Structural and Multidisciplinary Optimization. Accepted: June 10, DOI: 10.1007/s00158-010-0534-6

A 199-line Matlab Code for Pareto-Optimal Tracing in Topology Optimization

Krishnan Suresh

University of Wisconsin, Madison

Abstract

The paper 'A 99-line topology optimization code written in Matlab' by Sigmund (Structural and Multidisciplinary Optimization, 2001, 21) demonstrated that SIMP-based topology optimization can be easily implemented in less than hundred lines of Matlab code. The published method and code has been used even since by numerous researchers to advance the field of topology optimization.

Inspired by the above paper, we demonstrate here that, by exploiting the notion of *topological-sensitivity* (an alternate to SIMP), one can generate *pareto-optimal topologies* in about twice the number of lines of Matlab code. In other words, optimal topologies for various volume fractions can be generated in a highly efficient manner, by directly tracing the pareto-optimal curve.

1. INTRODUCTION[1]

Topology optimization is now a well established field. Indeed, numerous topology optimization methods such as homogenization [1], SIMP [2-4] and level-set [5-7], now exist. If a well-defined objective can be articulated, such methods can systematically generate insightful designs for complex engineering problems.

In this paper, we discuss a new and robust topology optimization method for multi-objective problems, based on the concept of *topological sensitivity* [8-16]. A salient feature of the proposed method is that, for compliance problems, one can trace the *pareto-optimal frontier* (see Figure 1) in a computationally efficient manner. In other words, the method can find pareto-optimal topologies [17] for various volume fractions with far fewer finite element analysis than classic SIMP methods.

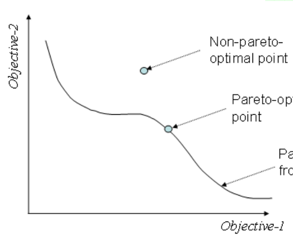


Figure 1: Pareto-optimal points, and pareto-frontier.

The remainder of the paper is organized as follows. In Section 2, multi-objective topology optimization is briefly reviewed. In Section 3, we introduce the notion of local pareto-optimality, review topological sensitivity, and finally establish fundamental results on pareto-optimal designs, and an associated algorithm. Then, in Section 4, the Matlab code (see Appendix) for generating pareto-optimal designs is explained. In Section 5, numerical results are presented, followed by conclusions and open issues in Section 6.

2. MULTI-OBJECTIVE TOPOLOGY OPTIMIZATION

There is significant amount of literature on topology optimization (for example, see review papers [18-21]), and multi-objective optimization [17, 22-25]. We focus here on the intersection of the two disciplines, specifically on topology optimization problems of the type:

$$\underset{\Omega\subset D}{Min}\left\{J,|\Omega|\right\} \tag{2.1}$$

where (see Figure 2, for example):

$$\begin{aligned} &J: \text{ Compliance}\\ &\Omega: \text{ Geometry/topology to be computed}\\ &D: \text{ Region within which the geometry must lie}\end{aligned} \tag{2.2}$$

It is implicitly assumed that, for any structure $\Omega\subseteq D$, the displacement must satisfy the elasticity equation [26].

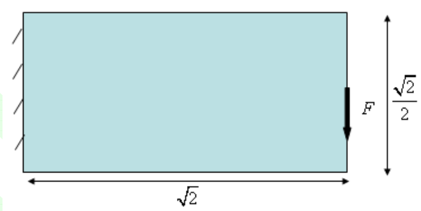


Figure 2: A structural problem on the domain D.

The objective is to find *pareto-optimal topologies* [17] for Equation (2.1). In the present context, a topology Ω is 'pareto-optimal' if no other topology Ω' exists with smaller compliance and identical volume.

2.1 Review of Methods

One approach to solving Equation (2.1) is to transform it into a series of single-objective optimization problems:

$$\begin{aligned}&\underset{\Omega\subset D}{Min}\, J\\ &|\Omega| = v_0\end{aligned} \tag{2.3}$$

where the volume is fixed at some desired value, and Equation (2.3) is solved to yield an optimal topology. Then, the volume desired is modified, and a fresh optimization problem is solved. This strategy is easy to implement, but can be computationally prohibitive since each 'run' of topology optimization entails numerous finite element analysis (FEA), and is therefore expensive.

An alternate approach to solving Equation (2.1) is through weighted optimization where the problem is transformed as follows:

$$\underset{\Omega\subset D}{Min}\, w_1 J + w_2|\Omega| \tag{2.4}$$

where the weights are prescribed *a priori*. By appropriately choosing the weights, a set of pareto-optimal designs may be obtained. This method has two inherent limitations: (1) not all pareto-optimal designs can be obtained via suitable weighting, and (2) finding suitable weights is non-trivial [23, 24, 27].

A variation is the 'compromise formulation' wherein the weights are supplemented by min and max for each objective function [28]. Yet another variation is the physical programming method where other aspects of the objectives, for example: the range within which it must lie, etc, are

[1] A preliminary version of this work will be presented at the 2010 ASME IDETC/CIE conference in Montreal, Canada.

Structural and Multidisciplinary Optimization. Accepted: June 10, DOI: 10.1007/s00158-010-0534-6

taken into account in the formulation [29]. Further, for specific applications, example: compliant mechanisms, other weighted-optimization methods have also been successfully implemented [30].

However, the most successful methods today for multi-objective optimization are based on the non-dominated evolutionary or genetic algorithms [22, 31-33].

The underlying principle behind this class of methods (there are numerous variations) is as follows. First, one generates a population of designs. Then, through stochastic methods, typically via genetic coding and mutation, some of the designs are modified. Non-pareto-designs are then eliminated, and the cycle is repeated. Specific examples of such methods for multi-objective topology optimization are reviewed below.

In [31], the authors deal with multi-objective topology optimum design, where the two objectives are minimization of volume and maximum displacement under given loading. Multi-objective evolutionary algorithms are used with Voronoi diagrams serving as a geometric representation.

In [34], the two objectives considered in topology optimization are the minimization of compliance, and maximization of the first eigen-value, with the amount of material to be used serving as a constraint. An additional parameter of penalty-timing is also considered.

While many of the methods are capable of generating pareto-optimal topologies, a common drawback is that they require numerous FEA-runs, and are therefore computationally prohibitive.

3. PROPOSED METHOD

The objective here is to develop a simple and efficient method to *directly trace the pareto-frontier* for the two-objective topology optimization problem in Equation (2.1).

The proposed method rests on: (1) the notion of *local pareto-optimality* discussed in Section 3.1, (2) *topological sensitivity*, reviewed in Section 3.2, and (3) *pareto-optimality criteria* stated and proved in Section 3.3. The method has been inspired by the pioneering work reported in [35, 36].

3.1 Local Pareto-Optimality

Since our objective is to trace the pareto-optimal curve, i.e., to move from pareto-optimal point to the next, we define in this Section the notion of "local pareto-optimality", by first defining the distance between 2 topologies.

