Journal of Applied Mechanics. Received November 13, 2019;
Accepted manuscript posted December 27, 2019. doi:10.1115/1.4045902
Copyright © 2020 by ASME

Topology Optimization with Many Right Hand Sides Using A Mirror Descent Stochastic Approximation - Reduction from Many to A Single Sample

Xiaojia Shelly Zhang[*]
Assistant Professor
Department of Civil and Environmental Engineering
Department of Mechanical Science and Engineering
University of Illinois at Urbana Champaign
Urbana, Illinois, 61801
Email: zhangxs@illinois.edu

Eric de Sturler
Professor
Department of Mathematics
Virginia Tech
Blacksburg, Virginia, 24061
Email: sturler@vt.edu

Alexander Shapiro
Professor
School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, Georgia, 30332
Email: ashapiro@isye.gatech.edu

Abstract *Practical engineering designs typically involve many load cases. For topology optimization with many deterministic load cases, a large number of linear systems of equations must be solved at each optimization step, leading to an enormous computational cost. To address this challenge, we propose a mirror descent stochastic approximation (MD-SA) framework with step size strategies to solve topology optimization problems with many load cases. This method requires only one linear system per optimization step. We reformulate the deterministic objective function and gradient into stochastic ones through randomization. The proposed MD-SA algorithm only requires low accuracy in the stochastic gradient and thus uses only a single sample per optimization step (i.e., the sample size is always one). We reduce the number of linear systems to solve per step from hundreds to one, thus drastically reducing the total computational cost. For example, for one of the design problems, the total number of linear systems to solve and wall clock time are reduced by factors of 223 and 22, respectively.*

[*]Correspondence author.

1 Introduction

Mechanics-driven topology optimization is a powerful tool that can generate materials and structures with unconventional and significantly improved performance in many engineering applications [1–12]. Real-world engineering designs, such as automobiles, high-rise buildings, and airplanes, involve numerous load scenarios. For instance, the challenge problem proposed in the 2019 topology optimization roundtable is to design a suspension upright of a race car subjected to 13 load cases, as shown in Fig. 1. A detailed solution of this challenge problem provided by SIMULIA from Dassault Systèmes can be found in [13]. Thus, it is important to account for many load cases in structural and mechanical topology optimization.

In the literature, several formulations incorporating many load cases have been explored [14–16]. A popular one is the weighted-sum formulation, which aims to minimize a weighted sum of the objective functions from all the load cases. Another common approach is the min-max formulation, which minimizes the maximum objective function among all load cases [16,17]. For all of these formulations, a

Journal of Applied Mechanics. Received November 13, 2019;
Accepted manuscript posted December 27, 2019. doi:10.1115/1.4045902
Copyright © 2020 by ASME

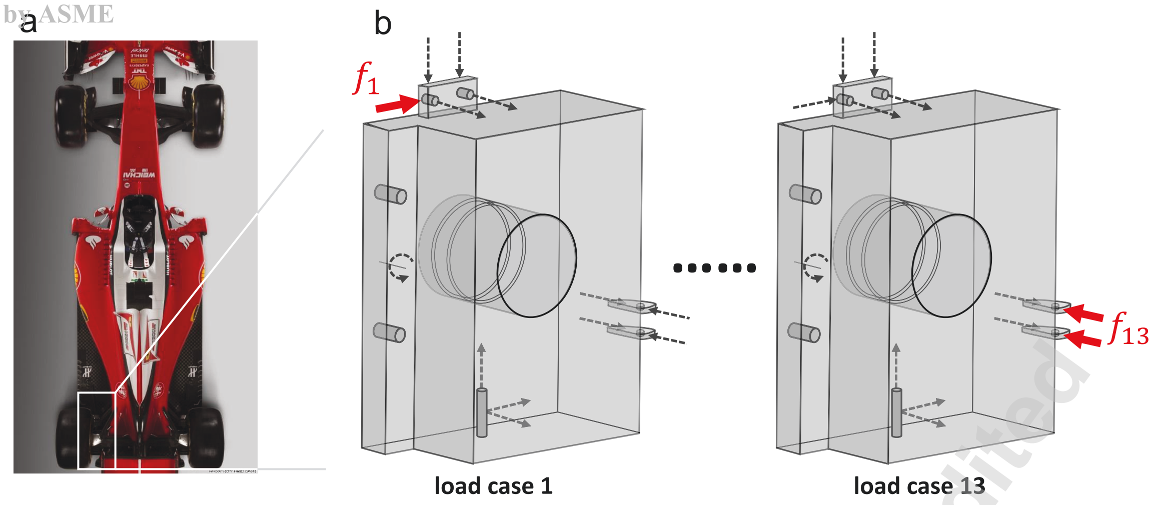

Fig. 1. 2019 topology optimization roundtable challenge problem: design of a suspension upright. (a) Formula 1 race car and the location of the design region. (b) Design domain and 13 load cases applied at the domain from cornering (6 cases) and breaking (7 cases) [13]. Dotted arrows are the schematic illustrations of non-active load cases. A detailed solution of this challenge problem provided by SIMULIA from Dassault Systèmes can be found in [13].

major challenge is the computational cost because the solution of the state equation for each load case is required. For large-scale problems with many load cases (e.g., hundreds or more), this leads to overwhelming computational costs.

To address this challenge, Zhang et al. [18] employed a randomized algorithm that drastically reduces computational cost for topology optimization with many load cases. The paper introduces randomization techniques to rewrite the deterministic optimization formulation into an equivalent stochastic one. A Sample Average Approximation (SAA)-based approach is then employed in [18] to estimate the corresponding gradients. Widely used in various contexts [19–21], the basic idea of the SAA approach is to generate a random sample $\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_n$ of size n, of the involved random vector $\boldsymbol{\xi}$, and consequently to approximate the original problem by the sample average problem. Different from the conventional SAA, which fixes the random sample throughout the optimization process, the SAA-based algorithm in [18] re-selects random samples at every optimization iteration.

The obtained stochastic gradient has to be combined with an optimization algorithm. For example, the Optimality Criteria (OC) method is used to update the vector of design variables in [18]. As a result, the SAA approach requires an adequate level of accuracy for the estimated problem (the objective function and its gradient) and, thus, a few samples (e.g. 5 or 6) are typically used. For the randomized formulation of topology optimization considering many load cases, $n = 6$ is typically sufficient to estimate the gradients when the SAA-based approach is used. This leads to a greatly reduced computational cost compared with the standard deterministic approach.

Unlike the SAA method, the Stochastic Approximation (SA) method is an algorithm designed to solve stochastic optimization problems, whose objective functions are given in terms of expectations. The SA approach does not require the estimated gradient to be accurate and it only requires one sample to evaluate the estimated gradient at each optimization iteration [22]. Because of this attractive property, the SA method and its variants have been widely adopted in many modern machine learning methods [23, 24]. The performance of the classical SA approach, however, is highly sensitive to the choice of step size. To address this step size dependence, a robust SA approach is proposed in [25], in which "robustness" with respect to the step size is achieved by averaging the iterates (vector of design variables) over a certain window size. The Mirror Descent (MD) approach originates from the work of Nemirovski and Yudin [26], and the Mirror Descent SA (MD-SA) approach can be found in [22] and is a generalization of the robust SA approach (which is intrinsically linked to Euclidean space). Through a general definition of a distance generating function [22], the MD-SA allows for the adjustment of the algorithm to the underlying geometry of the problem.

In this paper, we propose a tailored MD-SA algorithm to efficiently solve randomized topology optimization problems under many load cases based on compliance minimization. Tailored to the geometry of the underlying feasible design space, we introduce an entropic ℓ_1 norm MD-SA algorithm with the distance generating function being the entropy function. This algorithm requires only low accuracy in the stochastic gradient. Thus, we are able to use a single sample ($n = 1$) regardless of the number of load cases considered. To obtain effective step sizes and updates, we propose several algorithmic strategies, including step size policy and recalibration, iterates averaging, and a damping scheme to improve the performance and convergence of the proposed algorithm. We also demonstrate both theoretically and numerically that, compared with the robust SA approach based on the Euclidean norm (the commonly used method), the proposed entropic ℓ_1 norm MD-SA algorithm exhibits robustness in step size selection and better performance for various problem sizes, particularly large-scale problems. We further

Journal of Applied Mechanics. Received November 13, 2019;
Accepted manuscript posted December 27, 2019. doi:10.1115/1.4045902
Copyright © 2020 by ASME

compare the performance of an SAA variant [18] (with OC update) with the proposed SA algorithm described in this paper. Compared with the SAA-based algorithm, the proposed algorithm is able to perform high-quality updates with less accurate estimates of gradient and, as a result, leads to fewer linear systems (i.e., 1 solve per step vs. 6 solves per step) to solve. We also demonstrate that, when adopted in conjunction with an iterative solver, the proposed MD-SA algorithm allows for a significantly higher convergence tolerance in solving the state equation without influencing the performance of the optimization, which typically cannot be achieved with any deterministic algorithms.

The remainder of this paper is organized as follows. Section 2 briefly reviews the deterministic and equivalent randomized formulations for topology optimization with many load cases. Section 3 reviews the general MD-SA framework and presents an entropic version of MD-SA using the ℓ_1 norm. Section 4 proposes an entropic ℓ_1 norm MD-SA algorithm tailored for a randomized topology optimization formulation. In Section 5, we introduce the step size strategies, a damping scheme, the iterative solver, and discuss the algorithmic parameters. Section 6 demonstrates numerical examples in two and three dimensions, highlighting the efficiency and effectiveness of the proposed algorithm, and Section 7 provides concluding remarks.

2 Deterministic and randomized topology optimization with many load cases

In this section, we briefly review the deterministic formulation of topology optimization under many load cases and its equivalent randomized form. Throughout this work, we use compliance minimization with a weighted-sum formulation, and we focus on the popular density-based method.

2.1 Deterministic density-based topology optimization with many load cases

For a given finite element mesh containing M elements and N nodes in d dimensions, and assuming a total of m design load cases $\boldsymbol{f}_i \in \mathbb{R}^{dN}, i = 1,\ldots,m$, we denote by α_i ($\alpha_i > 0, \sum_{i=1}^m \alpha_i = 1$) and $\boldsymbol{u}_i \in \mathbb{R}^{dN}$ the weight and displacement vector associated with i-th load case $\boldsymbol{f}_i$, respectively. The standard weighted-sum topology optimization formulation for minimum end-compliance problems using the density-based method takes the form [16, 18],

$$\begin{aligned}
&\min_{\boldsymbol{x}} \left\{ C(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i \boldsymbol{f}_i^T \boldsymbol{u}_i(\boldsymbol{x}) \right\} \\
&\quad \text{s.t.} \sum_{e=1}^{M} \frac{v^{(e)} \bar{x}^{(e)}}{|\Omega|} - V_f = 0\,, \\
&\qquad 0 \le x^{(e)} \le 1, e = 1,\ldots,M, \\
&\text{with } \boldsymbol{u}_i(\boldsymbol{x}) = \boldsymbol{K}(\boldsymbol{E}(\bar{\boldsymbol{x}}))^{-1} \boldsymbol{f}_i, i = 1,\ldots,m, \\
&\qquad E^{(e)} = E_{\min} + [\bar{x}^{(e)}]^p (E_0 - E_{\min}), e = 1,\ldots,M.
\end{aligned} \tag{1}$$

In the above optimization problem, the objective function $C(\cdot)$ is the weighted compliance, $\boldsymbol{x} \in \mathbb{R}^M$ is the vector of design variables, $x^{(e)}$ is the design variable of element e, $\boldsymbol{K} \in \mathbb{R}^{dN \times dN}$ is the global stiffness matrix, and dN is the number of degrees of freedom (DOF). In order to avoid checkerboard instability, we apply a density filter on the vector of design variables to obtain the vector of filtered densities $\bar{\boldsymbol{x}}$ as $\bar{\boldsymbol{x}} = \boldsymbol{H}\boldsymbol{x}$, where $\boldsymbol{H}$ is the filter matrix [27]. Furthermore, the design problem is subject to a global volume constraint, with the volume (area) of an element e denoted by $v^{(e)}$, the prescribed volume fraction denoted by V_f, Ω and $|\Omega|$ being the problem domain and area/volume of the domain, respectively. To simplify the form of the volume constraint, we introduce $\bar{v}^{(e)} = \sum_j^M H_{e,j} v^{(j)}$ and rewrite the volume constraint as

$$\sum_{e=1}^{M} \frac{\bar{v}^{(e)} x^{(e)}}{|\Omega|} - V_f = 0. \tag{2}$$

Notice that $\bar{v}^{(e)}$ is a constant that does not change throughout the entire optimization process. We use the Solid Isotropic Material with Penalization (SIMP) [28, 29] approach, $E_{\min}$ and E_0 are the elastic moduli for Ersatz and solid materials, respectively, and p is a penalization parameter.

The gradient (sensitivity) $\nabla C_x^{(e)} = \frac{\partial C}{\partial x^{(e)}}$ of the objective function is the weighted sum of the sensitivities from each individual loading case:

$$\nabla C_x^{(e)} = -\sum_{i=1}^{m} \alpha_i \boldsymbol{u}_i^T \frac{\partial \boldsymbol{K}}{\partial x^{(e)}} \boldsymbol{u}_i. \tag{3}$$

We remark that the optimization problem in formulation (1) is convex when $p = 1$ [30]. With $p > 1$ the formulation becomes non-convex, and there is no guarantee that the optimization algorithm converges to the global minimum.

2.2 Randomized topology optimization

In this subsection, we introduce a randomized topology optimization formulation under many load cases that is equivalent to the deterministic density-based one (1). We first briefly review the Hutchinson trace estimator [31], which is a popular stochastic sampling technique used to estimate the trace of a matrix. For alternative trace estimators in the literature, the interested reader is referred to [20, 32] and the references therein.

For a given matrix $\boldsymbol{A} \in \mathbb{R}^{q \times q}$, the Hutchinson trace estimator uses a random vector, $\boldsymbol{\xi} \in \mathbb{R}^q$, whose entries are independent and identically distributed (i.i.d.) and follow the Rademacher distribution (± 1, each with probability $1/2$ [33]). It can be shown that the random vector $\boldsymbol{\xi}$ has the following properties,

$$\mathbb{E}(\boldsymbol{\xi}) = \boldsymbol{0} \text{ and } \mathbb{E}\left(\boldsymbol{\xi}\boldsymbol{\xi}^T\right) = \boldsymbol{I}_q, \tag{4}$$

where $\boldsymbol{I}_q$ is the $q \times q$ identity matrix. It follows then that

$$\mathbb{E}\left(\boldsymbol{\xi}^T \boldsymbol{A} \boldsymbol{\xi}\right) = \text{trace}\left(\boldsymbol{A}\right). \tag{5}$$

Using this concept, the following subsections introduce an equivalent randomized topology optimization formulation for the deterministic topology optimization formulation under many load cases. In the framework of discretized parameter estimation problems with PDE constraints, a randomized algorithm based on (5) was used in [19].

