Proceedings of the Seventh International Conference on
Artificial Intelligence, Soft Computing, Machine Learning and Optimization,
in Civil, Structural and Environmental Engineering
Edited by: P. Iványi, J. Kruis and B.H.V. Topping
Civil-Comp Conferences, Volume 11, Paper 5.2
Civil-Comp Press, Edinburgh, United Kingdom, 2025
ISSN: 2753-3239, doi: 10.4203/ccc.11.5.2

Forward and Inverse Topology Optimization via Deep Rank-Reduction Autoencoders

I. Ben-Yelun[1], M. El-Fallaki Idrissi[2], J. Mounayer[2], S. Rodríguez[2], F. J. Montans[1] and F. Chinesta[2]

[1] Escuela Técnica Superior de Ingeniería Aeronáutica y del Espacio, Universidad Politécnica de Madrid, Spain
[2] PIMM, ParisTech Arts et Métiers Institute of Technology, France

Abstract

Topology optimization has gained renewed attention thanks to the improvement of 3D printing techniques, especially for metals. While it is currently mainly limited to small-scale production and non-critical components, improvements in printing quality and mass production will make topology optimization a key engineering design technology for enhancing products. However, designing components through topology optimization is computationally expensive, as many finite element simulations are required. This is particularly important when performing inverse analysis. The purpose of this paper is to explore the use of deep rank-reduction autoencoders to accelerate both the forward and the inverse topology optimization schemes, solving also the well-known ill-posedness of the inverse problem.

Keywords: autoencoder, topology optimization, finite element method, machine learning, model order reduction, structural design

1 Introduction

Topology optimization is a mathematical approach to pursue the best distribution of material within a design domain to obtain a target mechanical property (e.g. minimum compliance) with minimum material [1, 2]. This technique is especially useful in the design of mechanical metamaterials, and in particular may be used in the design of functionally graded metamaterials. Topology optimization (TO) typically requires the search for a finite element solution for every topology iteration. For the case of metamaterial-made structures, this implies a topology design at two levels, the component level and the metamaterial level. Therefore, the computational cost is high, especially for large multiscale structures and nonlinear material behavior [3]. Current machine learning approaches can significantly reduce the computational cost [4]. In particular, the use of autoencoders for rank reduction enhances the interpolation on nonlinear manifolds [5].

Whereas the usual engineering problem is to obtain the solution for a given geometry, the principal design problem is to get the best geometry for a given "solution", i.e., the inverse problem. Inverse problems are often ill-posed, meaning that several geometries may be good choices for a given set of boundary conditions. This is especially crucial in the design of metamaterial-made components. A procedure that offers a reasonable solution for inverse problems is also valuable in automating the design objectives. The purpose of this paper is to demonstrate that the use of deep rank-reduction autoencoders, a machine learning technique, facilitates both the forward and inverse problems.

The employed methodology consists of creating three types of models, two of which are low-rank approximations of the geometry and the solution, respectively. The third type of model involves the latent coefficients of the geometry and solution. The purpose of these models is to train the latter model type from the latent variables of the geometry and the solution, navigating forward and backward. The procedure can be used not only for the recovery of trained correspondences between geometry and solution, but also to discover new design possibilities (generative design), thanks to the available general interpolative nature of the machine learning procedure used. Given the generality of the procedure, the proposed approach may result in further significant advances in topology optimization not only of mechanical metamaterial-based components, but also in multiphysics ones.

In the next section, we briefly review the topology optimization (TO) problem and the solution procedure. Thereafter, we recall the ideas behind the Rank Reduction Autoencoders (RRAEs) and the methodology being used. For classical readers of Civil-Comp, we will try to make conceptual parallelisms with mechanical and civil engineering computational procedures.