Definition 1: Topologies Ω and Ω' are utmost δ-apart if their symmetric volume difference is less than or equal to δ:

$$\Delta V(\Omega,\Omega') = |(\Omega-\Omega')| + |(\Omega'-\Omega)| \le \delta \tag{3.1}$$

♦

For example, consider the topology in Figure 3.

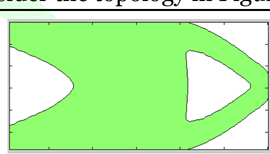


Figure 3: A given topology Ω .

Figure 4 illustrates topologies that are δ-apart from Figure 3, where δ denotes the volume of a small disc, shown in Figure 4. Thus, nearby-topologies can be constructed by: (a) subtracting or adding a single volume of δ, (b) subtracting or adding 2 volumes of $\delta/2$, (c) subtracting and adding a volume of $\delta/2$, (d) subtracting/adding multiple volumes of δ/N, such that Equation (3.1) is satisfied.

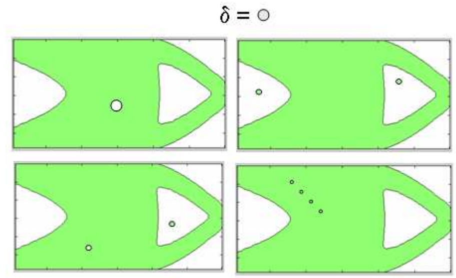


Figure 4: Topologies Ω' that are δ-apart from Figure 3c.

With this notion of nearby-topologies, we now define 'local' pareto-optimality as follows.

Definition 2: A topology Ω is said to be *locally pareto-optimal* if it is pareto-optimal with respect to all topologies that are within a distance δ apart from it, where δ is sufficiently small. ♦

This definition plays a crucial role in determining if a particular topology is pareto-optimal. In the next section, we provide necessary and sufficient conditions for a topology to be locally pareto-optimal. But, first we review the second concept of *topological sensitivity* (a.k.a. topological derivative).

3.2. Topological Sensitivity: A Review

The notion of topological derivative has its roots in the seminal paper by Eschenauer, et. al. [15]; this concept has been later explored by numerous authors, for example in [8-15, 35]. The classic topological sensitivity deals with the sensitivity of field problems to subtraction of infinitesimal but arbitrary shaped features [13]. However, in this paper, we shall restrict ourselves to subtraction of discs in 2-D, spheres in 3-D. Further, we assume that the discs/spheres lie in the interior of the domain (see [37, 38] for a treatment of boundary insertion/perturbation). For such shapes, topological sensitivity captures the first order impact of inserting a small circular hole within a domain on various quantities of interest [8-14, 39].

For the compliance J , the topological sensitivity field, for a 2-D plane-stress problem, is given by [40]:

$$\mathcal{T}^S(p) = \frac{4}{1+\nu}\sigma:\varepsilon - \frac{1-3\nu}{1-\nu^2}tr(\sigma)tr(\varepsilon) \tag{3.2}$$

It states that, once the stress and strain are determined for a given structural problem, the change in compliance due to insertion of a hole of area δ at any point p is given by:

$$\Delta J = \mathcal{T}^S(p)\delta + o(\delta) \tag{3.3}$$

where, by definition:

$$\lim_{\delta\to 0}\frac{o(\delta)}{\delta} \to 0 \tag{3.4}$$

In parallel, one can also consider the case of material addition and its impact on the compliance. In particular, let the region $D-\Omega$ be modeled using a soft-material $E_\varepsilon \ll 1$.

Structural and Multidisciplinary Optimization. Accepted: June 10, DOI: 10.1007/s00158-010-0534-6

Then, one can show that the topological sensitivity, for a 2-D plane-stress problem, with respect to addition is given by [41]:

$$\mathcal{T}^A(q) = -\frac{4}{3-\nu}\sigma:\varepsilon - \frac{1-3\nu}{(1+\nu)(3-\nu)}tr(\sigma)tr(\varepsilon) \tag{3.5}$$

It captures the first order impact of adding a small circular material in the soft-region on the compliance. Note that stresses & strains are related through E in Equation (3.2), and through E_ε in Equation (3.5).

The two fields $\mathcal{T}^S$ & $\mathcal{T}^A$ are, in general, discontinuous at the boundary of Ω. For convenience, one can combine the two into a single (discontinuous) topological sensitivity field $\mathcal{T}$ over the entire domain per:

$$\mathcal{T}(r) = \begin{cases} \mathcal{T}^S(r) \text{ if } r \in \Omega \\ -\mathcal{T}^A(r) \text{ if } r \in D - \Omega \end{cases} \tag{3.6}$$

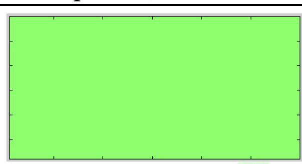
For compliance problems, since subtracting (or adding) material never decreases (or increases) the compliance:

$$\mathcal{T}^S(p) \ge 0 \ge \mathcal{T}^A(q), \forall p \in \Omega, \forall q \in D - \Omega \tag{3.7}$$

it follows that $\mathcal{T} \ge 0$.

3.3 Pareto-Optimality Criteria

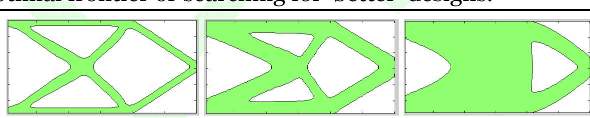
Consider now the multi-objective problem stated earlier in Equation (2.1). Various pareto-optimal solutions exist for Ω, for the example illustrated in Figure 2. For example, the special case of $\Omega = D$ is illustrated in Figure 5, i.e., where the structure occupies the entire space provided. This is a pareto-optimal design since it is impossible to find a structure of lower compliance, and identical volume.



$J = 39; v = 1.0$

Figure 5: A pareto-optimal topology of volume 1.

On the other hand, consider the structures illustrated in Figure 6; the compliances and volumes are also provided. *Are these topologies pareto-optimal* with respect to the objectives in Equation (2.1)? This question cannot be answered today without either computing the pareto-optimal frontier or searching for 'better' designs.



(a) $J = 162; v = 0.3$; (b) $J = 66; v = 0.5$; (c) $J = 56; v = 0.65$;

Figure 6: Are these topologies pareto-optimal with respect to Equation (2.1)?

The lack of an 'inherent' test to determine pareto-optimality of topologies is a serious short-coming. We now address this through an important claim on local pareto-optimality by combining the definitions of the above two Sections.

