

Optimal Design of Frame Structures with Mixed Categorical and Continuous Design Variables Using the Gumbel-Softmax Method*

Mehran Ebrahimi^{†,a}, Hyunmin Cheong^a, Pradeep Kumar Jayaraman^a, and Farhad Javid^a

^aAutodesk Research, 661 University Avenue, Toronto, ON M5G 1M1, Canada

Abstract

In optimizing real-world structures, due to fabrication or budgetary restraints, the design variables may be restricted to a set of standard engineering choices. Such variables, commonly called categorical variables, are discrete and unordered in essence, precluding the utilization of gradient-based optimizers for the problems containing them. In this paper, incorporating the Gumbel-Softmax (GSM) method, we propose a new gradient-based optimizer for handling such variables in the optimal design of large-scale frame structures. The GSM method provides a means to draw differentiable samples from categorical distributions, thereby enabling sensitivity analysis for the variables generated from such distributions. The sensitivity information can greatly reduce the computational cost of traversing high-dimensional and discrete design spaces in comparison to employing gradient-free optimization methods. In addition, since the developed optimizer is gradient-based, it can naturally handle the simultaneous optimization of categorical and continuous design variables. Through three numerical case studies, different aspects of the proposed optimizer are studied and its advantages over population-based optimizers, specifically a genetic algorithm, are demonstrated.

Keywords Categorical design variables, Gumbel-Softmax method, Frame structures, Structural optimization, Differentiable sampling, Sensitivity analysis

1 Introduction

The application of frame structures composed of interconnected beams and columns is ubiquitous in various fields of engineering making their simulation and optimization attractive subjects of interest in both academia and industry [1–4]. Commonly, the finite element (FE) analysis of frame structures is performed by modeling their components with beam elements which are capable of undergoing longitudinal (axial), transverse (bending) and torsional deformations, unlike truss

*This is a preprint of an article published in *Structural and Multidisciplinary Optimization*. The final authenticated version is available online at: <https://doi.org/10.1007/s00158-024-03745-7>.

[†]E-mail address: mehran.ebrahimi@autodesk.com

elements that can handle only the first one. Accurately capturing these deformation modes and their interactions makes the simulation of frame structures computationally more demanding than that for truss structures [4]. The difference becomes particularly more noticeable and exceedingly cumbersome in the optimal design of large-scale frame structures which has led to considerably fewer extensive investigations of this topic for such structures in the literature.

The focus of the present paper is the optimal design of frame structures with mixed categorical and continuous design variables. Categorical design variables are inherently discrete and their values belong to an unordered set of available choices. Examples of such variables in frame structures include beam cross-sections (profiles) and materials. In real-world applications, only a limited number of cross-sectional profiles and material choices may be available to design a structure due to various limitations such as manufacturing process and cost. For instance, the beam material may only be steel or aluminum, and its cross-section may only be I-profile or T-profile with limited options for their geometrical attributes. Therefore, for practical purposes, it is crucial to distinguish between categorical and continuous design variables, as the latter ones are free to take any value within their bounds and are more straightforward to handle.

The challenges associated with considering categorical design variables are not exclusive to frame structures and have been explored significantly in other application areas [5–9]. Of particular relevance to this paper are truss structures and the techniques developed for handling categorical variables therein. A worthwhile review of the available techniques for truss structures can be found in [10]. Although frame and truss structures behave differently, they bear a close resemblance in the way they are designed. Thus, the methodology and arguments presented in this paper are directly applicable to truss structures as well.

Broadly, existing optimization techniques applied to structural problems with categorical variables can be classified into gradient-free and gradient-based schemes. The majority of methods developed for these problems utilize gradient-free optimizers such as the branch-and-bound [2, 11], genetic algorithm (GA) [12, 13], simulated annealing [14, 15] and particle swarm optimization [16, 17], among others. It is well-known that due to the combinatorial nature of problems with categorical design variables, the computational cost associated with exploring their design space, which typically grows exponentially with the number of variables, becomes intractable for large-scale problems using the gradient-free optimization routines [18–20]. Hence, most cases studied in the literature involve structures with a small number of components and a few (predominantly two) categorical choices per element (component).

Employing gradient-based optimization techniques can be considered as a remedy to alleviate the scalability issue [10, 19, 21]. However, as the categorical variables are discrete and more importantly unordered, computing the gradients of objective and constraint functions with respect to these variables is problematic. There are hence rare precedents of incorporating gradient-based optimizers in the optimal design of beam and truss structures with categorical design variables [22–25]. The proposals in this area apply some form of approximation to compute the function gradients with respect to the categorical variables (e.g., relaxation of discrete to continuous variables [23], which only makes sense if the variables are ordered) along with rounding techniques to enforce the discreteness of the categorical variables. In another body of work [22, 24], the weights associated

with each categorical choice, in their case material types, are treated as design variables. Subsequently, in [22] the Heaviside function has been applied to penalize the intermediate weight values to a particular categorical choice.

One can think of several scenarios where these approaches fall short and a more robust technique is desired. For instance, if cross-sectional profiles are design variables in a problem, say I-, T- and U-profiles, it is mathematically challenging (if not impossible) to interpret the meaning of the gradient of a function with respect to these profiles in a direct sense. One possible solution to this problem is to work with the attributes of categorical variables. For example, instead of operating directly on the cross-sectional profiles, their areas can be taken as design variables. However, this becomes troublesome when categorical variables have multiple attributes influencing the optimization progress in a conflicting manner, such as for a cantilever beam under gravity where its cross-sectional area and its second moment of area can affect the beam's deflection in opposing ways. Furthermore, relaxing the discrete variables to continuous ones—inspired by the SIMP method for the topology optimization of solid structures [26]—is often limited to problems wherein categorical variables are ordered or have only two choices.

In this paper, we propose a technique for resolving the aforementioned issues in computing the gradients with respect to categorical variables, thus enabling the use of gradient-based optimizers in structural design problems. This novel optimization scheme is founded upon two main ideas. Firstly, we propose *reparametrizing* the categorical design variables and taking the probability of using them in the structure as their corresponding design variables. For instance, if a categorical variable is a beam's cross-section chosen from I-, T- and U-profiles, we propose taking the probabilities of having I-, T- and U-profiles for that beam as its corresponding design variables. As a result, instead of discrete variables, the optimizer deals with continuous ones. Consequently, the optimization functions must be reparameterized in terms of the probabilities and their sensitivity analysis must be carried out with respect to these probabilities, the details of which are presented in the forthcoming sections. Obviously, defining functions in terms of probabilities makes them probabilistic (i.e., stochastic) and in order to compute their values a sampling process is involved, which is in general nondifferentiable. Therefore, secondly, we propose using the Gumbel-Softmax (GSM) technique for making the sampling process differentiable [27, 28].

The GSM method is a relaxed version of the original Gumbel-Max (GM) method [29] that provides a simple mechanism for drawing differentiable samples from a categorical probability distribution parameterized by the unnormalized log-probabilities of the classes (choices) of that categorical variable [30]. In recent years, since the seminal papers [27] and [28], the GSM method and its variants have garnered significant attention in the machine learning community (e.g., [31–34]). The intrinsic commonalities between combinatorial problems in machine learning and (structural) engineering applications prompted us to explore the utilization of the GSM technique in frame structures. To the best of the authors' knowledge, this study is the first of its kind to employ the GSM method in optimization problems involving mixed continuous and categorical design variables.

The remainder of the present paper is organized as follows. Section 2 introduces the general problem formulation and the notations involved. In Section 3, we present the GSM method; the pivotal ingredient of the proposed optimization technique in this study. Then, in Section 4, the

details of the optimizer are laid out. Three numerical examples are provided in Section 5 to assess the performance of the developed optimizer and discuss its various aspects. Finally, the paper is concluded by making a few remarks about our proposal in this article and potential considerations for further improvement.

2 Problem statement

The optimal design problem of a frame structure can be posed as follows. Given a frame structure, find the optimum values of continuous design variables as well as optimum choices for the categorical design variables such that the target structural performance function is achieved and the governing linear elasticity equations due to a FE discretization and imposed constraints are satisfied. Accordingly, the topology of the structure does not change throughout the optimization routine. Denoting $\mathbf{x} := [x_1, \dots, x_{n_x}]^T$ and $\mathbf{c} := [c_1, \dots, c_{n_c}]^T$ as the vectors of continuous and categorical design variables, respectively, the optimization problem mathematically reads

$$\begin{aligned} & \min_{\mathbf{x}, \mathbf{c}} J(\mathbf{u}(\mathbf{x}, \mathbf{c}), \mathbf{x}, \mathbf{c}) \\ \text{subject to } & \mathbf{K}(\mathbf{x}, \mathbf{c}) \mathbf{u}(\mathbf{x}, \mathbf{c}) = \mathbf{f}(\mathbf{x}, \mathbf{c}), \\ & \mathbf{g}(\mathbf{u}(\mathbf{x}, \mathbf{c}), \mathbf{x}, \mathbf{c}) \leq 0, \\ & lb_i \leq x_i \leq ub_i, \quad i = 1, \dots, n_x, \\ & c_i \in \{1, \dots, N_i\}, \quad i = 1, \dots, n_c. \end{aligned} \tag{1}$$

In this equation, J is the scalar-valued objective function, $\mathbf{g} \in \mathbb{R}^{n_g}$ represents the vector of optimization constraint functions, $\mathbf{K} \in \mathbb{R}^{n_u \times n_u}$ indicates the structure's stiffness matrix, $\mathbf{f} \in \mathbb{R}^{n_u}$ specifies the external load vector, and $\mathbf{u} \in \mathbb{R}^{n_u}$ denotes the vector of nodal displacements (state variables). Also, lb_i and ub_i , $i = 1, \dots, n_x$, refer to the lower and upper bounds of the continuous variable x_i , respectively. Furthermore, N_i , $i = 1, \dots, n_c$, is the number of choices available for the categorical variable c_i . Therefore, without loss of generality, we assume the existing choices for each c_i , $i = 1, \dots, n_c$, are indexed as $\{1, \dots, N_i\}$.

Depending on the problem at hand and the capabilities of the FE solver, J and \mathbf{g} can be linear or nonlinear functions of their arguments. Other than continuity and (at least first order) differentiability, we make no further assumptions on J and \mathbf{g} . Given a structure, some examples of these functions include mass, compliance, displacement, von Mises stress and natural frequencies. Since the response of a structure is affected by the values of the design variables, the nodal displacements \mathbf{u} are implicit functions of \mathbf{x} and \mathbf{c} . As mentioned in the introduction, we aim to use gradient-based optimizers to solve the problem in (1). However, due to the challenges elaborated earlier, direct

computation of $\nabla_{\mathbf{c}} J$ and $\nabla_{\mathbf{c}} \mathbf{g}$ gives rise to mathematical hurdles. Thus, we restate (1) as

$$\begin{aligned} & \min_{\mathbf{x}, \boldsymbol{\theta}} J(\mathbf{u}(\mathbf{x}, \boldsymbol{\theta}), \mathbf{x}, \boldsymbol{\theta}) \\ \text{subject to } & \mathbf{K}(\mathbf{x}, \boldsymbol{\theta}) \mathbf{u}(\mathbf{x}, \boldsymbol{\theta}) = \mathbf{f}(\mathbf{x}, \boldsymbol{\theta}), \\ & \mathbf{g}(\mathbf{u}(\mathbf{x}, \boldsymbol{\theta}), \mathbf{x}, \boldsymbol{\theta}) \leq 0, \\ & lb_i \leq x_i \leq ub_i, \quad i = 1, \dots, n_x, \end{aligned} \tag{2}$$

where $\boldsymbol{\theta} := [\theta_{1,1}, \dots, \theta_{1,N_1}, \theta_{2,1}, \dots, \theta_{2,N_2}, \dots, \theta_{n_c,1}, \dots, \theta_{n_c,N_{n_c}}]^T$ denotes the vector of unnormalized log-probabilities of the categorical variables formulated in (3). The first and second indices in $\theta_{i,j}$ refer, respectively, to the variable index and its available choices' index. If for a categorical variable i , the probability of using its N_i choices in the structure is defined by $\mathbf{p}_i := [p_{i,1}, \dots, p_{i,N_i}]^T$, then

$$\theta_{i,j} = \text{logit}(p_{i,j}) := \ln \left(\frac{p_{i,j}}{1 - p_{i,j}} \right), \quad j = 1, \dots, N_i, \tag{3}$$

where $p_{i,j}$ is the likelihood of choice j for categorical variable i . Here, we opt for using unnormalized log-probabilities $\theta_{i,j}$, $i = 1, \dots, n_c$, $j = 1, \dots, N_i$, as the design variables instead of their corresponding probabilities $p_{i,j}$. Once the optimal values of $\theta_{i,j}$'s are found, their corresponding optimal probabilities $p_{i,j}$ can be computed through the Softmax function as

$$p_{i,j} = (\text{softmax}(\boldsymbol{\theta}_i))_j := \frac{\exp(\theta_{i,j})}{\sum_{k=1}^{N_i} \exp(\theta_{i,k})}, \quad i = 1, \dots, n_c, \quad j = 1, \dots, N_i. \tag{4}$$

The benefits of using $\theta_{i,j}$'s are twofold. Firstly, following (3), as opposed to $p_{i,j}$'s that are between 0 and 1, unnormalized log-probabilities $\theta_{i,j}$'s are unbounded continuous variables and can take any value between $-\infty$ and $+\infty$. Secondly, for each categorical variable i the probabilities must satisfy $\sum_{j=1}^{N_i} p_{i,j} = 1$ at each optimization iteration, whereas working with its associated $\theta_{i,j}$'s and eventually converting them to $p_{i,j}$'s using (4) would naturally satisfy this constraint. This reduces the number of optimization constraints significantly, especially in large-scale problems.