2 Topology optimization approach

The problems considered will be bi-dimensional. Therefore, the working domain Ω is a rectangular one with n_e $2D$ finite elements and n_d degrees of freedom. The constitutive behavior will be linear at infinitesimal strains. As it is usual in TO, the purpose is to minimize the compliance (i.e. displacements) using a target volume percentage fulfilling the equilibrium equation. In this work, the SIMP procedure will be employed. Given the design vector consistent of the densities of each element $\boldsymbol{\rho} \in \mathbb{R}^{ne}$, the objective ("cost") function to minimize, along the constraints, is (note that the $1/2$ factor is irrelevant here):

$$\underset{\rho}{\text{minimize}} \;\; J(\boldsymbol{\rho}) = \boldsymbol{d}^T(\boldsymbol{\rho})\boldsymbol{K}(\boldsymbol{\rho})\boldsymbol{d}(\boldsymbol{\rho}) \equiv \sum_{e=1}^{ne} J_e(\rho_e) = \sum_{e=1}^{ne} \boldsymbol{d}^T(\rho_e)\boldsymbol{K}_e(\rho_e)\boldsymbol{d}(\rho_e) \quad (1)$$

$$\underset{\rho}{\text{subject to}} \;\; \boldsymbol{d}(\boldsymbol{\rho}) = \boldsymbol{K}^{-1}\boldsymbol{p} \quad (2)$$

$$g(\boldsymbol{d}) = \frac{V(\boldsymbol{\rho})}{V_{max}} - f \leq 0 \quad (3)$$

$$\mathbf{0} \leq \boldsymbol{\rho} \leq \boldsymbol{I}_{n_e} \quad (4)$$

where $\boldsymbol{d} \in \mathbb{R}^{n_{dof}}$ is the global vector of displacements and $\boldsymbol{K}$ is the global stiffness matrix. Each one is obtained as usual following the typical finite elements assembly, e.g. $\boldsymbol{d} = \wedge_{e=1}^{n_e} \boldsymbol{d}_e$, where $\boldsymbol{d}_e$ is the displacement vector of element e; similarly $\boldsymbol{K} = \wedge_{e=1}^{n_e} \boldsymbol{K}_e$, where $\boldsymbol{K}_e$, is the stiffness matrix of element e.

In the SIMP method, it is assumed that the Young modulus of each element is related to the density in the form

$$E_e(\rho : e) = E_{min} + \rho_e^p(E_0 - E_{min}) \quad (5)$$

where E_0 is the reference modulus for the bulk material being used and E_{min} is the minimum allowed value to avoid a rank deficient stiffness matrix, because all elements are mounted regardless of being "void" or ''filled". The parameter p is a penalization parameter (a typical value of 3 is used) to force dichotomic solutions (material/void = $\rho = 1/\rho \simeq 0$). It is inmediate to show that

$$\boldsymbol{K}_e(\rho_e) = E_e(\rho)_e \hat{\boldsymbol{K}}_{e0} \quad (6)$$

with $\hat{\boldsymbol{K}}_{e0} = \boldsymbol{K}_{e0}/E_0$ being the stiffness matrix of element "e" considering a $E_0 = 1$.

To avoid pathologies typical of SIMP methods like checkerboard patterns, it is also customary to apply some smoothing or filtering. In this case, a neighbour weight-based density is used, where the weight depends on the nearby densities, say at a distance R,

$$\hat{\rho}_e = \frac{\sum_{i=1}^{n_e} w_{ei}\rho_i}{\sum_{i=1}^{n_e} w_{ei}} \tag{7}$$

To solve the optimization problem there are several options. The most common ones are gradient-based, e.g. an Optimality Criteria Method [6], or the Method of Moving Asymptotes [7]. To obtain a convenient form of the gradient, assuming that the loads do not depend on the densities, we can take the derivative of the equilibrium equation to get $\boldsymbol{K}(\boldsymbol{\rho})\partial\boldsymbol{d}(\boldsymbol{\rho})/\partial\boldsymbol{\rho} = -\boldsymbol{d}^T(\boldsymbol{\rho})\ \partial\boldsymbol{K}/\partial\boldsymbol{\rho}$. The jacobean is processed element by element, i.e. $\boldsymbol{K_e}(\rho_e)\partial\boldsymbol{d}_e(\rho_e)/\partial\rho_e = -\boldsymbol{d}_e^T(\rho_e)\ \partial\boldsymbol{K}_e/\partial\rho_e = -p(E_0 - E_{min})^{p-1}\boldsymbol{d}_e^T(\rho_e)\ \boldsymbol{K}_{0e}$. Therefore, considering also the replacement of the densities by the smoothed ones