2.2.1 Randomized formulation for the density-based method

Consider the standard topology optimization formulation in (1) with m load cases. We introduce the weighted load and displacement matrices $\boldsymbol{F}, \boldsymbol{U} \in \mathbb{R}^{dN \times m}$, defined as

$$\boldsymbol{F} = [\sqrt{\alpha_1}\, \boldsymbol{f}_1, ..., \sqrt{\alpha_m}\ \boldsymbol{f}_m] \text{ and } \boldsymbol{U} = [\sqrt{\alpha_1}\, \boldsymbol{u}_1, ..., \sqrt{\alpha_m}\, \boldsymbol{u}_m],$$

respectively. With these weighted matrices, we can rewrite the equilibrium equations as $\boldsymbol{U} = \boldsymbol{K}^{-1}\boldsymbol{F}$ and, consequently, the end-compliance and its sensitivities as

$$C(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i \boldsymbol{f}_i^T \boldsymbol{u}_i = \text{trace}\left(\boldsymbol{F}^T \boldsymbol{U}\right) = \text{trace}\left(\boldsymbol{F}^T \boldsymbol{K}^{-1} \boldsymbol{F}\right), \tag{6}$$

$$\begin{aligned}\nabla C_{\boldsymbol{x}}^{(e)} &= -\sum_{i=1}^{m} \alpha_i \boldsymbol{u}_i^T \frac{\partial \boldsymbol{K}}{\partial x^{(e)}} \boldsymbol{u}_i = -\text{trace}\left(\boldsymbol{U}^T \frac{\partial \boldsymbol{K}}{\partial x^{(e)}} \boldsymbol{U}\right) \\ &= -\text{trace}\left(\boldsymbol{F}^T \boldsymbol{K}^{-1} \frac{\partial \boldsymbol{K}}{\partial x^{(e)}} \boldsymbol{K}^{-1} \boldsymbol{F}\right).\end{aligned} \tag{7}$$

Furthermore, by introducing the random vector $\boldsymbol{\xi}$ and applying the Hutchinson trace estimator, we define the randomized topology optimization for the density-based method under m load cases as

$$\begin{aligned} &\min_{\boldsymbol{x}} \left\{ C(\boldsymbol{x}) = \mathbb{E}\left[(\boldsymbol{F}\boldsymbol{\xi})^T \boldsymbol{K}(\boldsymbol{E}(\bar{\boldsymbol{x}}))^{-1} (\boldsymbol{F}\boldsymbol{\xi}) \right] \right\} \\ &\text{s.t.} \sum_{e=1}^{M} \frac{\bar{v}^{(e)} x^{(e)}}{|\Omega|} - V_f = 0, \\ &\quad 0 \le x^{(e)} \le 1, e = 1, ..., M, \\ &\text{with } E^{(e)} = E_{\min} + [\bar{x}^{(e)}]^p (E_0 - E_{\min}),\, e = 1, ..., M. \end{aligned} \tag{8}$$

Again, $\bar{\boldsymbol{x}} = \boldsymbol{H}\boldsymbol{x}$. Accordingly, the sensitivity of the objective function in the above randomized formulation is also given in expectation form as

$$\begin{aligned}\nabla C_{\boldsymbol{x}}^{(e)} &= -\text{trace}\left(\boldsymbol{F}^T \boldsymbol{K}^{-1} \frac{\partial \boldsymbol{K}}{\partial x^{(e)}} \boldsymbol{K}^{-1} \boldsymbol{F}\right) \\ &= -\mathbb{E}\left[(\boldsymbol{F}\boldsymbol{\xi})^T \boldsymbol{K}^{-1} \frac{\partial \boldsymbol{K}}{\partial x^{(e)}} \boldsymbol{K}^{-1} (\boldsymbol{F}\boldsymbol{\xi}) \right].\end{aligned} \tag{9}$$

We remark that, because this randomized formulation (8) is equivalent to the standard one (1), it is convex when $p = 1$ and non-convex when $p > 1$, where p is the penalty parameter in the SIMP approach.

We also remark that, in recent work [18], a variant of the SAA algorithm is proposed where the compliance and its gradient in (8) are estimated as the sample average over n sampled load cases. For the density-based approach, the estimated compliance and its gradient are

$$\widehat{C}^{SAA}(\boldsymbol{x}) = \frac{1}{n} \sum_{k=1}^{n} (\boldsymbol{F}\boldsymbol{\xi}_k)^T \boldsymbol{K}(\boldsymbol{x})^{-1} (\boldsymbol{F}\boldsymbol{\xi}_k) \tag{10}$$

and

$$(\nabla \widehat{C}_{\boldsymbol{x}}^{SAA})^{(e)} = -\frac{1}{n} \sum_{k=1}^{n} (\boldsymbol{F}\boldsymbol{\xi}_k)^T \boldsymbol{K}^{-1} \frac{\partial \boldsymbol{K}}{\partial x^{(e)}} \boldsymbol{K}^{-1} (\boldsymbol{F}\boldsymbol{\xi}_k). \tag{11}$$

A standard deterministic optimization algorithm (i.e., the OC method) is then used to compute the design variable updates based on the estimated gradient. Thus, a sufficiently accurate gradient estimation is needed, and it is shown in [18] that a sample size of $n \approx 6$ provides this accuracy in the considered applications. By doing so, the number of linear system solves at each optimization iteration is reduced from m to n. In the present paper, we take a conceptually different approach. Instead of estimating the gradient using a few samples and then performing an optimization update using the standard algorithm, we treat the randomized formulation (8) as a stochastic optimization problem, and use SA algorithms to compute a design variable update. As a result, as shown below, the accuracy requirement on the gradient estimation is relaxed, and only one sample ($n = 1$) is needed at each optimization step.

For the remainder of this paper, we use the following notation. For a given vector $\boldsymbol{x} \in \mathbb{R}^n$, we denote by $\|\boldsymbol{x}\|_p$ its ℓ_p norm. In particular, $\|\boldsymbol{x}\|_2 = \sqrt{\boldsymbol{x}^T\boldsymbol{x}}$ denotes the Euclidean norm, and $\|\boldsymbol{x}\|_\infty = \max\{|x^{(1)}|, ..., |x^{(n)}|\}$ denotes the max-norm (infinity norm). In addition, we denote by Π_X the metric projection operator onto a closed convex set $X \subset \mathbb{R}^n$,

$$\Pi_X(\boldsymbol{x}) = \arg\min_{\boldsymbol{y} \in X} \|\boldsymbol{x} - \boldsymbol{y}\|_2. \tag{12}$$

3 Mirror descent stochastic approximation (MD-SA)

This section presents the mirror descent stochastic approximation approach for a general stochastic optimization problem. We first introduce the general framework of the MD-SA. We then restrict our attention to a version of the MD-SA algorithms, the Entropic MD-SA with the ℓ_1 norm, which is well-suited for our problem. Another version–the Euclidean MD-SA with the ℓ_2 norm and the comparison between these two algorithms are provided in the Appendices and Example 1.

3.1 General framework

As a direct descendant of the stochastic mirror descent method [26], the mirror descent SA algorithm, developed in [22], is an effective algorithm to solve stochastic optimization problems of the following form:

$$\min_{\boldsymbol{x}\in X}\left\{\phi(\boldsymbol{x})=\mathbb{E}[\Phi(\boldsymbol{x},\boldsymbol{\xi})]\right\}. \tag{13}$$

Here the feasible set $X\subset\mathbb{R}^n$ is assumed to be a nonempty bounded closed convex set, and $\boldsymbol{\xi}$ is a random vector with a given probability distribution. Suppose that the differentiation and expectation operators in the objective function of (13) commute (this holds under mild regularity conditions). Then, we can write the gradient of $\phi(\boldsymbol{x})$ as

$$\nabla\phi(\boldsymbol{x})=\mathbb{E}[\boldsymbol{G}(\boldsymbol{x},\boldsymbol{\xi})], \tag{14}$$

where $\boldsymbol{G}(\boldsymbol{x},\boldsymbol{\xi})=\nabla_x\Phi(\boldsymbol{x},\boldsymbol{\xi})$. Note that if $\Phi(\boldsymbol{x},\boldsymbol{\xi})$ is convex in $\boldsymbol{x}$, then the expected value function $\phi(\boldsymbol{x})$ is also convex. In that case, formula (14) also holds for the corresponding subgradients.

Note that the gradient of the objective function lies in the dual of the feasible space of the design variables. The key idea of the MD-SA algorithm is to obtain the updates of the vector of design variables by mapping the gradient descent into the dual of the primal space. At each optimization step, the mirror descent SA uses a single realization of the random vector $\boldsymbol{\xi}$, independent of the previous iterations. The stochastic gradient $\boldsymbol{G}(\boldsymbol{x},\boldsymbol{\xi})$, at the current vector of design variables, is computed and the algorithm takes a step in the descent direction of the stochastic gradient in the dual space. Finally, the update of the vector of design variables is obtained by "mirroring" the results back into the primal space (i.e., the feasible space of the optimization problem). The mirror descent SA algorithm is a generalization of the robust SA algorithm. For further details, the reader is referred to [22].

We will now describe the general framework of the MD-SA. For any norm $\|\cdot\|$, its dual norm is defined as $\|\boldsymbol{x}\|_*=\sup_{\|\boldsymbol{y}\|\le 1}\boldsymbol{y}^T\boldsymbol{x}$. A function $\omega: X\to\mathbb{R}$ is said to be a distance-generating function modulus $\alpha>0$ with respect to a norm $\|\cdot\|$, if $\omega(\cdot)$ is convex and continuous on X, continuously differentiable on the relative interior of X, and

$$\left(\boldsymbol{x}'-\boldsymbol{x}\right)^T\left(\nabla\omega\left(\boldsymbol{x}'\right)-\nabla\omega\left(\boldsymbol{x}\right)\right)\ge\alpha||\boldsymbol{x}'-\boldsymbol{x}||,\ \text{ for all }\ \boldsymbol{x},\boldsymbol{x}'\in X. \tag{15}$$

Note that (15) implies that ω is strongly convex. The corresponding prox-function $V(\cdot,\cdot)$ is defined as follows:

$$V(\boldsymbol{x},\boldsymbol{z})=\omega(z)-\left[\omega(\boldsymbol{x})+\nabla\omega(\boldsymbol{x})^T(z-\boldsymbol{x})\right]. \tag{16}$$

Here $\boldsymbol{z}$ is a point in the feasible space, and $V(\boldsymbol{x},\cdot)$ is nonnegative and strongly convex with respect to the chosen norm $\|\cdot\|$. The function $V(\cdot,\cdot)$ is also known as the Bergman distance generated by ω.

Based on the prox-function $V(\boldsymbol{x},\boldsymbol{z})$, the prox-mapping $P_{\boldsymbol{x}}:\mathbb{R}^n\to X$ is defined as follows:

$$P_{\boldsymbol{x}}(\boldsymbol{y})=\arg\min_{z\in X}\left\{\boldsymbol{y}^T(\boldsymbol{z}-\boldsymbol{x})+V(\boldsymbol{x},\boldsymbol{z})\right\}. \tag{17}$$

Because of the strong convexity of ω, the minimizer $\boldsymbol{z}^*(\boldsymbol{x},\boldsymbol{y})=P_{\boldsymbol{x}}(\boldsymbol{y})$ is unique. Note that $V(\boldsymbol{x},\boldsymbol{z})$ serves as a regularizer when mapping the point $\boldsymbol{y}$ (representing the gradient information in the dual space) to the primal space to obtain $\boldsymbol{z}$, and $\boldsymbol{x}$ is the current vector of design variables. One example of a distance-generating function is $\omega(\boldsymbol{x})=\frac{1}{2}\boldsymbol{x}^T\boldsymbol{x}$ (see 7). An important advantage of the MD algorithm is that it is possible to adjust the distance-generating function to the geometry of the set X (cf. [22]). The prox-mapping in this paper is defined in terms of the feasible set of the considered optimization problem (8).

Having introduced the concept of the prox-function and prox-mapping, the general update of the MD-SA at the current iterate $\boldsymbol{x}_k$ is

$$\boldsymbol{x}_{k+1}=P_{\boldsymbol{x}_k}\left(\gamma_k\boldsymbol{G}(\boldsymbol{x}_k,\boldsymbol{\xi}_k)\right), \tag{18}$$

where γ_k is a chosen step size and $\boldsymbol{\xi}_k$ is a sample of the random vector $\boldsymbol{\xi}$. A convergence analysis and convergence bounds for the so-defined MD-SA are provided in [22] for convex stochastic optimization problems. Our numerical studies, provided in Section 6, indicate that the MD-SA method works for non-convex problems (i.e. (8) with $p>1$) as well.

The recurrence (18) provides a general iteration framework for MD-SA. By selecting the norm $\|\cdot\|$ and various forms of distance generating functions ω, different forms of design update schemes can be obtained. Appropriate choices of the norm and distance generating function for a given application can substantially improve convergence by adjusting the geometry to the specific constraints. In the following section, the MD-SA algorithm with the ℓ_1 norm will be presented.

3.2 Entropic MD-SA with ℓ_1 norm

The MD-SA method with ℓ_1 norm has been studied in the context of linear programming and online learning, e.g., [34]. The dual of the ℓ_1 norm is the ℓ_∞ ($\|\cdot\|_\infty$) norm. A distance generating function for the ℓ_1 norm MD-SA is the entropy function given by (cf. [22])

$$\omega(\boldsymbol{x}) = \sum_{e=1}^{M} x^{(e)} \ln(x^{(e)}), \tag{19}$$

this leads to the entropic ℓ_1 norm MD-SA. Based on equation (19), the prox-function $V(\cdot,\cdot)$ can be written as

$$V(\boldsymbol{x},\boldsymbol{z}) = \sum_{e=1}^{M} \left[z^{(e)} \ln\left(\frac{z^{(e)}}{x^{(e)}}\right) + x^{(e)} - z^{(e)} \right], \tag{20}$$

and the corresponding prox-mapping

$$P_{\boldsymbol{x}}(\boldsymbol{y}) = \arg\min_{z\in X} \left\{ \sum_{e=1}^{M} \left[\left(y^{(e)}-1\right)\left(z^{(e)}-x^{(e)}\right) + z^{(e)} \ln\left(\frac{z^{(e)}}{x^{(e)}}\right) \right] \right\}. \tag{21}$$

When the feasible set is the standard simplex,

$$X = \left\{ \boldsymbol{x} \in \mathbb{R}^M : \sum_{e=1}^{M} x^{(e)} = 1,\ x^{(e)} \geq 0, \quad e = 1,...,M \right\}, \tag{22}$$

a closed form expression of the prox-function is

$$[P_{\boldsymbol{x}}(\boldsymbol{y})]^{(e)} = \frac{x^{(e)} e^{-y^{(e)}}}{\sum_{j=1}^{M} x^{(j)} e^{-y^{(j)}}}, \quad e = 1,...,M. \tag{23}$$

Thus the update $x_{k+1}^{(e)} = \left[P_{\boldsymbol{x}_k}\left(\gamma_k \boldsymbol{G}(\boldsymbol{x}_k,\boldsymbol{\xi}_k)\right)\right]^{(e)}$ for each component in the entropic ℓ_1 norm MD-SA can be written as

$$x_{k+1}^{(e)} = \frac{x_k^{(e)} e^{-\gamma_k \boldsymbol{G}(\boldsymbol{x}_k,\boldsymbol{\xi}_k)^{(e)}}}{\sum_{j=1}^{M} x_k^{(j)} e^{-\gamma_k \boldsymbol{G}(\boldsymbol{x}_k,\boldsymbol{\xi}_k)^{(j)}}}, \quad e = 1,...,M. \tag{24}$$

An important question for the MD-SA method is the choice of the step sizes γ_k; we will introduce the strategies for step size selection in Section 5.1.

4 Randomized topology optimization with an entropic ℓ_1 norm MD-SA update algorithm

In this section, we propose an MD-SA algorithm tailored to efficiently solve the randomized topology optimization formulation (8). As it is tailored to the geometry of the feasible design space, we use the entropic ℓ_1 norm MD-SA algorithm described in Section 3.2. Furthermore, in the derived update algorithm, we introduce a move limit for each design variable at each optimization step that allows us to effectively incorporate the damping scheme described in Section 5.2.