In order to utilize gradient-based optimizers, one needs to compute the sensitivities (gradients) of objective and constraint functions with respect to $\boldsymbol{\theta}$ and \mathbf{x} (i.e., $\nabla_{\mathbf{x}} J$, $\nabla_{\mathbf{x}} \mathbf{g}$, $\nabla_{\boldsymbol{\theta}} J$ and $\nabla_{\boldsymbol{\theta}} \mathbf{g}$). Most often, in structural engineering problems, J and \mathbf{g} are not explicit functions of the probabilities $\boldsymbol{\theta}$. For instance, for a categorical variable corresponding to a beam's material, J and \mathbf{g} are functions of the properties of the selected material such as its Young's modulus or density, not the material itself as an object. Therefore, for all categorical variables, a sampling process must be performed independently to draw a sample from each of their existing choices based on their probabilities and then compute the values of J and \mathbf{g} using that sample. The sampling process, however, in general breaks down the differentiability of these functions. This issue can be resolved by incorporating the GSM method. In the subsequent section, the GSM method and its application in having a differentiable sampling procedure are introduced.

3 Differentiable sampling via the GSM method

For conciseness, the subject of the current section is presented only for J . The same idea can be directly applied to the constraint functions \mathbf{g} in a similar fashion. The goal is to compute the value of J and its gradient with respect to the vector of unnormalized log-probabilities $\boldsymbol{\theta}$. For the sake of brevity, we assume $\boldsymbol{\theta}$ is associated with just one categorical variable with N choices and J is only a function of $\boldsymbol{\theta}$ (and not \mathbf{x} and \mathbf{u} as in the previous section). More general cases are provided in Section 4. Note that J is not an explicit function of $\boldsymbol{\theta}$ but rather the continuous attributes (properties) of a sample (i.e., choice) drawn from the categorical distribution function characterized by $\boldsymbol{\theta}$. In other words, once a sample is picked from the categorical distribution of $\boldsymbol{\theta}$, the value of J using the properties of this sample is computed.

A categorical distribution function is a discrete probability distribution specifying the likelihood of the choices (classes) for a categorical random variable [35]. For instance, applying (4), the categorical distribution of $\boldsymbol{\theta}$ is $p_{\boldsymbol{\theta}} := \{\text{softmax}(\boldsymbol{\theta})_1, \dots, \text{softmax}(\boldsymbol{\theta})_N\}$. A *one-hot* sample vector \mathbf{s} can be generated from $p_{\boldsymbol{\theta}}$ utilizing Algorithm 1. The size of \mathbf{s} is N and its entries are all 0 except for the index corresponding to the selected class at which its entry is 1. Suppose the attributes of the selected class are collected in a vector \mathbf{a} . Since J is a function of $\boldsymbol{\theta}$ through the drawn sample and its attributes, it can be formulated as $J(\mathbf{a}(\mathbf{s}(\boldsymbol{\theta})))$. The sensitivity of J with respect to $\boldsymbol{\theta}$ therefore can be written as

$$\nabla_{\boldsymbol{\theta}} J = (\nabla_{\boldsymbol{\theta}} \mathbf{a})^T \nabla_{\mathbf{a}} J = (\nabla_{\mathbf{s}} \mathbf{a} \quad \nabla_{\boldsymbol{\theta}} \mathbf{s})^T \nabla_{\mathbf{a}} J. \quad (5)$$

The sample vector \mathbf{s} , generated by incorporating the *common* sampling process described in Algorithm 1, is not differentiable with respect to $\boldsymbol{\theta}$ creating issues in computing $\nabla_{\boldsymbol{\theta}} \mathbf{s}$ and consequently $\nabla_{\boldsymbol{\theta}} J$ in (5). The step causing nondifferentiability is Step 5. The GSM method resolves this issue by a simple yet remarkably efficient solution [29, 36].

3.1 The GSM method

We first introduce the GM method from which the GSM method is derived (recall that GM stands for Gumbel-Max). Define $\mathcal{I}(i)$ as an operator that returns a one-hot vector of size N with an entry of 1 at index i and zero elsewhere. In the GM method, a sample vector $\bar{\mathbf{s}}$ is generated by

$$\bar{\mathbf{s}} = \mathcal{I} \left(\arg \max_{i \in \{1, \dots, N\}} \left(\theta_i + G^{(i)} \right) \right), \quad (6)$$

where θ_i , $i = 1, \dots, N$, are unnormalized log-probabilities of the classes and $G^{(i)}$, $i = 1, \dots, N$, are independent and identically distributed samples (noises) generated from the standard Gumbel distribution $\text{Gumbel}(0, 1)$. For each class index i , the noise $G^{(i)}$ can be obtained by independently sampling $\text{Gumbel}(0, 1)$ through drawing a uniformly distributed random real number r and setting $G^{(i)} = -\ln(-\ln(r))$ [27]. Therefore, instead of employing Algorithm 1, which is the common, nondifferentiable way of drawing samples from a categorical distribution, $\bar{\mathbf{s}}$ can be drawn from $\boldsymbol{\theta}$

Algorithm 1: The common, nondifferentiable way of drawing samples from a categorical distribution

Input: Unnormalized log-probabilities $\boldsymbol{\theta} \in \mathbb{R}^N$ associated with a categorical variable.

Output: One-hot sample vector \mathbf{s} .

- 1 Initialize $\mathbf{s} \in \mathbb{R}^N$ by zeros;
- 2 Compute the probabilities p_i , $i = 1, \dots, N$, using (4) and put them into a (not necessarily ordered) set $p_{\boldsymbol{\theta}} := \{p_1, \dots, p_N\}$;
- 3 Calculate the corresponding cumulative distribution function as

$$\bar{p}_{\boldsymbol{\theta}} := \{0, p_1, p_1 + p_2, \dots, \sum_{i=1}^{N-1} p_i\};$$

- 4 Choose a uniformly distributed random real number $r \in [0, 1]$;
 - 5 Find the largest index i in $\bar{p}_{\boldsymbol{\theta}}$ such that $(\bar{p}_{\boldsymbol{\theta}})_i \leq r$;
 - 6 Set the entry in \mathbf{s} corresponding to index i to 1;
-

by perturbing each entry of $\boldsymbol{\theta}$ via independently adding a Gumbel noise to them, selecting the index corresponding to the largest perturbed entry of $\boldsymbol{\theta}$, and returning the sample associated with that index. The sample vector $\bar{\mathbf{s}}$ produced this way has theoretically the same distribution as the categorical distribution associated with $\boldsymbol{\theta}$ (i.e., $\bar{\mathbf{s}} \sim p_{\boldsymbol{\theta}}$). For the sake of conciseness, the proof of this statement is provided in Appendix A. It is of the utmost importance to note that in (6) only the noise generated from the standard Gumbel distribution—and not any other types of probability distributions—results in a sample vector $\bar{\mathbf{s}} \sim p_{\boldsymbol{\theta}}$. Interested readers may refer to [27, 29, 30] for further details about the GM method and other versions of the proof.

Since the arg max function in (6) is not differentiable, Jang et al. [27] proposed a continuous, differentiable approximation (relaxation) to this function by introducing a temperature parameter τ and utilizing a so-called soft one-hot sample vector $\tilde{\mathbf{s}}$ with entries

$$\tilde{s}_i := \frac{\exp((\theta_i + G^{(i)})/\tau)}{\sum_{j=1}^N \exp((\theta_j + G^{(j)})/\tau)}, \quad i = 1, \dots, N, \quad (7)$$

or put it shortly $\tilde{\mathbf{s}} = \text{softmax}((\boldsymbol{\theta} + \mathbf{G})/\tau)$ with $\mathbf{G} := [G^{(1)}, \dots, G^{(N)}]^T$. Although \tilde{s}_i entries in (7) have the advantage of being differentiable with respect to θ_i , $i = 1, \dots, N$, there are two concerns with having a soft one-hot sample vector $\tilde{\mathbf{s}}$ in the GSM method. Firstly, due to the applied relaxation, $\tilde{\mathbf{s}}$ does not exactly follow the $p_{\boldsymbol{\theta}}$ distribution, as opposed to $\bar{\mathbf{s}}$ in (6). However, as $\tau \rightarrow 0$, this soft one-hot vector becomes a true one-hot vector and subsequently, the samples generated by the GSM method would have the exact $p_{\boldsymbol{\theta}}$ distribution [27]. Therefore, at the beginning of the optimization routine, one can set τ to a high temperature (in a relative sense) and using an annealing scheme reduce it to a small nonzero value as the optimization progresses. For machine learning applications, the typical range suggested for τ is between 100 to 0.1 (e.g., [33, 37–39]). More details about the annealing scheme and temperature range used in this study are provided

in Section 5.

Secondly, with regard to the concerns facing the GSM method, having a soft one-hot vector $\tilde{\mathbf{s}}$ leads to generating a sample vector that is a combination of all choices in the target categorical variable which oftentimes does not bear any physical realization, causing problems in computing the value of J and its gradients. For example, for the applications considered in this paper, we cannot have a beam cross-sectional profile that is a combination of I- and U-profiles. In such scenarios, the so-called *straight-through* GSM [27] can be adopted to generate a true one-hot sample vector

$$\hat{\mathbf{s}} := \mathcal{I} \left(\arg \max_{i \in \{1, \dots, N\}} (\tilde{s}_i) \right), \quad (8)$$

and to compute the value of J using this vector. Then, for the sensitivity analysis in (5), since $\hat{\mathbf{s}}$ is not differentiable with respect to $\boldsymbol{\theta}$, $\nabla_{\boldsymbol{\theta}} \tilde{\mathbf{s}}$ instead of $\nabla_{\boldsymbol{\theta}} \hat{\mathbf{s}}$ is utilized. As the optimization advances and $\tilde{\mathbf{s}}$ approaches toward being a true one-hot vector, the discrepancy between $\nabla_{\boldsymbol{\theta}} \tilde{\mathbf{s}}$ and $\nabla_{\boldsymbol{\theta}} \hat{\mathbf{s}}$ diminishes. Ultimately, following (5) and generating $\tilde{\mathbf{s}}$ using the GSM method, the gradient of J with respect to θ_j , $j = 1, \dots, N$, reads

$$\nabla_{\theta_j} J = \nabla_{\theta_j} \mathbf{a} \cdot \nabla_{\mathbf{a}} J = (\nabla_{\tilde{\mathbf{s}}} \mathbf{a} \quad \nabla_{\theta_j} \tilde{\mathbf{s}}) \cdot \nabla_{\mathbf{a}} J = \nabla_{\mathbf{a}} J \cdot \sum_{i=1}^N \frac{\partial \mathbf{a}}{\partial \tilde{s}_i} \frac{\partial \tilde{s}_i}{\partial \theta_j}, \quad j = 1, \dots, N, \quad (9)$$

where

$$\frac{\partial \tilde{s}_i}{\partial \theta_j} = \begin{cases} (1 - \tilde{s}_i) \tilde{s}_i, & \text{if } i = j \\ -\tilde{s}_i \tilde{s}_j, & \text{otherwise} \end{cases}, \quad (10)$$

is easily derived using (7). Note that although $G^{(i)}$, $i = 1, \dots, N$, which are random Gumbel noises appear in computing \tilde{s}_i values in (7), they do not cause any differentiability issues in (9) and (10). Assuming $\nabla_{\mathbf{a}} J$ and $\nabla_{\tilde{\mathbf{s}}} \mathbf{a}$ are known, Algorithm 2 encapsulates the steps involved in the GSM method for generating sample vectors $\tilde{\mathbf{s}}$ and $\hat{\mathbf{s}}$, as well as computing the value of a function J that through $\tilde{\mathbf{s}}$ is implicitly a function of unnormalized log-probabilities $\boldsymbol{\theta}$ corresponding to a categorical distribution and eventually calculating $\nabla_{\boldsymbol{\theta}} J$. The details of computing $\nabla_{\mathbf{a}} J$ and $\nabla_{\tilde{\mathbf{s}}} \mathbf{a}$ are laid out in the next section.

4 Optimization methodology

In Section 3, the GSM method was described in its most general form regardless of the application. Henceforth, to establish a more oriented dialogue, we limit the scope to structural problems and optimization functions arising in such applications. Considering the optimization problem stated in (2), in this section, we develop a scheme to compute the sensitivities of the optimization objective and constraint functions (J and \mathbf{g} , respectively) with respect to both \mathbf{x} and $\boldsymbol{\theta}$. Without loss of generality, we only focus on computing $\nabla_{\mathbf{x}} J$ and $\nabla_{\boldsymbol{\theta}} J$. Computing $\nabla_{\mathbf{x}} \mathbf{g}$ and $\nabla_{\boldsymbol{\theta}} \mathbf{g}$ proceeds similarly.