Lemma 1: A necessary condition for a domain Ω to be locally pareto-optimal with respect to Equation (2.1) is that its topological sensitivity fields must satisfy the inequality:

$$\min(\mathcal{T}^S) + \min(\mathcal{T}^A) \ge 0 \tag{3.8}$$

Proof: Let Ω' be a nearby topology with *identical* volume at a distance of δ from it, i.e., Ω' is constructed from Ω by subtracting M discs of areas $\delta_i^S > 0$, and adding N discs of areas $\delta_i^A > 0$, i.e.,

$$\Omega' = \Omega \setminus \sum_{i=1}^{M} B_{\delta_i^S}(p_i) \cup \sum_{j=1}^{N} B_{\delta_j^A}(q_j) \tag{3.9}$$

such that:

$$\sum_{i=1}^{M}\delta_i^S = \sum_{j=1}^{N}\delta_j^A = \delta/2 \tag{3.10}$$

Note that the change in compliance is:

$$\Delta J = \sum_{i=1}^{M}\mathcal{T}^S(p_i)\delta_i^S + \sum_{j=1}^{N}\mathcal{T}^A(q_j)\delta_j^A + o(\delta) \tag{3.11}$$

If Ω is locally pareto-optimal, Ω' cannot have a lower compliance implying:

$$\Delta J = \sum_{i=1}^{M}\mathcal{T}^S(p_i)\delta_i^S + \sum_{j=1}^{N}\mathcal{T}^A(q_j)\delta_j^A + o(\delta) \ge 0, \forall p_i, q_j, \delta_i^S, \delta_j^A \tag{3.12}$$

Considering Equation (3.7), as a particular choice subtract material of volume $\delta/2$ at a location where $\mathcal{T}^S$ takes a minimum, and add material of volume $\delta/2$ at a location where $\mathcal{T}^A$ also takes a minimum, i.e.,

$$\Delta J = \min(\mathcal{T}^S)\delta/2 + \min(\mathcal{T}^A)\delta/2 + o(\delta) \ge 0 \tag{3.13}$$

In the limit of $\delta \to 0$, from Equation (3.4)

$$\min(\mathcal{T}^S) + \min(\mathcal{T}^A) \ge 0 \tag{3.14}$$

♦

Thus, to determine if a topology Ω satisfies the necessary condition to be locally pareto-optimal, one must:

1. Solve the elasticity problem for u, ε & σ over D
2. Compute the topological sensitivity fields per Equations (3.2) and (3.5).
3. Check if Equation (3.8) is satisfied.

Indeed, from the above inherent test, one can easily determine that the topologies in Figure 6a and Figure 6c are *not* locally pareto-optimal. However, the topology in Figure 6b indeed satisfies the necessary condition for local pareto-optimality. In theory, second order checks are necessary to ensure local minimum. However, in practice, we found that the algorithms based on the above Lemma, trace the local minima.

From the above Lemma we have the following Corollary.

Corollary: If Ω is pareto-optimal, there exists a scalar l such that

$$\mathcal{T}^S(p) \ge l \ge -\mathcal{T}^A(q), \forall p \in \Omega, \forall q \in D - \Omega \tag{3.15}$$

Proof: From Equation (3.8), we have:

$$\min(\mathcal{T}^S) \ge -\min(\mathcal{T}^A) = \max(-\mathcal{T}^A) \tag{3.16}$$

i.e.,

$$\begin{aligned} &\mathcal{T}^S(p) \ge \min(\mathcal{T}^S) \ge \max(-\mathcal{T}^A) \ge -\mathcal{T}^A(q) \\ &\forall p \in \Omega, \forall q \in D - \Omega \end{aligned} \tag{3.17}$$

Thus Equation (3.15) follows.

Structural and Multidisciplinary Optimization. Accepted: June 10, DOI: 10.1007/s00158-010-0534-6

♦

Without a loss of generality, if we assume $l > \min(\mathcal{T}^S)$, then it follows from the above corollary and Equation (3.6) that a pareto-optimal domain satisfies the inverse relationship:

$$\Omega = \{r \mid \mathcal{T}(r) > l\} \tag{3.18}$$

This inverse relationship will be exploited in the algorithm below.

Observe that the scalar l in Equation (3.15) is not uniquely defined since the two fields $\mathcal{T}^S \,\&\, -\mathcal{T}^A$ are not necessarily continuous across the boundary $\partial\Omega$.

However, in the algorithm below, we apply a filter on the topological sensitivity fields in order to eliminate the 'checker-field' effect (similar to the filtering applied on the density in SIMP); this renders the two topological fields to be continuous. We then relax the constraint in Equation (3.15), and impose the volume constraint:

$$|\Omega| = v \tag{3.19}$$

In other words, the scalar l is determined such that the *filtered* topological sensitivity field $\mathcal{T}^*$ satisfies

$$\left|\{r \mid \mathcal{T}^*(r) > l\}\right| = v \tag{3.20}$$

The scalar l is now uniquely determined with the following exception: if there are two topologies that are equally optimal, then the algorithm may oscillate. Thus, in practice, one must detect and break such oscillation cycles. These implementation details are discussed further under Table 2 in Section 4.

3.4 Pareto-Frontier Tracing Algorithm

We now apply the above set of results to arrive at an algorithm for tracing pareto-optimal topologies. In particular, let Ω be a locally pareto-optimal topology. The objective is to compute a nearby pareto-optimal topology Ω' whose volume is less than that of Ω by Δv, i.e.,

$$|\Omega'| = |\Omega| - \Delta v \tag{3.21}$$

We rely on a fixed-iteration scheme similar to the ones discussed in [35, 36] in that: (1) given a domain Ω one can compute its topological field $\mathcal{T}$ via Equation (3.6), and (2) given a valid topological field $\mathcal{T}$, one can compute the corresponding Ω via Equation (3.18). Further, by imposing Equation (3.8) for local pareto-optimality, we arrive at the following algorithm.

Given : A pareto-optimal topology Ω, step size Δv
Find : A near-by pareto-optimal topology Ω'
  such that : $|\Omega'| = |\Omega| - \Delta v$
$v' \leftarrow |\Omega| - \Delta v$
$\Omega' \leftarrow \Omega$
Do
  $\mathcal{T} \equiv \left(\mathcal{T}^S, \mathcal{T}^A\right) \leftarrow \mathcal{T}(\Omega')$
  $l \leftarrow \left|\{r \mid \mathcal{T}(r) > l\}\right| = v'$
  $\Omega' \leftarrow \{r \mid \mathcal{T}(r) > l\}$
While $(\mathcal{T}^S_{\min} < \mathcal{T}^A_{\max}) \mid (\mathcal{T}^S_{\min} + \mathcal{T}^A_{\min} < 0)$

Algorithm 1: Finding a pareto-optimal topology with volume decrement of Δv.

The *existence* of the parameter l in the above algorithm hinges on the *existence* of a pareto-optimal domain Ω' satisfying Equation (3.21). If Ω' exists, then its topological sensitivity pair exists and can be computed. Since this pair satisfies Equation **(3.7)**, there exists an l satisfying:

$$\mathcal{T}^S > l > \mathcal{T}^A \tag{3.22}$$

Existence does not imply uniqueness or that the algorithm will converge to the correct value of l! Indeed, if a very large step-size is taken for Δv, the algorithm may never converge. In practice, $\Delta v \le 0.1$ is recommended.

To further optimize the step-size Δv, one can estimate the change in compliance at each step. Further, since the $\Omega = D$ is pareto-optimal, we have the following algorithm to trace the pareto-optimal curve.