According to (8), the feasible set for the design variable vector $\boldsymbol{x}$ is given by

$$X_{\boldsymbol{x}} = \left\{ \boldsymbol{x} \in \mathbb{R}^M : \sum_{e=1}^{M} \frac{\bar{v}^{(e)} x^{(e)}}{|\Omega|} - V_f = 0, \quad 0 \leq x^{(e)} \leq 1, \quad e = 1,...,M \right\}. \tag{25}$$

At iteration k with design variable vector $\boldsymbol{x}_k$, the compliance estimator $\widehat{C}^{SA}$ takes the form

$$\widehat{C}^{SA}(\boldsymbol{x}_k) = \left(\boldsymbol{F}\boldsymbol{\xi}_k\right)^T \boldsymbol{K}(\boldsymbol{x}_k)^{-1} \left(\boldsymbol{F}\boldsymbol{\xi}_k\right), \tag{26}$$

and the stochastic gradient $\boldsymbol{G}(\boldsymbol{x}_k,\boldsymbol{\xi}_k)$, using a *single* sample $\boldsymbol{\xi}_k$, is given in component form as

$$G^{(e)}(\boldsymbol{x}_k,\boldsymbol{\xi}_k) = (\nabla \widehat{C}^{SA}_{\boldsymbol{x}})^{(e)} = -\left(\boldsymbol{F}\boldsymbol{\xi}_k\right)^T \boldsymbol{K}^{-1} \frac{\partial \boldsymbol{K}}{\partial x^{(e)}} \boldsymbol{K}^{-1} \left(\boldsymbol{F}\boldsymbol{\xi}_k\right). \tag{27}$$

Here, we map the vector of design variables to a scaled feasible set (which is the standard simplex), compute the updated design variable vector using the proposed entropic ℓ_1 norm MD-SA update, and then map the updated design variable vector back to the original feasible set. Following this procedure, we introduce $\tilde{\boldsymbol{x}}$ as the scaled design variable vector, whose e-th component is given by $\tilde{x}^{(e)} = \frac{\bar{v}^{(e)} x^{(e)}}{|\Omega| V_f}$. The scaled feasible set $\widetilde{X}_{\boldsymbol{x}}$ for $\tilde{\boldsymbol{x}}$ is defined as

$$\widetilde{X}_{\boldsymbol{x}} = \left\{ \tilde{\boldsymbol{x}} \in \mathbb{R}^M : \sum_{e=1}^{M} \tilde{x}^{(e)} = 1, \quad 0 \leq \tilde{x}^{(e)} \leq \frac{\bar{v}^{(e)}}{|\Omega| V_f}, \quad e = 1,...,M \right\}. \tag{28}$$

Accordingly, the stochastic gradient with respect to $\tilde{\boldsymbol{x}}$, denoted by $\widetilde{\boldsymbol{G}}(\tilde{\boldsymbol{x}}_k,\boldsymbol{\xi}_k)$, is also scaled in component form as

$$\widetilde{G}^{(e)}(\tilde{\boldsymbol{x}}_k,\boldsymbol{\xi}_k) = \frac{\partial x^{(e)}}{\partial \tilde{x}^{(e)}} G^{(e)}(\boldsymbol{x}_k,\boldsymbol{\xi}_k) = \frac{|\Omega| V_f}{\bar{v}^{(e)}} G^{(e)}(\boldsymbol{x}_k,\boldsymbol{\xi}_k). \tag{29}$$

Given the scaled feasible set and scaled design variable vector, the prox-mapping generated by the entropy function (19) takes the form

$$P_{\tilde{\boldsymbol{x}}}(\tilde{\boldsymbol{y}}) = \arg\min_{\tilde{\boldsymbol{z}}\in \widetilde{X}_x} \left\{ \sum_{e=1}^{M} \left[\left(\tilde{y}^{(e)}-1\right)\left(\tilde{z}^{(e)}-\tilde{x}^{(e)}\right) + \tilde{z}^{(e)} \ln\left(\frac{\tilde{z}^{(e)}}{\tilde{x}^{(e)}}\right) \right] \right\}, \tag{30}$$

and the entropic ℓ_1 norm MD-SA update then is given by

$$\tilde{\boldsymbol{x}}_{k+1} = P_{\tilde{\boldsymbol{x}}_k}(\gamma_k \tilde{\boldsymbol{G}}(\tilde{\boldsymbol{x}}_k, \boldsymbol{\xi}_k)). \tag{31}$$

Comparing the above expression to (22), we notice that the scaled set $\widetilde{X}_x$ includes an additional upper bound for each design variable, $\tilde{x}^{(e)} \leq \frac{\bar{v}^{(e)}}{|\Omega|V_f}$. Therefore, the closed-form expression (24) cannot be applied directly, because the updated variables could violate their upper bounds. Thus, an iterative formula is derived here to solve for $\tilde{\boldsymbol{x}}_{k+1}$ in the constrained optimization problem (30)-(31). To achieve this, we first introduce a Lagrange multiplier $\tilde{\lambda}$ for the equality constraint in the definition of $\widetilde{X}$ and form the Lagrangian function

$$\mathcal{L}(\tilde{\boldsymbol{z}}, \tilde{\lambda}) = \sum_{e=1}^{M} \left[\left(\tilde{y}^{(e)} - 1\right)\left(\tilde{z}^{(e)} - \tilde{x}^{(e)}\right) + \tilde{z}^{(e)} \ln\left(\frac{\tilde{z}^{(e)}}{\tilde{x}^{(e)}}\right) \right] + \tilde{\lambda}\left(\sum_{e=1}^{M} \tilde{z}^{(e)} - 1\right), \tag{32}$$

where we assume $\tilde{\boldsymbol{y}}$ and $\tilde{\boldsymbol{x}}$ are given vectors in the optimization. The optimality condition of (32) with respect to $\tilde{z}^{(e)}$ states that

$$\frac{\partial \mathcal{L}}{\partial \tilde{z}^{(e)}}(\tilde{\boldsymbol{z}}, \tilde{\lambda}) = \tilde{y}^{(e)} + \ln\left(\frac{\tilde{z}^{(e)}}{\tilde{x}^{(e)}}\right) + \tilde{\lambda} = 0, \tag{33}$$

which gives

$$\tilde{z}^{(e)}(\tilde{\lambda}) = \tilde{x}^{(e)} e^{-\tilde{y}^{(e)}} e^{-\tilde{\lambda}}. \tag{34}$$

By introducing $\tilde{\mu} = e^{-\tilde{\lambda}}$ (notice that $\tilde{\mu} > 0$), we can simplify the above expression as

$$\tilde{z}^{(e)}(\tilde{\mu}) = \tilde{\mu}\tilde{x}^{(e)} e^{-\tilde{y}^{(e)}}, \tag{35}$$

and incorporating the upper bound (28) into equation (34) gives

$$\tilde{z}^{(e)}(\mu) = \min\left(\frac{\bar{v}^{(e)}}{|\Omega|V_f}, \tilde{\mu}\tilde{x}^{(e)} e^{-\tilde{y}^{(e)}}\right). \tag{36}$$

The stationary condition of (32) with respect to $\tilde{\mu}$ yields,

$$\frac{\partial \mathcal{L}}{\partial \tilde{\mu}}(z(\mu)) = \frac{\partial \mathcal{L}}{\partial \tilde{\lambda}}(z(\mu))\frac{\tilde{\lambda}}{\tilde{\mu}} = \sum_{e=1}^{M} z^{(e)}(\mu) - 1 = 0. \tag{37}$$

Observe that (37) is monotonic with respect $\tilde{\mu}$ and can be solved by various methods, e.g., the bisection method that is used in this work. Denote $\tilde{\mu}_k^*$ as the solution of (37) at step k. By plugging in $\tilde{\mu}_k^*$, step size γ_k, and the scaled stochastic gradient (29), the updated (scaled) design variable vector $\tilde{\boldsymbol{x}}_{k+1}$ takes the form,

$$\tilde{x}_{k+1}^{(e)} = \tilde{z}^{(e)}(\tilde{\mu}_k^*) = \min\left[\frac{\bar{v}^{(e)}}{|\Omega|V_f}, \tilde{\mu}_k^* \tilde{x}_k^{(e)} e^{-\gamma_k \widetilde{G}^{(e)}(\tilde{\boldsymbol{x}}_k, \boldsymbol{\xi}_k)}\right]. \tag{38}$$

Once $\tilde{\boldsymbol{x}}_{k+1}$ is obtained, the last step in the proposed entropic ℓ_1 norm MD-SA algorithm is to map it back to $\boldsymbol{x}_{k+1}$ in the original feasible set as

$$x_{k+1}^{(e)} = \frac{|\Omega|V_f}{\bar{v}^{(e)}}\tilde{x}_{k+1}^{(e)} = \min\left[1, \tilde{\mu}_k^* x_k^{(e)} e^{-\gamma_k \widetilde{G}^{(e)}(\tilde{\boldsymbol{x}}_k, \boldsymbol{\xi}_k)}\right]. \tag{39}$$

The update formula (39) is the final expression of the entropic ℓ_1 norm MD-SA algorithm for the randomized topology optimization problem. *Note that the gradient information in* (39) *is accessed through an estimate,* $\boldsymbol{G}(\boldsymbol{x}_k, \boldsymbol{\xi}_k)$*, by using a single sample* $\boldsymbol{\xi}_k$. The entropic ℓ_1-norm MD-SA algorithm (39) has advantages over standard gradient-based algorithms (e.g., OC and MMA) for stochastic optimization problems. Using a standard gradient-based update algorithm for a stochastic optimization problem requires moderate accuracy of the estimated gradient, which leads to an increase of the sample size required. On the other hand, the MD-SA update in (39) only needs low accuracy in the stochastic gradient, and thus only needs a single sample, meaning the sample size is always one. If the required gradient information is computationally expensive, the entropic ℓ_1 norm MD-SA algorithm is likely to be preferable compared with standard gradient-based algorithms.

Furthermore, for the structural optimization problems solved by the entropic ℓ_1 norm MD-SA algorithm in this paper, we propose a step size strategy with a damping scheme that creates a decaying step size set [18], which fits well with the structural optimization framework and stochastic algorithms. The proposed step size strategy and damping scheme is discussed in detail in Section 5. To incorporate the damping scheme, we introduce an additional move limit, denoted by *move* $\in (0,1]$, to the vector of design variables in the entropic ℓ_1 norm MD-SA update. At iteration k with design variable vector $\boldsymbol{x}_k$, the move limit modifies the lower bound (LB_k) and upper bound (UB_k) of the update of the vector of design variables as follows,

$$LB_k^{(e)} = \max\{x_k^{(e)} - move, 0\} \quad \text{and} \quad UB_k^{(e)} = \min\{x_k^{(e)} + move, 1\}. \tag{40}$$

Incorporating the modified LB_k and UB_k, (39) is changed to

$$x_{k+1}^{(e)} = \max\left\{LB_k^{(e)}, \min\left[UB_k^{(e)}, \tilde{\mu}_k^* x_k^{(e)} e^{-\gamma_k \widetilde{G}^{(e)}(\tilde{\boldsymbol{x}}_k, \boldsymbol{\xi}_k)}\right]\right\}. \tag{41}$$

We note that the specific form of update (39) is based on the feasible space defined by a simplex and the fact that the linear volume constraint is always active, the latter is always the case for compliance minimization problems. No specific form of the objective function is assumed. Thus, the update formula applies to other objective functions in forms of expectation of a random function, with linear (in terms of design variables) equality and active inequality constraints. For other constraint functions, one needs to adjust the projection (e.g., norm and distance generating function) of computed stochastic gradients to the geometry of the feasible space, which is determined by the specific constraint functions. For a detail discussion, we refer to [22] and [35].

5 Algorithmic strategies for randomized optimization

The choice of step size strategy is essential for SA algorithms. In this section, we propose a step size strategy with a damping scheme. The main idea is to calculate and re-calibrate step sizes in different stages of the optimization, and to reduce the move limit when the average progress per step drops below a tolerance. Furthermore, to improve the performance of the proposed algorithm, we introduce a technique that averages the history of design variable updates and a re-calibration strategy for step size. Finally, we review an iterative solver with recycling for solving the large linear systems to further reduce the computational cost.

5.1 Step size strategy

In SA algorithms, a key step is to determine the step size, γ_k, of the update (39). Thus, this subsection presents a formula to determine the step size γ_k. In general, there are two types of step size strategies, constant and varying [22]. The constant step size policy assumes a priori a total number of iterations, the step size is constant throughout the optimization, and the algorithm always ends at the prescribed maximum step. For the varying step size policy, the step size is a function of the iteration number and it decreases over the optimization process.

In [22], a constant step size is discussed for convex objective functions. The main idea is to calculate a step size that minimizes the error estimate of the stochastic optimization solutions. A formula for the constant step size policy is given as,

$$\gamma_k = \frac{\theta\sqrt{2}D_{\omega,X}}{B_*\sqrt{N_{\max}}}, \quad k = 1,\dots,N_s, \tag{42}$$

where $\theta > 0$ is a scaling parameter, $N_{\max}$ is the number of allowable iterations, B_* is an estimate of an upper bound on the stochastic gradient norm, i.e.,

$$B_*^2 \geq \mathbb{E}\left[||\boldsymbol{G}(\boldsymbol{x},\boldsymbol{\xi})||_\infty^2\right], \tag{43}$$

and $D_{\omega,X}$ is the ω-diameter of X defined as

$$D_{\omega,X} = \left[\max_{z\in X}\omega(z) - \min_{z\in X}\omega(z)\right]^{\frac{1}{2}}. \tag{44}$$

We adopt (42) to calculate the step size. In our case, $D_{\omega,X} = \sqrt{\log M}$, and we take the scaling parameter $\theta = 1$. Because the bound B_* for the stochastic gradient is unknown, we use the ℓ_∞ norm of the estimated gradient calculated by the sample average of a random sample with size $n = 6$ at the first optimization step, namely, $B_* = 1/n\|\sum_{i=1}^n \boldsymbol{G}(\boldsymbol{x}_0,\boldsymbol{\xi}_i)\|_\infty$, where $\boldsymbol{x}_0$ stands for the initial guess for the vector of design variables. This gives for the step size

$$\gamma_k = \frac{\sqrt{2\log M}}{B_*\sqrt{N_{\max}}}, \quad k = 1,\dots,N_{\max}, \tag{45}$$

which is used in the entropic ℓ_1 norm MD-SA.

5.2 Proposed damping scheme

Using the constant step size policy (45), the optimization terminates at the prescribed total number of steps, $N_{\max}$. To promote convergence before the maximum number of iterations is reached, we incorporate a damping scheme based on the one introduced in [18], which gradually reduces the move limit of the updates. The advantage of using the damping scheme is that it allows us to monitor the progress of the update throughout the optimization and damp the update appropriately.

Inspired by simulated annealing [36, 37], the damping scheme proposed in [18] evaluates the average progress per step and reduces the move limit whenever the average progress drops below a tolerance. The effective step ratio R_k for iteration k is defined as:

$$R_k = \frac{\frac{1}{N_D}\|\boldsymbol{x}_k - \boldsymbol{x}_{(k-N_D+1)}\|}{\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|}, \tag{46}$$

where N_D is the sample window size (see below). This effective step ratio serves as an indicator of the optimizer's status, i.e., if the ratio is relatively large, then the optimizer is making progress; and if the ratio is relatively small (typically smaller than 0.1), then the steps are not effective.

The damping scheme works as follows: once R_k is below a prescribed tolerance, i.e., $R_k < \tau_D$, we reduce the allowable move limit (i.e. *move*) (40) by a prescribed scale factor κ, namely *move* $=$ *move*$/\kappa$. To avoid damping the update too early because of insufficient history data, we do not damp the update in the first N_D optimization steps, where N_D is a chosen parameter. The damping scheme evaluates the average progress of the optimizer to determine damping, thus the window size for the damping scheme N_D needs to be large enough to have a conservative damping and a smooth convergence. However, as the window size increases, we average over more steps. If N_{D} is too large, it slows down convergence because the damping algorithm adapts more slowly. In practice, we have found $N_{\mathrm{D}} = 100$ is sufficient for problems containing more than one thousand design variables.

5.3 Averaging the iterates

The classical SA and MD-SA algorithms, going back to [38], are sensitive to the step size and may perform poorly in practice. In order to make the SA (and MD-SA) algorithm (more) robust, it was suggested in [39, 40] to use larger step sizes and to weighted average the resulting iterates over a chosen window (even earlier this was discussed in [26]). If we denote the window size as N_{A}, then current iteration k is the end of the window and step $j = k - N_{\mathrm{A}} + 1$ is the start of the window. The weighted averaged design variable vector at step k, denoted as $\hat{\boldsymbol{x}}_k^j$, is calculated as follows,

$$\hat{\boldsymbol{x}}_k^j = \sum_{t=j}^{k} \nu_t \boldsymbol{x}_t, \quad \text{with} \quad \nu_t = \frac{\gamma_t}{\sum_{t=j}^{k} \gamma_t}, \tag{47}$$

where γ_t is the step size in iteration t and ν_t is the associated weight. In case of the constant step size strategy, we have that $\hat{\boldsymbol{x}}_k^j = \frac{1}{N_{\mathrm{A}}} \sum_{t=j}^{k} \boldsymbol{x}_t$.

In this algorithm, the weighted averaged design variable vector, $\hat{\boldsymbol{x}}_k^j$, represents the current solution. As discussed in [22], the weighted averaged design variable vector converges to an optimal solution for convex stochastic optimization problems, thus we use the weighted averaged design variable vector to determine the current estimate of the optimal objective value and the optimized structure. On the other hand, the design variable vector $\boldsymbol{x}_k$ is used in computing the stochastic gradient and performing the design variable updates, as suggested in [22]. The window size for averaging solutions (N_{A}) directly determines the current solution, thus it needs to be smaller than N_{D} so as to achieve faster progress, because larger N_{A} leads to slow progress during optimization. In practice, we have found N_{A} = 50 is sufficient for problems containing more than one thousand design variables.

5.4 Re-calibration of step size

Because the (constant) step size is evaluated based on the initial guess at the start of the algorithm and kept constant thereafter, it may perform poorly at later iterations. Therefore, we propose a step size re-calibration scheme which re-evaluates the step size periodically during the optimization procedure. The prescribed parameter *calibration* defines the number of times we perform the re-calibration. In each re-calibration, we take the final $\hat{\boldsymbol{x}}_k$ (after the optimization is converged) as the initial guess $\boldsymbol{x}_0$ and restart the optimization process by re-calculating the step size according to (45) with an updated B_* and $\boldsymbol{x}_0$ (replaced by $\hat{\boldsymbol{x}}_k$).