$$\frac{\partial J}{\partial \boldsymbol{\rho}} = \bigwedge_{e=1}^{n_e} \frac{\partial J}{\partial \rho_e} = \bigwedge_{e=1}^{n_e}\sum_{k=1}^{n_e} \frac{\partial J}{\partial \hat{\rho}_k}\frac{\partial \hat{\rho}_k}{\partial \rho_e} = \bigwedge_{e=1}^{n_e}\sum_{k=1}^{n_e} \frac{\partial J}{\partial \hat{\rho}_k}\frac{w_{ke}}{\sum_i w_{ki}} \tag{8}$$

3 Rank-Reduction Autoencoders

The ideas behind Rank-Reduction Autoencoders (RRAEs) and the formulation employed herein can be found in [5]. Additionally, the reader can see a comparison with traditional autoencoders.

The purpose of autoencoders is to learn a reduced dimensionality of the problem, representing data efficiently by capturing the main aspects while neglecting the less relevant ones. There are many versions, but the most used ones are based on multilayer perceptrons (the basis of neural networks). When compared to structural dynamics, autoencoders are similar to a procedure for generating simple models with equivalent structural responses to a larger structure. Autoencoders first "encode" (reduce) the features and then "decode" to reconstruct the response. The best "model" (latent space $\boldsymbol{Y}$) generates the decoded response closest to the encoded one. The learned reduced model (latent space) may be used in similar ways to the simplified dynamics model. The generated mapping is $\tilde{\boldsymbol{X}} = \mathcal{D}(\mathcal{E}(\boldsymbol{X}))$, where the encoding and decoding mapping operations are denoted by $\mathcal{E}(\cdot)$ and $\mathcal{D}(\cdot)$ respectively, where $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$ is the input data with of n samples and $\boldsymbol{x}_i$ is the vector (or matrix) of features. In essence, the autoencoder operation is

$$\underset{D\times n}{\boldsymbol{X}} \xrightarrow[\triangleright]{\mathcal{E}} \underset{L\times n}{\boldsymbol{Y}} \xrightarrow[\triangleleft]{\mathcal{D}} \underset{D\times n}{\tilde{\boldsymbol{X}}} \tag{9}$$

Rank-reduction autoencoders do not only hinge on the reduction of the encoder, but also incorporate a further reduction of the latent space by the use of the Singular Value Decomposition (SVD), such that

$$\underset{L\times n}{\boldsymbol{Y}} = \underset{L\times r}{\boldsymbol{U}} \quad \underset{r\times r}{\boldsymbol{\Sigma}} \quad \underset{r\times n}{\boldsymbol{V}^T} \longrightarrow \underset{L\times n}{\tilde{\boldsymbol{Y}}} = \underset{L\times k_{max}}{\tilde{\boldsymbol{U}}} \quad \underset{k_{max}\times k_{max}}{\tilde{\boldsymbol{\Sigma}}} \quad \underset{k_{max}\times n}{\tilde{\boldsymbol{V}}^T} = \underset{L\times k_{max}}{\tilde{\boldsymbol{U}}} \quad \underset{k_{max}\times n}{\boldsymbol{A}} \tag{10}$$

or

$$\underset{L\times n}{\{\tilde{\boldsymbol{y}}_1, ..., \tilde{\boldsymbol{y}}_n\}} = \underset{L\times k_{max}}{\{\boldsymbol{u}_1, ..., \boldsymbol{u}_{k_{max}}\}} \times \underset{k_{max}\times n}{\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n\}} \tag{11}$$

where r is the number of singular values and k_{max} the number of them chosen in the reduction. The dimension of the latent space is $L << D$, the dimension of the features. The number of used modes is $k_{max} << r$.

The performed operation is ideally similar to choosing only the lower or higher modes of the simplified model above to reproduce the main dynamic characteristics of the main system in structural dynamics.