Given : A domain D, $\Delta v_{\max}, \Delta J_{\max}$
Find : The pareto-optimal curve for (J, v)
$i \leftarrow 0$
${}^{(i)}\Omega \leftarrow D$
${}^{(i)}v \leftarrow |D|$
Do
  ${}^{(i)}\mathcal{T} \equiv \left({}^{(i)}\mathcal{T}^S, {}^{(i)}\mathcal{T}^A\right) \leftarrow \mathcal{T}({}^{(i)}\Omega)$
  ${}^{(i)}J_{,v} \leftarrow -\min({}^{(i)}\mathcal{T}^S)$
  $\Delta v \leftarrow \min(\Delta v_{\max}, \Delta J_{\max} / {}^{(i)}J_{,v})$
  ${}^{(i+1)}\Omega \leftarrow \text{Algorithm-1}({}^{(i)}\Omega, \Delta v)$
  ${}^{(i+1)}v \leftarrow {}^{(i)}v - \Delta v$
  $i \leftarrow i + 1$
While $({}^{(i)}v > 0)$

Algorithm 2: Tracing the pareto-optimal curve.

The strength and weakness of the proposed algorithm is that it moves from one local minimum to the next closest local minimum (with a new set of volume constraints) on the pareto-optimal curve. By exploiting the closeness of locally pareto-optimal solutions (as defined in the paper), the computational expense of pareto-tracing is reduced dramatically, as demonstrated via numerical experiments. The short-coming is that a 'far away' and alternate pareto-optimal solution cannot be detected via the proposed method.

4. MATLAB CODE

For convenience, the Matlab code for the above algorithm is listed in the Appendix; it can also be downloaded from the Matlab file exchange website (www.mathworks.com/matlabcentral/fileexchange/). The finite element assumptions, syntax and conventions closely follow those of Sigmund [4]. The input parameters are summarized in Table 1 below. The user can run the code with the default parameters via the Matlab call `ParetoOptimalTracing;`

Table 1: Description of input parameters

	Description	*Default*

Structural and Multidisciplinary Optimization. Accepted: June 10, DOI: 10.1007/s00158-010-0534-6

`nely`	The number of quad elements in the vertical direction.	30
`nely`	The number of quad elements in the horizontal direction.	60
`desVolFrac`	The final volume fraction desired (0.1 to 1.0).	0.5
`problem`	Choose between the two topology optimization problems (see experiments below). Additional problems can be modeled via appropriate boundary conditions [4].	1
`volDecrMax`	The maximum volume decrement Δv allowed during pareto-tracing.	0.05
`JIncMac`	The maximum compliance decrement ΔJ allowed during pareto-tracing.	3.0
`filterRadius`	This is used for smoothening the topological sensitivity field, similar to the use of filters in SIMP [4].	0.8

A brief explanation of the Matlab code is given in the following table. The reader is also encouraged to study the conventions in [4].

Table 2: Description of the Matlab code.

Lines	*Description*
`1-16`	Function call with default parameters
`17-26`	We carry out a FEA for the full domain, and compute the filtered topological sensitivity field. Gaussian filters work the best; other filters may however be used.
`27-31`	Check if the final volume fraction has been reached; if not, decrement volume fraction.
`32`	Terminate after 10 fixed-point iterations to avoid oscillation/ cycles. In almost all experiments, the iteration converged in 6~8 iterations.
`34-37`	If the current topology is pareto-optimal, terminate fixed-point iteration.
`39`	Compute the contour value of a 'nearby' topology with desired volume.
`40-42`	Eliminate quad elements that do not lie within the above topology.
`43-47`	Carry out a FEA over current topology; also compute filtered topological sensitivity.
`48`	After a pareto-optimal topology has been determined, compute its compliance.
`49-52`	Extract optimal topology and plot.
`53-54`	Determine the next volume decrement.
`56-57`	Final plot of pareto-optimal curve.
`58-97`	FEA code; see [4].
`98-127`	A function to compute the topological sensitivity fields for elements inside and outside; see Equations (3.2) and (3.5).
`128-141`	Compliance computation; see [4].
`142-166`	Given a pair of topological sensitivity fields, and a desired volume fraction, compute the appropriate level-set value. A binary search is used to find the level-set value. Extra rows and columns are added around the field to obtain closed-contours.
`167-179`	For a given buffered-field (see above), and a level-set value, find the area enclosed.
`180-192`	Implementation of Equations **(3.7)** & (3.8)
`193-199`	A function for plotting the contour / topology.

5. NUMERICAL EXPERIMENTS

In this Section, we illustrate the above algorithm through numerical experiments, using the default parameters, unless otherwise stated. For all experiments, the domain D is discretized into (60,30) linear quad elements.

For SIMP, we use the Matlab code described in [4] with the following parameters:

- For each volume fraction v, we determine the optimal topology by minimizing compliance (i.e., single objective minimization).
- We use a filtering radius of 1.5 to smoothen the sensitivity field and use a penalty of 3.0 for density.
- The termination criteria is when the density change is less than 0.01 [4].

In addition, to trace the pareto-optimal curve via SIMP, we use two strategies:

1. **SIMP-restart**: Here, for each volume fraction, the density is initialized to a uniform density of specified volume fraction.
2. **SIMP-continuous**: Here, for each volume fraction, the density is initialized to the termination density of optimal topology of the previous volume fraction.

In the proposed method and the two strategies of SIMP, the most dominating cost is finite element analysis (FEA). Therefore, in the experiments below, we note the number of FEA runs required to reach an optimal topology, for a given volume fraction, via the proposed algorithm, and via single-objective SIMP-restart & SIMP-continuous.

5.1 Problem-1

The first set of experiments is on the classic cantilevered beam-problem posed below. We assume that Ω must lie within a space D, illustrated in Figure 7 of unit volume, and dimensions as shown. The structure Ω is to be fixed on one edge, and must carry a unit-load F on the other end as illustrated; the material properties are E = 1, and $\nu = 0.3$.

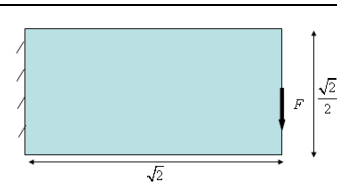
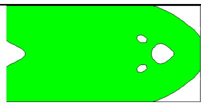
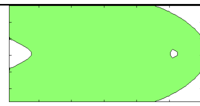
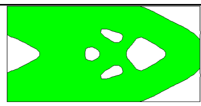
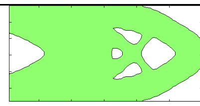
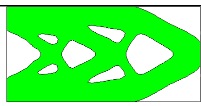
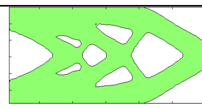
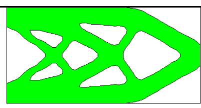
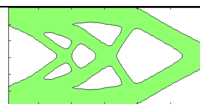
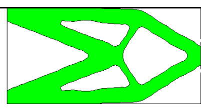
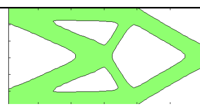


Figure 7: A cantilevered beam-problem

The table below summarizes the compliances for various volume fractions. For each volume fraction, we provide the compliance as predicted by the proposed method, and that from SIMP-restart (SIMP-continuous yielded almost identical result, and are therefore not included). There are minor differences between the topologies generated by the two methods, but the compliances are almost identical, confirming that we have achieved pareto-optimality.