5.5 Iterative solver: MINRES with recycling

Although the number of linear systems to solve is drastically reduced by the MD-SA method, we still have a long sequence of large linear systems to solve, with the systems changing only modestly from one optimization step to the next. Hence, we use a preconditioned iterative method with recycling, the recycling MINRES (RMINRES) method [41–43], which was derived from the MINRES method [44, 45]. RMINRES uses approximate invariant subspaces computed during previous linear solves to accelerate the convergence for subsequent linear systems. Recycling an approximate invariant subspace effectively removes the corresponding eigenvalues from the spectrum of the matrix over the resulting Krylov space [42, 46]. In particular, if we remove the largest or smallest eigenvalues for a symmetric, positive definite matrix in this fashion, this improves the theoretical convergence bounds of the MINRES method [45]. The most common choice, also used here, is to effectively remove the smallest eigenvalues by approximating the corresponding invariant subspace. Compared with MINRES or with the method of conjugate gradients (CG) [47, 48] this generally leads to substantially faster convergence [41–43, 49].

We outline the main steps in the RMINRES algorithm for solving the linear system $\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}$, in a sequence of linear systems. For details, including a MATLAB® code, see [42, 43].

Let the columns of the matrix $\boldsymbol{W} \in \mathbb{R}^{dN\times k}$ span the approximate invariant subspace that we recycle from previous linear systems, and assume that $\boldsymbol{W}$ has been computed such that $\boldsymbol{Y} = \boldsymbol{K}\boldsymbol{W}$ has orthonormal columns, that is, $\boldsymbol{Y}^T\boldsymbol{Y} = \boldsymbol{I}$ (the identity matrix). Given an initial guess, $\tilde{\boldsymbol{u}}_0$, and its residual, $\tilde{\boldsymbol{r}}_0 = \boldsymbol{f} - \boldsymbol{K}\tilde{\boldsymbol{u}}_0$, we compute $\boldsymbol{u}_0 = \tilde{\boldsymbol{u}}_0 + \boldsymbol{W}\boldsymbol{Y}^T\tilde{\boldsymbol{r}}_0$ and the updated residual $\boldsymbol{r}_0 = \tilde{\boldsymbol{r}}_0 - \boldsymbol{Y}\boldsymbol{Y}^T\tilde{\boldsymbol{r}}_0$, set $\boldsymbol{v}_1 = \boldsymbol{r}_0/\|\boldsymbol{r}_0\|_2$, and we carry out a Lanczos iteration with additional orthogonalization against $\boldsymbol{Y}$,

$$\begin{aligned} \boldsymbol{v}_2 t_{2,1} &= \boldsymbol{K}\boldsymbol{v}_1 - \boldsymbol{Y}\boldsymbol{b}_1 - \boldsymbol{v}_1 t_{1,1}, \\ \boldsymbol{v}_{j+1} t_{j+1,j} &= \boldsymbol{K}\boldsymbol{v}_j - \boldsymbol{Y}\boldsymbol{b}_j - \boldsymbol{v}_j t_{j,j} - \boldsymbol{v}_{j-1} t_{j-1,j}, \end{aligned} \tag{48}$$

where $\boldsymbol{b}_j = \boldsymbol{Y}^T\boldsymbol{K}\boldsymbol{v}_j$, $t_{j,j} = \boldsymbol{v}_j^T\boldsymbol{K}\boldsymbol{v}_j$, and $t_{j-1,j} = t_{j,j-1} = \|\boldsymbol{K}\boldsymbol{v}_{j-1} - \boldsymbol{Y}\boldsymbol{b}_{j-1} - \boldsymbol{v}_{j-1}t_{j-1,j-1} - \boldsymbol{v}_{j-2}t_{j-2,j-1}\|_2$ was computed in the previous iteration. At the mth iteration, with $\boldsymbol{V}_{m+1} = [\boldsymbol{v}_1\ \boldsymbol{v}_2\ \ldots\ \boldsymbol{v}_{m+1}]$ and $\boldsymbol{V}_{m+1}^T\boldsymbol{V}_{m+1} = \boldsymbol{I}_{m+1}$, $\boldsymbol{B}_m = [\boldsymbol{b}_1\ \boldsymbol{b}_2\ \ldots\ \boldsymbol{b}_m]$, and $\underline{\boldsymbol{T}}_m \in \mathbb{R}^{(m+1)\times m}$ a tridiagonal matrix with the $t_{i,j}$ above as coefficients and its leading $m \times m$ part symmetric, we have

$$\boldsymbol{K}\boldsymbol{V}_m = \boldsymbol{Y}\boldsymbol{B}_m + \boldsymbol{V}_{m+1}\underline{\boldsymbol{T}}_m. \tag{49}$$

Minimizing the residual for a solution of the form

$$\boldsymbol{u}_m = \boldsymbol{u}_0 + \boldsymbol{W}\boldsymbol{z} + \boldsymbol{V}_m\boldsymbol{y}, \tag{50}$$

gives

$$\begin{aligned} \boldsymbol{r}_m &= \boldsymbol{f} - \boldsymbol{K}\boldsymbol{u}_0 - \boldsymbol{K}\boldsymbol{W}\boldsymbol{z} - \boldsymbol{K}\boldsymbol{V}_m\boldsymbol{y} &(51)\\ &= \boldsymbol{r}_0 - \boldsymbol{Y}\boldsymbol{z} - (\boldsymbol{Y}\boldsymbol{B}_m + \boldsymbol{V}_{m+1}\underline{\boldsymbol{T}}_m)\boldsymbol{y} \\ &= \boldsymbol{V}_{m+1}(\boldsymbol{e}_1\|\boldsymbol{r}_o\|_2 - \underline{\boldsymbol{T}}_m\boldsymbol{y}) - \boldsymbol{Y}(\boldsymbol{z}+\boldsymbol{B}_m\boldsymbol{y}). &(52)\end{aligned}$$

Since the matrix $[\boldsymbol{Y}\ \boldsymbol{V}_{m+1}]$ has all orthonormal columns, minimizing $\|\boldsymbol{r}_m\|_2$ involves (i) solving for $\boldsymbol{y}$ in exactly the same way as in standard MINRES, which can be done efficiently with a short term recurrence, (ii) choosing $\boldsymbol{z} = -\boldsymbol{B}_m\boldsymbol{y}$, and (iii) updating $\boldsymbol{u}_m$ according to (50) at the end of the linear solve.

The implementation details, including updating the matrix $\boldsymbol{W}$ that defines the recycle space for subsequent linear systems, are outside the scope of this paper; for this we refer to [42, 43].

In addition, as the MD-SA algorithm using a single sample will only provide gradient estimates with low accuracy, there is no need to solve the linear systems very accurately. Hence, we can reduce the computational cost further by using a low relative convergence tolerance. We provide results of these experiments for a 3D design in Section 6.3.

5.6 Entropic ℓ_1 norm MD-SA algorithm

The proposed entropic ℓ_1 norm MD-SA algorithm including algorithmic parameters for randomized topology optimization is summarized in Algorithm 1.

Algorithm 1 Randomized topology optimization using entropic ℓ_1 norm MD-SA

```
1: Initialize: $x_0$, $N_{\max}$, $\tau_{opt}$, $\tau_D$, $N_D$, move, calibration
2: for $k = 0, 1, ..., N_{\max}$ do
3:     if $k = 0$ then
4:         Evaluate $\gamma_k$ based on (45)
5:     end if
6:     Select $\xi_k$ and evaluate $G(x_k, \xi_k)$
7:     Compute $x_{k+1}$ based on (41)
8:     Calculate $\hat{x}_{k+1}$ based on (47)
9:     Evaluate the effective step ratio $R_k$
10:    if $R_k < \tau_D$ and $k > N_D$ then
11:        $move = move/\kappa$
12:    end if
13:    if $\|\hat{x}_{k+1} - \hat{x}_k\|_\infty < \tau_{opt}$ then
14:        quit
15:    end if
16: end for
17: if $calibration > 0$ then
18:     $calibration = calibration - 1$
19:     $x_0 \leftarrow \hat{x}_{k+1}$ and goto step 2
20: else
21:     Evaluate final compliance $C(\hat{x}_{k+1})$ and plot final topology
22: end if
```

6 Numerical examples

In this section, we present several numerical examples in both two and three dimensions to demonstrate the effectiveness and the computational efficiency of the proposed entropic ℓ_1 norm MD-SA algorithm for topology optimization.

The first example compares the entropic ℓ_1 norm MD-SA with the ℓ_2 norm MD-SA (see Appendix A). The second example compares the entropic ℓ_1 norm MD-SA and the randomized algorithm proposed in [18], which is a variant of the SAA algorithm, coupled with a common optimization update algorithm (i.e., OC). In the last example, we use a 3D problem to demonstrate the use of an iterative method with a relatively low convergence tolerance for the entropic ℓ_1 norm MD-SA algorithm to further reduce the computational cost. In all examples, we also compare the results from these stochastic algorithms with those from the standard deterministic algorithm. The examples are summarized in Table 1.

To quantify the computational cost of the standard and stochastic optimization algorithms, we define the total number of linear systems of equations to solve in the optimization process as $N_{\text{solve}} = m \times N_{\text{step}}$ for the standard deterministic approach, and $N_{\text{solve}} = n \times N_{\text{step}}$ for stochastic algorithms, where N_{step} is the number of optimization steps. This is a measure of the computational efficiency of the optimization formulation. For large 3D systems, where iterative solvers are required, N_{solve} is a good proxy for computational cost. In addition, we include the wall clock time of the entire optimization process for comparison. All the examples are performed on a machine with an Intel(R) Xeon(R) CPU E5-1660 v3, 3.00 GHz processor and 256 GB of RAM, running Matlab R2018b. The optimization is considered converged if the current step size (bounded by the move limit) is below a prescribed tolerance τ_{opt} for the optimization process, that is, $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_\infty < \tau_{\text{opt}}$.

We have incorporated the proposed MD-SA algorithm into the computer program PolyTop [50]. All the problems are initialized as follows. The initial guess is taken as $x_0^{(e)} = V_f$. The convergence tolerance is $\tau_{\text{opt}} = 10^{-2}$; the initial move limit is chosen as $move = 0.1$. The maximum number of optimization steps for the standard deterministic approach is $N_{max} = 400$.

The entropic ℓ_1 norm MD-SA algorithm uses the following parameters. The sample size is chosen to be $n = 1$, the window size to weighted average iterates is $N_A = 50$; for step size calculation, $\theta = 1$, $N_{\max} = 400$, the initial move limit is $move = 0.1$; the calibration parameter is $calibration = 1$; the sample size to estimate the initial gradient for stepsize calculation is 6. In the damping scheme, the window size is $N_D = 100$, and we use a step size reduction factor $\kappa = 2$, and the tolerance for the damping $\tau_D = 0.05$. For the ℓ_2 norm MD-SA algorithm, $\theta = 50$ is used for step size calculation, all other parameters are the same as for the entropic ℓ_1 norm MD-SA algorithm. For the SAA-based randomized algorithm, the sample size is chosen to be $n = 6$, other parameters are as follows: $move = 0.1$, $N_D = 100$, $\kappa = 2$, and $\tau_D = 0.05$.

Journal of Applied Mechanics. Received November 13, 2019;
Accepted manuscript posted December 27, 2019. doi:10.1115/1.4045902
Copyright © 2020 by ASME

Let $\boldsymbol{x}^*$ and $\boldsymbol{x}^*_{SA}$ represent the optimal solutions obtained from the standard formulation in (1) and the MD-SA algorithms, respectively. For the stochastic algorithms, we present the true values of the objective function $C(\boldsymbol{x}^*_{SA})$ at the solution $\boldsymbol{x}^*_{SA}$ (instead of its estimator $\widehat{C}^S(\boldsymbol{x}^*_{SA})$) and compare these with the ones obtained from the standard algorithm, $C(\boldsymbol{x}^*)$, to evaluate the quality of the solutions. The relative difference is defined as $\Delta C = \left(C(\boldsymbol{x}^*_{SA}) - C(\boldsymbol{x}^*)\right)/C(\boldsymbol{x}^*)$. To distinguish between different stochastic algorithms, we use $\boldsymbol{x}^*_{L1-SA}$, $\boldsymbol{x}^*_{L2-SA}$, and $\boldsymbol{x}^*_{SAA}$ to denote the optimal solutions from the entropic ℓ_1 norm MD-SA algorithm, ℓ_2 norm MD-SA algorithm, and the SAA-based randomized algorithm, respectively.

6.1 Example 1: 2D disk with 200 load cases

In this example, we demonstrate the computational efficiency as well as the effectiveness of two MD-SA algorithms, entropic ℓ_1 norm and ℓ_2 norm (Euclidean) MD-SA, for various problem sizes. We show that, as the problem size increases, the entropic ℓ_1 norm MD-SA performs better than the ℓ_2 norm MD-SA.

Here, we consider a 2D disk whose design domain, boundary conditions, and passive zone (non-designable solid) are shown in Fig. 2a. The domain is discretized with continuum polygonal elements [50]. We enforce the symmetry of the density distribution in both horizontal and vertical axes. A total of 200 linearly independent and equally weighted load cases are applied at the outer circle of the domain (Figure 2b). For this design example, the active volume fraction (excluding the passive zone) is $V_f = 0.25$, the radius of the density filter is chosen as 0.05, and the penalization factor, p, is taken to be 3.0. Moreover, to demonstrate the difference between designs considering a combined single load case and multiple load cases, we include the case where all the 200 loads are applied simultaneously (i.e., 1 deterministic load case with all the 200 loads acting together). The corresponding optimized topology is shown in Fig. 3(a), which is distinct from all other designs which consider 200 load cases (Figs. 3(b)-(d)).

To investigate the performance and computational efficiency of the entropic ℓ_1 norm MD-SA ($n = 1$) and the ℓ_2 norm MD-SA ($n = 1$) algorithms versus the standard deterministic approach ($m = 200$), we consider four problem sizes: $M = 10,000; 40,000; 70,000;$ and $100,000$. The optimized topologies (from representative trials) obtained by these three methods (the standard deterministic approach, the entropic ℓ_1 norm MD-SA, and the ℓ_2 norm MD-SA) for problem size $M = 40,000$ are shown in Figs. 3 (b)-(d). For this problem size, the entropic ℓ_1 norm MD-SA algorithm leads to a topology with a similar (true) compliance value as the standard deterministic formulation. The ℓ_2 norm MD-SA algorithm, on the other hand, produces a final topology with a slightly higher (true) compliance value.

In order to evaluate the stability of the stochastic algorithms, we run each of the MD-SA algorithms (the entropic ℓ_1 norm MD-SA and the ℓ_2 norm MD-SA) 50 times and present the statistics of the results in Table 2 (this is not needed in practice). "Mean" and "Dev" stand for the mean and the standard deviation, over 50 trials, of the objective function value evaluated at the obtained solutions. Note that one trial is one run of the numerical experiment. For the standard deterministic approach, we present two sets of results in Table 2. The first set of results is obtained, if we stop the optimization when $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_\infty < \tau_{\text{opt}} = 10^{-2}$ or when the number of optimization steps reaches the given maximum, $N_{max} = 400$. This strategy is also used in Examples 2 and 3. The second set of results is obtained, if we terminate the optimization only when $\tau_{\text{opt}} = 10^{-2}$ is reached. In the comparison with other algorithms, we mainly use the results from the first strategy in the standard deterministic approach, because we also use $N_{max} = 400$ for the MD-SA algorithms.

Figures 4 and 5 depict the final (true) compliances versus the corresponding number of design variables for the entropic ℓ_1 norm MD-SA (for 50 trials) and the ℓ_2 norm MD-SA (for 50 trials) algorithms, respectively. In these figures, the final compliances from the standard deterministic algorithm with 200 load cases are included for comparison. Representative optimized topologies obtained from the entropic ℓ_1 norm and ℓ_2 norm MD-SA algorithms are provided as well.

The entropic ℓ_1 norm MD-SA algorithm, while offering similar solutions (in terms of the objective function values) to the standard approach (-7.60%, -1.06%, 0.14%, and 1.41% relative differences, respectively for $M = 10,000$, $M = 40,000$, $M = 70,000$, and $M = 100,000$), drastically reduces the computational cost for all the problem sizes considered. As shown in Table 2, for the largest problem ($M = 100,000$), the number of linear systems to solve (N_{solve}) on average is 223 times smaller than for the standard deterministic algorithm (i.e., 358 solves vs. 80,000 solves), and the total wall clock time is 22 times lower (i.e., 20 minutes vs. 7.5 hours). The ℓ_2 norm MD-SA algorithm provides comparable solutions with the standard approach for the smallest problem size ($M = 10,000$), as shown in Fig. 5. However, as the problem size increases, the performance of the ℓ_2 norm MD-SA algorithm deteriorates.