3.1 Topology Optimization methodology using RRAEs

Let $\boldsymbol{X}$ be the discretized geometry data and $\boldsymbol{S}$ the discretized solutions. If we have an already trained RRAE, we have an encoding for the geometry and an encoding for the solutions; refer to them respectively as $\mathcal{E}^g$ and $\mathcal{E}^s$. For a given geometry sample $\boldsymbol{x}_j$, we can express it in the latent space by the corresponding general $\tilde{\boldsymbol{U}}$–matrix and the specific sample coefficients $\boldsymbol{a}_i$ using

$$\boldsymbol{y}_i = \mathcal{E}^g(\boldsymbol{x}_i) \longrightarrow \boldsymbol{a}_i = \tilde{\boldsymbol{U}}^{\sim 1}\boldsymbol{y}_i \tag{12}$$

where $(\cdot)^{\sim 1}$ is the pseudoinverse (in the case $\tilde{\boldsymbol{U}}$ is square, its pseudoinverse is the transpose because $\tilde{\boldsymbol{U}}$ is orthogonal). The decoding is

$$\tilde{\boldsymbol{x}}_i = \mathcal{D}^g(\tilde{\boldsymbol{y}}_i = \tilde{\boldsymbol{U}}\boldsymbol{a}_i) \tag{13}$$

In a similar way, for the solution, taking similar symbols

$$\boldsymbol{z}_i = \mathcal{E}^g(\boldsymbol{s}_i) \longrightarrow \boldsymbol{b}_i = \tilde{\boldsymbol{V}}^{\sim 1}\boldsymbol{z}_i \tag{14}$$

The decoding is

$$\tilde{\boldsymbol{s}}_i = \mathcal{D}^g(\tilde{\boldsymbol{z}}_i = \tilde{\boldsymbol{V}}\boldsymbol{b}_i) \tag{15}$$

Recall that multi-layer preceptors are used to generate the mappings, as well as the mappings between the coefficients of the geometries and the solutions, namely, $\mathcal{NN}_*^{\lceil} : \boldsymbol{a}_i \rightarrow \boldsymbol{b}_i$ for the direct mapping, and $\mathcal{NN}_*^{\rangle} : \boldsymbol{b}_i \rightarrow \boldsymbol{a}_i$ for the inverse one.

With the trained models, the overall procedure can be seen in Figure 1. The computational pipeline of the direct and inverse problems is as follows.

$$\boldsymbol{y}_i = \mathcal{E}^g(\boldsymbol{x}_i) \longrightarrow \boldsymbol{a}_i = \tilde{\boldsymbol{U}}^T \boldsymbol{y}_i \longrightarrow \boldsymbol{b}_i = \mathcal{NN}^d(\boldsymbol{a}_i) \longrightarrow \tilde{\boldsymbol{s}}_i = \mathcal{D}^s(\tilde{\boldsymbol{V}}\boldsymbol{b}_j) \tag{16}$$

and

$$\boldsymbol{z}_i = \mathcal{E}^s(\boldsymbol{s}_i) \longrightarrow \boldsymbol{b}_i = \tilde{\boldsymbol{V}}^T \boldsymbol{z}_i \longrightarrow \boldsymbol{a}_i = \mathcal{NN}^i(\boldsymbol{b}_i) \longrightarrow \tilde{\boldsymbol{x}}_i = \mathcal{D}^g(\tilde{\boldsymbol{U}}\boldsymbol{a}_j) \tag{17}$$

Note that the solution may be, for example the von Mises stress or any other scalar or vector variable. The cases with a scalar variable are the most challenging ones because the less bijective case.