Table 3: Optimal topologies and compliances for Figure 7.

	Proposed Method	***SIMP (restart and continuous)***
$v = 0.9$	 J=40.8	 J=40.64
$v = 0.8$	 J=44	 J=44
$v = 0.7$	 J=48.96	 J=48.98
$v = 0.6$	 J=55.3	 J=56.1
$v = 0.5$	 J=68	 J=66.3

The figure below captures the number of *cumulative* FEA runs (the most dominating cost) for the three strategies, as a function of the volume fraction removed.

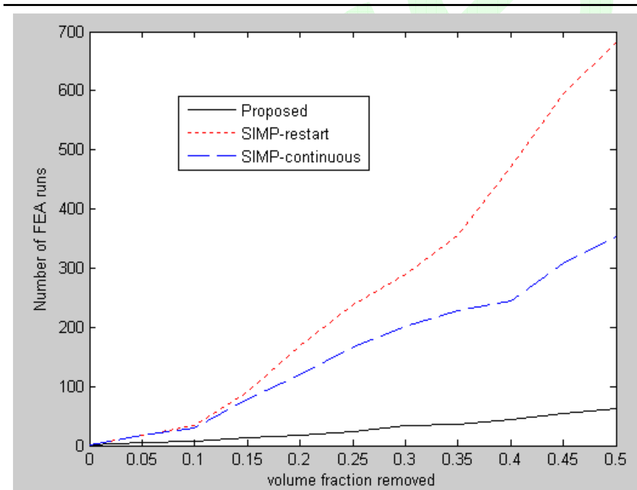


Figure 8: Number of cumulative FEA runs as a function of the volume fraction removed.

Observe that the number of FEA runs to achieve a given volume fraction for the proposed method is significantly smaller than that of either SIMP strategies. Thus, one can generate the entire set of pareto-optimal topologies with 62 FEA runs as illustrated, SIMP-restart takes 628 FEA runs, and SIMP-continuous takes 353 runs.

The figure below illustrates the pareto-optimal curve as generated by the proposed method.

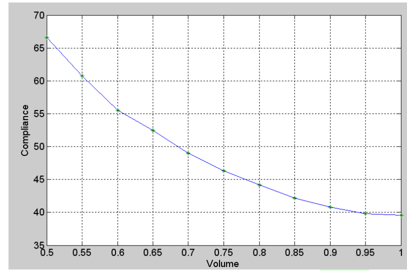


Figure 9: The pareto-optimal curve for the cantilevered beam problem.

We will now vary some of the parameters and study their impact. For example, the figure below compares the pareto-optimal curve for various mesh densities, namely (40,20), (60,30) (default), and (80,40); all other parameters being identical to the default parameters. Not surprisingly, finer density meshes yield lower pareto-optimal curves, and topologies with more 'holes'.

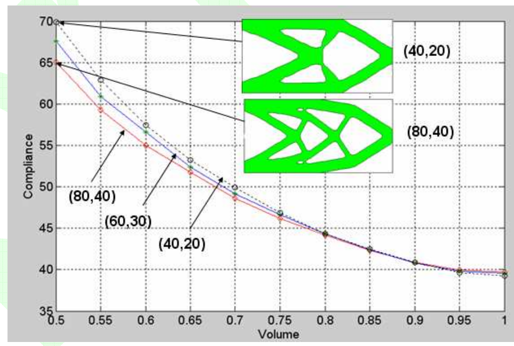


Figure 10: The pareto-optimal curves for the cantilevered beam problem for various mesh sizes.

Next we consider three volume step-sizes $\Delta v = 0.025$, $\Delta v = 0.05$ (default), and $\Delta v = 0.025$, and compare the pareto-optimal curves; all other parameters being identical. Since the curves in Figure 11 are almost identical, we have not identified them; the topologies were also identical.

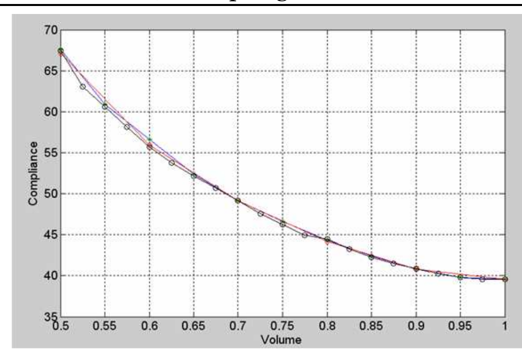


Figure 11: The pareto-optimal curves for the cantilevered beam problem for various volume step-sizes.

Finally, the pareto-optimal curve was found to be relatively insensitive to the filter radius (ranging from 0.5 to 2.0). But, for lower filter radius, topologies with larger number of holes were generated, as expected.

5.2 Problem-2

The next set of experiments is on the beam problem posed below with material parameters, etc. as before.

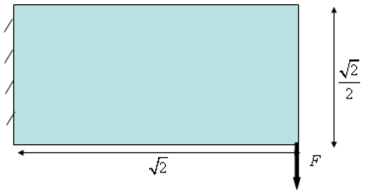
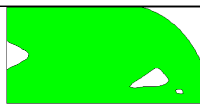
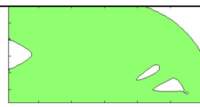
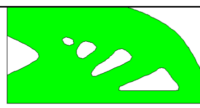
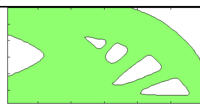
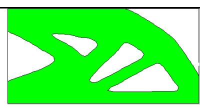
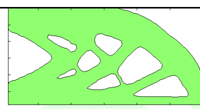
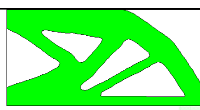
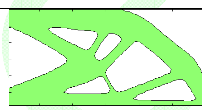
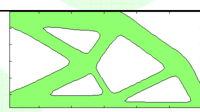


Figure 12: Problem-2

The table below compares the results from the proposed method and SIMP-restart for various volume fractions. Once again, there are minor differences between the topologies generated by the two methods, but the compliances are almost identical, confirming that we have achieved pareto-optimality.

Table 4: Optimal topologies and compliances.

	Proposed Method	***SIMP (restart and continuous)***
$v = 0.9$	 J=46.33	 J=46.22
$v = 0.8$	 J=49.8	 J=49.6
$v = 0.7$	 J=54.3	 J=54.8
$v = 0.6$	 J=61.77	 J=62.2
$v = 0.5$	 J=73.13	 J=73.1

The figure below captures the number of *cumulative* FEA runs (the most dominating cost) for the three strategies, as a function of the volume fraction removed.