Moreover, the comparison between the two MD-SA algorithms suggests that the entropic ℓ_1 norm algorithm is more accurate than the ℓ_2 norm algorithm for larger problems. This observation confirms the theoretical comparison of error estimates between the two MD-SA algorithms in Appendix B. It should also be noted that, while the numbers of optimization steps are similar in both MD-SA algorithms, the wall clock time of the entropic ℓ_1 norm MD-SA algorithm on average is 8% less than that of the ℓ_2 norm one for all the problem sizes considered. The slight difference in wall clock time comes from the different ways to compute the projections ((30) vs (55)) in the two MD-SA algorithms.

6.2 Example 2: Two-dimensional bracing design with 102 load cases

In this example, we aim to compare the proposed entropic ℓ_1 norm MD-SA algorithm ($n = 1$) in (41) with the standard deterministic algorithm ($m = 102$) in (1) with OC

Journal of Applied Mechanics. Received November 13, 2019;
Accepted manuscript posted December 27, 2019. doi:10.1115/1.4045902

Table 1. Brief description of the numerical examples.

Example	Dimension	Description	Feature
1	2D	Disk design with 200 load cases	Comparison between deterministic, entropic ℓ_1 norm MD-SA and ℓ_2 norm MD-SA algorithms
2	2D	Bracing design with 102 load cases	Comparison among standard deterministic formulation with OC, entropic ℓ_1 norm MD-SA algorithm, and SAA-based randomized algorithm with OC
3	3D	Bridge design with 441 load cases	Combination of entropic ℓ_1 norm MD-SA with RMINRES iterative solver

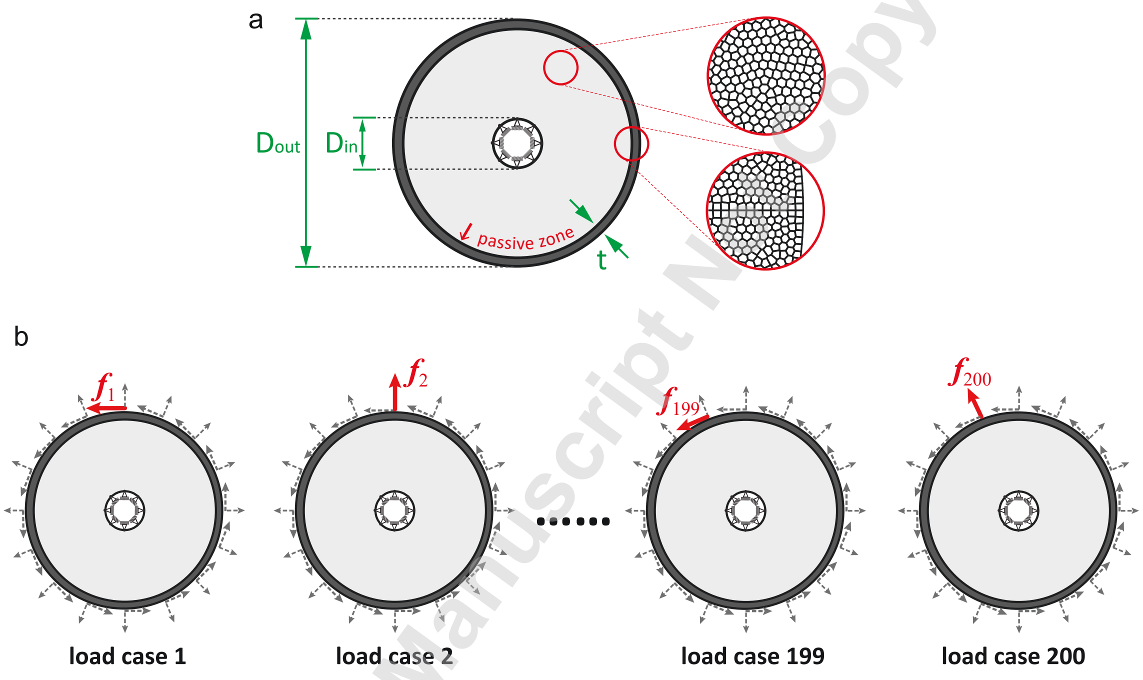

Fig. 2. Example 1. 2D disk with (a) the domain geometry and non-designable layer, $D_{\text{in}} = 0.2$ and $D_{\text{out}} = 1.0$; the domain is discretized by a mesh with continuum polygonal elements (mesh sizes: 10,000; 40,000; 70,000; and 100,000); (b) a total of 200 equal-weighted load cases are applied at the outer boundary of the domain (the dotted arrows are the schematic illustrations of non-active load cases).

update and with the randomized algorithm ($n = 6$) proposed by Zhang et al [18] (which uses a variant of the SAA technique with OC update). The design domain, boundary conditions, and the passive zone (non-designable solid region) are shown in Figure 6a. A total of 102 linearly independent and equally weighted load cases are applied on the two sides of the box (Figure 6b). The problem domain is modeled using 153,600 continuum quadrilateral (Q4) elements, which gives 309,442 degrees of freedom (DOFs). For this example, the active volume fraction (excluding the passive zone) is $V_f = 0.25$, the radius of the linear density filter is 3, and the penalization factor, p, is taken to be 3.0.

First, we perform topology optimization using the standard deterministic formulation (1) with OC update scheme. The final topology obtained is shown in Fig. 7(c), which has an objective function value of $C(\boldsymbol{x}^*) = 50.65$. The final topology is obtained with 400 optimization steps (maximum number of optimization steps), and in each optimization step we solve 102 linear systems (corresponding to 102 load cases), which leads to a total of $N_{solve} = 40,800$ solves.

Next, we use the entropic ℓ_1-norm MD-SA algorithm with the update proposed in (41), which uses a single sample to estimate the gradient (reselected every iteration), and the SAA-based randomized algorithm with the OC update algo-

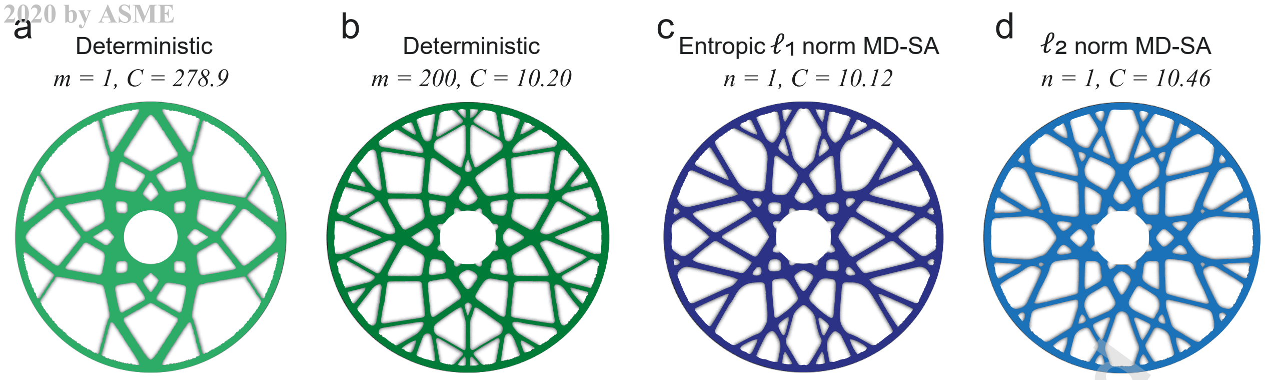

Fig. 3. Results for Example 1 with 40,000 design variables. The optimized topologies obtained by (a) the standard deterministic approach with 1 load case and $m = 1$ (i.e., all 200 loads are applied simultaneously); (b) the standard deterministic approach with 200 load cases and $m = 200$ (each of 200 loads is applied independently); (c) the proposed entropic ℓ_1 norm MD-SA with $n = 1$ (one representative trial); (d) the ℓ_2 norm MD-SA with $n = 1$ (one representative trial).

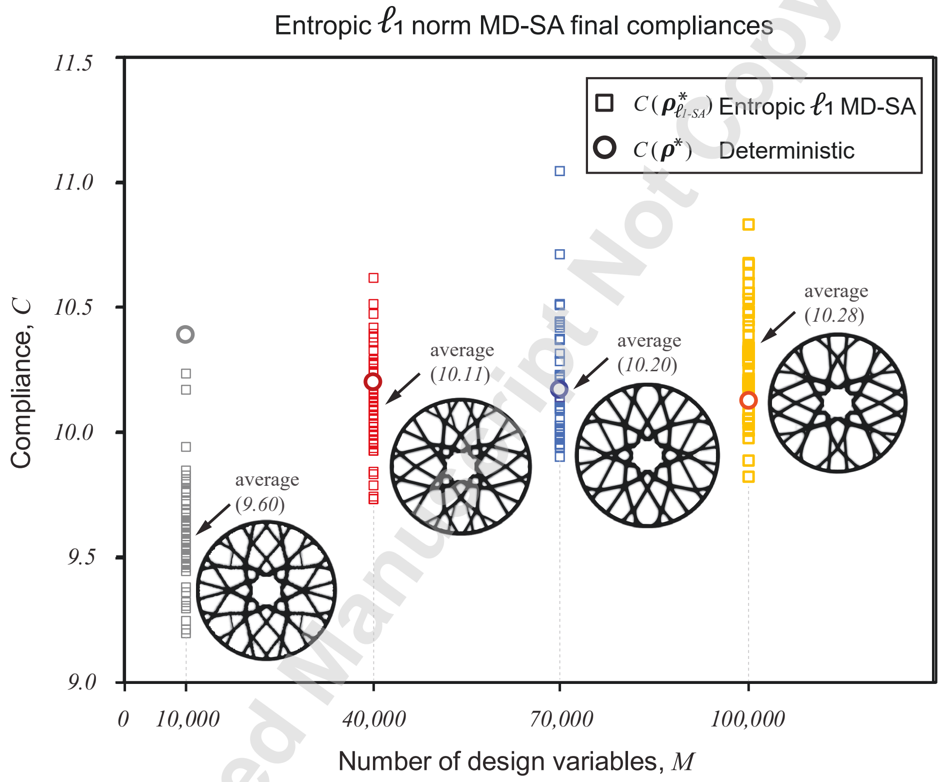

Fig. 4. Performance of the entropic ℓ_1 norm MD-SA algorithm ($n = 1$) for various problem sizes ($M = 10,000; 40,000; 70,000; 100,000$): Number of design variables versus the final (true) compliance. The final compliances for each problem size for the standard deterministic approach ($m = 200$) are included for comparison. The final topologies obtained by the entropic ℓ_1 norm MD-SA from representative trials are also included. The final compliances for the entropic ℓ_1 norm MD-SA algorithm are shown for 50 trials for each problem size.

rithm, which uses six samples to estimate the gradient (reselected every iteration). Because the final topology from the standard algorithm is y-symmetric, we enforce the symmetry of the topologies in both cases by enforcing symmetry of the density distribution with respect to the y-axis. Similar to the first example, we run each of them 50 times, the resulting statistics are shown in Table 3. “Mean” and “Dev” are the mean and the standard deviation, over 50 trials, of the objective function evaluated at the obtained solutions, respectively. The optimized topologies (from representative trials) obtained by the entropic ℓ_1 norm MD-SA and the SAA-based randomized algorithms are shown in Fig. 7(a) and 7(b), respectively.

For each method, the convergence history of the objective function for a representative trial is shown in Fig. 7(d). For the stochastic cases, the objective function estimators

Journal of Applied Mechanics. Received November 13, 2019;
Accepted manuscript posted December 27, 2019. doi:10.1115/1.4045902
Copyright © 2020 by ASME

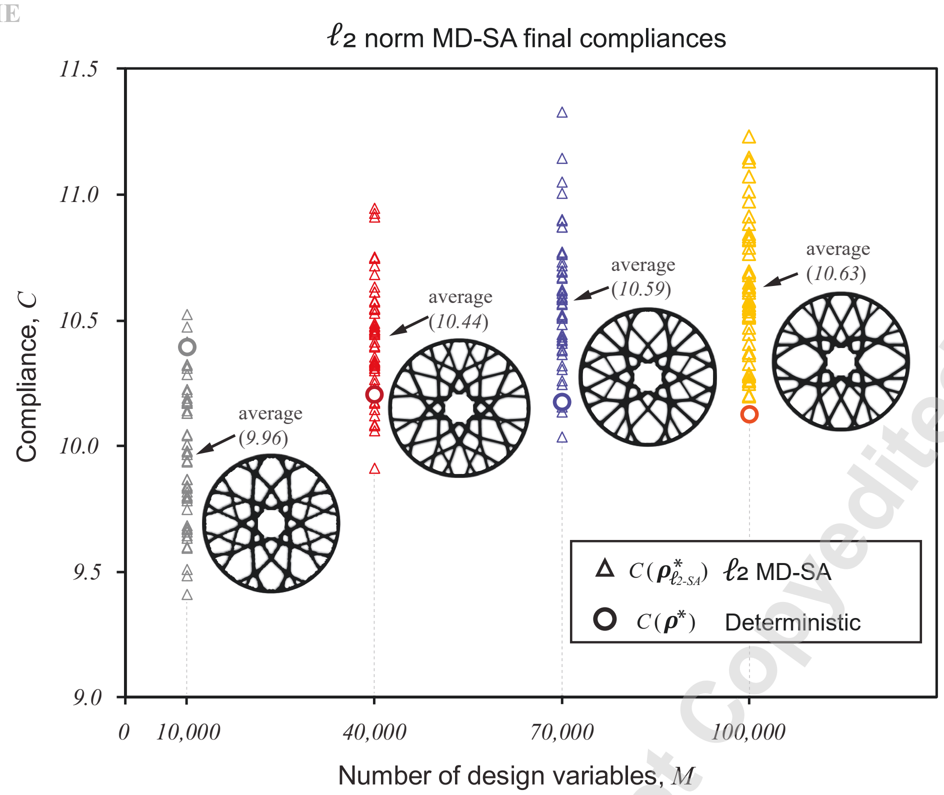

Fig. 5. Performance of the ℓ_2 norm MD-SA algorithm ($n = 1$) for four problem sizes ($M = 10,000; 40,000; 70,000; 100,000$): Number of design variables versus the final (true) compliance. The final compliances for each problem size for the standard deterministic approach ($m = 200$) are included for comparison. The final topologies obtained by the ℓ_2 norm MD-SA from representative trials are also included. The final (true) compliances for the ℓ_2 norm MD-SA algorithm are shown for 50 trials for each problem size.

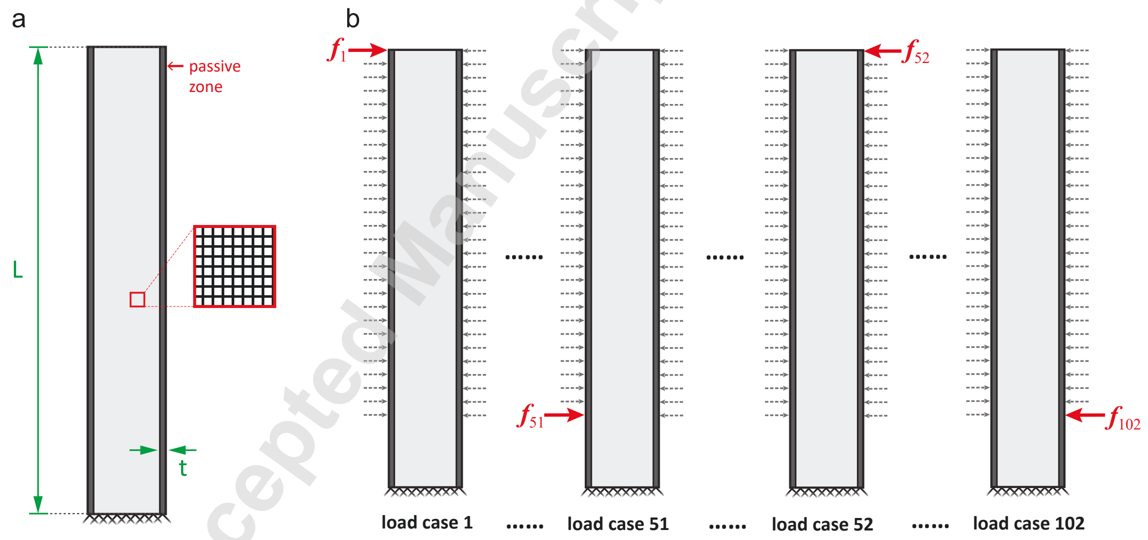

Fig. 6. Example 2. Two-dimensional bracing design domain with (a) the geometry and the non-designable layer, the domain is modeled by a mesh with 153,600 continuum quadrilateral elements and 309,442 DOFs; (b) a total of 102 linearly independent equal-weighted load cases are applied at the domain (dotted arrows are the schematic illustrations of non-active load cases).