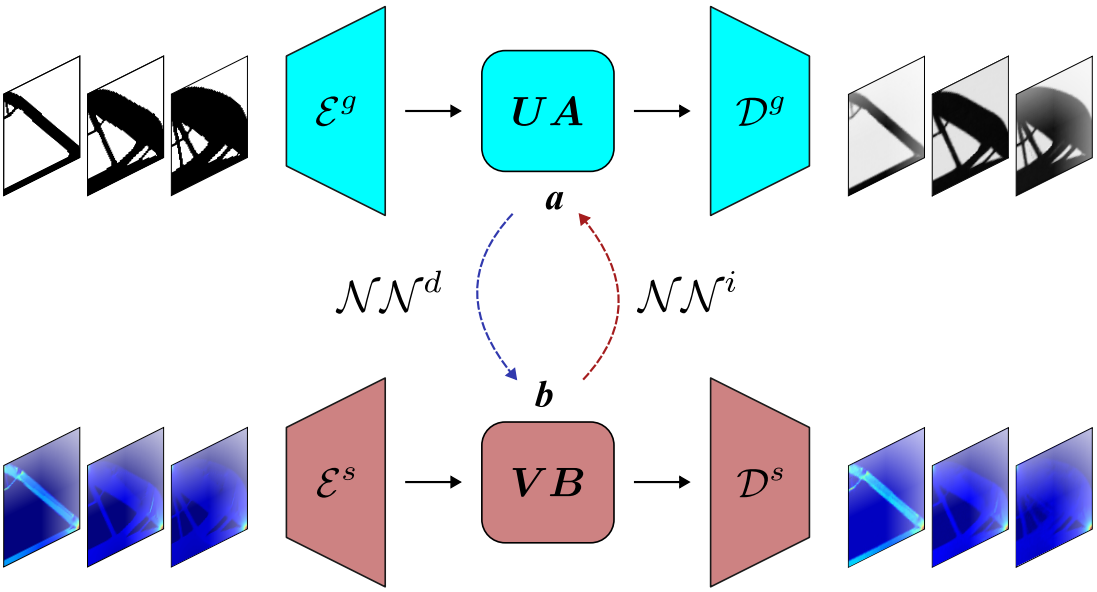

Figure 1: Procedure to address the direct geometry-to-solution and inverse solution-to-geometry problems. Three NN models are separately trained: the geometry $RRAE(\boldsymbol{UA})$, the solution $RRAE(\boldsymbol{VB})$ and the latent cofficient direct and inverse mappings $\mathcal{NN}^d$ and $\mathcal{NN}^i$.

4 Example

For the example, the von Mises stress is selected as the solution variable. The domain is discretized in 80×80 2D four-node finite elements. $E_0 = 1$, $\nu = 0.3$, $E_{min} = 10^{-9}$, $p = 3$, filter radius for the weigths of the smoothed densities of 1.5, 100 volume fraction contraint f values equispaced between 0.1 and 0.1. The Optimality criteria have been chosen, along with the well-known `top88.m` 88-lines Matlab code [8].

Figure 3 shows the training and test results of the CNN-RRAE geometries model. After the training, it is seen that the model is capable of recovering a solution to the

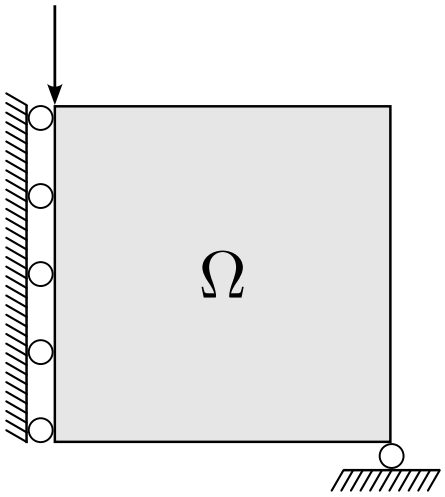

Figure 2: Domain and boundary conditions for the topology optimization problem consisting on designing a half beam

problem for a given volume fraction f without the need to perform finite element simulations.

Figure 4 shows the training and test results of the inverse problem, namely, if the von Mises stress distribution is given, the topology in the original design domain that corresponds to that distribution is obtained. This means that if we desire a stress distribution in a given domain, we could be capable of identifying the structure that would result in such pursued stress distribution.

5 Conclusions

In this work, we have presented the use of deep Rank-Reduction Autoencoders (RRAEs) in topology optimization, both to solve the forward problem (determine the geometry and solution given a design domain and boundary conditions) and the inverse problem (given a solution like a stress distribution in a design domain, obtain the geometry that would result in such a distribution.

RRAEs facilitate the cross-mappings between the latent spaces of the geometry and solution autoencoders via the relation of the singular value decompositions. This brings a promising approach to solving the usually ill-posed inverse problem.

