, such as Euler buckling. This can be incorporated in Stage 2 by adding the following constraint (after [41, 42]):

$$q_i \geq -\alpha \frac{a_i^2}{l_i^2}, \text{ for } i = 1, 2, ...m, \tag{3}$$

where α is a cross-sectional constant calculated using the shape of the cross-section, elastic modulus, and effective length factor. For circular hollow sections, α can be calculated from:

$$\alpha = \frac{\pi E}{4k^2} \cdot \frac{1+\beta^2}{1-\beta^2} \tag{4}$$

where E is the elastic modulus of the material employed, k is the effective length factor (taken as 1 for a pin-ended strut), and β is the ratio between the outer and inner radii of the section. Note that the moment of inertia of the cross-section is expressed in terms of area variable a, which is possible for some cross section types.

3. Implementation: Rhino-Grasshopper plug-in

To facilitate use at the conceptual design stage, the proposed global-local design framework has been implemented in the Peregrine [43] plug-in for Rhino-Grasshopper. The problem data, e.g., design domain, loads and support conditions, are specified via parametric controls, facilitating rapid exploration of the influence of the parameters involved. The LP and NLP problems are respectively solved via the Mosek [37] and IPOPT [44] solvers, both of which employ efficient interior point methods [45]. In the case of NLP problems, first- and second-order derivatives of the functions involved are computed via an automatic differentiation routine.

The main workflow is illustrated in Fig. 7, and comprises the following main elements:

- Specification: All problem data can be created via parametric controls in Rhino & Grasshopper. For example, various geometric entities can be defined in terms of lines, polygons, NURBS surfaces, and complex BReps that are controlled by user-defined parameters. These can then be used to create problem data such as the design domain, load, and support conditions. A number of bespoke components have been programmed to aid this process.
- Stage 1 (layout optimization): Numerical layout optimization is used to generate an idealized solution for use as the starting point for Stage 2. A relatively coarse nodal grid can be used for this purpose, with a finer nodal grid used to generate an accurate benchmark against which other designs can be judged.
- Stage 2 (improving practicality): Various tools can be utilized to improve the practicality of the designs. In addition to the rationalization techniques described in Section 2.4.1, the automatic simplification method described in [22, 46] can be used to e.g. control the number of members, connections and/or the complexity of connections.

In this way the designer can freely adjust one or more of a number of controlling parameters to generate a wide range of optimized designs, each of which can be evaluated against the benchmark identified in Stage 1. Candidate designs can also easily be analysed in more detail using tools such as Karamba 3D [47] structural analysis plugin, to allow in-depth evaluation of the designs generated.

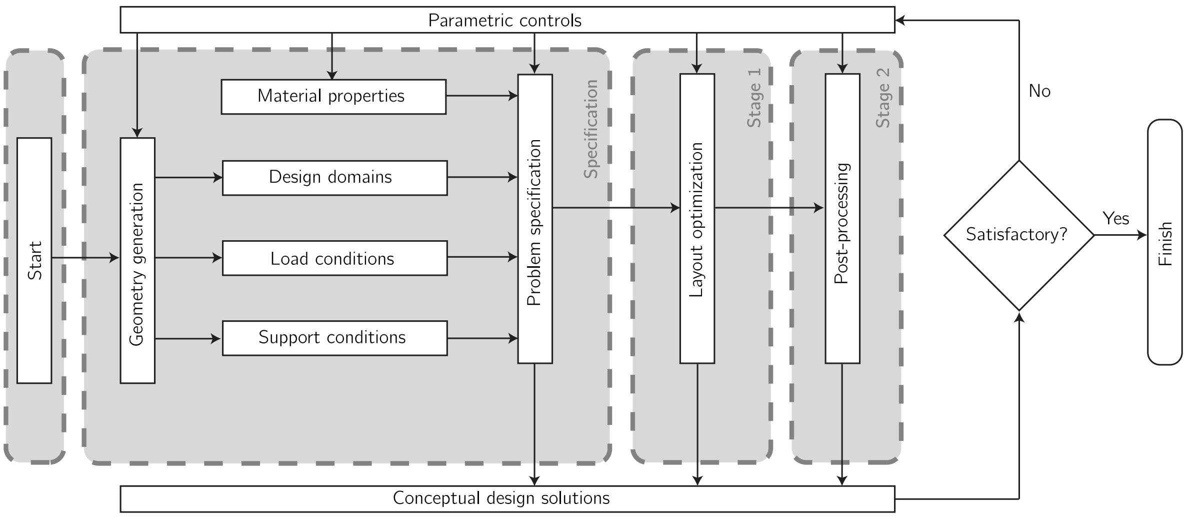


Figure 7: Workflow utilizing the proposed global-local optimization framework.

4. Design examples

A number of design examples are now used to demonstrate the efficacy of the proposed global-local design framework and the underlying layout and geometry optimization methods. All examples were solved using a laptop computer equipped with an Intel i7-7700HQ CPU, 32GB of memory and running Windows 10. For sake of simplicity

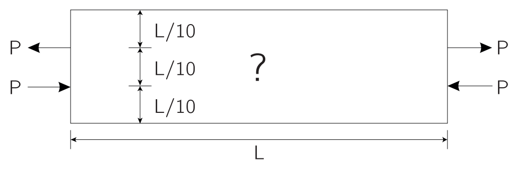


Figure 8: Simple load path example: problem specification, showing extent of design domain and the applied loads.

self-weight loads are ignored in all design examples. Note that although for sake of simplicity the examples presented involve two-dimensional design domains, either single domains or multiple intersecting domains, the methodology described can equally be applied to fully three-dimensional design problems.

4.1. Simple load path example

The first design example involves finding the form of the least volume structure to transmit a pair of opposing forces across a rectangular design domain, as shown in Fig. 8. In the case of this design problem, for comparative purposes the Genetic Algorithm (GA) meta-heuristic method is used to generate solutions in addition to the proposed global-local optimization framework.

An initial intuitive solution to the problem comprises two horizontal members connecting the pairs of forces, as shown in Fig. 9(a). However, layout optimization methods are useful in helping the designer to identify more efficient solutions, which may not be intuitively obvious. In this case a more complex alternative solution is indeed found, as shown in Fig. 9(d). This Stage 1 benchmark solution was obtained using approximately 3,000 nodes and 3,000,000 potential members, and can be observed to require 62% less material than the solution involving two horizontal members. Once a designer is shown this alternative solution, it starts to make sense: each pair of opposing forces can be seen to form a couple (i.e., an applied moment) and the optimized solution incorporates tensile and compressive members that are pushed to opposite edges of the domain in order to maximise bending resistance. Since the Stage 1 benchmark solution is rather complex in form, the Stage 2 rationalization process can be used to furnish more practical solutions, as shown in Fig. 9(b) and Fig. 9(c).