($\widehat{C}^{SA}(\boldsymbol{x})$ and $\widehat{C}^{SAA}(\boldsymbol{x})$) are plotted during the optimization, while the true values of the objective function ($C(\boldsymbol{x}^*_{L2-SA})$ and $C(\boldsymbol{x}^*_{SAA})$) are only evaluated at the end of the optimization (indicated with markers). Notice that, because only one sample load case is used to estimate the compliance in the entropic ℓ_1 norm MD-SA algorithm rather than six sample load cases in the SAA-based one, the history of the estimator $\widehat{C}^{SA}(\boldsymbol{x})$ is more oscillatory than $\widehat{C}^{SAA}(\boldsymbol{x})$. However, in terms of the true objective function values obtained at the end of the optimization, the one from the entropic ℓ_1 norm MD-SA

Table 2. Results for Example 1, disk design with 200 load cases. Results are averaged over 50 trials.

Algorithm	M	C	$Mean(C)$	ΔC avg.	$Dev(C)$	N_{step} avg.	N_{solve} avg.	Time avg. (sec)	$\frac{\text{Time}}{N_{\text{step}}}$ (sec)
	10,000	10.39	-	-	-	228	45,600	1,401	6.2
Std.	40,000	10.22	-	-	-	400(max)	80,000	10,425	26.1
Deterministic	70,000	10.18	-	-	-	400(max)	80,000	18,520	46.3
$m = 200$	100,000	10.14	-	-	-	400(max)	80,000	26,888	67.2
Std.	10,000	10.39	-	-	-	228	45,600	1,401	6.2
Deterministic	40,000	10.17	-	-	-	821	164,200	21,397	26.1
(reach τ_{opt})	70,000	10.11	-	-	-	1,056	211,200	48,893	46.3
$m = 200$	100,000	10.07	-	-	-	1,449	289,800	97,402	67.2
Entropic	10,000	-	9.60	-7.60%	0.21	349	349	91	0.26
ℓ_1 norm	40,000	-	10.11	-1.06%	0.20	351	351	426	1.21
MD-SA	70,000	-	10.20	0.14%	0.22	357	357	815	2.29
$n = 1$	100,000	-	10.28	1.41%	0.22	358	358	1,194	3.40
	10,000	-	9.96	-4.10%	0.27	330	330	95	0.29
ℓ_2 norm	40,000	-	10.44	2.10%	0.23	344	344	471	1.37
MD-SA	70,000	-	10.59	3.96%	0.26	350	350	888	2.54
$n = 1$	100,000	-	10.63	4.90%	0.32	355	355	1,298	3.65

algorithm is smaller than the one from the SAA-based algorithm.

To quantify the accuracy and efficiency of the stochastic algorithms over multiple trials, Fig. 8 plots the final (true) compliance of the entropic ℓ_1 norm MD-SA algorithm (averaged over 50 trials), the SAA-based algorithm (averaged over over 50 trials), and the standard deterministic algorithm versus the corresponding total number of linear system solves. In terms of the accuracy, the entropic ℓ_1 norm MD-SA algorithm provides a similar topology to the one obtained from the standard deterministic formulation with OC update and yields a slightly smaller mean compliance value (-1.24% relative difference). The SAA-based approach with OC update also leads to a design similar to the one obtained from the standard deterministic formulation, but with a slightly larger mean compliance value ($+1.88\%$ relative difference). In terms of the efficiency, both stochastic algorithms use substantially fewer linear system solves and less wall clock time than the standard deterministic one. For the entropic ℓ_1 norm MD-SA algorithm, the number of linear systems to solve is on average (over 50 trials) 110 times fewer than for the standard algorithm (i.e., 372 solves vs. 40,800 solves), and the total wall clock time is 14.1 times shorter (i.e., 11.2 minutes vs. 2.6 hours). Moreover, the convergence for both the entropic ℓ_1 norm MD-SA algorithm and the standard one is comparable. For the SAA-based algorithm, the number of linear systems to solve is on average 27 times fewer than for the standard algorithm (1,530 solves vs. 40,800 solves), and the total wall clock time is 12.8 times shorter (12.3 minutes vs. 2.6 hours). Moreover, the convergence of the SAA-based algorithm is more rapid than that of the standard algorithm. Comparing the two stochastic algorithms, the entropic ℓ_1 norm MD-SA leads to a 3.06% lower (averaged) compliance and 9.2% less wall clock time than the SAA-based one on average, because the MD-SA algorithm solves fewer linear systems (372 solves vs. 1,530 solves). According to the standard deviation of both methods, the entropic ℓ_1 norm MD-SA gives more consistent solutions than the SAA-based algorithm. In summary, among all algorithms in this example, the entropic ℓ_1 norm MD-SA offers the lowest (averaged) compliance while being the most computationally efficient, it takes the least amount of time and the smallest number of solves.

6.3 Example 3: Three-dimensional bridge design with the RMINRES iterative solver

In the third example, we use a three-dimensional (3D) bridge design to demonstrate the performance of the pro-

Journal of Applied Mechanics. Received November 13, 2019;
Accepted manuscript posted December 27, 2019. doi:10.1115/1.4045902

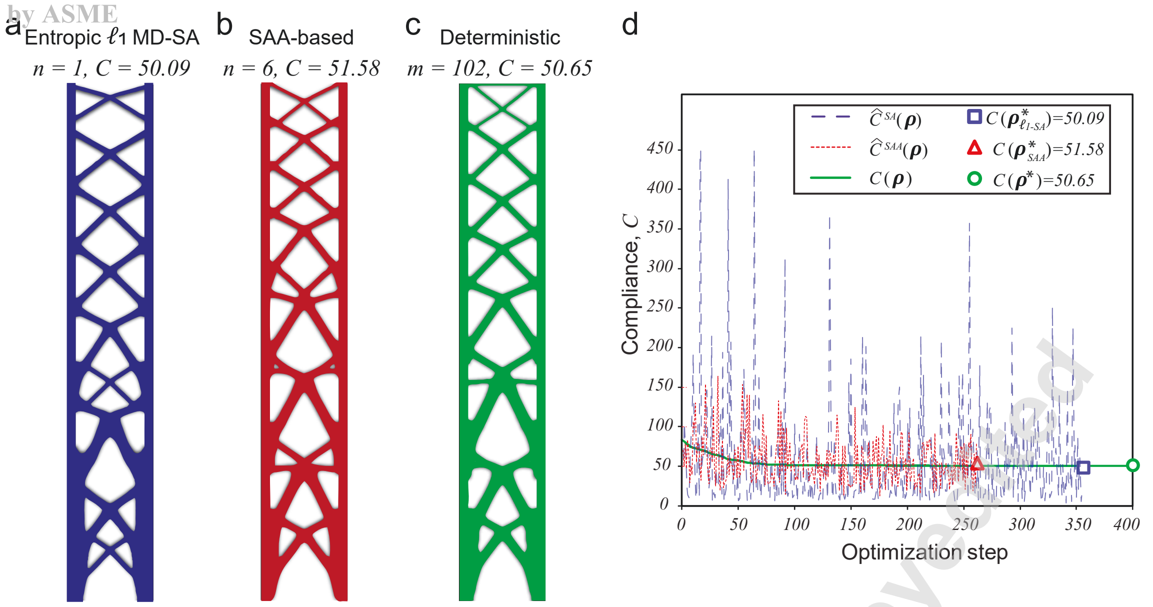

Fig. 7. Results for Example 2. (a) The optimized topology obtained by the proposed entropic ℓ_1 norm MD-SA algorithm with $n = 1$ (one representative trial); (b) the optimized topology obtained by the SAA-based randomized algorithm [18] (with OC update) with $n = 6$ (one representative trial); (c) the optimized topology obtained by the standard deterministic algorithm [16] (with OC update), where $m = 102$; (d) the history of compliance estimators ($\widehat{C}^{SA}(\boldsymbol{x})$ and $\widehat{C}^{SAA}(\boldsymbol{x})$) and deterministic compliance ($C(\boldsymbol{x})$) for corresponding cases (a)-(c), where also the true values of the objective function ($C(\boldsymbol{x}^*_{SA}), C(\boldsymbol{x}^*_{SAA})$, and $C(\boldsymbol{x}^*)$) at the optimized solutions ($\boldsymbol{x}^*_{SA}$, $\boldsymbol{x}^*_{SAA}$, and $\boldsymbol{x}^*$) in corresponding cases (a)-(c) are given.

Table 3. Results for Example 2, bracing design with 102 load cases. Results are averaged over 50 trials. "*Dev*" represents standard deviation.

Algorithm	$C(\boldsymbol{x}^*)$	$Mean(C)$	ΔC (avg.)	$Dev(C)$	N_{step} (avg.)	N_{solve} (avg.)	Time (avg.) (sec)	$\frac{\text{Time}}{N_{\text{step}}}$ (avg.) (sec)
Std. Deterministic $m = 102$	50.65	-	-	-	400 (max)	40,800	9,480	23.70
Entropic MD-SA $n = 1$	-	50.02	-1.20%	0.50	372	372	671	1.80
SAA-based [18] $n = 6$	-	51.60	1.92%	0.87	255	1,530	739	2.90

posed entropic ℓ_1 norm MD-SA algorithm for 3D problems. In this example, we also combine the entropic ℓ_1 MD-SA algorithm with a fast iterative solver for linear systems to further reduce the computational cost of topology optimization with many load cases. Specifically, we use the RMINRES iterative solver [42, 43], which recycles selected subspaces with an incomplete Cholesky preconditioner. For the RMINRES parameters, we choose the number of vectors to be recycled as 10, the number of Lanczos vectors kept in cycle as 100, and the max number of iterations as 1000.

The bridge design problem is set up as follows. The design domain, load cases, and boundary conditions are shown in Fig. 9(a). A total of 1764 equally-weighted load cases are applied to the bridge deck, which is taken to be a non-designable solid layer. Due to the symmetry of the design problem, we optimize a quarter of the domain with $m = 441$ load cases, as shown in Fig. 9(b). A total of 248,832 brick elements are used to discretize the quarter domain, resulting in 793,875 DOFs. For this design example, the volume fraction is chosen as $V_f = 0.08$ (excluding the passive zone), the radius of the density filter is 6, and the penalization factor is taken to be $p = 3$. For the entropic ℓ_1 norm MD-SA algo-

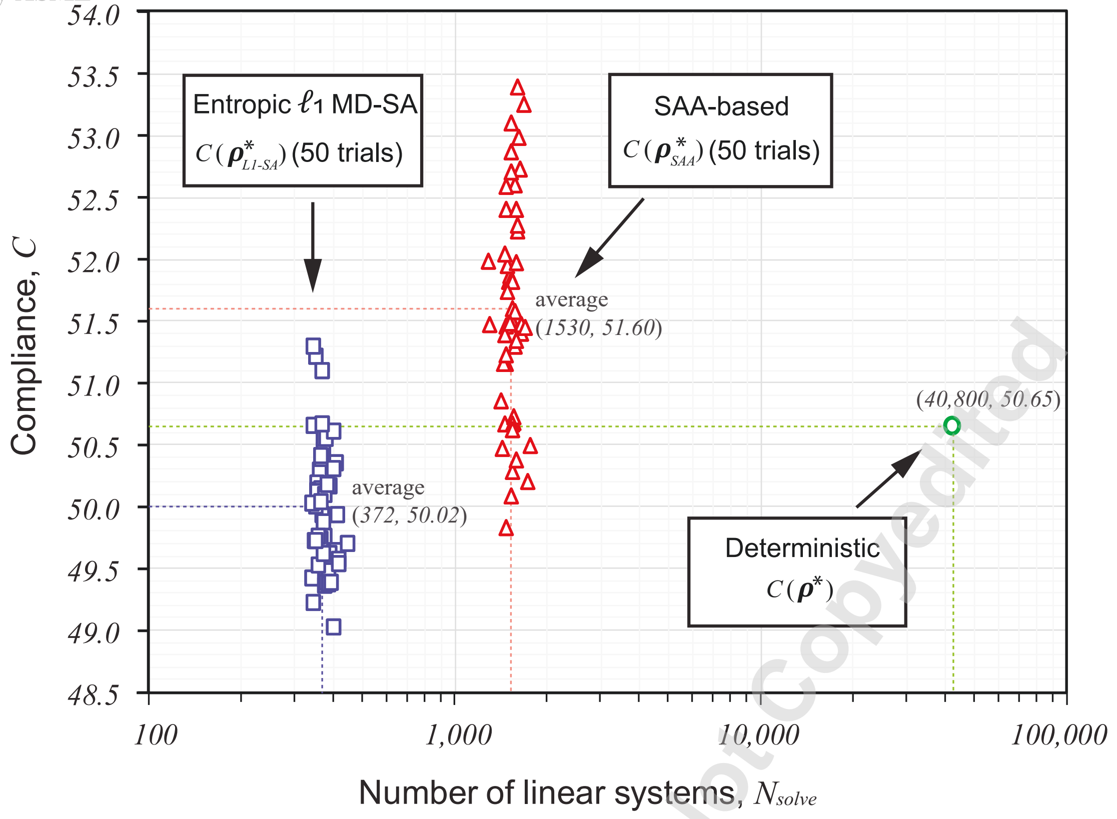

Fig. 8. Performance comparison of the proposed entropic ℓ_1 norm MD-SA (41), SAA-based [18] (with OC method), and the standard deterministic (1) (with OC method) algorithms: Number of linear systems to solve in the optimization process versus the resulting final compliance for each algorithm. (The entropic ℓ_1 norm MD-SA and SAA-based algorithms each include 50 trials.) In this example, the entropic ℓ_1 norm MD-SA offers the lowest (averaged) compliance while being the most computationally efficient, i.e., solving the least number of linear systems.

rithm, the following parameters are adopted: the sample size $n = 1$, the window size $N_D = 100$, and $\tau_{step} = 0.05$.

We perform topology optimization using the standard deterministic formulation (1). The final topology obtained is shown in Fig. 10(a), which has an objective function value of $C(\boldsymbol{x}^*) = 13.56$. This final topology is obtained with 400 optimization steps (maximum number of optimization steps) and, in each optimization step, Matlab's sparse direct solver is used to solve the linear systems with 441 right hand sides. Note that Matlab's sparse direct solver uses compiled code, while our iterative solver is run as interpreted code (so it runs much more slowly).

We then compare the design using the entropic ℓ_1 norm MD-SA algorithm with $n = 1$. Three cases are considered: one using a sparse direct solver, and the other two using the RMINRES iterative solver. For the latter two cases, we test two convergence tolerance values, $\tau_{iter} = 10^{-8}$ and $\tau_{iter} = 10^{-4}$. For each case, we perform the optimization 10 times, and present the results in Table 4. In addition, the optimized structures for the three cases (each obtained from one representative trial) are shown in Figures 10 (b)-(d).

First, we observe that all three cases using the entropic ℓ_1 norm MD-SA algorithm with $n = 1$ produce designs similar to the standard deterministic formulation in terms of the final topology and on average 4.2% higher true compliance values. However, the entropic ℓ_1 norm MD-SA algorithm is able to yield the final design with a greatly reduced wall clock time compared with the standard algorithm. For example, with the RMINRES iterative solver with a tolerance of $\tau_{iter} = 10^{-4}$, the total wall clock time is 87.1% lower than with the standard formulation (i.e., 5.6 hours vs. 43.6 hours).

Comparing the three cases using the entropic ℓ_1 norm MD-SA algorithm, with the RMINRES iterative solver we obtain almost identical designs and compliances as with the sparse direct solver. However, the algorithm with the RMINRES iterative solver is more efficient than the algorithm with the sparse direct solver. In terms of the wall clock time, the RMINRES runtimes are 72% and 60% faster than the sparse direct solver when the tolerance values $\tau_{iter} = 10^{-4}$ and $\tau_{iter} = 10^{-8}$ are used, respectively.

Finally, based on the performance comparison between the RMINRES iterative solver with two tolerance values, we highlight that, the single sample in entropic ℓ_1 norm MDSA leads to a relatively inaccurate gradient estimate, and hence there is no need to solve the linear system very accurately. This is an advantage of the entropic ℓ_1 norm MD-SA algorithm that we exploit with low accuracy iterative solves. The features of the proposed entropic ℓ_1 norm MD-SA algorithm and the RMINRES iterative solver with a large convergence tolerance allow us to achieve a further reduction in the computational time in addition to reducing the number of linear systems to solve.