Acknowledgements

IB and FM acknowledges the funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 101007815.

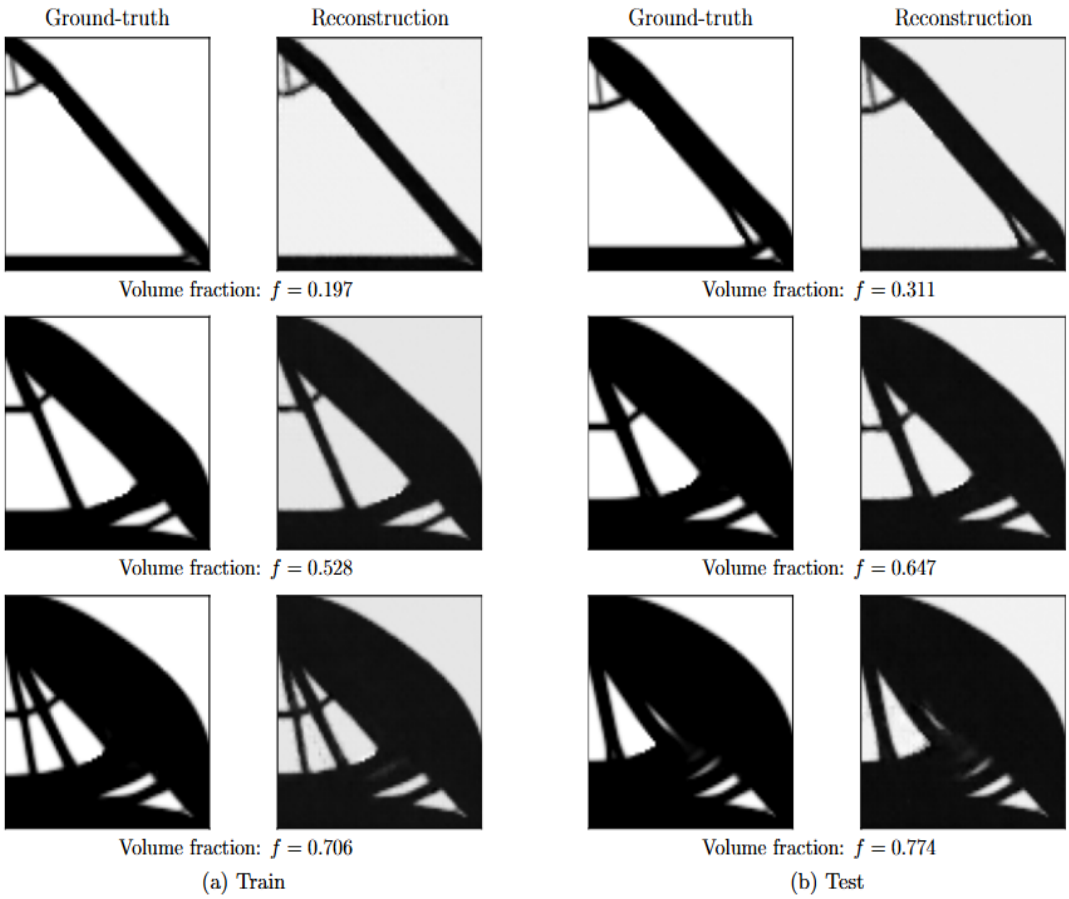

Figure 3: Convolutional Neural Network RRAE geometries model: (a) Training, (b) Testing.

References

[1] M.P. Bendsøe, N. Kikuchi, "Generating optimal topologies in structural design using a homogenization method", Computer Methods in Applied Mechanics and Engineering, 71(2), 197–224, 1988.

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

[3] M.P. Bendsøe, O. Sigmund, Topology Optimization: Theory, Methods, and Applications, Springer, 2004.

[4] F.J. Montáns, F. Chinesta, R. Gómez-Bombarelli, J.N. Kutz, "Data-driven modeling and learning in science and engineering", Comptes Rendus Mécanique, 11(347), 845-855, 2019.