Alternatively, the same problem can be solved using a GA based method, in this case using the `ga` function from the Global Optimization Toolbox in Matlab R2019b. In this approach, binary variables are used to indicate the existence of members in the ‘ground structure’. Finite element analysis is performed to calculate internal forces in the members, allowing the volume of material consumed to be evaluated as the objective function. Two population sizes (100 and 1,000) are considered, and each is tested with 10 independent runs, with the best reported. Results are shown in Fig. 9(e) and (f), with the reported CPU time only including time spent evaluating the objective (i.e., fitness) function (programmed using vectorized commands in the interests of computational efficiency). It is evident that the resulting designs are both indistinct in form and consume significantly higher volumes of material than the benchmark. It was also observed that these solutions were not markedly improved by adjusting control parameters (e.g., the crossover fraction number, elite count). This stems from the number of design variables (603 in total), which is large for a meta-heuristic algorithm like GA to deal with. To use such methods efficiently, designer assumptions are often required to reduce the design space. For example, by grouping variables and removing long diagonal members, only 75 binary variables are required, which allows an improved solution to be obtained, as shown in Fig. 10. In this case, the solution is only 7% above the benchmark and the associated structure is much clearer. Also, the CPU cost reduces to 3.2 seconds. This indicates that the efficacy of meta-heuristic methods depend heavily on designer assumptions. However, if used without a known benchmark, such methods are prone to identify solutions that are highly sub-optimal, though with the designer oblivious to the degree of this sub-optimality.

4.2. Simple bridge example

The problem specification for a simple bridge example is shown in Fig. 11(a) and the complete design process is shown in Fig. 11(b). Taking advantage of symmetry conditions, only half of the design domain is considered, with the line of symmetry modelled using roller supports. Circular hollow sections with $\beta = 0.9$ were assumed, with the tensile and compressive yield stress and elastic modulus of the steel material used taken as 350MPa and 210GPa respectively. In Stage 1 a relatively dense nodal grid (approx. 1900 nodes) was employed to generate a

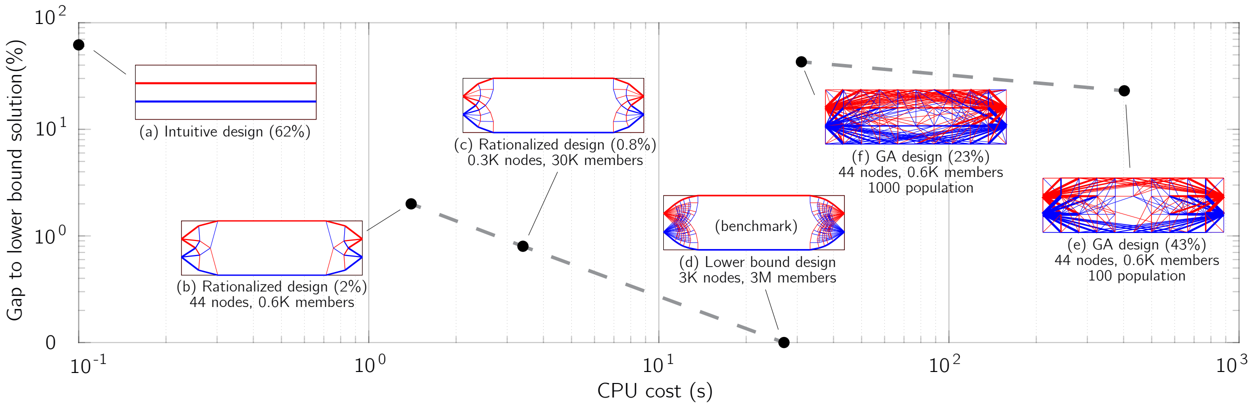


Figure 9: Simple load path example: (a) initial intuitive design; (b)-(d) designs obtained using the proposed framework; (e)-(f) using GA. Dashed trend lines drawn between data points provide an indication of the relative computational costs associated with mathematical programming vs GA based optimization schemes (% gap to the benchmark solution shown in parenthesis).

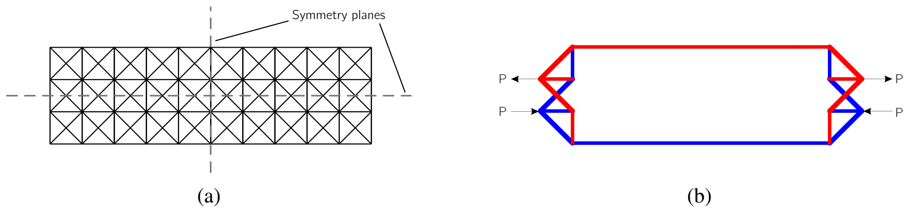


Figure 10: Simple load path example: GA solution identified after reducing the number of binary design variables, (a) ground structure (using top-right quadrant); (b) identified solution, 7% above the benchmark in Fig. 9(d)