7 Concluding remarks and perspective

In this paper, we propose a tailored stochastic approximation algorithm to solve mechanics-driven topology optimization problems with many load cases based on compliance minimization formulation at a drastically reduced computational cost. We first reformulate the deterministic topology optimization problem with many load cases into a stochastic one, whose objective function takes the form of the expectation of a random variable. We propose a tailored MD-SA algorithm (see Algorithm 1), which adopts an entropic distance generating function using the ℓ_1 norm to mimic the underlying geometry of the feasible design space (i.e., a design space with a linear volume constraint). Unlike commonly used optimization algorithms, the proposed entropic ℓ_1 norm MD-SA algorithm requires only moderate accuracy in the stochastic gradient, which enables the use of a single sample per optimization step (i.e., the sample size is always one) to estimate the stochastic gradient. With the proposed MD-SA algorithm, we reduce the number of linear systems to solve per iteration from hundreds to one. To the authors' knowledge, this is the first work in the literature that tailors SA algorithms to solve deterministic topology optimization problems.

To improve the performance and convergence of the proposed algorithm, we propose several algorithmic strategies for obtaining effective step sizes and updates, including the step size policy and re-calibration, iterates averaging, and a damping scheme inspired by simulated annealing. The main idea is to calculate and re-calibrate step sizes adaptively in different stages of the optimization. We also adopt an iterative solver with recycling for solving the large linear systems, this allows the MD-SA algorithm to use a significantly higher convergence tolerance in solving large state equations without influencing the performance of the design update to further reduce computational cost. With the proposed algorithmic strategies, the convergence rate of the proposed MD-SA algorithm is shown to be comparable to that of the standard algorithm.

Through numerical examples, we demonstrate the effectiveness, efficiency, and potential of the proposed entropic ℓ_1 norm MD-SA algorithm. Compared to the standard deterministic approach, the entropic ℓ_1 norm MD-SA algorithm allows to solve large-scale topology optimization problems with hundreds of load cases at a drastically reduced computational cost while obtaining similar design quality. For instance, in example 1, compared to the standard deterministic algorithm ($m = 200$), the number of linear system solves is 223 times smaller (i.e., 358 solves vs. 80,000 solves) and the average wall clock time is 22 times faster (i.e., 20 minutes vs. 7.5 hours) with the proposed entropic ℓ_1 norm MD-SA algorithm ($n = 1$). In addition, the proposed entropic ℓ_1 norm MD-SA ($n = 1$) is shown to outperform both the ℓ_2 norm MD-SA ($n = 1$) and the SAA-based algorithm [18] ($n = 6$) with OC update in terms of both efficiency and effectiveness. The first example investigates the performance of two MD-SA algorithms, namely the entropic ℓ_1 norm and the ℓ_2 norm, in a design problem with various problem sizes ($M = 10,000$, $M = 40,000$, $M = 70,000$, and $M = 100,000$). The entropic ℓ_1 norm algorithm is found to be more accurate than the ℓ_2 norm one for larger problems, which confirms the theoretical comparison of error estimates between the two MD-SA algorithms (see Appendix B). In addition, the wall clock time of the entropic ℓ_1 norm MD-SA algorithm on average is 9.21% less than that of the ℓ_2 norm algorithm for all the problem sizes investigated. The second example shows that the entropic ℓ_1 norm MD-SA leads to a 3.06% lower (averaged) compliance and 9.2% lower (averaged) wall clock time than the SAA-based algorithm with OC, because the MD-SA algorithm solves fewer linear systems (i.e., 1 solve per step vs. 6 solves per step). In the third example, the entropic ℓ_1 norm MD-SA algorithm is employed in conjunction with the RMINRES iterative solver for a 3D bridge design under 1,764 load cases (441 load cases for a quarter of the domain). We demonstrate that, as compared to the deterministic optimization algorithm, the proposed entropic ℓ_1 MD-SA algorithm can produce good-quality designs even when working with an iterative solver with a relaxed tolerance (i.e., 10^{-4} instead of the more common 10^{-8}), which is especially desirable for large-scale problems to further reduce the computational time.

Future research directions include studying the optimal choices of parameters for both overall computational cost and quality of design. Another important direction is to tailor the proposed algorithm to other objective functions and constraints as well as applying it to practical and complex design examples.

Acknowledgements

The authors would like to acknowledge Glaucio H. Paulino for his help, insightful discussions, and suggestions. The work by X. S. Zhang was supported in part by the startup fund from the University of Illinois. The work by E. de Sturler was supported in part by the grant NSF DMS 1720305. The work by A. Shapiro was supported in part by the grant NSF 1633196. The information provided in this paper is the sole opinion of the authors and does not necessarily reflect the view of the sponsoring agencies.

References

[1] Bendsoe, M. P., Guedes, J. M., Haber, R. B., Pedersen, P., and Taylor, J. E., 1994. "An Analytical Model to Predict Optimal Material Properties in the Context of Optimal Structural Design". Journal of Applied Mechanics, **61**(4), 12, pp. 930–937.

[2] Sigmund, O., and Torquato, S., 1997. "Design of materials with extreme thermal expansion using a three-phase topology optimization method". Journal of the

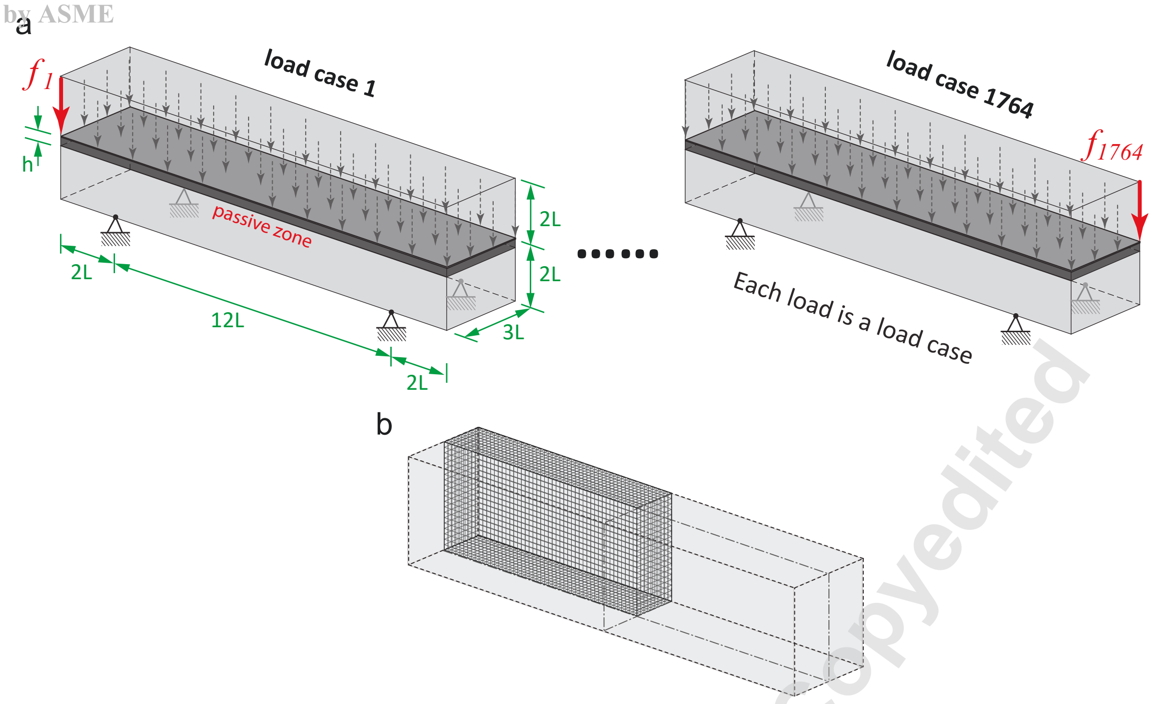

Fig. 9. Example 3. Three-dimensional bridge design with (a) the geometry and boundary conditions, $L = 12, h = 2$, with 1764 load cases (441 load cases for the quarter domain); (b) the quarter domain is modeled by a mesh with 248,832 brick elements ($144 \times 24 \times 72$) and 793,875 DOFs.

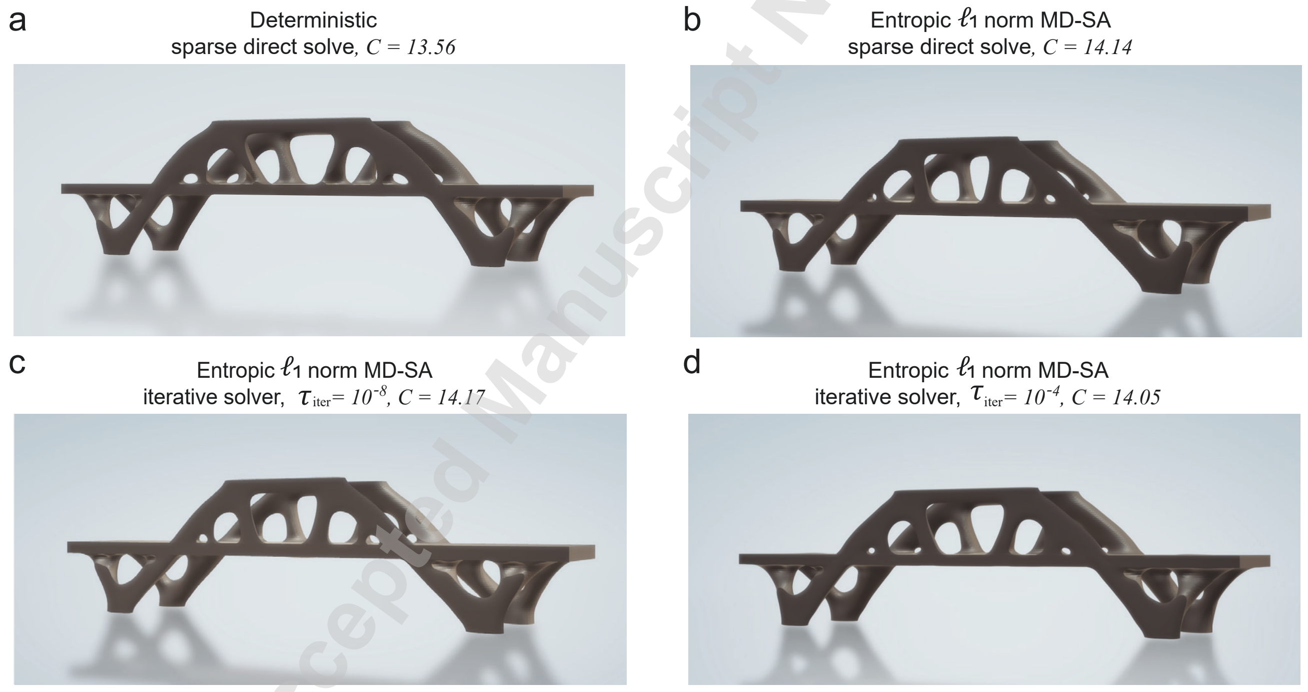
a
Deterministic
sparse direct solve, $C = 13.56$
b
Entropic ℓ_1 norm MD-SA
sparse direct solve, $C = 14.14$
c
Entropic ℓ_1 norm MD-SA
iterative solver, $\tau_{iter} = 10^{-8}$, $C = 14.17$
d
Entropic ℓ_1 norm MD-SA
iterative solver, $\tau_{iter} = 10^{-4}$, $C = 14.05$

Fig. 10. Rendering of optimized structures for the 3D bridge design obtained by (a) standard deterministic approach and by entropic ℓ_1 norm MD-SA using (b) a sparse direct solver; (c) the iterative solver with $\tau_{\text{iter}} = 10^{-8}$; (d) the iterative solver with $\tau_{\text{iter}} = 10^{-4}$.

Mechanics and Physics of Solids, **45**(6), pp. 1037–1067.
[3] Wang, F., Sigmund, O., and Jensen, J. S., 2014. "Design of materials with prescribed nonlinear properties". Journal of the Mechanics and Physics of Solids, **69**, pp. 156–174.
[4] Clausen, A., Wang, F., Jensen, J. S., Sigmund, O., and Lewis, J. A., 2015. "Topology optimized architectures with programmable poisson's ratio over large deformations". Advanced Materials, **27**(37), pp. 5523–5527.
[5] Ma, Z.-D., Kikuchi, N., Pierre, C., and Raju, B., 2005. "Multidomain Topology Optimization for Structural

Table 4. Results for Example 3, 3D density-based bridge design. Results are averaged over 10 trials. $\mathrm{Time}_{\mathrm{solve}}$ represents the wall clock time of solving the linear systems.

	τ_{iter}	C	$mean(C)$	ΔC	N_{step}	Time (sec)	$\mathrm{Time}_{\mathrm{solve}}$(sec)	$\frac{\mathrm{Time}}{N_{\mathrm{step}}}$
Deterministic & dir. sol.	-	13.56	-	-	400 (max)	156,988	142,694	392
MD-SA & dir. sol.	-	-	14.14	4.25%	293	73,269	71,156	250
MD-SA & iter. sol.	10^{-8}	-	14.17	4.49%	278	29,254	26,461	105
MD-SA & iter. solver	$\mathbf{10^{-4}}$	-	**14.08**	**3.80%**	**278**	**20,246**	**17,350**	**72**

and Material Designs". Journal of Applied Mechanics, **73**(4), 10, pp. 565–573.

[6] Aage, N., Andreassen, E., Lazarov, B. S., and Sigmund, O., 2017. "Giga-voxel computational morphogenesis for structural design". Nature, **550**, p. 84.

[7] Liu, C., Du, Z., Zhang, W., Zhu, Y., and Guo, X., 2017. "Additive Manufacturing-Oriented Design of Graded Lattice Structures Through Explicit Topology Optimization". Journal of Applied Mechanics, **84**(8), 06. 081008.

[8] Zhang, X. S., Paulino, G. H., and Ramos Jr., A. S., 2018. "Multi-material topology optimization with multiple volume constraints: A ground structure approach involving material nonlinearity". Structural and Multidisciplinary Optimization, **57**, pp. 161–182. Submitted.

[9] Sutradhar, A., Paulino, G., Miller, M., and Nguyen, T., 2010. "Topological optimization for designing patient-specific large craniofacial segmental bone replacements". Proceedings of the National Academy of Sciences, **107**(30), pp. 13222–13227.

[10] Nanthakumar, S., Lahmer, T., Zhuang, X., Park, H. S., and Rabczuk, T., 2016. "Topology optimization of piezoelectric nanostructures". Journal of the Mechanics and Physics of Solids, **94**, pp. 316–335.

[11] Yun, K.-S., and Youn, S.-K., 2018. "Microstructural topology optimization of viscoelastic materials of damped structures subjected to dynamic loads". International Journal of Solids and Structures, **147**, pp. 67–79.

[12] Chen, W., and Huang, X., 2019. "Topological design of 3d chiral metamaterials based on couple-stress homogenization". Journal of the Mechanics and Physics of Solids, **131**, pp. 372–386.

[13] 2019 USACM thematic conference topology optimization roundtable: Challenge problem competition. http://paulino.ce.gatech.edu/topopt%20 workshop%20website/challenge.php.

[14] Diaz, A. R., and Bendsøe, M. P., 1992. "Shape optimization of structures for multiple loading conditions using a homogenization method". Structural Optimization, **4**(1), pp. 17–22.

[15] Bendsøe, M. P., Ben-Tal, A., and Zowe, J., 1994. "Optimization methods for truss geometry and topology design". Structural optimization, **7**(3), pp. 141–159.

[16] Bendsøe, M. P., and Sigmund, O., 2003. Topology optimization: theory, methods, and applications. Springer, Berlin, Germany.

[17] Achtziger, W., 1998. "Multiple-load truss topology and sizing optimization: Some properties of minimax compliance". Journal of optimization theory and applications, **98**(2), pp. 255–280.

[18] Zhang, X. S., de Sturler, E., and Paulino, G. H., 2017. "Stochastic sampling for deterministic structural topology optimization with many load cases: Density-based and ground structure approaches". Computer Methods in Applied Mechanics and Engineering, **325**, pp. 463–487.

[19] Haber, E., Chung, M., and Herrmann, F., 2012. "An effective method for parameter estimation with PDE constraints with multiple right-hand sides". SIAM Journal on Optimization, **22**(3), pp. 739–757.

[20] Roosta-Khorasani, F., and Ascher, U., 2015. "Improved bounds on sample size for implicit matrix trace estimators". Foundations of Computational Mathematics, **15**(5), pp. 1187–1212.

[21] Neelamani, R., Krohn, C. E., Krebs, J. R., Romberg, J. K., Deffenbaugh, M., and Anderson, J. E., 2010. "Efficient seismic forward modeling using simultaneous random sources and sparsity". Geophysics, **75**(6), pp. WB15–WB27.

[22] Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A., 2009. "Robust stochastic approximation approach to stochastic programming". SIAM Journal on Optimization, **19**(4), pp. 1574–1609.

[23] Bottou, L., Curtis, F. E., and Nocedal, J., 2018. "Optimization methods for large-scale machine learning". Siam Review, **60**(2), pp. 223–311.