For the structural problems considered in this article, continuous design variables \mathbf{x} could be components' cross-sectional dimensions (or properties), their orientation, length and material properties.

Algorithm 2: The GSM method for computing the value and gradient of a function of a categorical distribution

Input: Function J and its gradients $\nabla_{\mathbf{a}} J$ and $\nabla_{\tilde{\mathbf{s}}} \mathbf{a}$, unnormalized log-probabilities $\boldsymbol{\theta} \in \mathbb{R}^N$ associated with a categorical variable, temperature parameter τ .

Output: J and $\nabla_{\boldsymbol{\theta}} J$.

- 1 Generate samples $G^{(i)}$, $i = 1, \dots, N$, from the standard Gumbel distribution $\text{Gumbel}(0, 1)$;
 - 2 Compute \tilde{s}_i , $i = 1, \dots, N$, through (7);
 - 3 Calculate $\hat{\mathbf{s}}$ using (8);
 - 4 Get J using $\hat{\mathbf{s}}$;
 - 5 Compute $\partial \tilde{s}_i / \partial \theta_j$, $i, j = 1, \dots, N$, using (10);
 - 6 Get $\nabla_{\boldsymbol{\theta}} J$ via (9);
-

The categorical variables may be material and cross-sectional profile choices of each component. The categorical variables are determined by their corresponding continuous attributes. For example, material choices are characterized through their Young's modulus, Poisson's ratio and density. The objective and constraint functions may be the structure's mass, its compliance, nodal displacement and maximum stress to name a few. The proposed optimization framework and required derivations are elucidated in a general format such that other types of design variables and functions can be incorporated in a straightforward manner.

For the sake of brevity, suppose the aim is to compute the sensitivities of J with respect to a single continuous variable x_l , $l = 1, \dots, n_x$ and unnormalized log-probabilities $\boldsymbol{\theta}_m := [\theta_{m,1}, \dots, \theta_{m,N_m}]^T$ associated with a single categorical variable c_m , $m = 1, \dots, n_c$. Subsections 4.1 and 4.2 present the procedures for computing $\nabla_{x_l} J$ and $\nabla_{\boldsymbol{\theta}_m} J$, respectively. Once sensitivities of J are calculated, they can be utilized for optimizing the frame structures using the optimization approach proposed in Subsection 4.3.

4.1 Sensitivity analysis for continuous variables

Recall from Section 2 that J can explicitly and implicitly be a function of x_l (and $\boldsymbol{\theta}_m$) through \mathbf{u} . Accordingly, the gradient of J with respect to x_l is expressed as

$$\nabla_{x_l} J = \frac{\partial J}{\partial x_l} + \nabla_{x_l} \mathbf{u} \cdot \nabla_{\mathbf{u}} J, \quad l = 1, \dots, n_x, \quad (11)$$

where $\partial J / \partial x_l$ and $\nabla_{\mathbf{u}} J$ which are, respectively, explicit derivatives of J with respect to x_l and \mathbf{u} can be carried out in a simple manner knowing the function formulation, thus not further elaborated in this paper. For example, if $J = 0.5 \mathbf{u}^T \mathbf{K} \mathbf{u}$, then $\partial J / \partial x_l = 0.5 \mathbf{u}^T (\nabla_{x_l} \mathbf{K}) \mathbf{u}$ and $\nabla_{\mathbf{u}} J = \mathbf{K} \mathbf{u}$. The onerous term in (11) is $\nabla_{\mathbf{u}} J$ for which we appeal to the *adjoint* method [21, 40] well-known to be advantageous over the direct differentiation method for optimizing problems with several design variables. The process is presented for a general J that may be a linear or nonlinear function of its arguments. Using the discretized governing equation $\mathbf{K} \mathbf{u} - \mathbf{f} = \mathbf{0}$ and differentiating it with

respect to x_l yields

$$\mathbf{K} \nabla_{x_l} \mathbf{u} + (\nabla_{x_l} \mathbf{K}) \mathbf{u} - \nabla_{x_l} \mathbf{f} = \mathbf{0}, \quad (12)$$

where it is assumed that the external force vector \mathbf{f} is only a function of x_l not \mathbf{u} . Since the left-hand side of (12) is zero, multiplying it by any arbitrary vector of a conforming size to the governing equation also results in zero. Denote this multiplier vector, called the adjoint variable vector, by $\boldsymbol{\lambda}_J \in \mathbb{R}^{n_u}$. Applying it to (12) and employing the result in (11) leads to

$$\nabla_{x_l} J = \frac{\partial J}{\partial x_l} + \nabla_{x_l} \mathbf{u} \cdot \nabla_{\mathbf{u}} J - \boldsymbol{\lambda}_J \cdot (\mathbf{K} \nabla_{x_l} \mathbf{u} + (\nabla_{x_l} \mathbf{K}) \mathbf{u} - \nabla_{x_l} \mathbf{f}), \quad l = 1, \dots, n_x. \quad (13)$$

This equation holds regardless of the value of $\boldsymbol{\lambda}_J$; ergo, it can be chosen such that $\nabla_{x_l} \mathbf{u}$ in (13) vanishes. Meaning

$$\mathbf{K} \boldsymbol{\lambda}_J = \nabla_{\mathbf{u}} J. \quad (14)$$

Once $\boldsymbol{\lambda}_J$ is found through (14), putting it in (13) gives

$$\nabla_{x_l} J = \frac{\partial J}{\partial x_l} - \boldsymbol{\lambda}_J \cdot ((\nabla_{x_l} \mathbf{K}) \mathbf{u} - \nabla_{x_l} \mathbf{f}), \quad l = 1, \dots, n_x. \quad (15)$$

This is the final equation for computing the sensitivity of J with respect to continuous design variables. Note that solving (14) can be carried out quite efficiently as factorizing \mathbf{K} , the arduous step in the solution process, has already been performed in solving the governing equation $\mathbf{K}\mathbf{u} = \mathbf{f}$. Furthermore, there is no dependence on x_l or any other continuous design variable in (14); hence (14) is solved only once (and not for each variable individually), justifying the application of the adjoint method for problems with several design variables. The adjoint equation for each constraint function g_i , $i = 1, \dots, n_g$, is formulated as

$$\mathbf{K} \boldsymbol{\lambda}_{g_i} = \nabla_{\mathbf{u}} g_i. \quad (16)$$

4.2 Sensitivity analysis for categorical variables

With respect to $\boldsymbol{\theta}_m$, the gradient of J using (5) reads

$$\nabla_{\boldsymbol{\theta}_m} J = (\nabla_{\tilde{\mathbf{s}}_m} \mathbf{a} \ \nabla_{\boldsymbol{\theta}_m} \tilde{\mathbf{s}}_m)^T \nabla_{\mathbf{a}} J = \nabla_{\mathbf{a}} J \cdot (\nabla_{\tilde{\mathbf{s}}_m} \mathbf{a} \ \nabla_{\boldsymbol{\theta}_m} \tilde{\mathbf{s}}_m), \quad m = 1, \dots, n_c, \quad (17)$$

where $\tilde{\mathbf{s}}_m$ is the sample vector drawn from the categorical probability distribution characterized by $\boldsymbol{\theta}_m$ employing the GSM method. Calculating $\nabla_{\boldsymbol{\theta}_m} \tilde{\mathbf{s}}_m$ can be performed incorporating the procedure developed in Section 3 and (10). We first focus on handling $\nabla_{\tilde{\mathbf{s}}_m} \mathbf{a}$ and then $\nabla_{\mathbf{a}} J$ in (17). Naturally, each entry of $\tilde{\mathbf{s}}_m$ corresponds to a class which can be characterized by some continuous attributes (properties). If $\tilde{\mathbf{s}}_m$ is associated with an isotropic material its entries are determined by Young's modulus and Poisson's ratio. On the other hand, if $\tilde{\mathbf{s}}_m$ is linked to a cross-sectional profile, the relevant continuous attributes are cross-sectional properties such as area and second moments of area. It is important to note that all the choices considered for a categorical variable must be characterized using the same set of attributes.

Suppose $\tilde{\mathbf{s}}_m$ corresponds to a categorical variable with N_m choices. Let $\mathbf{a}_k := [a_{1,k}, \dots, a_{n_m,k}]^T$, $k = 1, \dots, N_m$, be the vector of size n_m containing the continuous attributes of each choice. Also, define $\mathbf{A}_m \in \mathbb{R}^{n_m \times N_m}$ the attribute matrix of this categorical variable as

$$\mathbf{A}_m := \begin{bmatrix} a_{1,1} & \cdots & a_{1,N_m} \\ \vdots & \ddots & \vdots \\ a_{n_m,1} & \cdots & a_{n_m,N_m} \end{bmatrix}. \quad (18)$$

If $\tilde{\mathbf{s}}_m$ is a hard one-hot vector, the attributes of the selected class in this categorical variable can be picked out by

$$\mathbf{a} = \mathbf{A}_m \tilde{\mathbf{s}}_m. \quad (19)$$

As mentioned earlier in Section 3, for running the structural simulation and computing the optimization functions, $\hat{\mathbf{s}}_m$ (given by (8)) instead of $\tilde{\mathbf{s}}_m$ needs to be used in (19). During the sensitivity analysis, however, $\tilde{\mathbf{s}}_m$ itself is employed leading to $\nabla_{\tilde{\mathbf{s}}_m} \mathbf{a} = \mathbf{A}_m$. As the optimization advances, $\tilde{\mathbf{s}}_m$ approaches $\hat{\mathbf{s}}_m$. Finally, inserting this relation into (17) yields

$$\nabla_{\theta_m} J = \nabla_{\mathbf{a}} J \cdot (\nabla_{\tilde{\mathbf{s}}_m} \mathbf{a} \nabla_{\theta_m} \tilde{\mathbf{s}}_m) = \nabla_{\mathbf{a}} J \cdot (\mathbf{A}_m \nabla_{\theta_m} \tilde{\mathbf{s}}_m), \quad m = 1, \dots, n_c. \quad (20)$$

In this equation, $\nabla_{\mathbf{a}} J$ is the sensitivity of J with respect to the \mathbf{a} associated with the continuous attributes of the class with the highest unnormalized log-probability at a given optimization iteration (i.e., the attributes of the sample generated by the straight-through GSM). Since \mathbf{a} is a continuous variable, computing $\nabla_{\mathbf{a}} J$ can be carried out by adopting the adjoint method described in the earlier subsection. In other words,

$$\nabla_{a_i} J = \frac{\partial J}{\partial a_i} - \boldsymbol{\lambda}_J^T ((\nabla_{a_i} \mathbf{K}) \mathbf{u} - \nabla_{a_i} \mathbf{f}), \quad i = 1, \dots, n_m, \quad (21)$$

with the same $\boldsymbol{\lambda}_J$ found by (14). Equation (20) is the final equation for computing the sensitivity of J with respect to θ_m . To further clarify the presented procedure for computing $\nabla_{\theta_m} J$, we provide the following example. Suppose an optimization problem whose categorical design variables are the cross-sectional profiles of its beams. Assume for one of the variables the choices are circular and rectangular profiles each characterized by their area A , second moments of area I_{yy} and I_{zz} and torsion constant K . Also, let $\boldsymbol{\theta} = [\theta_1, \theta_2]^T$ be the unnormalized log-probabilities associated with each profile at a given optimization iteration. For this variable, the attribute matrix \mathbf{A} reads

$$\mathbf{A} := \begin{bmatrix} A_{\text{circle}} & A_{\text{rectangle}} \\ I_{yy\text{circle}} & I_{yy\text{rectangle}} \\ I_{zz\text{circle}} & I_{zz\text{rectangle}} \\ K_{\text{circle}} & K_{\text{rectangle}} \end{bmatrix}. \quad (22)$$

There may be two scenarios for θ_1 and θ_2 at any optimization iteration:

1. $\theta_1 \geq \theta_2$: leading to $\tilde{s}_1 \geq \tilde{s}_2$ using (7) and $\hat{\mathbf{s}} = [1, 0]^T$ according to (8).

2. $\theta_1 < \theta_2$: resulting in $\tilde{s}_1 < \tilde{s}_2$ based on (7) and $\hat{\mathbf{s}} = [0, 1]^T$ utilizing (8).

In the former situation, J and $\nabla_{\mathbf{a}}J$ in (20) are computed employing the circular profile and its continuous attributes, otherwise, the rectangular profile is adopted. Finally, $\nabla_{\boldsymbol{\theta}}J$ is obtained through $\nabla_{\tilde{\mathbf{s}}_m}\mathbf{a} = \mathbf{A}$, finding $\nabla_{\mathbf{a}}J$ via (21) and calculating $\nabla_{\boldsymbol{\theta}}\tilde{\mathbf{s}}$ applying the GSM method. We once again highlight that as the optimization moves toward convergence, $\tilde{\mathbf{s}}$ approaches a true one-hot vector and the errors due to the applied relaxations inherent in the GSM method dwindle.

4.3 Optimization scheme

The proposed optimization routine—named GSMO standing for Gumbel-Softmax optimization—for frame (or truss) structures with mixed categorical and continuous design variables is presented in Algorithm 3. Although this scheme is described for a particular class of structures undergoing linear elasticity behavior, it can be applied with minor modifications to other problems (not necessarily structural) with mixed categorical and continuous design variables governed by a different set of physics equations.