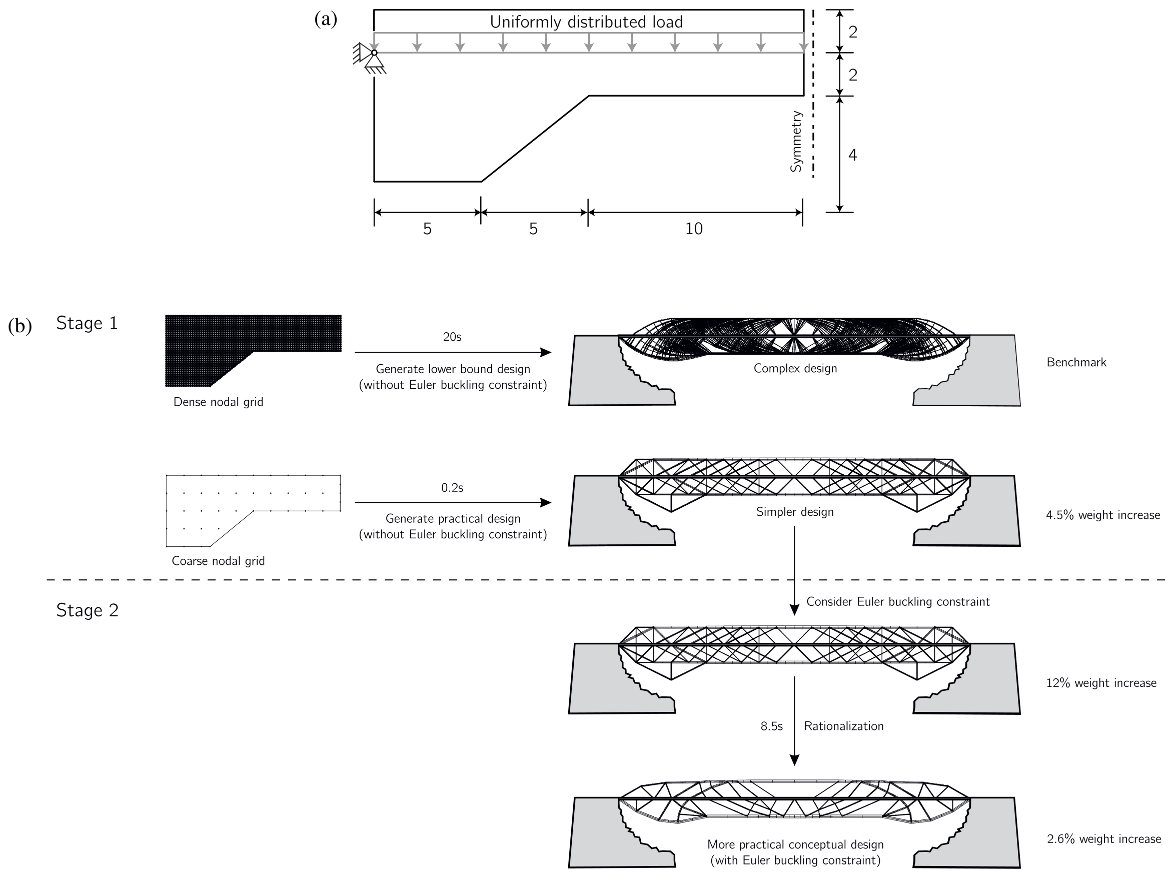

(a)
Uniformly distributed load
Symmetry
2
2
4
5
5
10
(b)
Stage 1
Dense nodal grid
20s
Generate lower bound design
(without Euler buckling constraint)
Complex design
Benchmark
Coarse nodal grid
0.2s
Generate practical design
(without Euler buckling constraint)
Simpler design
4.5% weight increase
Stage 2
Consider Euler buckling constraint
12% weight increase
8.5s
Rationalization
More practical conceptual design
(with Euler buckling constraint)
2.6% weight increase

Figure 11: Simple bridge example: (a) problem specification (all dimensions in metres and uniformly distributed load of 100kN/m discretized via 11 point loads); (b) two stage optimization process; in Stage 1 fine and coarse grids are used to generate a benchmark solution and the starting point for Stage 2, which involves use of NLP to obtain more practical solutions, after [38].

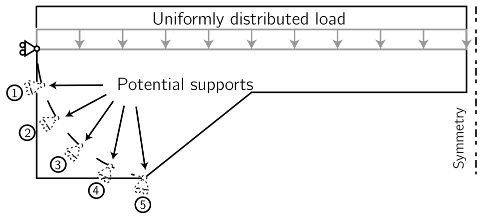
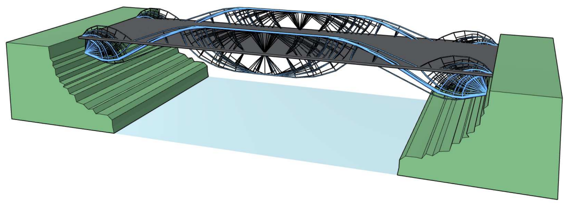
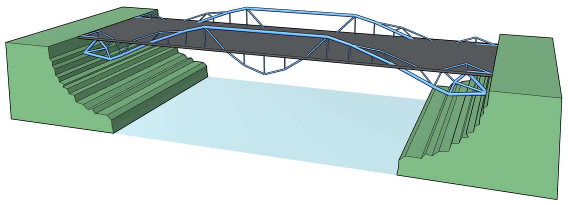


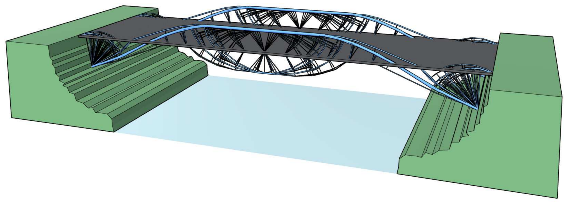
Figure 12: Simple bridge: parametric study considering five potential roller support locations.

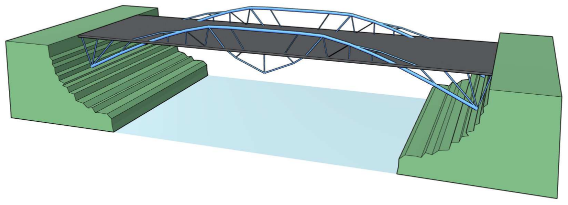
benchmark solution, against which other designs can be judged. Since layout optimization is highly efficient, this step took only around 20 seconds to complete. A relatively coarse nodal grid was then used to generate a simpler design (nearly instantaneously, in 0.2 seconds). After the initial Stage 1 solution was identified, local Euler buckling constraints were imposed, with all members resized to satisfy constraint (3). This led to an initial consumed volume of material increase of 12%. Rationalization was then performed to improve the design (taking 8.5 seconds). The subsequently generated structure was found to be both more rational and more efficient (volume just 2.5% higher than the benchmark design).

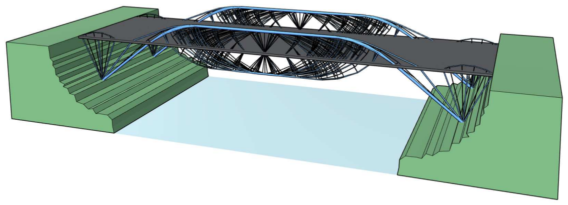
Note that the entire design process shown in Fig. 11(b) was completed within a minute. This rapid run time makes the proposed optimization process amenable to use within an interactive parametric modelling environment, with the designer able to quickly explore a range of design scenarios at the conceptual design stage. For example, in Fig. 12

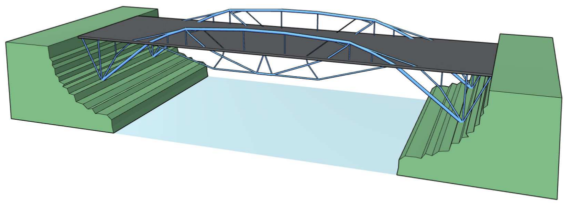

(a) Support location 1, benchmark (vol. = $1.67V_0$)

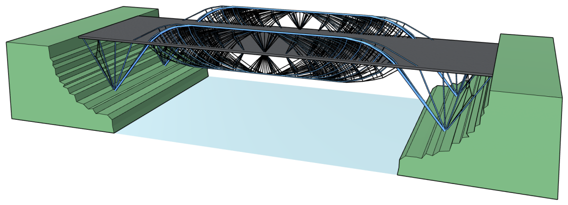

(b) Rationalized (vol. diff. = +4.3%)

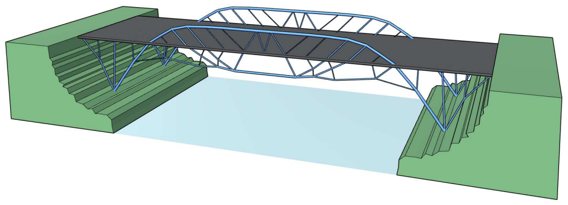

(c) Support location 2, benchmark (vol. = $1.28V_0$)

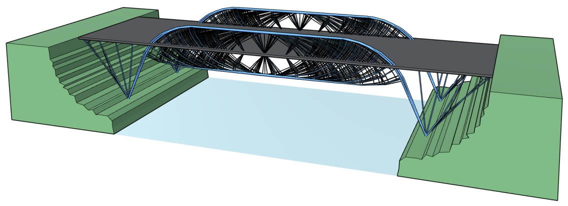

(d) Rationalized (vol. diff. = +7.4%)


(e) Support location 3, benchmark (vol. = $1.08V_0$)


(f) Rationalized (vol. diff. = +23%)


(g) Support location 4, benchmark (vol. = $1.03V_0$)


(h) Rationalized (vol. diff. = +10%)


