Engineering with Computers
https://doi.org/10.1007/s00366-023-01904-w

ORIGINAL ARTICLE



DMF-TONN: direct mesh-free topology optimization using neural networks

Aditya Joglekar[1] · **Hongrui Chen**[1] · **Levent Burak Kara**[1]

Received: 2 June 2023 / Accepted: 16 September 2023
© The Author(s) 2023

Abstract
We propose a direct mesh-free method for performing topology optimization by integrating a density field approximation neural network with a displacement field approximation neural network. We show that this direct integration approach can give comparable results to conventional topology optimization techniques, with an added advantage of enabling seamless integration with post-processing software, and a potential of topology optimization with objectives where meshing and Finite Element Analysis (FEA) may be expensive or not suitable. Our approach (DMF-TONN) takes in as inputs the boundary conditions and domain coordinates and finds the optimum density field for minimizing the loss function of compliance and volume fraction constraint violation. The mesh-free nature is enabled by a physics-informed displacement field approximation neural network to solve the linear elasticity partial differential equation and replace the FEA conventionally used for calculating the compliance. We show that using a suitable Fourier Features neural network architecture and hyperparameters, the density field approximation neural network can learn the weights to represent the optimal density field for the given domain and boundary conditions, by directly backpropagating the loss gradient through the displacement field approximation neural network, and unlike prior work there is no requirement of a sensitivity filter, optimality criterion method, or a separate training of density network in each topology optimization iteration.

Keywords Topology optimization · Physics-informed neural network · Implicit neural representations · Mesh-free

1 Introduction

Topology optimization approaches like SIMP (Solid Isotropic Material with Penalization) ([1, 2]) find the optimum structure for a given set of boundary conditions by meshing the design domain and using an iterative process where each iteration involves an FEA calculation for computing the objectives such as compliance. Therefore, removing these iterations completely or creating a new class of solvers with a reparameterization of the design variables in this optimization problem is highly desirable. Advances in neural networks, both in learning from large amounts of data and in learning implicit representations of complex signals show great promise to bring about this transformation, and hence many new approaches trying to utilize neural networks for topology optimization have been recently developed.

Data-driven approaches perform instant optimal topology generation during inference time. However, they require a large training database generation, a long training time and face generalization issues. Online training approaches use a neural network to represent the density field of designs for an alternative parameterization. They do not face any generalization issues. However, meshing and FEA is still required.

One of the first online training topology optimization approaches, TOuNN, was proposed by Chandrasekhar and Suresh [3]. The neural network takes in as inputs the domain coordinates and outputs the density value at each of these coordinates. The loss function consists of the compliance and volume fraction constraint violation. This loss gradient is backpropagated and used for updating the weights of the neural network such that it learns the optimal density distribution for minimizing the loss. The compliance for the density field is calculated as in the traditional SIMP method using FEA. For removing the meshing requirement of FEA

Aditya Joglekar and Hongrui Chen have contributed equally to this work.

✉ Levent Burak Kara
lkara@cmu.edu

[1] Department of Mechanical Engineering, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213, USA

Published online: 28 October 2023

and creating a new class of solvers for various partial differential equations (PDEs) in computational mechanics, there have been recent advances and promising results in using physics-informed neural networks (PINNs). Samaniego et al. [4] propose an energy approach for solving the linear elasticity PDEs. The displacement field is parameterized by a neural network which takes as input the domain coordinates and outputs the displacements in each direction at each of these coordinates. The loss function consists of the potential energy, which when minimized, will give static equilibrium displacements.

Though the computational time for the neural network PDE approximation frameworks is worse than the current state of the art FEA solvers, there are several potential advantages of this approach, including the mesh-free nature and an easy modelling of non-linear PDEs. Incorporation of these neural network PDE approximation frameworks in online training topology optimization enables mesh-free topology optimization and a new class of solvers for this complex inverse design problem.

Zehnder et al. [5] were the first to propose such a mesh-free framework for topology optimization with compliance as an objective, where in addition to the density field, the displacement field is also parameterized using a neural network. However, they conclude that connecting the two neural networks directly leads to bad local minima. Hence, they propose using the optimality criterion method and sensitivity filtering for calculating target densities. As such, the density neural network needs to be trained for estimating these target densities in every topology optimization iteration.

In this work, we show that using directly connected displacement field estimation and density field estimation neural networks is indeed an effective approach for mesh-free topology optimization. In particular, we argue that using just one gradient descent step of the density network in each topology optimization iteration without any sensitivity or density filtering leads to comparable results to conventional topology optimization. Moreover, after the initial run of the displacement network, we significantly reduce the number of iterations in each topology optimization iteration. We show that transfer learning applies here and in this high dimensional and non-convex optimization problem setting, approximate loss and gradients can work well.