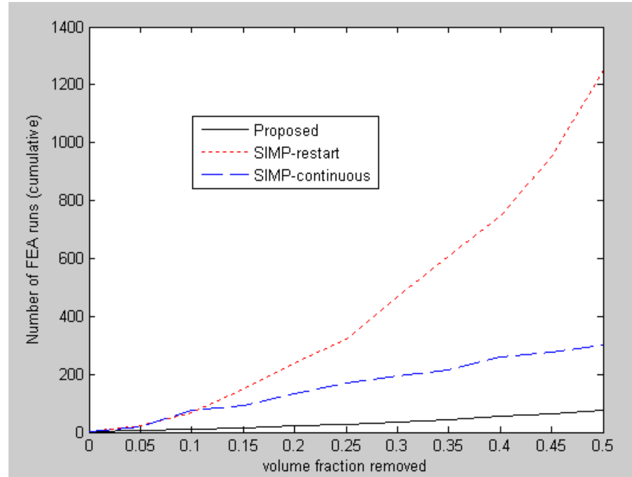


Figure 13: Number of cumulative FEA runs as a function of the volume fraction removed.

Observe that, once again, the number of FEA runs to achieve a given volume fraction for the proposed method is significantly smaller than that of either SIMP strategies.

One can generate the entire set of pareto-optimal topologies with 75 FEA runs via the proposed method, while SIMP-continuous requires 250 FEA runs. The figure below illustrates the pareto-optimal curve as generated by the proposed method.

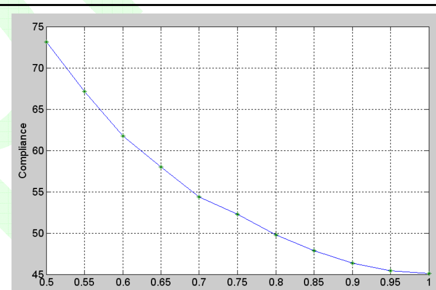


Figure 14: The pareto-optimal curve.

6. CONCLUSIONS

The three most significant contributions of the paper are: (1) a theoretical framework for determining if a topology satisfies the necessary condition for local pareto-optimality, (2) an efficient algorithm for tracing pareto-optimal curves for compliance-related objectives, and (3) a compact Matlab code for generating pareto-optimal topologies.

Future work will focus on: (a) non-compliance objectives since the topological sensitivity concept is well defined for a large class of problems, (b) further improving the efficiency of the proposed algorithm, and (c) considering arbitrary shaped features (rather than uniform discs and spheres) for the topological sensitivity field.

6. REFERENCES

1. Bendsøe, M.P., Kikuchi, N, *Generating optimal topologies in optimal design using a homogenization method*. Computer Methods in Applied Mechanics and Engineering, 1988. **71**: p. 197–224.
2. Bendsøe, M.P., *Optimal shape design as a material distribution problem*. Structural Optimization, 1989. **1**(193-202).

3. Zhou, M., Rozvany, G.I.N., *The COC algorithm, part II: Topological, geometry and generalized shape optimization*. Computer Methods in Applied Mechanics and Engineering, 1991. **89**: p. 197-224.
4. Sigmund, O., *A 99 line topology optimization code written in Matlab*. Structural and Multidisciplinary Optimization, 2001. **21**(2): p. 120-127.
5. Allaire, G., Jouve, F., Toader, A., *Structural Optimization using Sensitivity Analysis and a Level-set Method*. Journal of Computational Physics, 2004. **194**(1): p. 363-393.
6. Burger, M., Hackl, B., Ring, W., *Incorporating Topological Derivatives into Level Set Methods*. Journal of Computational Physics, 2004. **194**(1): p. 344-362.
7. Wang, M.Y., Wang, X., Guo, D., *A level set method for structural topology optimization*. Computer Methods in Applied Mechanics and Engineering, 2003. **192**: p. 227-246.
8. Feijóo, R.A., Novotny, A. A., Taroco, E., Padra, C., *The Topological Derivative for the Poisson's Problem*. Mathematical Models and Methods in Applied Sciences, 2003. **13**(12): p. 1825-1844.
9. Novotny, A.A., Feijóo, R. A., Taroco, E., Padra, C., *Topological-Shape Sensitivity Analysis*. Computer Methods in Applied Mechanics and Engineering, 2003. **192**(7): p. 803-829.
10. Novotny, A.A., Feijóo, R. A., Taroco, E., Padra, C. *Topological Sensitivity Analysis for Three-dimensional Linear Elasticity Problem*. in *6th World Congress on Structural and Multidisciplinary Optimization*. 2005. Rio de Janeiro.
11. Novotny, A.A., Feijóo, R. A., Taroco, E., Padra, C., *Topological-Shape Sensitivity Method: Theory and Applications*. Solid Mechanics and its Applications, 2006. **137**: p. 469-478.
12. Belytschko, T., Xiao, S. P., Parimi, C., *Topology Optimization with Implicit Functions and Regularization*. International Journal for Numerical Methods in Engineering, 2003. **57**(8): p. 1177–1196.
13. Sokolowski, J., Zochowski, A., *On Topological Derivative in Shape Optimization*. SIAM journal on control and optimization, 1999. **37**(4): p. 1251-1272.
14. Sokolowski, J., Zochowski, A., *Optimality Conditions for Simultaneous Topology and Shape Optimization*. SIAM journal on control and optimization, 2003. **42**(4): p. 1198-1221.
15. Eschenauer, H.A., Kobelev, V. V., Schumacher, A, *Bubble method for topology and shape optimization of structures*. Structural Optimization, 1994. **8**: p. 42-51.
16. Céa J., G., S., Guillaume, P., Masmoudi, M., *The shape and topological optimizations connection*. Computer Methods in Applied Mechanics and Engineering, 2000. **188**: p. 713-726.
17. Cohon, J.L., *Multiobjective programming and planning*. Vol. 140. 1978, London: Academic Press, Inc.
18. Eschenauer, H.A., Olhoff, N., *Topology optimization of continuum structures: A review*. Applied Mechanics Review, 2001. **54**(4): p. 331-389.
19. Rozvany, G.I.N., *Aims, scope, methods, history and unified terminology of computer-aided topology optimization in structural mechanics*. Structural and Multidisciplinary Optimization, 2001. **21**(2): p. 90-108.
20. Papalambros, P.Y., *The optimization paradigm in engineering design: promises and challenges*. Computer-Aided Design, 2002. **34**(12): p. 939-951.
21. Rozvany, G.I.N., *Stress ratio and compliance based methods in topology optimization – a critical review*. Structural and Multidisciplinary Optimization, 2001. **21**(2): p. 109-119.
22. Deb, K., *Multi-Objective Optimization Using Evolutionary Algorithms*. 2001, Chichester,: John Wiley & Sons,.
23. Messac, A., Ismail-Yahaya, A., *Required Relationship Between Objective Function and Pareto Frontier Orders: Practical Implications*. AIAA JOURNAL, 2001. **39**(11): p. 2168-2174.
24. Messac, A., Sundararaj, G.J., Tappeta, R.V., Renaud, J.E., *Ability of Objective Functions to Generate Points on Non-Convex Pareto Frontiers*. AIAA JOURNAL, 2000. **38**(6): p. 1084-1091.
25. Zhang, W.H., Yang, H.C., *Efficient gradient calculation of the Pareto optimal curve in multicriteria optimization*. Structural and Multidisciplinary Optimization, 2002. **23**: p. 311–319.
26. Zienkiewicz, O.C., Taylor, R. L., *The Finite Element Method for Solid and Structural Mechanics*. 6th ed. 2005: Elsevier.
27. Das, I., Dennis, J.E., *A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems*. Structural Optimization, 1997. **14**: p. 63-69.
28. Chen, T.Y., Wu, S-C, *Multiobjective optimal topology design of structures*. Computational Mechanics, 1998. **21**: p. 483-492.
29. Lin, J., Luo, Z., Tong, L., *A new multi-objective programming scheme for topology optimization of compliant mechanisms*. Structural and Multidisciplinary Optimization, 2010. **30**: p. 241–255.
30. Luo, Z., Chen, L. Yang, J., Zhang, Y., Abdel-Malek, K., *Compliant mechanism design using multi-objective topology optimization scheme of continuum structures*. Structural and Multidisciplinary Optimization, 2005. **30**: p. 142–154.
31. Hamda, H., Roudenko, O., Schoenauer, M. *Application of a multi-objective evolutionary algorithm to topology optimum design*. in *Fifth International Conference on Adaptive Computing in Design and Manufacture*. 2002.
32. Madeira, J.F.A., Rodrigues, H., Pina, H., *Multi-objective optimization of structures topology by genetic algorithms*. Advances in Engineering Software, 2005. **36**: p. 21-28.
33. Padhye, N. *Topology Optimization of Compliant Mechanism using Multi-Objective Particle SwarmOptimization*. in *GECCO'08, July 12–16*. 2008. Atlanta, Georgia, USA.: ACM.
34. Chen, T.-Y., Wu, S.-C., *Multiobjective optimal topology design of structures*. Computational Mechanics, 2002. **21**: p. 483-492.
35. Céa J, G., S, Guillaume, P, Masmoudi, M., *The shape and topological optimizations connection*. Computer Methods in Applied Mechanics and Engineering, 2000. **188**: p. 713-726.
36. Norato, J.A., Bendsøe, M. P., Haber, R.B., Tortorelli, D.A., *A topological derivative method for topology optimization*. Structural and Multidisciplinary Optimization, 2007. **33**: p. 375–386.