(i) Support location 5, benchmark (vol. = V_0)

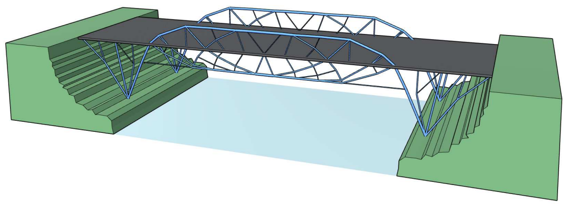
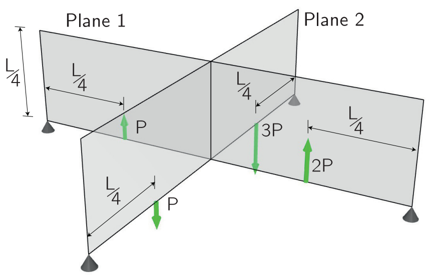

(j) Rationalized (vol. diff. = +15%)

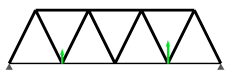
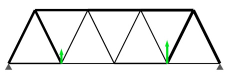
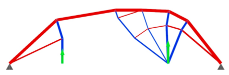
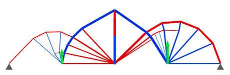
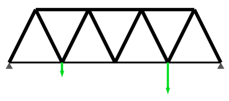
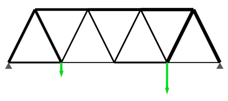
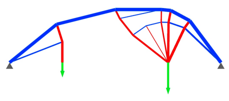
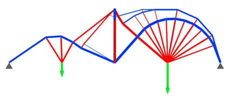
Figure 13: Simple bridge example: parametric study solutions obtained by varying the roller support locations (see Fig. 12).

the pin support in Fig. 11(a) is replaced with two roller supports, the lower of which is placed at one of a number of possible locations along a circular arc of radius 5m. Repeating the optimization process of Fig. 11(b) leads to the generation of a range of benchmarks and more practical designs, as shown in Fig. 13. In most cases the volumes of the Stage 2 designs are within 15% of the benchmark volume. However, in one case the gap is greater (Fig. 13(f)), likely due to the influence of the Euler buckling constraint. In this case, the designer has evidence to suggest that it may be worthwhile to rerun the problem with alternative inputs, or to intervene manually (e.g., as was done in [48]).



(a)

Figure 14: Two intersecting planar trusses example: design domain formed by two ($L \times L/4$) rectangular planar domains

	#1 Separated design (Warren truss, 3 groups[†])	#2 Separated design (Warren truss[‡])	#3 Separated design (optimized)	#4 Holistic design
Plane 1	 $1.36V_0$	 $0.81V_0$	 $0.64V_0$	 NA
Plane 2	 $1.96V_0$	 $1.13V_0$	 $0.88V_0$	 NA
Total Volume	$3.32V_0$	$1.94V_0$	$1.52V_0$	$1.005V_0$

† Using three member groups in top chord, bottom chord, and diagonals; members in a group have the same cross-sectional area.
‡ Sizing optimization of all members.

Figure 15: Two intersecting trusses example: traditional and optimized solutions

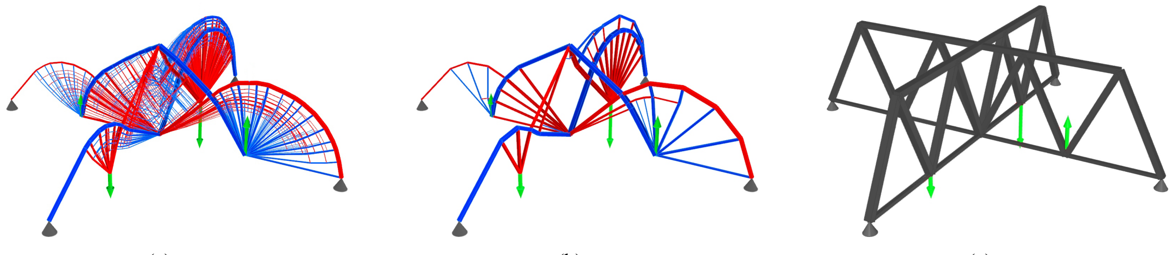
4.3. *Two intersecting trusses example*

Figure 14 shows details of another relatively simple design problem, this time comprising two intersecting planar design domains. In this case, fixed pin supports are present at the ends of each domain and all loads are assumed to be applied simultaneously, in a single load case. In standard design practice a truss for each domain would be designed separately. For example, Warren trusses, each employing three cross-sections (one cross-section for each of the top and bottom chords and one for the diagonals) may be selected. When this was done for this problem the total volume of material consumed was found to be $3.32V_0$ (#1 in Fig. 15), where V_0 is the corresponding benchmark solution (Fig. 16(a)). By sizing each member in each truss separately, a lower structural volume of $1.94V_0$ (#2) was obtained, whilst by using layout optimization to design each truss separately, a much reduced volume of $1.52V_0$ was obtained (#3). Finally, when the problem was considered in a holistic manner, a much more efficient means of transferring the forces was identified, with a reduced structural volume of $1.005V_0$ (#4), which is only 0.5% larger than the benchmark in Fig. 16(a).

For this problem, potential interactions between the two trusses may have not been foreseen by the designer. Thus traditional engineering design solutions that do not take advantage of these interactions may be proposed (e.g., Fig. 16(c)), which have been shown to be inefficient. The proposed two stage optimization framework can in this case generate significantly more efficient conceptual designs that, although not necessarily practical in this case, can serve as inspiration to the designer.

4.4. *Multiple intersecting trusses example*

Many of the structures generated for the previous example problems can be observed to be in unstable equilibrium with the applied loading. This in part results from the fact that the structures have only been required to carry a single set of loads, whereas real-world structures are generally required to deal with a number of sets of loads or load cases.



(a) (b) (c)

Figure 16: Two intersecting trusses example: (a) Stage 1 benchmark obtained with approx. 6,000 nodes, $V = V_0$; (b) Stage 2 conceptual design obtained using the proposed optimization framework, $V = 1.005V_0$; (c) traditional Warren truss design comprising three member cross-sections (one each for the top and bottom chords, and one for the diagonals), $V = 3.32V_0$.

The next example involves the conceptual design of a series of intersecting trusses for a real world project and involves multiple load cases. An overview of the problem is shown in Fig. 17 (a). A holistic study involving simultaneous optimization of the trusses in all domains is performed since, as demonstrated in the previous example, this should ensure the most materially efficient design is found. The four load cases considered are shown in Fig. 18; these comprise two primary load cases involving vertical point loads and two secondary load cases designed to stabilize the structure (though this is not guaranteed; for details of a formulation that ensures the generated structure is globally stable see e.g. [27, 28]). In addition, square hollow sections are utilised in the Euler buckling constraint (3), where the cross-sectional constant α for square hollow sections is calculated from $\alpha = \frac{\pi^2 E}{12k^2} \cdot \frac{1+\beta^2}{1-\beta^2}$, where here $\beta = 0.9$. For this problem the tensile and compressive yield stress and elastic modulus were taken as 350MPa and 210GPa respectively.