[24] Goodfellow, I., Bengio, Y., and Courville, A., 2016. Deep Learning. MIT Press.

[25] Nemirovski, A., and Yudin, D., 1978. "On Cezari's convergence of the steepest descent method for approximating saddle point of convex-concave functions". Doklady Akademii Nauk SSSR, **239**, pp. 1056–1059.

[26] Nemirovski, A., and Yudin, D., 1983. Problem Complexity and Method Efficiency in Optimization. John Wiley.

[27] Bourdin, B., 2001. "Filters in topology optimization". International Journal for Numerical Methods in Engineering, **50**(9), pp. 2143–2158.

[28] Bendsøe, M. P., 1989. "Optimal shape design as a material distribution problem". Structural optimization, **1**(4), pp. 193–202.

[29] Zhou, M., and Rozvany, G., 1991. "The COC algorithm, part II: Topological, geometrical and generalized shape optimization". Computer Methods in Applied Mechanics and Engineering, **89**(1-3), pp. 309–336.

[30] Petersson, J., 1999. "A finite element analysis of optimal variable thickness sheets". SIAM Journal on Numerical Analysis, **36**(6), pp. 1759–1778.

[31] Hutchinson, M., 1989. "A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines". Communication in Statistics-Simulation and Computation, **18**(3), pp. 1059–1076.

[32] Avron, H., and Toledo, S., 2011. "Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix". Journal of the ACM, **58**(8.).

[33] Dilworth, S. J., and Montgomery-Smith, S. J., 1993. The distribution of vector-valued Rademacher series. JSTOR.

[34] Shalev-Shwartz, S., 2011. "Online learning and online convex optimization". Foundations and Trends in Machine Learning, **4**, pp. 107–194.

[35] Shapiro, A., Dentcheva, D., and Ruszczyński, A., 2009. Lectures on stochastic programming: modeling and theory. SIAM, Philadelphia.

[36] Kirkpatrick, S., Gelatt Jr., C. D., and Vecchi, M. P., 1983. "Optimization by simulated annealing". Science, **220**(4598), pp. 671–680.

[37] Salamon, P., Sibani, P., and Frost, R., 2002. Facts, conjectures, and improvements for simulated annealing. SIAM, Philadelphia.

[38] Robbins, H., and Monro, S., 1951. "A stochastic approximation method". The Annals of Mathematical Statistics, **22**(3), pp. 400–407.

[39] Polyak, B. T., 1990. "New stochastic approximation type procedure". Automat. i Telemekh., **7**, pp. 98–107.

[40] Polyak, B. T., and Juditsky, A. B., 1992. "Acceleration of stochastic approximation by averaging". SIAM J. Control Optim., **30**, pp. 838–855.

[41] Kilmer, M., and de Sturler, E., 2006. "Recycling subspace information for diffuse optical tomography". SIAM J. Sci. Comput., **27**(6), pp. 2140–2166.

[42] Wang, S., de Sturler, E., and Paulino, G. H., 2007. "Large-scale topology optimization using preconditioned Krylov subspace methods with recycling". International Journal for Numerical Methods in Engineering, **69**(12), pp. 2441–2468.

[43] Mello, L. A. M., de Sturler, E., Paulino, G. H., and Silva, E. C. N., 2010. "Recycling Krylov subspaces for efficient large-scale electrical impedance tomography". Comput. Methods Appl. Mech. Engrg., **199**, pp. 3101–3110.

[44] Paige, C. C., and Saunders, M. A., 1975. "Solutions of sparse indefinite systems of linear equations". SIAM Journal on Numerical Analysis, **12**(4), pp. 617–629.

[45] Saad, Y., 2003. Iterative Methods for Sparse Linear Systems, Second ed. SIAM, Philadelphia.

[46] Eiermann, M., Ernst, O. G., and Schneider, O., 2000. "Analysis of acceleration strategies for restarted minimal residual methods". Journal of Computational and Applied Mathematics, **123**(1-2), pp. 261–292.

[47] Hestenes, M. R., and Stiefel, E., 1952. "Methods of conjugate gradients for solving linear systems". Journal of Research of the National Bureau of Standards, **49**, pp. 409–436 (1953).

[48] Meijerink, J., and Van der Vorst, H., 1977. "An iterative solution method for linear equations systems of which the coefficient matrix is a symmetric M-matrix". Mathematics of Computation, **31**, pp. 148–162.

[49] Gaul, A., and Schlömer, N., 2015. "Preconditioned recycling Krylov subspace methods for self-adjoint problems". Electron. Trans. Numer. Anal., **44**, pp. 522–547.

[50] Talischi, C., Paulino, G. H., Pereira, A., and Menezes, I. F. M., 2012. "PolyTop: a Matlab implementation of a general topology optimization framework using unstructured polygonal finite element meshes". Structural and Multidisciplinary Optimization, **45**(3), pp. 329–357.

Nomenclature

α_i	Weighting factor for the ith load case
$\boldsymbol{H}$	Filter matrix for density
$\boldsymbol{x}^*_{SAA}$	Optimal solution obtained by the SAA variant algorithm [18] in the density-based method
$\boldsymbol{x}^*$	Optimal solution obtained by the standard algorithm in the density-based method
$\hat{\boldsymbol{x}}^*_{\ell_2-SA}$	Optimal solution obtained by the entropic ℓ_2 norm MD-SA algorithm in the density-based method with weighted averaged iterates
$\hat{\boldsymbol{x}}^*_{\ell_1-SA}$	Optimal solution obtained by the ℓ_1 norm MD-SA algorithm in the density-based method with weighted averaged iterates
$\bar{\boldsymbol{x}}$	Filtered density vector
$\tilde{\boldsymbol{x}}$	Scaled density vector
$\boldsymbol{\xi}$	Random vector with its entries drawn from the Rademacher distribution
$\boldsymbol{f}_i$	External force vector for the ith load case
$\boldsymbol{G}(\boldsymbol{x}_k, \boldsymbol{\xi}_k)$	Stochastic gradient at optimization step k
$\widetilde{G}(\tilde{\boldsymbol{x}}_k, \boldsymbol{\xi}_k)$	Stochastic gradient with respect to $\tilde{\boldsymbol{x}}$ at optimization step k
$\boldsymbol{u}_i$	Displacement vector for the ith load case

γ_k	Step size at optimization step k
κ	Scale factor in the damping scheme
$\omega(\boldsymbol{x})$	Distance generation function
$V(\cdot,\cdot)$	Prox function generated by $\omega(\boldsymbol{x})$
$\widetilde{X}_{\boldsymbol{x}}$	Feasible set of the scaled design variable $\tilde{\boldsymbol{x}}$ in the density-based appraoch
$R_{\min}$	Radius of the density filter
N_D	Sample window size in the damping scheme
N_A	Window size to obtain the weighted averaged density $\hat{\boldsymbol{x}}$
τ_D	Tolerance for the damping scheme
τ_{opt}	Tolerance for the optimization process
$\boldsymbol{F}$	Weighted external force matrix
$\boldsymbol{K}$	Global stiffness matrix
$\boldsymbol{U}$	Weighted displacement matrix
$\tilde{\boldsymbol{a}}$	Scaled cross-sectional area
$\widehat{C}^{SA}$	Expected objective function by the SA algorithm
$\widehat{C}^{SAA}$	Expected objective function by the SAA variant algorithm [18]
C	Weighted compliance (objective function)
calibration	Number of times the step size re-calibration process is performed
d	Number of dimensions
E_0	Young's modulus of the solid material
$E_{\min}$	Young's modulus of the Ersatz material
M	Number of elements in the mesh
m	Number of load cases
move	Move limit in the design update
N	Number of nodes in the mesh
n	Sample size
N_{step}	Total number of optimization steps to obtained the final topology
$N_{\max}$	Maximum number of steps in the optimization
N_{solve}	Total number of linear systems solves in the optimization process
p	Penalization parameter in the density-based method
$P_{\boldsymbol{x}}(\cdot)$	Prox mapping defined by $V(\cdot,\cdot)$
R_k	Effective step ratio in the damping scheme evaluated at optimization step k
$x^{(e)}$	Density of element e (the e-th design variable in the density-based method)
$v^{(e)}$	Volume of element e
$V_{\max}$	Prescribed maximum volume of the design
V_f	Prescribed allowable volume fraction in the density-based method
$X_{\boldsymbol{x}}$	Feasible set of the design variable $\boldsymbol{x}$ in the density-based approach

Appendix A: Euclidean MD-SA with ℓ_2 norm

In this appendix, we present the MD-SA algorithm with the ℓ_2 norm (also known as the Euclidean SA). In this variant, we use the Euclidean (ℓ_2) norm, and define the distance generating function as follows, $\omega(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{x}$. In that case the prox-function $V(\cdot,\cdot)$ becomes

$$V(\boldsymbol{x},\boldsymbol{z}) = \frac{1}{2}\boldsymbol{z}^T\boldsymbol{z} - \left[\frac{1}{2}\boldsymbol{x}^T\boldsymbol{x} + \boldsymbol{x}^T(\boldsymbol{z}-\boldsymbol{x})\right] = \frac{1}{2}\|\boldsymbol{z}-\boldsymbol{x}\|_2^2, \quad (53)$$

and the prox-mapping $P_{\boldsymbol{x}}(\boldsymbol{x}) = \Pi_X(\boldsymbol{x}-\boldsymbol{y})$; see (12). As a result, we obtain the following update formula (18) for MD-SA algorithm with the ℓ_2 norm,

$$\boldsymbol{x}_{k+1} = \Pi_X\left(\boldsymbol{x}_k - \gamma_k G(\boldsymbol{x}_k,\boldsymbol{\xi}_k)\right). \quad (54)$$

That is, using the Euclidean norm, the MD-SA is equivalent to the robust SA algorithm.

When applied to randomized topology optimization formulations, the feasible sets are given by

$$X_{\boldsymbol{x}} = \left\{\boldsymbol{x}\in\mathbb{R}^M : \sum_{e=1}^{M}\frac{v^{(e)}x^{(e)}}{|\Omega|} - V_f = 0, \quad 0 \le x^{(e)} \le 1, \quad e = 1,\ldots,M\right\}. \quad (55)$$

The ℓ_2 norm MD-SA framework gives the update formula

$$\boldsymbol{x}_{k+1} = \Pi_{X_{\boldsymbol{x}}}\left(\boldsymbol{x}_k - \gamma_k G(\boldsymbol{x}_k,\boldsymbol{\xi}_k)\right). \quad (56)$$

We note that in the above update formula, a subproblem needs to be solved to find the corresponding Euclidean projection.

The ℓ_2 norm MD-SA adopts the same algorithmic framework as the entropic ℓ_1 norm MD-SA described in Section 5, except for a different formula for γ_k. A general formula for ℓ_2 norm MD-SA is suggested in [22] which takes the form

$$\gamma_k = \frac{\theta D_{\omega,X}}{B\sqrt{N_{\max}}}, \quad k = 1,\ldots,N_{\max}, \quad (57)$$

where

$$B^2 \ge \mathbb{E}\left[\|\boldsymbol{G}(\boldsymbol{x},\boldsymbol{\xi})\|_2^2\right], \quad (58)$$

is a bound on the ℓ_2 norm squared of the stochastic gradient. Similar to the entropic ℓ_1 norm MD-SA case, the above formula is used in this paper to determine the step size for the ℓ_2 norm MD-SA. In our case, $D_{\omega,X} = \sqrt{M}$ and the scaling parameter θ is taken to be 50. Because the bound B for the stochastic gradient is unknown, we estimate B using a few samples as $B = 1/n\|\sum_{i=1}^{n}\boldsymbol{G}(\boldsymbol{x}_0,\boldsymbol{\xi}_i)\|_2$, which is the sample averaged ℓ_2 norm of the estimated gradient at the initial step. This gives the following formula for the constant step size γ_k for the ℓ_2 norm MD-SA,

$$\gamma_k = \frac{50\sqrt{M}}{B\sqrt{N_{\max}}}, \quad k = 1,\ldots,N_{\max}. \quad (59)$$

As a final remark, unlike in the entropic ℓ_1 norm MD-SA, we find that the performance of ℓ_2 norm MD-SA is much more sensitive to the step size, especially to the scaling parameter θ.

In Algorithm 2, we summarize the ℓ_2 norm MD-SA algorithm for randomized topology optimization considered in this work.

Algorithm 2 Randomized topology optimization using ℓ_2 norm MD-SA

```
1: Initialize: $x_0$, $N_{\max}$, $\tau_{opt}$, $\tau_D$, $N_D$, $move$, $calibration$
2: for $k = 0, 1, \ldots, N_{\max}$ do
3:     if $k = 0$ then
4:         Evaluate $\gamma_k$ based on (59)
5:     end if
6:     Select a single sample $\boldsymbol{\xi}_k$ and evaluate $\mathbf{G}(\boldsymbol{x}_k, \boldsymbol{\xi}_k)$
7:     Compute $\boldsymbol{x}_{k+1}$ based on (56)
8:     Calculate $\hat{\boldsymbol{x}}_{k+1}$ by averaging using (47)
9:     Evaluate the effective step ratio $R_k$
10:    if $R_k < \tau_D$ and $k > N_D$ then
11:        $move = move/\kappa$
12:    end if
13:    if $\|\hat{\boldsymbol{x}}_{k+1} - \hat{\boldsymbol{x}}_k\|_\infty < \tau_{opt}$ then
14:        quit
15:    end if
16: end for
17: if $calibration > 0$ then
18:     $calibration = calibration - 1$
19:     $\boldsymbol{x}_0 \leftarrow \hat{\boldsymbol{x}}_{k+1}$ and goto step 2
20: else
21:     Evaluate final compliance $C(\hat{\boldsymbol{x}}_{k+1})$ and plot final topology
22: end if
```

Appendix B: Comparison of entropic ℓ_1 norm MD-SA and ℓ_2 norm MD-SA

Here, we provide an error estimate comparison of the entropic ℓ_1 norm MD-SA and the ℓ_2 norm MD-SA algorithms for a generic convex function f [22]. The error estimate for the ℓ_2 norm MD-SA is as follows,

$$\mathbb{E}\left[f(\bar{\boldsymbol{x}}_{N\text{step}}) - f(\boldsymbol{x}_*)\right] \leq O(1)\max\left\{\theta, \theta^{-1}\right\} B N_{\text{step}}^{-\frac{1}{2}}, \quad (60)$$

where $\bar{\boldsymbol{x}}_{N_{\text{step}}}$ is the weighted averaged design variable vector after N_{step} steps using the corresponding MD-SA algorithm, $\boldsymbol{x}_*$ is the optimal design variable vector, $O(1)$ is a generic constant, θ is the chosen parameter to determine the step size γ (see previous page), and B is a bound on the ℓ_2 norm of the stochastic gradient,

$$B^2 \geq \mathbb{E}\left[\|G(\boldsymbol{x}, \boldsymbol{\xi})\|_2^2\right]. \quad (61)$$

The error estimate for the entropic ℓ_1 norm MD-SA is as follows,

$$\mathbb{E}\left[f(\bar{\boldsymbol{x}}_{N_{\text{step}}}) - f(\boldsymbol{x}_*)\right] \leq O(1)\max\left\{\theta, \theta^{-1}\right\} \sqrt{\ln(M)} B_* N_{\text{step}}^{-\frac{1}{2}}, \quad (62)$$

where M is the number of design variables, and B_* is a bound on the ℓ_∞ norm of the stochastic gradient,

$$B_*^2 \geq \mathbb{E}\left[\|G(\boldsymbol{x}, \boldsymbol{\xi})\|_\infty^2\right]. \quad (63)$$

Notice that the following relation holds for every optimization step,

$$\sqrt{\frac{1}{\ln M}} \leq \frac{B}{B_*\sqrt{\ln M}} \leq \sqrt{\frac{M}{\ln M}}, \quad (64)$$

and

$$1 \leq \frac{B}{B_*} \leq \sqrt{M}. \quad (65)$$

According to equations 60, 62 and 65, the entropic ℓ_1 norm MD-SA can be more accurate (in terms of estimated error) than the ℓ_2 norm MD-SA for large M. Therefore, MD-SA with ℓ_1 is preferred for problems with larger dimensions. In Section 6.3, the numerical comparison between entropic $\ell 1$ and ℓ_2 norm MD-SA algorithms shows that the entropic ℓ_1 norm MD-SA is more accurate than the ℓ_2 norm MD-SA for large problem sizes. This observation agrees with the theoretical comparison between these two algorithms regarding the error estimates. An important benefit of the ℓ_1 norm MD-SA algorithm over the ℓ_2 norm MD-SA (Euclidean SA) is the possibility to reduce the constant factor by adjusting the norm and the distance-generating function to the geometry of the problem [22].