We devise DMF-TONN as a method not for replacing SIMP, but for adding to and improving the current class of mesh-free solvers for topology optimization, using the advancements in neural networks. We use Fourier Features and a fully connected layer as the architecture of both our neural networks. We verify the effectiveness of our approach with case studies with different boundary conditions and volume fractions. The implementation of this work is available at: https://github.com/AdityaJoglekar/DMF-TONN.

Our main contributions include:

- A framework for achieving mesh-free topology optimization by directly combining displacement field approximation and density field approximation neural networks, with just one gradient descent step of density approximation network in each topology optimization iteration.
- Extensive analysis and comparison with SIMP for 3D examples, unlike prior related works which focus on 2D examples
- A method with an improved computational time compared to prior related works.

2 Literature review

Topology optimization: Bendsøe and Kikuchi [6] introduced the homogenization approach for topology optimization. The SIMP method ([1, 2]) considers the relative material density in each element of the Finite Element (FE) mesh as design variables, allowing for a simpler interpretation and optimized designs with more clearly defined features. Other common approaches to topology optimization include the level-set method ([7, 8]) and evolutionary algorithms. Improving the optimization results and speed of these approaches using neural networks has seen a lot of development recently and Woldseth et al. [9] provide an extensive overview on this topic.

Neural networks for solving PDEs: Driven by a neural network's ability to approximate functions, there have been several recent works proposing novel solvers for PDEs. Raissi et al. [10] propose PINNs, neural networks that are trained to solve supervised learning tasks while respecting the laws of physics described by general nonlinear partial differential equations. Samaniego et al. [4] propose an approach using neural networks which does not require labelled data points, and just uses domain coordinates and boundary conditions as input to solve computational mechanics PDEs. Nguyen-Thanh et al. [11] develop a deep energy method for finite deformation hyperelasticity. Sitzmann et al. [12] leverage periodic activation functions for implicit neural representations and demonstrate that these networks are ideally suited for representing complex natural signals and their derivatives and solving PDEs. Tancik et al. [13] show that passing input points through a simple Fourier feature mapping enables a multilayer perceptron to learn high-frequency functions in low-dimensional problem domains. We utilize this concept of Fourier Feature mapping for finding good approximations of the displacement field and density field in the low-dimensional coordinate domain.

Neural networks for topology optimization: Several data-driven methods for topology optimization using neural networks [14–20] have been proposed. In this review we focus on the online training topology optimization methods, i.e. those methods which do not use any prior

data, rather train a neural network in a self-supervised manner for learning the optimal density distribution and topology. Chandrasekhar and Suresh [3] explore an online approach where the density field is parameterized using a neural network. Fourier projection based neural network for length scale control ([21]) and application for multi-material topology optimization ([22]) has also been explored. Deng and To [23] propose topology optimization with Deep Representation Learning, with a similar concept of re-parametrization, and demonstrate the effectiveness of their method on compliance minimization and stress-constrained problems. Zhang et al. [24] also employ a design reparameterization approach and improve accuracy of physical response and sensitivity analysis by proposing the BA-MFEA as a substitution for conventional FEA. Hoyer et al. [25] use CNNs for density parameterization and directly enforce the constraints in each iteration, reducing the loss function to compliance only. Chen et al. [26] propose a neural network based approach to topology optimization that aims to reduce the use of support structures in additive manufacturing. Chen et al. [27] demonstrate that by using a prior initial field on the unoptimized domain, the efficiency of neural network based topology optimization can be improved. He et al. [28] and Jeong et al. [29] approximate displacement fields using PINNs, but a continuous density field is not learned and the frameworks are not mesh-free. Lu et al. [30] demonstrate the effectiveness of hard constraints over soft constraints for solving PDEs in various topology optimization problems. Zehnder et al. [5] effectively leverage neural representations in the context of mesh-free topology optimization and use multilayer perceptrons to parameterize both the density and displacement fields. Mai et al. [31] develop a similar approach for optimum design of truss structures. We show that unlike in [5], sensitivity filtering, optimality criterion method and separate training of density network in each topology optimization epoch is not necessary for mesh-free topology optimization using neural networks.

3 Proposed method

We parameterize the displacement field as well as the density field using neural networks and integrate them as shown in Fig. 1.