A Stage 1 benchmark design was initially obtained by solving a standard layout optimization problem with approx. 5,000 nodes, leading to the structural layout shown in Fig. 17(b) which has a structural volume of 25.020m^3. The nodal density in the ground structure was then reduced and a Stage 2 rationalization step performed, leading to generation of the simpler design shown in Fig. 17(c). This design has a volume of 26.112m^3, which is only 4.4% higher than the benchmark solution shown in Fig. 17(b). The design can be further simplified if fewer nodes are utilised in Stage 1; Fig. 17(d) shows a more practical design obtained using approx. 100 nodes, though the associated volume of material consumed is in this case 10% higher than the benchmark.

4.5. *Staircase support design example*

The final example involves another real-world design problem, this time involving the design of the support for a staircase, as shown in Fig. 19. The applied loads include a line load of 120kN/m from the stair, a 900kN point load (at point E) transferred from structures above, and a 90kN point load (at point F) from the stair arch. The structure is supported by a pin support at point A, and a pin/roller support at point B. In addition to strength requirements, a maximum deflection of 10mm has been specified. Since the problem involves a single primary load case, the optimal layouts obtained when using stress or deflection design constraints are equivalent [49]. Typically, either stress or deflection limits are critical; in the latter case the limiting stress can be scaled down to ensure the generated structure satisfies the prescribed deflection limit. For this reason, the structural volume for a given deflection limit can be calculated even when using stress-based formulation (1) (e.g., see [50]). A post optimization elastic analysis can also be carried out using the Karamba3D plugin [47], in line with the workflow shown in Fig. 7.

An initial design for this problem was produced manually using steel (yield stress 355MPa; elastic modulus 210GPa; density 7850kg/m^3), as shown in Fig. 19(a). The line load was modelled as point loads, lumped at five points (D, E, C, F and the midpoint of line segment DE). To evaluate the material efficiency of this design, a Stage 1 benchmark solution was found using layout optimization with approx. 2,500 nodes. The benchmark design is shown in Fig. 20(a), revealing a very different structural form. The weight of steel required for this benchmark solution was found to be 0.99t. In contrast the weight of the design developed manually (Fig. 20(b)) was found to be 1.78t, 80% higher than the benchmark. This assumes the use of bespoke sections; if catalogue sections are used (e.g., British

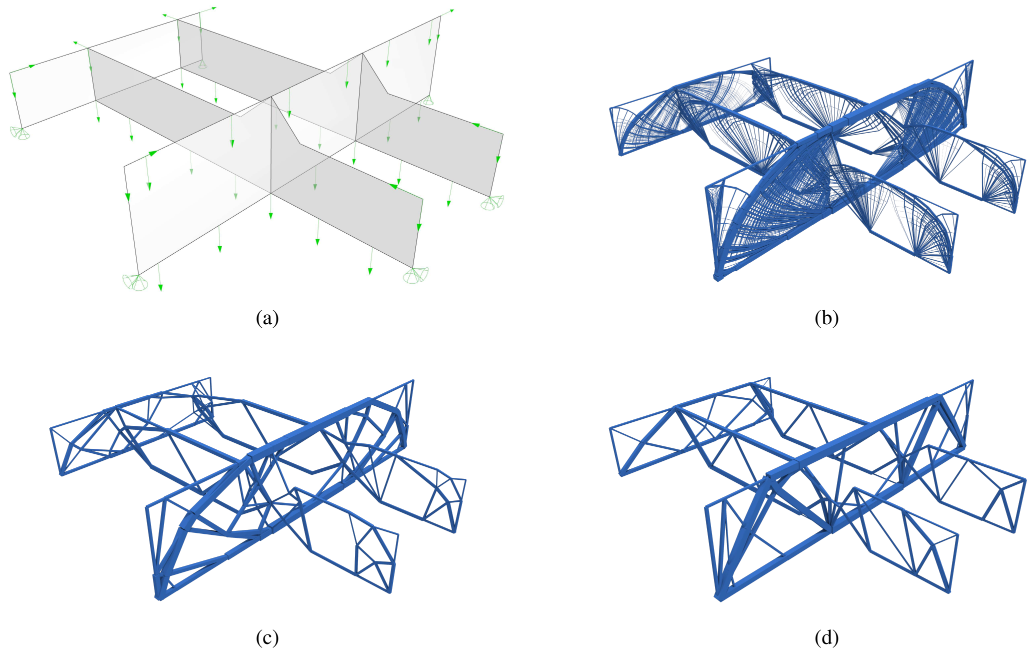


Figure 17: Multiple intersecting trusses example: (a) design domain, load and support conditions; (b) design benchmark obtained with 5,000 nodes, 25.020m^3; (c) rationalized conceptual design derived with 400 nodes, 26.112m^3; (d) more practical design derived with 100 nodes, 27.581m^3

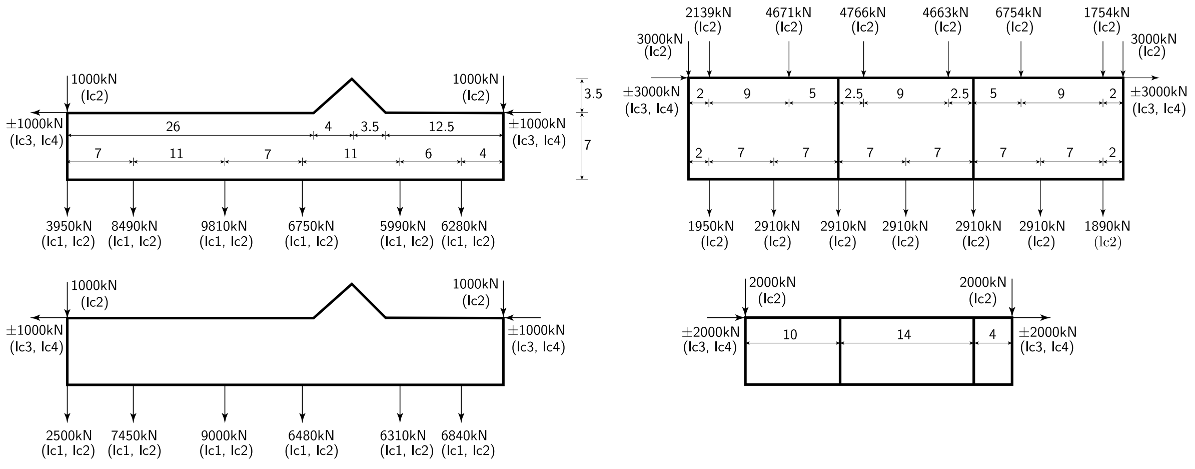
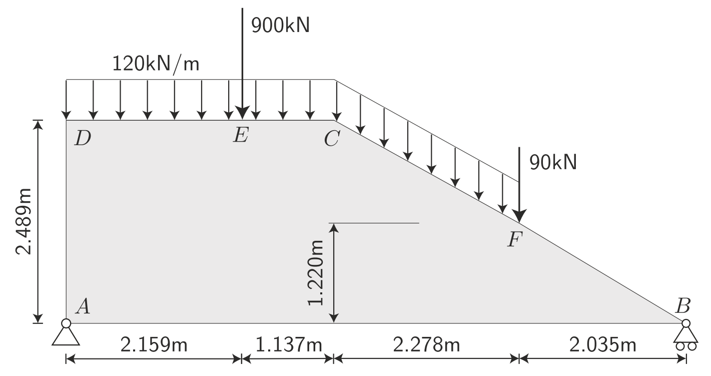
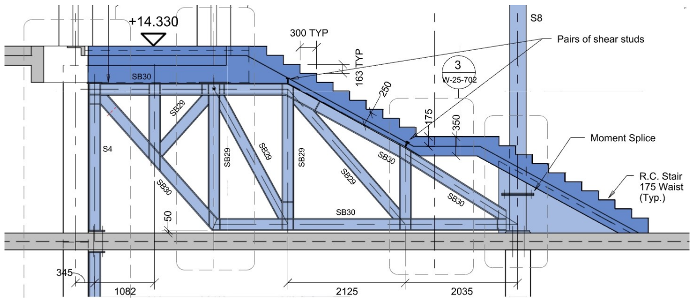