An important fact about GSMO in Algorithm 3 is that there is no stratification between continuous and categorical design variables. In other words, both classes of variables are handled concurrently in each optimization iteration, thus giving no priority to either of them. This is unlike the bilevel optimization routines where categorical and continuous variables are treated at different levels (stages) [25, 41–43]. In such techniques, first, the categorical variables are taken into account and the structure is optimized updating only those. Then, in the second level, the continuous design variables are handled. A shortcoming of these approaches is that each variable type is optimized sequentially at one level at a time, which may lead to underperformance. Nevertheless, a bilevel version of GSMO, named BiGSMO, is provided in Algorithm 4 of Appendix B and assayed against GSMO in the forthcoming section.

5 Case studies

Prior to introducing the case studies, we note a few remarks. Since the GSM method relies on generating Gumbel samples which is a stochastic process, the optimum solution found through GSMO (and BiGSMO) may vary every time the optimization is executed. Therefore, GSMO (and BiGSMO) shares a similarity in this aspect with stochastic optimizers such as GA. Hence, to make a fair comparison with stochastic methods, we run each of the case studies 10 times and report the best, average and standard deviation of optimum solutions. As for the temperature τ in (7) and its annealing scheme, an initial temperature of 100 scaled by 0.9 per iteration is adopted. The minimum temperature is kept at 0.01. Other options for the temperature and its annealing scheme are described in [30]. Also, a step size of 10^{-3} is set for the optimization routine. These choices can be seen as hyper-parameters for GSMO. Moreover, an identical initial probability value is assigned to the available choices of each categorical variable. Further remarks about them are provided in the conclusion section.

Algorithm 3: The GSMO scheme for optimizing frame/truss structures with mixed categorical and continuous design variables

Input: Objective function J , constraint functions \mathbf{g} , continuous design variables \mathbf{x} and their bounds, categorical design variables \mathbf{c} and their available choices, GSM annealing scheme.

Output: Optimum design.

```

1 Initialize  $\mathbf{x}$ ;
2 Initialize  $\boldsymbol{\theta}_i$ ,  $i = 1, \dots, n_c$ , for each categorical variable  $c_i$  by entry values of zero assuming equal probabilities for the choices in each  $c_i$ ;
3 while not converged do
4   for  $i = 1, \dots, n_c$  do
5     Generate Gumbel noises  $G^{(j)}$ ,  $j = 1, \dots, N_i$ , from the standard Gumbel distribution  $\text{Gumbel}(0, 1)$ ;
6     Compute  $(\tilde{\mathbf{s}}_i)_j$ ,  $j = 1, \dots, N_i$ , using (7) and  $\nabla_{\boldsymbol{\theta}_i} \tilde{\mathbf{s}}_i$  via (10);
7     Calculate  $\hat{\mathbf{s}}_i$  through (8);
8   end
9   Solve the governing equations  $\mathbf{K}\mathbf{u} = \mathbf{f}$  using  $\hat{\mathbf{s}}_i$ ,  $i = 1, \dots, n_c$ , and get  $\mathbf{u}$ ;
10  Compute  $J$  and  $\mathbf{g}$  values using  $\hat{\mathbf{s}}_i$ ,  $i = 1, \dots, n_c$ , and  $\mathbf{u}$ ;
11  Calculate  $\nabla_{\mathbf{u}} J$  then find  $\boldsymbol{\lambda}_J$  by solving (14);
12  Compute  $\nabla_{\mathbf{u}} g_i$ ,  $i = 1, \dots, n_g$ , then find  $\boldsymbol{\lambda}_{g_i}$ ,  $i = 1, \dots, n_g$ , by solving (16);
13  Calculate  $\partial J / \partial x_i$ ,  $\partial \mathbf{g} / \partial x_i$ ,  $\partial \mathbf{K} / \partial x_i$  and  $\partial \mathbf{f} / \partial x_i$ ,  $i = 1, \dots, n_x$ ;
14  Get  $\nabla_{\mathbf{x}} J$  and  $\nabla_{\mathbf{x}} \mathbf{g}$  incorporating the corresponding adjoint vectors and (15);
15  for  $i = 1, \dots, n_c$  do
16    Form the attribute matrix  $\mathbf{A}_i$  as in (18);
17    Calculate  $\partial J / \partial \mathbf{a}_i$ ,  $\partial \mathbf{g} / \partial \mathbf{a}_i$ ,  $\partial \mathbf{K} / \partial \mathbf{a}_i$  and  $\partial \mathbf{f} / \partial \mathbf{a}_i$  using the attributes of the selected class for this categorical variable;
18    Compute  $\nabla_{\mathbf{a}_i} J$  and  $\nabla_{\mathbf{a}_i} \mathbf{g}$  utilizing the adjoint vectors found in Steps 11 and 12,
       $\nabla_{\boldsymbol{\theta}_i} \tilde{\mathbf{s}}_i$  found in Step 6 and employing (21);
19    Get  $\nabla_{\boldsymbol{\theta}_i} J$  and  $\nabla_{\boldsymbol{\theta}_i} \mathbf{g}$  through (20);
20  end
21  Update  $\mathbf{x}$  and  $\boldsymbol{\theta}_i$ ,  $i = 1, \dots, n_c$ , using the sensitivities found in Steps 14 and 19;
22 end
```

Unfortunately, the relevant literature to this article lacks a repository of well-documented benchmarks, particularly for beam structures [10]. Nevertheless, to study the performance of GSMO, its results are compared to those reported in [20] using different optimization techniques (which includes GA, particle swarm optimization (PSO) [16], mine blast optimization (MBO) [44], water cycle optimization (WCO) [45], colliding bodies optimization (CBO) [46], differential evolution (DE) [47] and neighborhood search (NS) [20]) for a 72-bar truss structure (first case study) and those produced by a GA scheme for the other case studies. For the implemented GA, the population size of 10 times the number of design variables, crossover rate of 0.9 and mutation rate of

0.1 are selected. Furthermore, the constraints are enforced using a penalty method with a penalty factor of 1000. The maximum number of iterations for both GSMO and GA is set to 100. The case studies with a mix of continuous and categorical variables are also run by BiGSMO considering 10 iterations for both outer and inner loops (Lines 3 and 4 in Algorithm 4). All tests are run on a single desktop computer with an Intel Core i9 Processor at 2.40GHz with 8 cores and 16 threads.

5.1 72-bar truss structure

The 72-bar truss structure [20, 48] depicted in Figure 1 undergoes two load cases. In the first one, a force of $\mathbf{f} = [5000, 5000, 0]^T$ lbf is applied to Node 1, and in the second load case, a force of $\mathbf{f} = [0, 0, -5000]^T$ lbf is exerted on Nodes 1, 2, 3 and 4. The objective function for this problem is to minimize the structure's mass and the design variables are the cross-sectional areas of all trusses (hence 72 variables in aggregate) selected from a discrete set given in Table 1. All members are made of an aluminum alloy with a density of 0.1 lb/in³, Young's modulus of 10⁷ psi and yield stress of 25000 psi. For both load cases, the stress in all trusses must remain below the given yield stress and the displacement of Nodes 1, 2, 3 and 4 along X and Y axes must be bounded by ± 0.25 in.

Figure 2 shows the convergence plot of the mean and standard deviation of the 10 GSMO and GA runs. Accordingly, the optimization routine advances smoothly and reaches convergence quite rapidly using both algorithms. Table 2 presents the optimum solution obtained by GSMO (the best one amongst the 10 runs), GA and those of other approaches reported in [20]. For this problem, since the design variables are only categorical, in essence, there is no difference between GSMO and BiGSMO and the result of the latter is not provided. As can be seen, the best solution generated by GSMO has a slight lead compared to those reported previously. The mean and standard deviation of the 10 optimal results by GSMO are respectively 394.31 and 11.14, which may be taken as an affirmation of the fact that although a sampling process is involved in GSMO, the method consistently finds close optimum solutions (we have reviewed this claim in the subsequent case studies as well). In this problem, perhaps the most significant advantage of GSMO is its computational time. Since GSMO is a gradient-based approach, it requires only one FE solve per

Table 1: Categorical choices of the cross-sectional areas for the 72-bar truss structure

Areas (in ²)
0.111, 0.141, 0.196, 0.250, 0.307, 0.391, 0.442, 0.563, 0.602, 0.766, 0.785, 0.994, 1.000, 1.228, 1.266, 1.457, 1.563, 1.620, 1.800, 1.990, 2.130, 2.380, 2.620, 2.630, 2.880, 2.930, 3.090, 3.130, 3.380, 3.470, 3.550, 3.630, 3.840, 3.870, 3.880, 4.180, 4.220, 4.490, 4.590, 4.800, 4.970, 5.120, 5.740, 7.220, 7.970, 8.530, 9.300, 10.85, 11.50, 13.50, 13.90, 14.20, 15.50, 16.00, 16.90, 18.80, 19.90, 22.00, 22.90, 24.50, 26.50, 28.00, 30.00, 33.50

optimization iteration leading to a total of 100 FE solves for each run. For this example, each GMSO run takes about 0.9 seconds on average on the computer specified earlier.

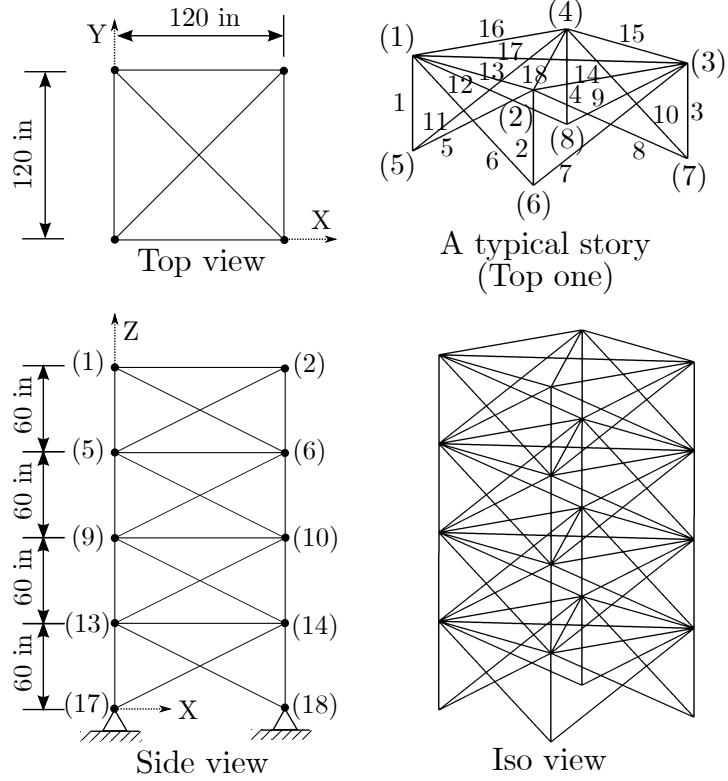


Figure 1: The 72-bar truss structure. The typical story is repeated 4 times along the Z axis. The node and element numbering also follows suit.

5.2 812-bar lattice structure

In this example, the goal is to optimize the lattice structure shown in Figure 3 to have *nonpositive* Poisson's ratios in X and Y directions. The lattice is symmetric with respect to XY, XZ and YZ planes and is composed of 4 by 4 unit cells depicted in Figure 3. It is pulled along the Z axis by 0.2 m, while the nodes on the outer edges of the mid-plane are expected to satisfy

$$\frac{x_{f,i}}{x_{0,i}} \geq 1, \quad \frac{y_{f,i}}{y_{0,i}} \geq 1, \quad i = 1, \dots, 16, \quad (23)$$

where $x_{f,i}$ and $x_{0,i}$ are respectively final and initial x-coordinates, and $y_{f,i}$ and $y_{0,i}$ are respectively final and initial y-coordinates of Node i on the outer edges of the plane. Equation 23 means that the lattice structure must not shrink and possibly expand from the middle in both X and Y directions

Table 2: Categorical choices of the cross-sectional areas (in^2) for the truss structure found by various optimization methods in the literature

Member ID	PSO [16]	MBO [44]	CBO [46]	WCO [45]	DE [47]	NS [20]	GA	GSMO
1-4	0.196	1.800	0.196	0.196	0.196	0.196	0.196	0.141
5-12	0.563	0.602	0.563	0.563	0.563	0.563	0.563	0.563
13-16	0.442	0.111	0.391	0.391	0.391	0.391	0.391	0.391
17-18	0.563	0.111	0.563	0.563	0.563	0.563	0.563	0.563
19-22	0.563	1.266	0.563	0.563	0.563	0.563	0.563	0.563
23-30	0.563	0.563	0.563	0.563	0.563	0.563	0.563	0.563
31-34	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111
35-36	0.250	0.111	0.111	0.111	0.111	0.111	0.111	0.111
37-40	1.228	0.442	1.228	1.228	1.228	1.228	1.228	1.228
41-48	0.563	0.442	0.442	0.442	0.563	0.563	0.442	0.442
49-52	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111
53-54	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111
55-58	1.800	0.196	1.990	1.990	1.990	1.990	1.990	1.990
59-66	0.442	0.563	0.563	0.563	0.442	0.442	0.563	0.563
67-70	0.141	0.442	0.111	0.111	0.111	0.111	0.111	0.111
71-72	0.111	0.602	0.111	0.111	0.111	0.111	0.111	0.111
Optimum mass (lbm)	393.38	390.73	389.33	389.33	389.33	389.33	389.33	388.01

while being pulled along the Z direction. Note that even though the deformations are large, we still model the structure assuming linear elasticity as the focus of this paper is optimization rather than simulation. In this example, the sole purpose is to satisfy the constraints expressed in (23), hence the constant function $J = 0$ is considered for the objective function. The lattice is discretized by Euler-Bernoulli beam elements all of which are made of a steel alloy with a Young's modulus of 210 GPa and a Poisson's ratio of 0.3.