[5] J. Mounayer, S. Rodriguez, C. Ghnatios, C. Farhat, F. Chinesta, "Rank reduction autoencoders-enhancing interpolation on nonlinear manifolds, arXiv preprint arXiv:2405.13980, 2024.

[6] M.P. Bensøe, "Optimization of Structural Topology, Shape, and Material", Springer, 1995.

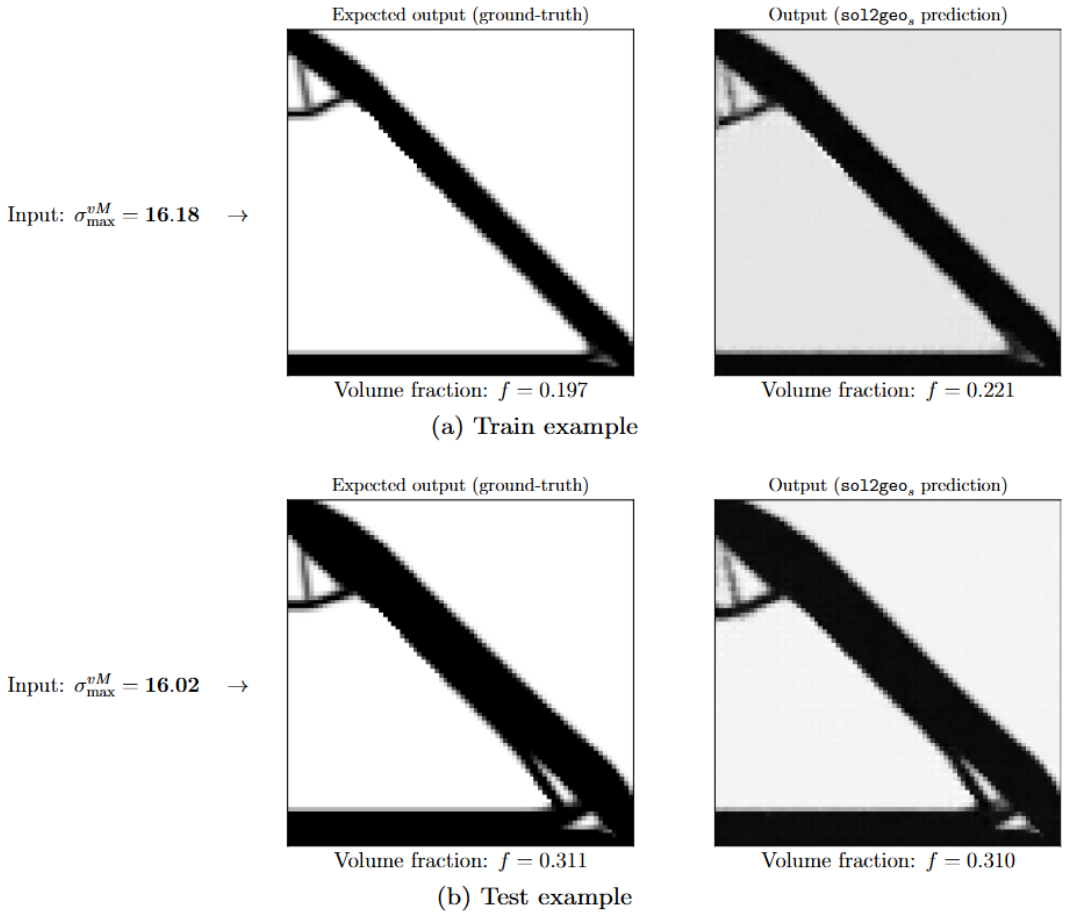

Figure 4: Example of the solution of the inverse problem: given a distribution of von Mises stress in the design domain (80×80 elements), the optimized structure is obtained. (a) Training samples, (b) Testing samples.

[7] K. Svanberg, "The method of moving asymptotes — a nw method for structural optimization", International Journal for Numerical Methods in Engineering, 24(2), 359-373, 1987.

[8] E. Andreassen, A. Clausen, M. Schevenels, B..S. Lazarov, O. Sigmund, "Efficient topology optimization in matlab using 88 lines of code", Structural and Multidisciplinary Optimization, 43, 1-16, 2011.