Figure 18: Multiple intersecting trusses example: details of the loads in the four load cases considered (all distances in m).



(a)



(b)

Figure 19: Staircase support design example: (a) problem specification; (b) details of initial manually obtained design.

Standard Universal Column sections, identified via Karamba3D), the total weight increases to 2.35t, more than twice the weight of the benchmark design.

To improve the material efficiency of the manually derived layout, a geometry optimization rationalization process can be performed. This reduces the weight of the structure down to 1.39t and 1.67t, when using bespoke and catalogue sections respectively, Fig. 20(c). However, to take advantage of all the available design freedom, the proposed global-local framework workflow can be employed, using approx. 100 nodes in the Stage 1 nodal grid. The resulting structure is shown in Fig. 20(d), which, though much simpler, closely resembles the benchmark design in Fig. 20(a). If bespoke sections are used, its weight is only slightly (3%) heavier than the benchmark; however, when catalogue member sections are used the weight increases more significantly (by 42%). This design (weight = 1.44t) is however still 38.7% lighter than the initial manually produced design.

Finally, to take full advantage of the parametric modelling functionality of the Rhino/Grasshopper environment, the initial design assumptions can be changed to examine various 'what if' scenarios. Here the potential benefit of introducing a horizontal support at the top left of the design domain will be considered (i.e., at point D in Fig. 19). The magnitude of the reaction force can be constrained in the optimization by adding bounded force variables to $\mathbf{f}^{(w)}$ in problem (1)(b). Therefore, by varying the allowable support reaction force, a range of design options can be generated rapidly. As the limiting support force increases, the benchmark weight can be observed to reduce gradually until the reaction force that can be resisted is large enough not to influence the results; see Fig. 21(f). Whilst the simplified designs are nearly as efficient as the benchmarks, when catalogue sections are used the associated weight increase is significant, as was observed in the case of the example shown in Fig. 20(d). Nevertheless, the solutions help inform the designer of the trade-off associated with each design decision, helping them to identify the best design for use in practice.

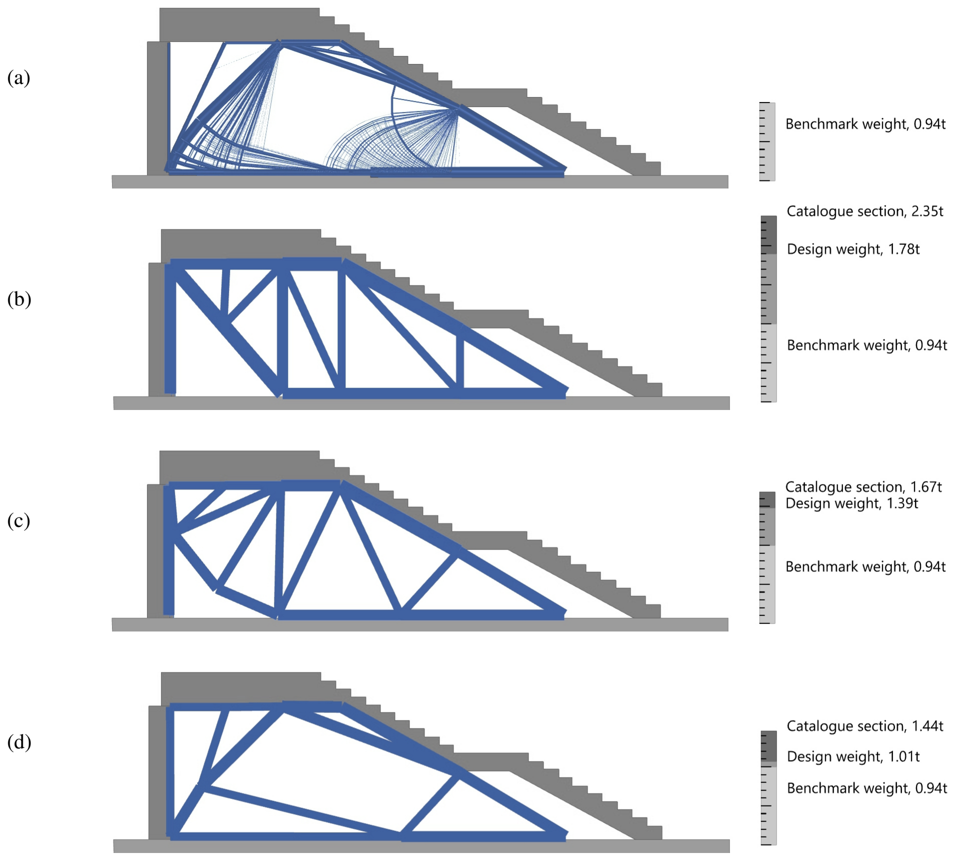


Figure 20: Staircase support design example: comparison for the same deflection limit of 1cm, with vertical bars showing weight of the benchmark, simplified design, and that after adopting catalogue sections: (a) benchmark design; (b) initial design proposed by designer; (c) limited design freedom, optimized solution using the provided initial layout; (d) full design freedom, optimized solution using the proposed framework.