37. Samet, B., *The topological asymptotic with respect to a singular boundary perturbation*. Comptes Rendus Mathematique, 2003. **336**(12): p. 1033-1038.
38. Dambrine, M., Vial, G., *Influence of a boundary perforation on the Dirichlet energy*. Control and Cybernetics, 2005. **34**(1): p. 117-136.
39. Gopalakrishnan, S.H., Suresh, K.,, *Feature Sensitivity: A Generalization of Topological Sensitivity*. Finite Elements in Analysis and Design, 2008. **44**(11): p. 696-704.
40. Feijóo, R.A., Novotny, A.A., Taroco,E., Padra, C., *The topological-shape sensitivity method in two-dimensional linear elasticity topology design*, in *Applications of Computational Mechanics in Structures and Fluids*, V.S. S.R. Idelsohn, Editor. 2005, CIMNE.
41. Amstutz, S., *Sensitivity analysis with respect to a local perturbation of the material property*. Asymptotic Analysis, 2006. **49**(1-2): p. 87-108.

Appendix: Matlab Code

```
function ParetoOptimalTracing(nelx,nely,desVolFrac,problem,volDecrMax,JIncMax,filterRadius)
% Generate pareto-optimal topologies via fixed point iteration
% Author: Krishnan Suresh; suresh@engr.wisc.edu
% Reference: "A 199-line Matlab Code for Pareto-Optimal Tracing in Topology Optimization",
%             K. Suresh, Vol X, pp xy, Structural and Multidisciplinary Optimization,
% Acknowledgements: "A 99 line topology optimization code written in Matlab"
%                   by Ole Sigmund (2001), Structural and Multidisciplinary Optimization,
%                   Vol 21, pp. 120--127.
if (nargin == 0) % default values
    nelx = 60;nely = 30; % The grid size for topology optimization
    desVolFrac = 0.5; % The final volume fraction desired
    problem = 1; % 1 or 2 for cantilevered beam problems
    volDecrMax = 0.05; % step-size for pareto-tracing
    JIncMax = 3; % For steep change in pareto-curve, use additional constraint
    filterRadius = 0.8; % Use for smoothening the topological sensitivity field
end
voidEps = 1e-4; % Relative Young's Modulus of void region
filter = fspecial('gaussian', [3 3],filterRadius); % smoothen topological sensitivity  field
totalIter = 0;
elemsIn(1:nely,1:nelx) = 1; % intialize the domain
U = FE(nelx,nely,elemsIn,voidEps,problem); % Solve FEA problem
T = ComputeT(U,elemsIn,voidEps); % Compute topological sensitivity
T = filter2(filter,T); % smoothen the field
J(1) = computeCompliance(nelx,nely,elemsIn,voidEps,U); % compute & store compliance
volIndex = 1;volFractions(1) = 1; volfrac = 1; % initialization
volDecrement = volDecrMax; % current decrement of volume fraction
while (volfrac > desVolFrac)
    volfrac = volfrac-volDecrement; % move to the next volume fraction
    volIndex = volIndex+1;
    volFractions(volIndex) = volfrac; % store the volume fraction
    iter = 0;
    while (iter < 10) % to avoid cycles; typically 10 iterations is sufficient
        totalIter= totalIter+1;
        [isValid,isParetoOptimal] = analyzeTopology(T,elemsIn);
        if ((iter > 0)&&(isValid)&&(isParetoOptimal)) % done with current vol
            break
        end
        % Find the level-set value such that the contour has given vol fraction
        value = findContourValueWithVolumeFraction(T,volfrac);
        [index] = find(T < value); % eliminate all elements less than this value
        elemsIn(1:nely,1:nelx) = 1; % start with the full domain
        elemsIn(ind2sub(size(T),index)) = 0; % remove elements
        U = FE(nelx,nely,elemsIn,voidEps,problem); % FEA
```

```
        T = ComputeT(U,elemsIn,voidEps); % Topological Sensitivity
        T = filter2(filter,T); % Smoothen the field
        iter= iter+1;
    end
    J(volIndex) = computeCompliance(nelx,nely,elemsIn,voidEps,U);
    value = findContourValueWithVolumeFraction(T,volfrac); % as above
    plotContour(T,value,figure(1));
    title(['v=' num2str(volfrac) '; J = ' num2str(J(volIndex)) '; #FEA = ' num2str(totalIter)]);
    pause(0.001);
    dJ =  J(volIndex)- J(volIndex-1);
    volDecrement = max(volDecrement/5,min(volDecrement,JIncMax*volDecrement/dJ));
end
figure(2); plot(volFractions,J,volFractions,J,'*');
xlabel('Volume'); ylabel('Compliance'); grid on;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [U]=FE(nelx,nely,elemsIn,voidEps,problem)
if (problem == 1) % Cantilevered beam;
    fixeddofs = 1:2*(nely+1); % left edge
    forcedDof = 2*(nelx+1)*(nely+1)-nely; % y force
elseif (problem == 2) % MBB beam
    fixeddofs = 1:2*(nely+1); % left edge
    forcedDof = 2*(nelx+1)*(nely+1)-2*nely; % y force
end
K = sparse(2*(nelx+1)*(nely+1), 2*(nelx+1)*(nely+1));
F = sparse(2*(nely+1)*(nelx+1),1); U = zeros(2*(nely+1)*(nelx+1),1);
[KE] = lk;
for elx = 1:nelx
  for ely = 1:nely
    n1 = (nely+1)*(elx-1)+ely;
    n2 = (nely+1)* elx   +ely;
    edof = [2*n1-1 2*n1 2*n2-1 2*n2 2*n2+1 2*n2+2 2*n1+1 2*n1+2]';
    alpha = (1-elemsIn(ely,elx))*voidEps + elemsIn(ely,elx);
    K(edof,edof) = K(edof,edof) + alpha*KE;
  end
end
F(forcedDof,1) = -1;
alldofs     = 1:2*(nely+1)*(nelx+1);