This problem contains a mix of continuous and categorical design variables. The former variables include the orientation of all the 812 beams and the spatial position of the nodes interior to the structure (i.e., those not lying on the lattice's top, bottom, front, back and side planes; 91 nodes in total). This leads to a total of 1085 continuous design variables. For the categorical variables, the 812 beams of the lattice are classified into 4 groups; those along the X, Y and Z axes as well as those along the unit cells' diagonal directions. For each group, 4 cross-sectional choices exist. The

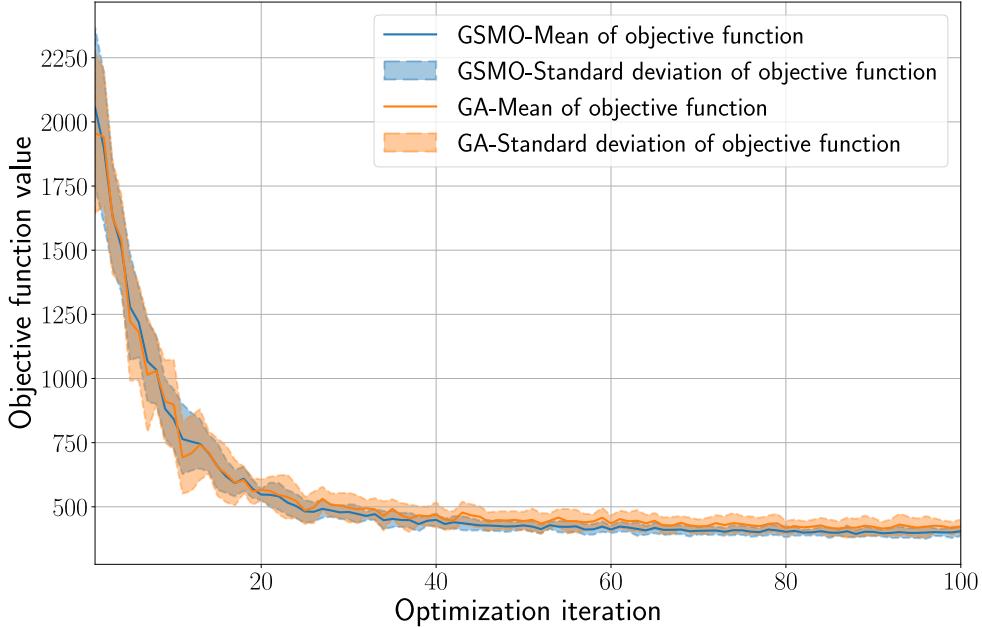


Figure 2: Convergence plot of the 10 GSmo and GA runs for the truss structure

details of these cross-sections are provided in Figure 4.

All three methods, GSmo, BiGSmo and GA, were able to produce a feasible solution albeit different from one another. Out of the 10 runs of GSmo and BiGSmo, all converged to the same solution. This once again demonstrates that although a sampling process is involved in these algorithms the optimum solutions generated in different runs are similar (in this example identical). Unlike GSmo and BiGSmo, GA had a hard time finding a feasible solution. For GA, only 3 of the runs led to a solution satisfying the constraints in (23) (All GA runs could find a feasible solution after about 300 iterations). Figure 5 illustrates the optimum solutions obtained by GSmo and BiGSmo in their undeformed and deformed configurations. The optimum value of cross-sectional choices is provided in Table 3.

Table 3: Optimum cross-sectional choices of the lattice structure

Member group	GSmo	BiGSmo	GA
Along X	CS4	CS2	CS1
Along Y	CS4	CS2	CS4
Along Z	CS2	CS1	CS4
Along diagonal	CS3	CS3	CS3

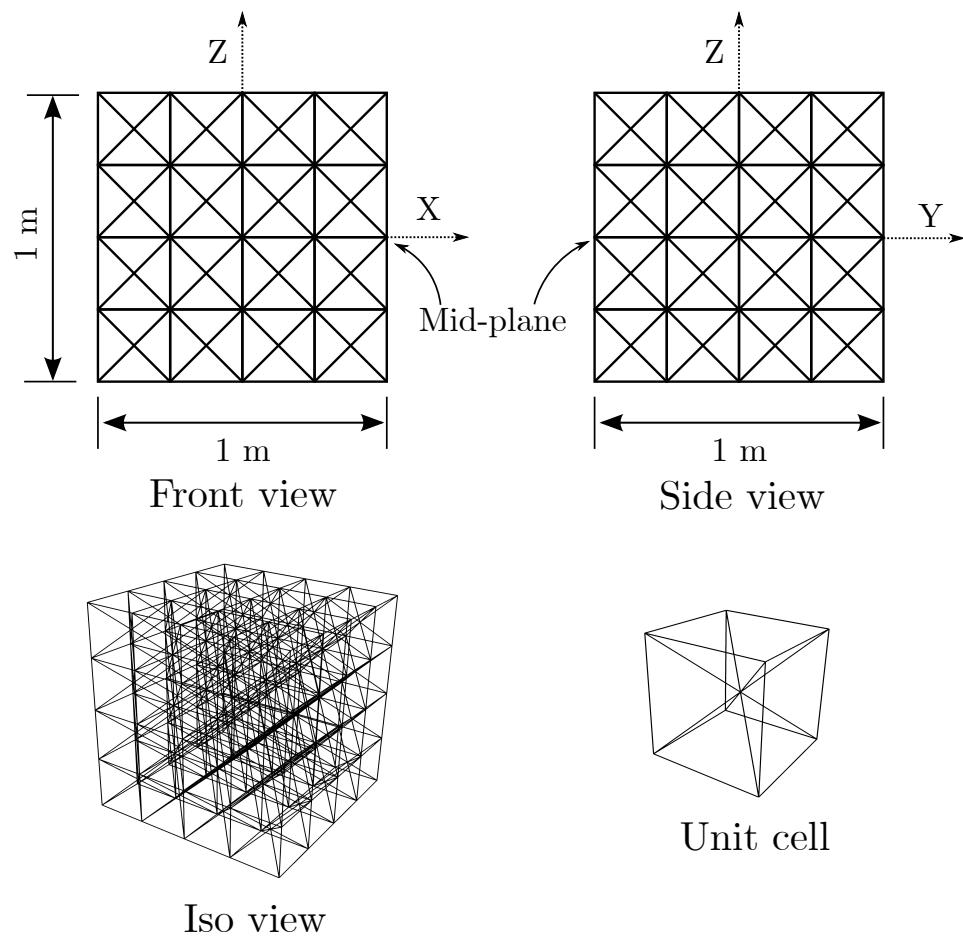


Figure 3: Geometry of the lattice structure

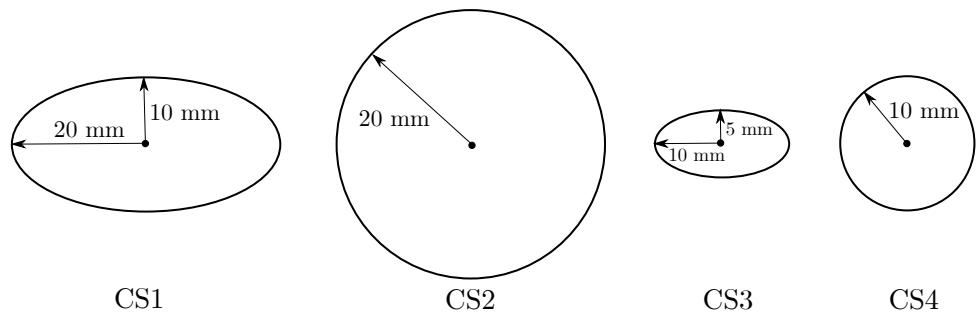


Figure 4: Available cross-sectional choices for the beams in the lattice structure

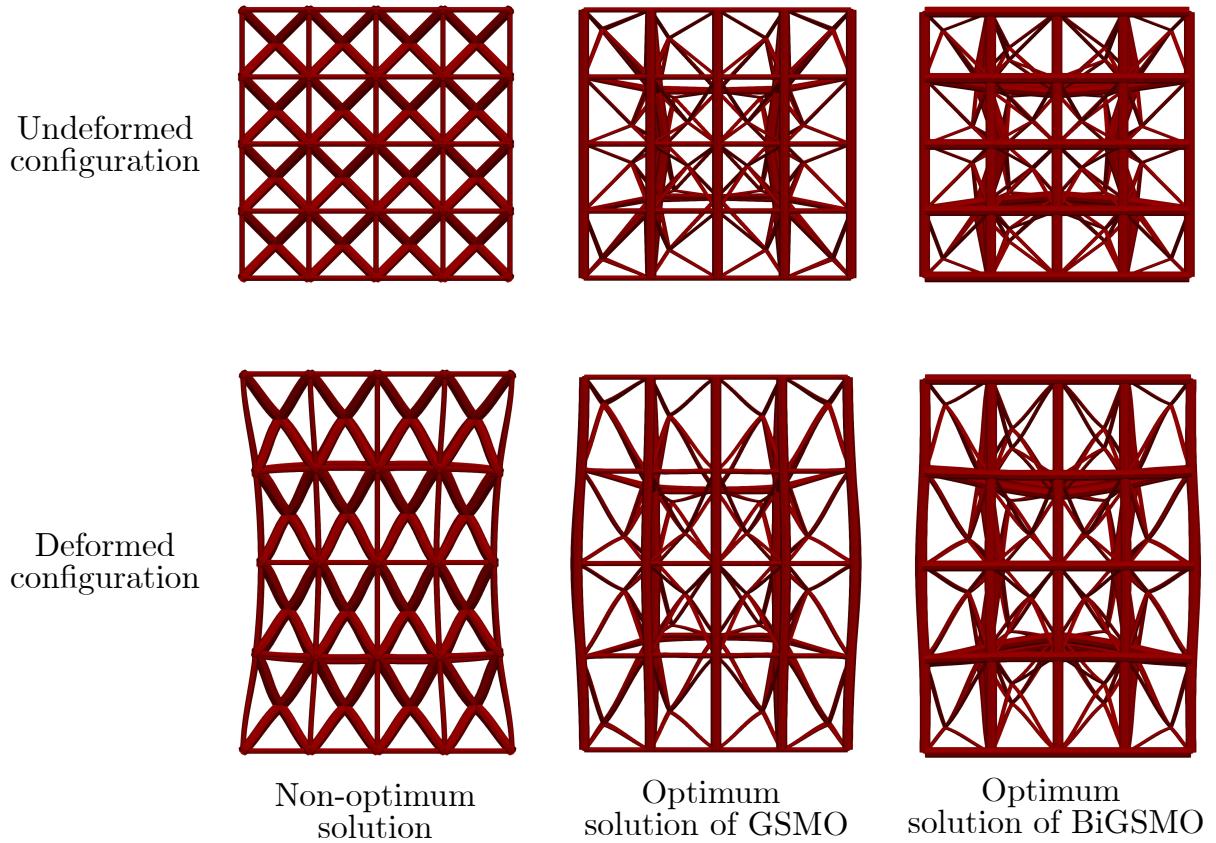


Figure 5: Optimum solutions of the lattice structure due to GSmo and BiGSmo

Similar to the previous case study, a major superiority of GSmo and BiGSmo relative to GA is their computational cost. Both GSmo and BiGSmo require only one FE solve per optimization iteration regardless of the number of design variables. For GA on the other hand, since there are 1089 design variables (1085 continuous and 4 categorical variables), with a population size of 10 times the number of design variables, 10890 FE solves are needed per iteration. Even if the FE routine is parallelized with GA, with the computational resource used for the experiment (16 threads), the number of FE solves required for every optimization iteration per CPU core would be $10890/16 \approx 680$. For this example, each FE solve takes about 0.8 seconds on the computer specified earlier, leading to 95 seconds for each GSmo and BiGSmo run (encompassing gradient calculations and other associated overheads). Each GA run with parallelization, on the other hand, extends to around 55000 seconds. Subsequently, the computation cost of GA for this problem is significantly larger than that of GSmo and BiGSmo making it prohibitive for larger-scale problems. The same adversity is expected for other population-based optimization approaches as well.

5.3 258-bar bridge structure

The bridge structure depicted in Figure 6 is composed of 258 components modeled by Euler-Bernoulli beam elements. The bridge spans 14 m along the X axis, has a width of 1 m and a maximum height of 3.45 m. All of its components are made of steel with a Young's modulus of 210 GPa, a yield stress of 360 MPa and a Poisson's ratio of 0.3. The members lying on the bridge floor are subject to a downward (along the negative Z axis) uniform load of 1000 N/m. Furthermore, the entire bridge is under the gravity load. The bridge is clamped from Nodes 1 to 4.