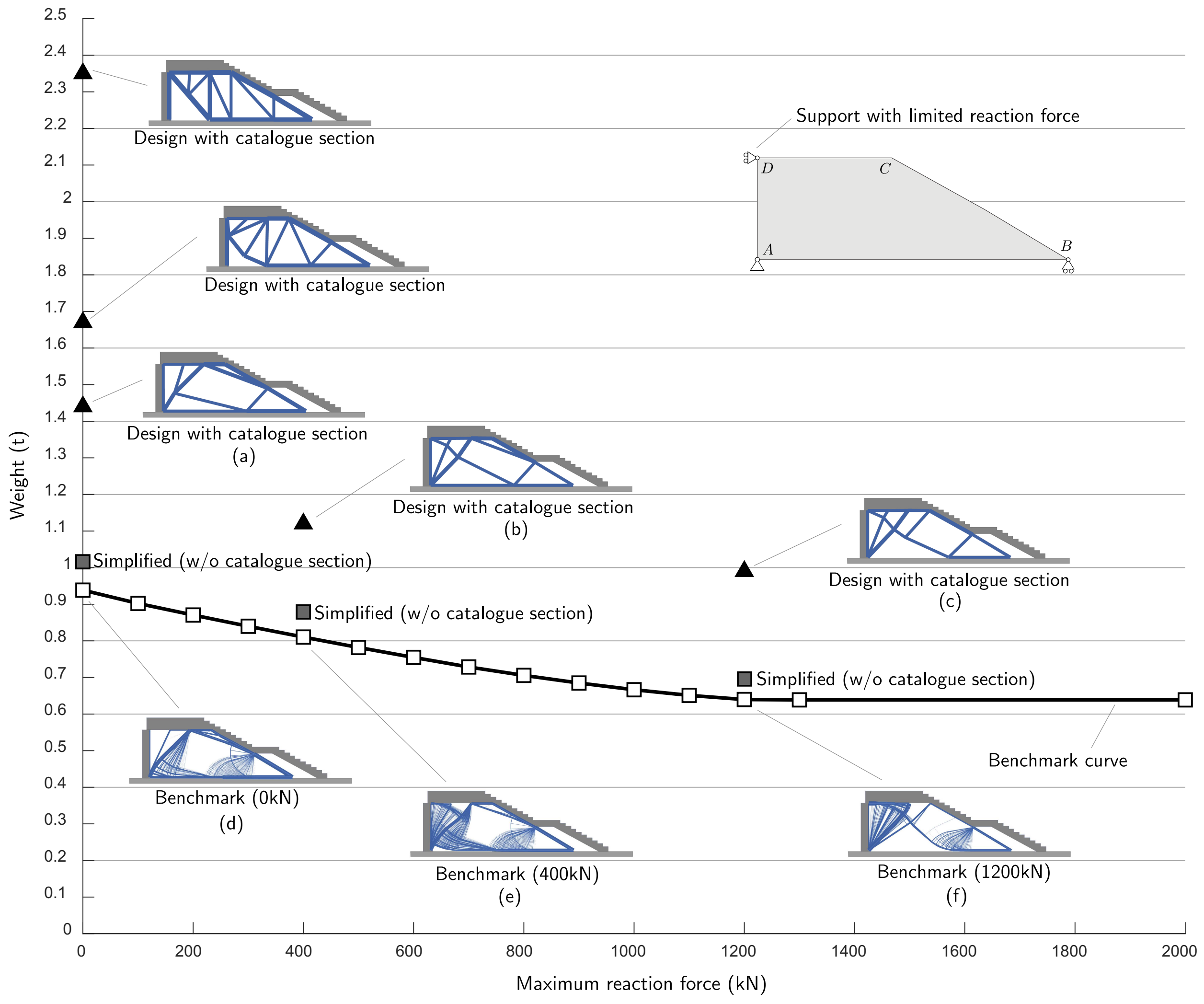


Figure 21: Staircase support design example: alternative designs obtained by adding a horizontal wall support at point D with varying allowable reaction forces, (a)-(c) simplified designs with catalogue sections; (d)-(f) corresponding benchmarks designs.

5. Conclusions

In this paper, a two stage global-local optimization framework has been proposed that uses layout and geometry optimization methods in sequence to identify conceptual designs for building and bridge structures. The main conclusions are as follows:

- Layout optimization is a highly efficient numerical method for identifying truss structures that consume as little material as possible. This is particularly important at present, given the high embodied carbon associated with common construction materials and the pressing need to reduce the carbon footprint associated with the built environment in the face of the current climate emergency. Problems with significant design freedom can be tackled, with the most materially-efficient designs quickly identified from amongst vast numbers of alternatives (e.g., problems involving 3 million potential members can be solved in a matter of seconds; see Fig. 9). These designs are beyond the reach of meta-heuristic optimization methods. When the design domain is discretized using a relatively dense nodal grid, highly accurate solutions suitable for use as design benchmarks (e.g., $< 0.1\%$ error to the theoretical minimum) can be generated. When coarser nodal grids are used, layout optimization can be used to generate designs that can subsequently be made more practical and can serve as inspiration to designers at the conceptual design stage.
- This observation has stimulated the development of a global-local optimization framework: in Stage 1 a benchmark design is generated via layout optimization, with linear optimization used to obtain a globally optimal solution to a simplified problem that neglects various practical considerations; in Stage 2 the practicality of the design is improved via the use of non-linear optimization methods. This means that more rational forms can be identified, which take account of considerations such as Euler buckling. Although the Stage 2 solutions are only locally optimal, they can be generated rapidly, allowing the influence of various other design considerations to be readily evaluated. Significantly, given the availability of the benchmark established in Stage 1, the structural efficiency of a given practical design can readily be evaluated. This also means that when examining various 'what if' scenarios as part of the design process, inefficient designs are clearly identifiable.
- The framework allows rapid exploration of the design space at the initial, conceptual, design stage. The framework has been incorporated in the Peregrine plugin for the Rhino / Grasshopper parametric modelling environment for use by practitioners. Various design examples have been presented in the paper to demonstrate the power and flexibility of the approach developed. In a real-world design problem involving a staircase support structure it is demonstrated that large material savings can be achieved by using the Peregrine plugin (38.6% reduction).

Finally, the flexibility of the proposed framework has been demonstrated via introduction in the Peregrine plugin [43] of additional post-processing tools that are beyond the scope of the present paper. These tools have been designed to facilitate further improvements in the practicality of the generated designs (e.g., by automatically simplifying the designs and/or including global stability constraints).

6. Acknowledgements

The financial support provided by the Engineering and Physical Research Council (EPSRC) for projects EP/N023471/1 and EP/N023269/1 and a follow-up Knowledge Exchange project is gratefully acknowledged. The development of the Peregrine Rhino / Grasshopper plugin has also been supported by the INTEGRADDE project, funded from the European Union's Horizon 2020 research and innovation programme under grant agreement No 820776.

References

[1] Holland J (1975) Adaptation in natural and artificial systems. Michigan: The University of Michigan Press
[2] Kirkpatrick S, Gelatt Jr CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680
[3] Dorigo M, Maniezzo V, Colorni A (1996) Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 26:29–41

[4] Azad S, Bybordiani M, Azad S, Jawad F (2018) Simultaneous size and geometry optimization of steel trusses under dynamic excitations. Structural and Multidisciplinary Optimization 58:2545–2563, DOI 10.1007/s00158-018-2039-7

[5] Yang XS (2009) Firefly algorithms for multimodal optimization. In: Watanabe O, Zeugmann T (eds) Stochastic Algorithms: Foundations and Applications, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 169–178

[6] Wolpert DH, Macready WG (1997) No free lunch therems for optimization. IEEE Transactions on Evolutionary Computation 1(1):67–82

[7] Mueller CT, Ochsendorf JA (2015) Combining structural performance and designer preferences in evolutionary design space exploration. Automation in Construction 52:70 – 82, DOI 10.1016/j.autcon.2015.02.011