freedofs    = setdiff(alldofs,fixeddofs);
U(freedofs,:) = K(freedofs,freedofs) \ F(freedofs,:);
U(fixeddofs,:)= 0;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [KE]=lk % element stiffness
E = 1.; nu = 0.3;
k=[ 1/2-nu/6   1/8+nu/8 -1/4-nu/12 -1/8+3*nu/8 ...
   -1/4+nu/12 -1/8-nu/8  nu/6       1/8-3*nu/8];
KE = E/(1-nu^2)*[ k(1) k(2) k(3) k(4) k(5) k(6) k(7) k(8)
                  k(2) k(1) k(8) k(7) k(6) k(5) k(4) k(3)
                  k(3) k(8) k(1) k(6) k(7) k(4) k(5) k(2)
                  k(4) k(7) k(6) k(1) k(8) k(3) k(2) k(5)
                  k(5) k(6) k(7) k(8) k(1) k(2) k(3) k(4)
                  k(6) k(5) k(4) k(3) k(2) k(1) k(8) k(7)
                  k(7) k(4) k(5) k(2) k(3) k(8) k(1) k(6)
                  k(8) k(3) k(2) k(5) k(4) k(7) k(6) k(1)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [T] = ComputeT(U,elemsIn,voidEps)
```

```
% Compute the topological sensitivity at the center of each element
[nely,nelx] = size(elemsIn);
gradN =0.5*[-1 1 1 -1;-1 -1 1 1]; % at center
E0 = 1;nu = 0.3;
D0 = 1/(1-nu^2)*[1 nu 0; nu 1 0;0 0 (1-nu)/2]; % plane stress
T(1:nely,1:nelx) = 0; % initialize to 0
for elx = 1:nelx
  for ely = 1:nely
    n1 = (nely+1)*(elx-1)+ely;
    n2 = (nely+1)* elx   +ely;
    edof = [2*n1-1 2*n1 2*n2-1 2*n2 2*n2+1 2*n2+2 2*n1+1 2*n1+2]';
    uGrad = gradN*U(edof(1:2:end));
    vGrad = gradN*U(edof(2:2:end));
    strains = [uGrad(1); vGrad(2); (uGrad(2)+vGrad(1)) ];
    alpha = (1-elemsIn(ely,elx))*voidEps + elemsIn(ely,elx);
    E = E0*alpha;
    stresses = D0*E*strains;
    stressTensor = [stresses(1) stresses(3); stresses(3) stresses(2)];
    strainTensor = [strains(1) strains(3)/2; strains(3)/2 strains(2)];
    if (elemsIn(ely,elx))
        T(ely,elx) = 4/(1+nu)*sum(sum(stressTensor.*strainTensor))- ...
         (1-3*nu)/(1-nu^2)*trace(stressTensor)*trace(strainTensor);
    else
        T(ely,elx) = 4/(3-nu)*sum(sum(stressTensor.*strainTensor))+...
          (1-3*nu)/((1+nu)*(3-nu))*trace(stressTensor)*trace(strainTensor);
    end
  end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [J]=computeCompliance(nelx,nely,elemsIn,voidEps,U)
% Compute the compliance of the entire mesh
[KE] = lk;J = 0;
for elx = 1:nelx
  for ely = 1:nely
    n1 = (nely+1)*(elx-1)+ely;
    n2 = (nely+1)* elx   +ely;
    edof = [2*n1-1 2*n1 2*n2-1 2*n2 2*n2+1 2*n2+2 2*n1+1 2*n1+2]';
    alpha = (1-elemsIn(ely,elx))*voidEps + elemsIn(ely,elx);
    Ue = U(edof);
    J = J + alpha*Ue'*KE*Ue;
  end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function value = findContourValueWithVolumeFraction(field,volfrac)
 % Find the level-set value such that the contour has given vol fraction
 % The code computes the level-set value with desired external volume
[nely,nelx] = size(field);
externalVolumeDesired = nelx*nely*(1-volfrac);
field = -field; % reverse the sign to compute external volume
valMax = 0; valMin = min(field(:));
bufferedField = valMin*ones(nely+2,nelx+2);% Add buffer to get closed contours
bufferedField(2:end-1,2:end-1) = field;
iterMax = 50;iter = 1;
while (1) % A binary search is used to find the optimal level-set value
    valMid = (valMax+valMin)/2;
```

```
    extVol = computeAreaInContour(bufferedField,valMid);
    err = abs(extVol-externalVolumeDesired)/(extVol);
    if (err < 0.001) || (iter > iterMax)
        value = -valMid; % change the sign before return
        return;
    end
    if (extVol > externalVolumeDesired)
        valMin = valMid;
    else
        valMax = valMid;
    end
    iter = iter+1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function area = computeAreaInContour(bufferedField,value)
% For a given level-set value, compute the area enclosed
% It is assumed that the field has been buffered; see code above
[cntr,h] = contours(bufferedField,[value value]);
indices = find(cntr(1,:) == value);area = 0;
for i = 1:numel(indices)
    startCol = indices(i)+1;
    endCol = startCol+ cntr(2,indices(i))-1;
    xPoly = cntr(1,startCol:endCol);
    yPoly = cntr(2,startCol:endCol);
    area = area + polyarea(xPoly,yPoly);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [isValid,isParetoOptimal] = analyzeTopology(T,elemsIn)
% Check if the Topological field is valid and/or pareto-optimal
T_SMin = min(T(elemsIn==1)); % Min of topological field inside the domain
T_AMax = max(T(elemsIn==0)); % Max of topological field outside the domain
T_AMin = min(T(elemsIn==0)); % Min of topological field outside the domain
isValid = 0; isParetoOptimal = 0;
if (T_SMin > 0.8*T_AMax) % See paper
    isValid = 1;
end
if (T_AMin+T_SMin >= 0) % See paper
    isParetoOptimal = 1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function plotContour(T,value,fig)
%Use Matlab's built-in contour command to draw the optimal topology.
[nely,nelx] = size(T);
figure(fig);clf;fill([1 nelx nelx 1],[1 1 nely nely],'b'); hold on;
[cntr,h] =contourf(-T,[-value -value]); % the second argument is essential
axis('equal'); axis tight;axis off;
```