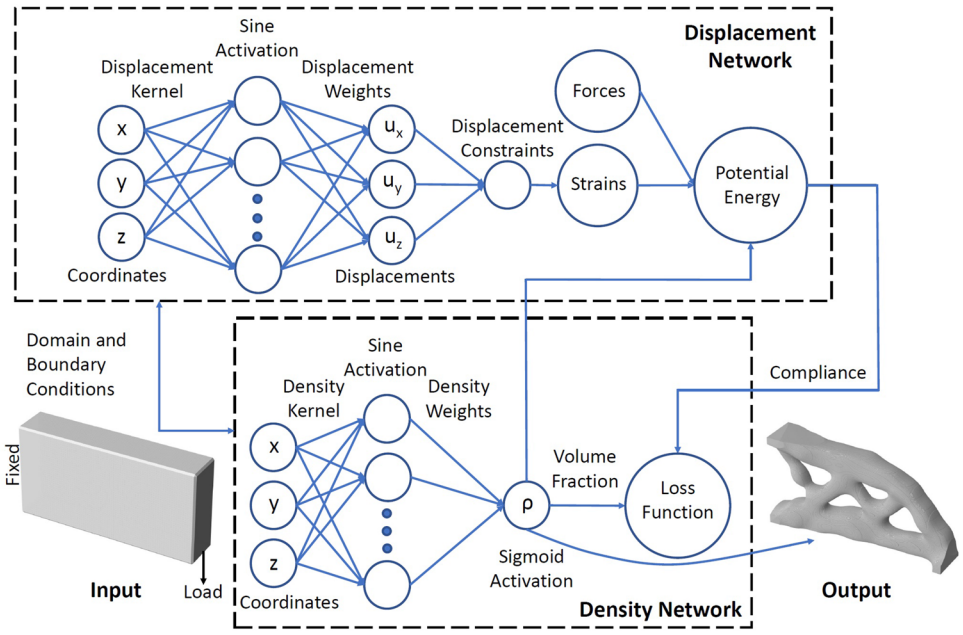


Fig. 1 Our proposed framework. Each topology optimization iteration consists of: (1) training the displacement network with current density field, randomly sampled domain coordinates in each of its iterations and boundary conditions to obtain static equilibrium displacements. (2) Sampling domain coordinates and performing a forward pass through the density network to obtain current topology output, and displacement network to obtain current compliance, which are passed to the density network loss function (3) back-propagating density network loss and performing a gradient descent step on density network weights

Algorithm 1 DMF-TONN

Initialize neural networks: $Den_{W_{den}}$, $Disp_{W_{disp}}$
Initialize Adam optimizers: Opt_{den}, Opt_{disp}
Initialize domain ρ_{init}
for $n_{initdisp}$ iterations **do**
    Sample domain coordinates X_{disp}
    $u_{temp} \leftarrow Disp_{W_{disp}}(X_{disp})$
    $W_{disp} \leftarrow Opt_{disp}.step(W_{disp}, \frac{\partial L_{disp}(u_{temp},\rho_{init})}{\partial W_{disp}})$
end for
for n_{opt} iterations **do**
    for n_{disp} iterations **do**
        Sample domain coordinates X_{disp}
        $\rho_{temp} \leftarrow Den_{W_{den}}(X_{disp})$
        $u_{temp} \leftarrow Disp_{W_{disp}}(X_{disp})$
        $W_{disp} \leftarrow Opt_{disp}.step(W_{disp}, \frac{\partial L_{disp}(u_{temp},\rho_{temp})}{\partial W_{disp}})$
    end for
    Sample domain coordinates X_{den}
    $\rho \leftarrow Den_{W_{den}}(X_{den})$
    $u \leftarrow Disp_{W_{disp}}(X_{den})$
    $c \leftarrow L_{disp}(u, \rho) + EW$
    $W_{den} \leftarrow Opt_{den}.step(W_{den}, \frac{\partial L_{den}(\rho,c)}{\partial W_{den}})$
end for

3.1 Density neural network

The density neural network $\text{Den}(\mathbf{X}_{\text{den}})$ can be represented as follows:

$$\text{Den}(\mathbf{X}_{\text{den}}) = \sigma(\sin(\mathbf{X}_{\text{den}}\mathbf{K}_{\text{den}} + \mathbf{b})\mathbf{W}_{\text{den}}) \tag{1}$$