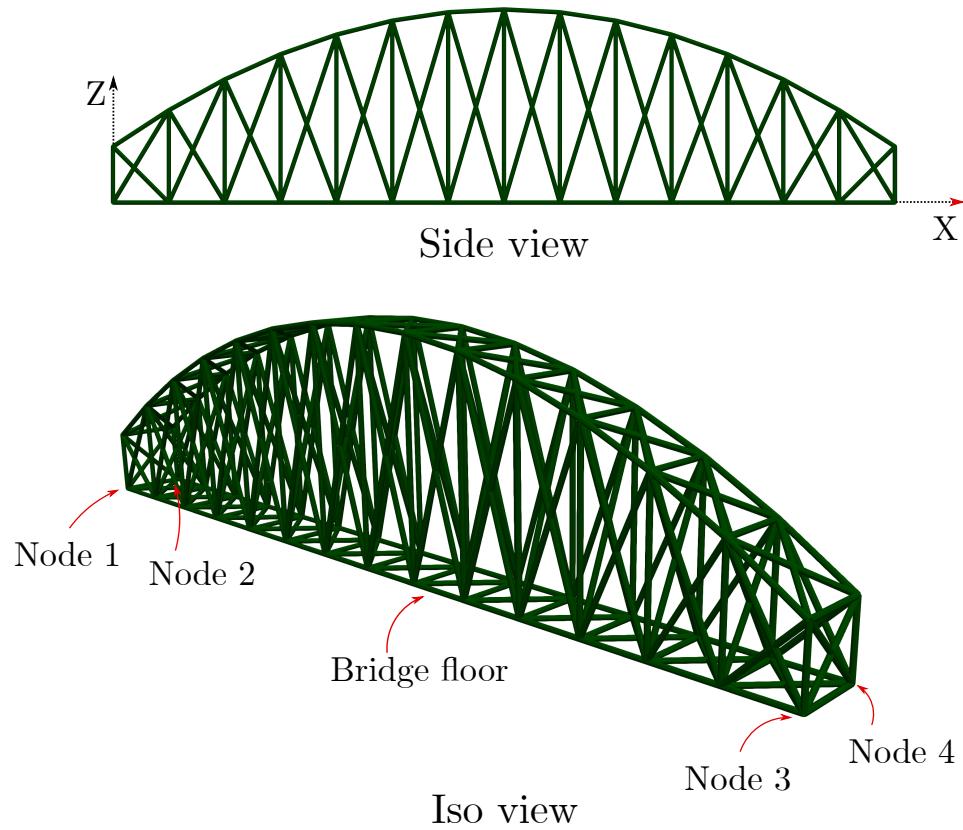


Figure 6: The 258-bar bridge structure

The objective function of this problem is to minimize the total strain energy of the structure due to the applied loads while the maximum stress in all elements remains below their yield strength. Also, the smallest natural frequency of the structure must be larger than 50 Hz. Similar to the previous case study, in this problem too we deal with a combination of continuous and categorical design variables. The former variables include the orientation of all 258 beam elements and the length of the beams not lying on the bridge floor (187 beams in total). Hence, there are 445 continuous design variables in aggregate. The categorical design variables contain the cross-sectional choices of

all the elements which are allowed to change independently (i.e., 258 categorical design variables). The available 5 cross-sectional profiles are illustrated in Figure 7 and their associated parameters are provided in Table 4.

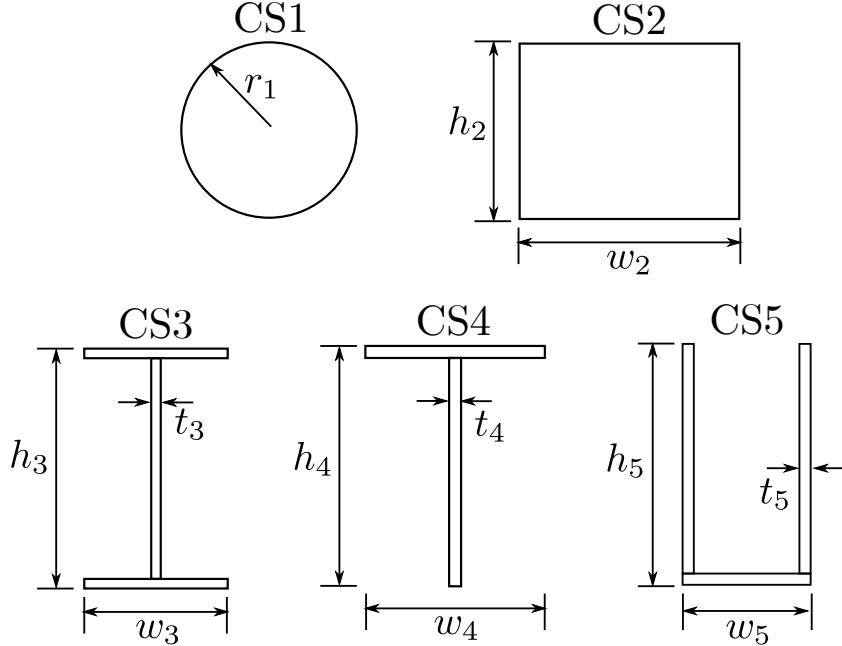


Figure 7: Available cross-sectional choices for the beams in the bridge structure

Figure 8 portrays the convergence plot of GSMO, BiGSMO and GA for the 10 runs. Accordingly, GSMO and BiGSMO have a slightly better convergence rate compared to that of GA for this particular problem. both GSMO and BiGSMO exhibit rapid convergence during the initial optimization stages, followed by a gradual deceleration as they approach a local optimum. This slowdown in convergence can be attributed to the diminishing gradient magnitude as the optimization nears a local optimum, leading to smaller solution updates at each iteration. Furthermore, as perhaps expected, two distinct stages can be observed in the BiGSMO plot; a somewhat flat region followed by a steep decline in every 10 iterations. This behavior is due to the fact that in BiGSMO the

Table 4: Parameter values associated with available cross-sectional choices for the beams in the bridge structure

Parameter name	r_1	h_2	w_2	h_3	w_3	t_3	h_4	w_4	t_4	h_5	w_5	t_5
Value (mm)	40	80	100	125	75	5	200	150	10	150	80	7

categorical and continuous design variables are handled in different levels of the optimization routine. As detailed in Algorithm 4, considering each 20-iteration interval, the first 10 iterations are dedicated to evolving the categorical variables and the second 10 iterations advance the continuous variables.

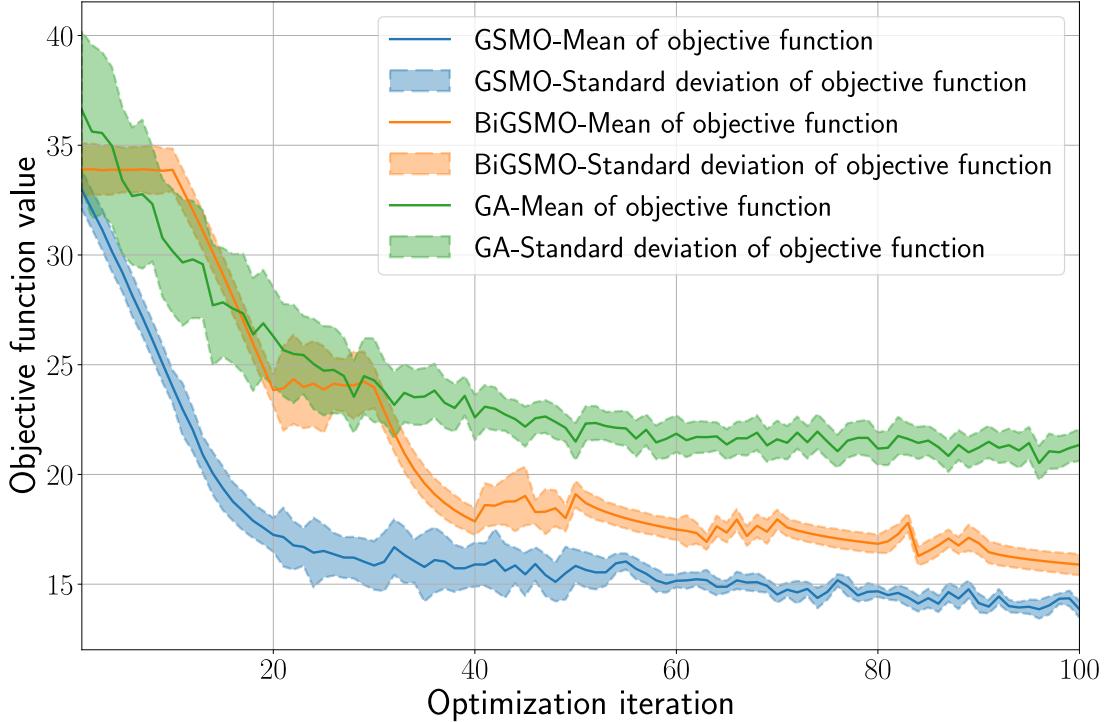


Figure 8: Convergence plot of GSmo, BiGSmo and GA averaged over the 10 runs for the bridge structure problem

The best objective value among the 10 runs as well as their average and standard deviations for the three optimization methods are presented in Table 5. Accordingly, GSmo outperforms BiGSmo and GA in all three aspects. The advantage of GSmo over GA for this example may be mainly attributed to the existence of a considerably large number of design variables, particularly the categorical variables. Since GSmo relies on sensitivity analysis, it is able to explore the high-dimensional design space more effectively compared to GA. Therefore, although GA is by construction capable of finding the global optimum, it falls short of finding one in this problem. On the other hand, the best solution found via BiGSmo is not as good as those of GSmo and GA. The reason could be due to the bilevel nature of BiGSmo. Since the categorical and continuous variables are handled in two different stages in this algorithm, at each stage, a portion of the design space is obscured and inadmissible for exploration by the optimizer. For this example, this has led to a less qualified solution compared to that generated by the other two methods.

Other noteworthy considerations in Table 5 are due to average and standard deviation values.

Unlike the previous case study, for this problem, the optimum solutions produced in the 10 runs using GSmo and BiGSmo are not unique, pointing to the fact that the sampling involved in these two approaches may lead to generating different solutions every time they are run. However, as can be seen, the average values are quite close to their corresponding best optimum values and the solutions are highly clustered around the best solutions. In this problem, that is not the case for GA. Looking at GA’s average and standard deviation values, there is a high chance of getting a solution relatively far from the best possible solution every time GA is executed. While this trait can sometimes help in obtaining hard-to-find global optima, such is not the case in this experiment as GA failed to find the best solution compared to GSmo. This is in fact a well-known phenomenon for other population-based optimizers as well.

Table 5: Performance of GSmo, BiGSmo and GA on the bridge structure

Method	Best Value (N.m)	Average (N.m)	Standard Deviation (N.m)	Approximate Execution Time (s)
GSmo	12.998	13.643	0.348	140
BiGSmo	15.121	15.853	0.350	140
GA	13.642	20.520	2.712	49000

Similar to other case studies, the advantage of GSmo and BiGSmo over GA can be appreciated more from the computational perspective. For this example, the computational cost per iteration of GA is about 7000 times larger than that of both GSmo and BiGSmo. The difference once again is due to requiring one linear elasticity and one modal analysis solve per optimization iteration for GSmo and BiGSmo as compared to 7030 of each solve for GA (total number of design variables is $445 + 258 = 703$). For this problem, one FE linear elasticity solve and one FE modal analysis solve combined take about 1.1 seconds on the computer specified earlier. This leads to approximately 140 seconds per run on average (including other overloads such as gradient calculations) for both GSmo and BiGSmo and about 49000 seconds per run (executing 16 solves in parallel per iteration) for GA. Figure 9 shows the GSmo’s best optimum solution and nonoptimum (initial) solution in their undeformed and deformed configurations.