[8] Altair (2019) OptiStruct. URL `https://www.altair.com/optistruct`

[9] Zhou Q, Shen W, Wang J, Zhou YY, Xie YM (2018) Ameba: A new topology optimization tool for architectural design. In: Proceedings of IASS Annual Symposia, International Association for Shell and Spatial Structures (IASS), 19, pp 1–8

[10] Michalatos P, Kaijima S (2019) Millipede. URL `http://www.sawapan.eu`

[11] Tcherniak D, Sigmund O (2001) A web-based topology optimization program. Structural and Multidisciplinary Optimization 22(3):179–187

[12] Aage N, Nobel-Jørgensen M, Andreasen CS, Sigmund O (2013) Interactive topology optimization on hand-held devices. Structural and Multidisciplinary Optimization 47(1):1–6

[13] Larsen S, Sigmund O, Groen J (2018) Optimal truss and frame design from projected homogenization-based topology optimization. Structural and Multidisciplinary Optimization 57(4):1461–1474

[14] Zakian P, Kaveh A (2020) Topology optimization of shear wall structures under seismic loading. Earthq Eng Eng Vib 19:105–116

[15] Stromberg LL, Beghini A, Baker WF, Paulino GH (2012) Topology optimization for braced frames: Combining continuum and beam/column elements. Engineering Structures 37:106–124, DOI 10.1016/j.engstruct.2011.12.034

[16] Dorn WS, Gomory RE, Greenberg HJ (1964) Automatic design of optimal structures. Journal de Mècanique 3:25–52

[17] Gilbert M, Tyas A (2003) Layout optimization of large-scale pin-jointed frames. Engineering Computations 20(8):1044–1064

[18] 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

[19] Sokół T (2011) A 99 line code for discretized Michell truss optimization written in Mathematica. Structural and Multidisciplinary Optimization 43(2):181–190

[20] Zegard T, Paulino GH (2014) GRAND—ground structure based topology optimization for arbitrary 2d domains using MATLAB. Structural and Multidisciplinary Optimization 50(5):861–882

[21] He L, Gilbert M, Song X (2019) A Python script for adaptive layout optimization of trusses. Structural and Multidisciplinary Optimization 60(2):835–847, DOI 10.1007/s00158-019-02226-6

[22] Fairclough HE, He L, Pritchard TJ, Gilbert M (2021) LayOpt: an educational web-app for truss layout optimization. Structural and Multidisciplinary Optimization 64:2805–2823

[23] Park P, Gilbert M, Tyas A, Popovic-Larsen O (2012) Potential use of structural layout optimization at the conceptual design stage. International Journal of Architectural Computing 10(1):13–32

[24] Fairclough H, Gilbert M, Thirion C, Tyas A, Winslow P (2019) Optimisation-driven conceptual design: case study of a large transfer truss. The Structural Engineer October:20–26

[25] Hayashi K, Ohsaki M (2019) Fdmopt: force density method for optimal geometry and topology of trusses. Advances in Engineering Software 133:12–19

[26] Zegard T, Hartz C, Mazurek A, Baker WF (2020) Advancing building engineering through structural and topology optimization. Structural and Multidisciplinary Optimization pp 1–21

[27] Weldeyesus AG, Gondzio J, He L, Gilbert M, Shepherd P, Tyas A (2019) Adaptive solution of truss layout optimization problems with global stability constraints. Structural and Multidisciplinary Optimization 60:2093—-2111, DOI 10.1007/s00158-019-02312-9

[28] Weldeyesus AG, Gondzio J, He L, Gilbert M, Shepherd P, Tyas A (2020) Truss geometry and topology optimization with global stability constraints. Structural and Multidisciplinary Optimization 62:1721—-1737, DOI 10.1007/s00158-020-02634-z

[29] Poulsen PN, Olesen JF, Baandrup M (2020) Truss optimization applying finite element limit analysis including global and local stability. Structural and Multidisciplinary Optimization 62(1):41–54

[30] Vanderbei RJ (2001) Linear programming: foundations and extensions, 2nd edn. Springer Verlag

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

[32] Bolbotowski K, He L, Gilbert M (2018) Design of optimum grillages using layout optimization. Struct Multidisc Optim 58(3):851–868

[33] Michell AGM (1904) The limits of economy of material in frame-structures. Philosophical Magazine 8:589–597

[34] Tyas A, Gilbert M, Pritchard T (2006) Practical plastic layout optimization of trusses incorporating stability considerations. Computers & Structures 84(3-4):115–126

[35] Achtziger W (1999) Local stability of trusses in the context of topology optimization. part ii: a numerical approach. Structural Optimization 17(4):247–258

[36] Gurobi (2018) Gurobi optimizer reference manual. URL `http://www.gurobi.com`

[37] MOSEK (2015) The MOSEK optimization toolbox for MATLAB. Version 7.1. URL `http://docs.mosek.com/7.1/toolbox/index.html`

[38] He L, Gilbert M (2015) Rationalization of trusses generated via layout optimization. Structural and Multidisciplinary Optimization 52(4):677–694

[39] Achtziger W (2007) On simultaneous optimization of truss geometry and topology. Structural and Multidisciplinary Optimization 33(4-5):285–304

[40] Descamps B, Coelho RF (2013) A lower-bound formulation for the geometry and topology optimization of truss structures under multiple loading. Structural and Multidisciplinary Optimization 48(1):49–58

[41] Achtziger W (1999) Local stability of trusses in the context of topology optimization part II a numerical approach. Structural Optimization 17(4):247–258, DOI 10.1007/BF01207000

[42] Schwarz J, Chen T, Shea K, Stanković T (2018) Efficient size and shape optimization of truss structures subject to stress and local buckling constraints using sequential linear programming. Structural and Multidisciplinary Optimization 58(1):171–184, DOI 10.1007/s00158-017-1885-z

[43] He L, Pritchard T, Maggs J, Gilbert M, Lu H (2021) Peregrine user manual, v6.0. LimitState Ltd, Sheffield, URL https://www.food4rhino.com/app/peregrine

[44] Wächter A, Biegler LT (2006) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming 106:25–57

[45] Terlaky T (2013) Interior point methods of mathematical programming, vol 5. Springer Science & Business Media

[46] He L, Gilbert M, Shepherd P, Ye J, Koronaki A, Fairclough HE, Davison B, Tyas A, Gondzio J, Weldeyesus AG (2018) A new conceptual design optimization tool for frame structures. In: Proceedings of IASS Annual Symposia, International Association for Shell and Spatial Structures (IASS), 19, pp 1–8

[47] Preisinger C (2013) Linking structure and parametric geometry. Architectural Design 83(2):110–113

[48] Fairclough H, Gilbert M, Thirion C, Tyas A (2018) Balancing complexity and structural efficiency in the design of optimized trusses. In: Mueller C, Adriaenssens S (eds) Proceedings of the IASS Symposium 2018

[49] Achtziger W, Bendsøe M, Ben-Tal A, Zowe J (1992) Equivalent displacement based formulations for maximum strength truss topology design. IMPACT of Computing in Science and Engineering 4(4):315 – 345

[50] Baker W, Beghini L, Mazurek A, Carrion J, Beghini A (2015) Structural innovation: Combining classic theories with new technologies. Engineering Journal 52:203–217