6 Conclusion

The optimal design of real-world frame structures presents a significant challenge due to the presence of a large number of categorical design variables, such as cross-sectional profiles and material choices. The categorical nature of these variables precludes the employment of gradient-based optimizers and necessitates the use of gradient-free optimizers that are known to be less efficient. To address this challenge, this paper proposes a gradient-based optimizer that leverages the GSM method. The GSM method represents categorical choices as continuous probability distributions and enables the

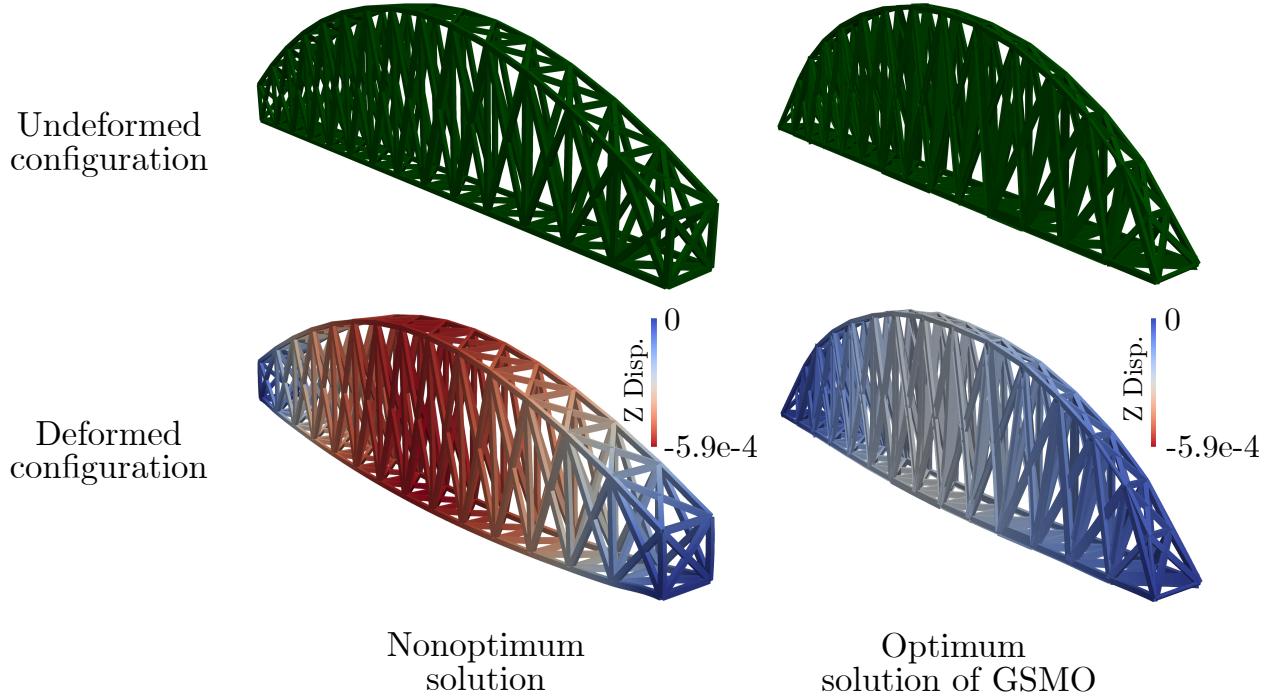


Figure 9: Optimum solution of the bridge structure due to GSmo. The deformations are scaled by a factor of 1000.

sampling process from such distributions to be differentiable. This in turn allows computing the sensitivities of objective and constraint functions with respect to the categorical design variables. This information can be combined with the sensitivities with respect to the continuous design variables to enable either simultaneous or bilevel optimization of categorical and continuous design variables, corresponding to the development of GSmo and BiGSmo, respectively.

Relative to GA, we have demonstrated that both GSmo and BiGSmo can find optimal solutions in a significantly shorter amount of time because the number of FE solves required is orders of magnitudes smaller. Specifically, our case studies showed that compared to GA, the computational costs of GSmo and BiGSmo were $\mathcal{O}(10^3)$ lower for the lattice and bridge structure problems. This remarkable improvement can be ascribed to two primary factors. Firstly, we have essentially transformed combinatorial optimization problems involving categorical design variables—which typically have exponential complexity—into problems involving only continuous design variables with a polynomial complexity [49]. Secondly, the incorporation of continuous design variables enables the application of gradient-based optimizers, which are well-known for their superior efficiency and scalability in comparison to gradient-free optimization methods [50].

In addition to the computational advantages, our case studies also showed that GSmo and BiGSmo found better solutions with more consistency than GA. In the 72-bar truss structure, GSmo generated an optimal solution that outperformed those of other methodologies reported in various

studies. For the 812-bar lattice structure, both GSmo and BiGSmo consistently found a unique optimum solution in all 10 runs, while GA only managed to find an optimum solution in 3 out of the 10 runs. Lastly, in the 258-bar bridge structure, both GSmo and BiGSmo resulted in a small standard deviation in optimal objective function values across the 10 runs, with GSmo delivering a noticeably better solution compared to BiGSmo and GA.

These findings were somewhat surprising because GA is known to be more adept at finding global optimum solutions than gradient-based approaches due to its ability to explore the solution space, albeit at the expense of efficiency. We hypothesize several reasons for this outcome. Assuming the general understanding that the gradient-based optimizers are susceptible to falling into local optima, it is likely that the consistent sets of solutions found by our method were indeed local minima. Conversely, because of the high complexity of frame structure design problems, GA was unable to get close to any optimal solution (neither local nor global) within the given number of iterations. Hence, the best solutions found by GA in our case studies tended to have high variance and subpar objective values. Had we run GA for a longer period of time (as was done in the second case study), it might have been possible that GA would have eventually converged to a better optimal solution. However, we conjecture that this would require an extraordinary amount of computational time for solving real-world design problems with even greater complexity than those studied in this paper.

Another important advantage of our method is that we are able to simultaneously optimize both categorical and continuous design variables. Typically, the problems involving both variable types are solved in a bilevel manner, wherein a gradient-free optimizer is used to solve for categorical variables at the outer level while a gradient-based optimizer is used to solve for continuous variables at the inner level. In this paper, we have investigated how this bilevel approach performs using a gradient-based optimizer for both levels. The last case study of bridge structure design demonstrated that the simultaneous approach employed by GSmo outperforms the bilevel approach employed by BiGSmo. This superior performance can likely be attributed to GSmo's enhanced efficacy in exploring the design space by considering both categorical and continuous design variables concurrently, while BiGSmo solves for each variable type sequentially.

Regarding the limitations, a distinctive aspect of our method is the requirement for each categorical variable to be represented by a set of continuous attributes (properties) to compute sensitivities. For instance, computing the gradients of an objective function with respect to cross-sectional profiles requires the gradients of the objective function with respect to the cross-sectional areas and moments of inertia, as expressed in (20). In certain applications, such characterization of categorical variables might not be readily available (e.g., the choice of joint types for a multi-component structural problem). Nonetheless, we assert that the majority of categorical choices involved in engineering design problems are associated with continuous parameters, enabling sensitivity computation and making our approach applicable to a wide range of applications.

Another limitation is that since GSmo and BiGSmo are gradient-based approaches, they are susceptible to getting trapped in local minima. However, the inherent stochastic nature of the GSmo method provides a means to control the exploration capability of our method, thereby increasing the likelihood of finding global optima. Specifically, the Gumbel temperature annealing schedule

can be varied such that the optimizer can use more iterations to sample a variety of solutions before converging. In addition, one can take the initial and final Gumbel temperatures as design variables and determine their optimum values for a given problem alongside other design variables. Furthermore, adopting a multi-start strategy where the problem is run using various initial solutions might mitigate the risk of being trapped in a local minimum. Such extensions should be explored in future work.

Furthermore, our broader application experience beyond the cases showcased in this study indicates that as the number of choices for individual categorical variables increases, both GSmo and BiGSmo exhibit oscillatory convergence behavior and, occasionally, may even fail to converge to a mathematically optimal solution. This phenomenon stems from the fact that when numerous choices are available for a variable, generating a sole sample from the distribution might not adequately capture the actual distribution associated with that variable. A plausible resolution could involve generating multiple samples for each categorical variable and subsequently selecting the sample generated most frequently. Lower Gumbel temperatures can also reduce the variance of the generated samples.

Lastly, in the presented second and third case studies, we compared our method to only one specific derivative-free method, GA. While GA is a well-known method that has been successfully applied to various engineering design applications, there are other derivative-free methods suitable for solving problems with categorical variables such as Tabu search, Monte Carlo tree search and estimation of distribution algorithms. However, all of these methods require a large number of function evaluations as they rely on either unguided (i.e., not guided by gradients) search moves or population-based improvements. Therefore, we are confident that the proposed method in this study will outperform such alternatives in terms of computational efficiency.

It is important to note that while we developed GSmo and BiGSmo in this study with structures undergoing linear elasticity behavior in mind, the algorithms presented in Algorithms 3 and 4 can be adapted to any optimization problem with mixed categorical and continuous design variables subject to other types of governing equations, with only minor modifications. The core principles of the presented algorithms remain unchanged; only the adjoint equations need to be adjusted accordingly. This versatility further underscores the potential of GSmo (and BiGSmo) for a wide range of applications, including structures with nonlinear behaviors.

Appendix A

Proposition 1. *The samples generated through (6) have the same probability distribution as $\boldsymbol{\theta}$.*

Proof. Let us first introduce the standard Gumbel distribution $\text{Gumbel}(0, 1)$. Its cumulative distribution function F and probability density function f read

$$F_X(x) = P(X \leq x) = \exp(-\exp(-x)), \quad f_X(x) = \frac{d}{dx} F_X(x) = \exp(-(x + \exp(-x))), \quad (24)$$

where the subscript X is the random variable associated with $\text{Gumbel}(0, 1)$. Defining Y and Θ as two other random variables satisfying $Y = \Theta + X$, we have

$$F_Y(\bar{y}) = P(Y \leq \bar{y}) = P(\Theta + X \leq \bar{y}) = P(X \leq \bar{y} - \Theta) = F_X(\bar{y} - \Theta) = \exp(-\exp(\Theta - \bar{y})), \quad (25)$$

leading to

$$f_Y(\bar{y}) = \frac{d}{d\bar{y}} F_Y(\bar{y}) = \frac{d}{d\bar{y}} \exp(-\exp(\Theta - \bar{y})) = \exp(\Theta - \bar{y}) \exp(-\exp(\Theta - \bar{y})). \quad (26)$$

Now, defining $z_i := \theta_i + G^{(i)}$ as in (6), to prove the proposition, it suffices to show that the probability of z_k , $k \in [1, N]$ being the maximum of set $\{z_1, \dots, z_N\}$ is $[\text{softmax}(\boldsymbol{\theta})]_k$ as given in (4). In other words,

$$P(z_k = \max\{z_1, \dots, z_N\}) = \frac{\exp(\theta_k)}{\sum_{i=1}^N \exp(\theta_i)}, \quad k \in [1, N]. \quad (27)$$

We proceed as follows: Another way of interpreting $z_k = \max\{z_1, \dots, z_N\}$ is to set $\bar{z} := z_k$ and write $z_k = \max\{z_1, \dots, z_N\}$ as

$$z_1 \leq \bar{z}, \dots, z_{k-1} \leq \bar{z}, z_k = \bar{z}, z_{k+1} \leq \bar{z}, \dots, z_N \leq \bar{z}. \quad (28)$$

Then, since the samples z_i are generated independently, using (25) and (26) we have

$$\begin{aligned} P(z_k = \max\{z_1, \dots, z_N\}) &= P(z_1 \leq \bar{z}, \dots, z_{k-1} \leq \bar{z}, z_k = \bar{z}, z_{k+1} \leq \bar{z}, \dots, z_N \leq \bar{z}) \\ &= \int_{-\infty}^{\infty} \left[\exp(\theta_k - \bar{z}) \exp(-\exp(\theta_k - \bar{z})) \prod_{i=1, i \neq k}^N \exp(-\exp(\theta_i - \bar{z})) \right] d\bar{z} \\ &= \int_{-\infty}^{\infty} \left[\exp(\theta_k - \bar{z}) \prod_{i=1}^N \exp(-\exp(\theta_i - \bar{z})) \right] d\bar{z} \\ &= \int_{-\infty}^{\infty} \left[\exp(\theta_k - \bar{z}) \exp\left(-\exp(-\bar{z}) \sum_{i=1}^N \exp(\theta_i)\right) \right] d\bar{z} \\ &= \frac{\exp(\theta_k) \exp\left(-\exp(-\bar{z}) \sum_{i=1}^N \exp(\theta_i)\right)}{\sum_{i=1}^N \exp(\theta_i)} \Big|_{-\infty}^{\infty} = \frac{\exp(\theta_k)}{\sum_{i=1}^N \exp(\theta_i)}. \end{aligned} \quad (29)$$

Therefore, (27) holds and the proposition is proved. \square

Appendix B

Algorithm 4: The BiGSMO scheme

Input: Objective function J , constraint functions \mathbf{g} , continuous design variables \mathbf{x} and their bounds, categorical design variables \mathbf{c} and their available choices, GSM annealing scheme.

Output: Optimum design.

```

1 Initialize  $\mathbf{x}$ ;
2 Initialize  $\boldsymbol{\theta}_i$ ,  $i = 1, \dots, n_c$ , for each categorical variable  $c_i$  by entry values of zero;
3 while outer optimizer not converged do
4   while inner optimizer not converged do
5     for  $i = 1, \dots, n_c$  do
6       Generate Gumbel noises  $G^{(j)}$ ,  $j = 1, \dots, N_i$ , from  $\text{Gumbel}(0, 1)$ ;
7       Compute  $(\tilde{\mathbf{s}}_i)_j$ ,  $j = 1, \dots, N_i$ , using (7) and  $\nabla_{\boldsymbol{\theta}_i} \tilde{\mathbf{s}}_i$  via (10);
8       Calculate  $\hat{\mathbf{s}}_i$  through (8);
9     end
10    Solve the governing equations  $\mathbf{K}\mathbf{u} = \mathbf{f}$  using  $\hat{\mathbf{s}}_i$ ,  $i = 1, \dots, n_c$ , and get  $\mathbf{u}$ ;
11    Compute  $J$  and  $\mathbf{g}$  values using  $\hat{\mathbf{s}}_i$ ,  $i = 1, \dots, n_c$ , and  $\mathbf{u}$ ;
12    Calculate  $\nabla_{\mathbf{u}} J$  then find  $\boldsymbol{\lambda}_J$  by solving (14);
13    Compute  $\nabla_{\mathbf{u}} g_i$ ,  $i = 1, \dots, n_g$ , then find  $\boldsymbol{\lambda}_{g_i}$ ,  $i = 1, \dots, n_g$ , by solving (16);
14    for  $i = 1, \dots, n_c$  do
15      Form the attribute matrix  $\mathbf{A}_i$  as in (18);
16      Calculate  $\partial J / \partial \mathbf{a}_i$ ,  $\partial \mathbf{g} / \partial \mathbf{a}_i$ ,  $\partial \mathbf{K} / \partial \mathbf{a}_i$  and  $\partial \mathbf{f} / \partial \mathbf{a}_i$  using the attributes of the selected class for this categorical variable;
17      Compute  $\nabla_{\mathbf{a}_i} J$  and  $\nabla_{\mathbf{a}_i} \mathbf{g}$  utilizing the adjoint vectors found in Steps 12 and 13,
          $\nabla_{\boldsymbol{\theta}_i} \tilde{\mathbf{s}}_i$  found in Step 7 and employing (21);
18      Get  $\nabla_{\boldsymbol{\theta}_i} J$  and  $\nabla_{\boldsymbol{\theta}_i} \mathbf{g}$  through (20);
19    end
20    Update  $\boldsymbol{\theta}_i$ ,  $i = 1, \dots, n_c$ , using the sensitivities found in Step 18;
21  end
22  Update  $\tilde{\mathbf{s}}_i$  and  $\hat{\mathbf{s}}_i$ ,  $i = 1, \dots, n_c$ , using the new  $\boldsymbol{\theta}_i$  values;
23  Solve the governing equations  $\mathbf{K}\mathbf{u} = \mathbf{f}$  using  $\hat{\mathbf{s}}_i$ ,  $i = 1, \dots, n_c$ , in Step 22 and get  $\mathbf{u}$ ;
24  Compute the new values of  $J$ ,  $\nabla_{\mathbf{u}} J$ ,  $\boldsymbol{\lambda}_J$ ; and  $g_i$ ,  $\nabla_{\mathbf{u}} g_i$  and  $\boldsymbol{\lambda}_{g_i}$ ,  $i = 1, \dots, n_g$ ;
25  Calculate  $\partial J / \partial x_i$ ,  $\partial \mathbf{g} / \partial x_i$ ,  $\partial \mathbf{K} / \partial x_i$  and  $\partial \mathbf{f} / \partial x_i$ ,  $i = 1, \dots, n_x$ ;
26  Get  $\nabla_{\mathbf{x}} J$  and  $\nabla_{\mathbf{x}} \mathbf{g}$  incorporating the corresponding adjoint vectors and (15);
27  Update  $\mathbf{x}$  using the sensitivities found in Steps 26;
28 end
```

References

- [1] W. McGuire, R. H. Gallagher, and H. Saunders, *Matrix structural analysis*. John Wiley & Sons, 1982.
- [2] M.-W. Huang and J. S. Arora, “Optimal design of steel structures using standard sections,” *Structural Optimization*, vol. 14, no. 1, pp. 24–35, 1997.
- [3] A. Kaveh and S. Talatahari, “Charged system search for optimal design of frame structures,” *Applied Soft Computing*, vol. 12, no. 1, pp. 382–393, 2012.
- [4] M. Ebrahimi, A. Butscher, and H. Cheong, “A low order, torsion deformable spatial beam element based on the absolute nodal coordinate formulation and bishop frame,” *Multibody System Dynamics*, vol. 51, no. 3, pp. 247–278, 2021.
- [5] C. D. Chapman, K. Saitou, and M. J. Jakiel, “Genetic algorithms as an approach to configuration and topology design,” *Journal of Mechanical Design*, vol. 116, pp. 1005–1012, 1994.
- [6] P. P. Angelov, Y. Zhang, J. A. Wright, V. I. Hanby, and R. A. Buswell, “Automatic design synthesis and optimization of component-based systems by evolutionary algorithms,” in *Genetic and Evolutionary Computation Conference*, pp. 1938–1950, Springer, 2003.
- [7] E. Lund and J. Stegmann, “On structural optimization of composite shell structures using a discrete constitutive parametrization,” *Wind Energy*, vol. 8, no. 1, pp. 109–124, 2005.
- [8] H. Cheong, M. Ebrahimi, A. Butscher, and F. Iorio, “Configuration design of mechanical assemblies using an estimation of distribution algorithm and constraint programming,” in *IEEE Congress on Evolutionary Computation*, pp. 2339–2346, 2019.
- [9] C. Piacentini, H. Cheong, M. Ebrahimi, and A. Butscher, “Multi-speed gearbox synthesis using global search and non-convex optimization,” in *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pp. 381–398, Springer, 2020.
- [10] M. Stolpe, “Truss optimization with discrete design variables: a critical review,” *Structural and Multidisciplinary Optimization*, vol. 53, no. 2, pp. 349–374, 2016.
- [11] M. Stolpe and A. Kawamoto, “Design of planar articulated mechanisms using branch and bound,” *Mathematical Programming*, vol. 103, no. 2, pp. 357–397, 2005.
- [12] W. Jenkins, “Plane frame optimum design environment based on genetic algorithm,” *Journal of Structural Engineering*, vol. 118, no. 11, pp. 3103–3112, 1992.
- [13] A. Kaveh and V. Kalatjari, “Size/geometry optimization of trusses by the force method and genetic algorithm,” *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik: Applied Mathematics and Mechanics*, vol. 84, no. 5, pp. 347–357, 2004.

- [14] H. S. Park and C. W. Sung, “Optimization of steel structures using distributed simulated annealing algorithm on a cluster of personal computers,” *Computers & Structures*, vol. 80, no. 14-15, pp. 1305–1316, 2002.
- [15] M. Kripka, “Discrete optimization of trusses by simulated annealing,” *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 26, no. 2, pp. 170–173, 2004.
- [16] A. Kaveh and S. Talatahari, “A particle swarm ant colony optimization for truss structures with discrete variables,” *Journal of Constructional Steel Research*, vol. 65, no. 8-9, pp. 1558–1568, 2009.
- [17] L. Li, Z. Huang, and F. Liu, “A heuristic particle swarm optimization method for truss structures with discrete variables,” *Computers & Structures*, vol. 87, no. 7-8, pp. 435–443, 2009.
- [18] D. Yates, A. Templeman, and T. Boffey, “The complexity of procedures for determining minimum weight trusses with discrete member sizes,” *International Journal of Solids and Structures*, vol. 18, no. 6, pp. 487–495, 1982.
- [19] I. Fister Jr, X.-S. Yang, I. Fister, J. Brest, and D. Fister, “A brief review of nature-inspired algorithms for optimization,” *arXiv preprint arXiv:1307.4186*, 2013.
- [20] M. Shahabsafa, A. Mohammad-Nezhad, T. Terlaky, L. Zuluaga, S. He, J. T. Hwang, and J. R. Martins, “A novel approach to discrete truss design problems using mixed integer neighborhood search,” *Structural and Multidisciplinary Optimization*, vol. 58, no. 6, pp. 2411–2429, 2018.
- [21] M. Ebrahimi, A. Butscher, H. Cheong, and F. Iorio, “Design optimization of dynamic flexible multibody systems using the discrete adjoint variable method,” *Computers & Structures*, vol. 213, pp. 82–99, 2019.
- [22] J. Yan, Z. Duan, E. Lund, and G. Zhao, “Concurrent multi-scale design optimization of composite frame structures using the Heaviside penalization of discrete material model,” *Acta Mechanica Sinica*, vol. 32, pp. 430–441, 2016.
- [23] C. Krogh, M. H. Jungersen, E. Lund, and E. Lindgaard, “Gradient-based selection of cross sections: a novel approach for optimal frame structure design,” *Structural and Multidisciplinary Optimization*, vol. 56, no. 5, pp. 959–972, 2017.
- [24] Z. Duan, J. Yan, I. Lee, E. Lund, and J. Wang, “Discrete material selection and structural topology optimization of composite frames for maximum fundamental frequency with manufacturing constraints,” *Structural and Multidisciplinary Optimization*, vol. 60, pp. 1741–1758, 2019.
- [25] P.-J. Barjhoux, Y. Diouane, S. Grihon, D. Bettebghor, and J. Morlier, “A bi-level methodology for solving large-scale mixed categorical structural optimization,” *Structural and Multidisciplinary Optimization*, vol. 62, no. 1, pp. 337–351, 2020.

- [26] M. P. Bendsoe and O. Sigmund, *Topology optimization: theory, methods, and applications*. Springer Science & Business Media, 2003.
- [27] E. Jang, S. Gu, and B. Poole, “Categorical reparameterization with Gumbel-Softmax,” in *The International Conference on Learning Representations*, 2017.
- [28] C. Maddison, A. Mnih, and Y. Teh, “The concrete distribution: A continuous relaxation of discrete random variables,” in *The International Conference on Learning Representations*, 2017.
- [29] E. J. Gumbel, *Statistical theory of extreme values and some practical applications: a series of lectures*, vol. 33. US Government Printing Office, 1954.
- [30] I. A. Huijben, W. Kool, M. B. Paulus, and R. J. Van Sloun, “A review of the Gumbel-max trick and its extensions for discrete stochasticity in machine learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.
- [31] N. Cesa-Bianchi, C. Gentile, G. Lugosi, and G. Neu, “Boltzmann exploration done right,” *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [32] J. Gu, D. J. Im, and V. O. Li, “Neural machine translation with Gumbel-greedy decoding,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.
- [33] J. Yang, Q. Zhang, B. Ni, L. Li, J. Liu, M. Zhou, and Q. Tian, “Modeling point clouds with self-attention and Gumbel subset sampling,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3323–3332, 2019.
- [34] W. Kool, H. Van Hoof, and M. Welling, “Stochastic beams and where to find them: The Gumbel-top-k trick for sampling sequences without replacement,” in *International Conference on Machine Learning*, pp. 3499–3508, PMLR, 2019.
- [35] V. K. Rohatgi and A. M. E. Saleh, *An introduction to probability and statistics*. John Wiley & Sons, 2015.
- [36] C. J. Maddison, D. Tarlow, and T. Minka, “A* sampling,” *Advances in Neural Information Processing Systems*, vol. 27, 2014.
- [37] A. Baevski, S. Schneider, and M. Auli, “VQ-WAV2VEC: Self-supervised learning of discrete speech representations,” in *International Conference on Learning Representations*, 2019.
- [38] P. Guo, C.-Y. Lee, and D. Ulbricht, “Learning to branch for multi-task learning,” in *International Conference on Machine Learning*, pp. 3854–3863, 2020.
- [39] M. Kang and B. Han, “Operation-aware soft channel pruning using differentiable masks,” in *International Conference on Machine Learning*, pp. 5122–5131, 2020.

- [40] Y. Cao, S. Li, L. Petzold, and R. Serban, “Adjoint sensitivity analysis for differential-algebraic equations: The adjoint dae system and its numerical solution,” *SIAM Journal on Scientific Computing*, vol. 24, no. 3, pp. 1076–1089, 2003.
- [41] G. Allaire and G. Delgado, “Stacking sequence and shape optimization of laminated composite plates via a level-set method,” *Journal of the Mechanics and Physics of Solids*, vol. 97, pp. 168–196, 2016.
- [42] M. J. Islam, X. Li, and K. Deb, “Multimodal truss structure design using bilevel and niching based evolutionary algorithms,” in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 274–281, 2017.
- [43] H. Cheong, M. Ebrahimi, and T. Duggan, “Optimal design of continuum robots with reachability constraints,” *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3902–3909, 2021.
- [44] A. Sadollah, A. Bahreininejad, H. Eskandar, and M. Hamdi, “Mine blast algorithm for optimization of truss structures with discrete variables,” *Computers & Structures*, vol. 102, pp. 49–63, 2012.
- [45] A. Sadollah, H. Eskandar, A. Bahreininejad, and J. H. Kim, “Water cycle, mine blast and improved mine blast algorithms for discrete sizing optimization of truss structures,” *Computers & Structures*, vol. 149, pp. 1–16, 2015.
- [46] A. Kaveh and M. I. Ghazaan, “A comparative study of CBO and ECBO for optimal design of skeletal structures,” *Computers & Structures*, vol. 153, pp. 137–147, 2015.
- [47] V. Ho-Huu, T. Nguyen-Thoi, T. Vo-Duy, and T. Nguyen-Trang, “An adaptive elitist differential evolution for optimization of truss structures with discrete design variables,” *Computers & Structures*, vol. 165, pp. 59–75, 2016.
- [48] R. T. Haftka and Z. Gürdal, *Elements of structural optimization*, vol. 11. Springer Science & Business Media, 2012.
- [49] J. Lenstra and A. Rinnooy Kan, “Computational complexity of discrete optimization problems,” in *Discrete Optimization I* (P. Hammer, E. Johnson, and B. Korte, eds.), vol. 4 of *Annals of Discrete Mathematics*, pp. 121–140, Elsevier, 1979.
- [50] J. R. R. A. Martins and A. Ning, *Engineering design optimization*. Cambridge University Press, Jan 2022.