The input is a batch of domain coordinates $\mathbf{X}_{\text{den(batchsize×3)}}$. We use the domain center as the origin for the coordinates, and the coordinates are normalized with the longest dimension coordinates ranging from −0.5 to 0.5. We use the concept proposed in Tancik et al. [13] and a neural network architecture similar to the one used in Chandrasekhar and Suresh [21]. The first layer weights (kernel $\mathbf{K}_{\text{den(3×kernelsize)}}$) are fixed, which create Fourier features after passing through the sine activation. We add a bias term $\mathbf{b}$ consisting of ones before applying the sine activation to break off the symmetry about the origin. The kernel is created using a grid of number of dimensions same as the number of domain dimensions, and then reshaping the grid coordinates to the matrix $\mathbf{K}_{\text{den(3×kernelsize)}}$. The grid size in each dimension dictates how good it can represent topological features, and the grid's range of values control the frequency of the output topology, with higher ranges of values giving a topology with more intricate features. Note that this grid is not a mesh structure, and consists solely of coordinates. We find that making the kernel trainable can slightly improve compliance. However, we keep it fixed for all the experiments in this paper assuming that the slight increase in performance may not be preferable to the large number of trainable weights. The next layer weights ($\mathbf{W}_{\text{den(kernelsize×1)}}$) are trainable and the output is passed through a sigmoid activation (σ). This ensures output values are between 0 and 1, which represent the density, for each of the coordinates in the input batch. We use Adam (Kingma and Ba [32]) as the optimizer, with a learning rate of 2.0×10^{-3} for all the experiments.

3.2 Displacement neural network

We use a neural network similar to the density neural network for approximating the displacement field. The physics-informed components shown in Samaniego et al. [4] are then added on the displacement output by this neural network $\text{Disp}(\mathbf{X}_{\text{disp}})$. This can be represented as follows:

$$\text{Disp}(\mathbf{X}_{\text{disp}}) = \sin(\mathbf{X}_{\text{disp}}\mathbf{K}_{\text{disp}} + \mathbf{b})\mathbf{W}_{\text{disp}} \tag{2}$$

In each displacement network iteration, we randomly sample domain coordinates $\mathbf{X}_{\text{disp}}$. The frequency determined by $\mathbf{K}_{\text{disp}}$ should be greater than or equal to the frequency determined by $\mathbf{K}_{\text{den}}$. This is due to the fact that if the displacement network in unable to capture and pass fine changes in displacement to the density network, and the density network is attempting to create very fine features, incorrect features are created and disconnections are observed in the final topology. For all our experiments, we use the same frequencies

and grid sizes for $\mathbf{K}_{\text{disp}}$ and $\mathbf{K}_{\text{den}}$ and find this setting works well. Multiplying the Fourier features with $\mathbf{W}_{\text{disp(kernelsize}\times 3)}$ gives the displacements in each direction.

3.2.1 Displacement constraints

Boundary conditions on displacements such as fixed sides, are implemented as hard constraints. The output of $\text{Disp}(\mathbf{X}_{\text{disp}})$ is multiplied with a differentiable function that is 0 at the fixed boundary and 1 elsewhere. We use the exponential function for this. For example, with a cuboidal domain that has the side with all x coordinates equal to zero, fixed (zero displacement in all three directions), such as in Fig. 2a, the hard constraint function takes the form $2(\frac{1}{(1+\exp(-m(c_x+0.5)))} - 0.5)$ where c_x is all the x coordinates in the domain, and m is a constant which dictates the slope of this function. We find empirically $m = 20$ works well and use it for all our experiments. For multiple fixed sides, the displacements output by the neural network are multiplied by the functions for each fixed side.

3.2.2 Minimum potential energy loss function

The principle of minimum potential energy is used for approximating the displacement field as proposed in Samaniego et al. [4]. The neural network learns the weights that output the displacements that minimize the potential energy, and thus learns to output static equilibrium displacements. With Monte-Carlo sampling, the loss function of the displacement neural network, $L_{\text{disp}} = \text{Potential Energy}$, for 3d problems, is defined as follows:

$$L_{\text{disp}} = \text{ISE} - \text{EW} \tag{3}$$

$$\text{ISE} = \frac{V}{N}\sum_{i}^{N}\left(\mu\varepsilon_i \;:\; \varepsilon_i + \frac{\lambda(\text{trace}(\varepsilon_i))^2}{2}\right) \tag{4}$$

$$\text{EW} = \frac{A}{N_b}\sum_{i}^{N_b} Tu_i \tag{5}$$

where

ISE = Internal Strain Energy
EW = External Work
V = domain volume
N = number of sample points in domain
$\mu = \frac{E}{2(1+\nu)}$, $\lambda = \frac{E\nu}{((1+\nu)(1-2\nu))}$
E = Young's Modulus
ν = Poisson's ratio
ϵ_i = strain matrix at *ith* point
A = area on which traction is applied
N_b = number of sample points on boundary
T = traction
u_i = displacement at *ith* point

The symbol ' : ' indicates element wise multiplication, followed by a summation.

The strains are calculated using automatic differentiation in TensorFlow (Abadi et al. [33]). We use Adam as the optimizer, with a learning rate of 5.0×10^{-6} for all the experiments.

3.3 Integration of density and displacement neural networks

A topology optimization epoch starts by training the displacement network with randomly sampled coordinates, the corresponding current topology (found by a forward pass through the density network) and the boundary conditions. The conventional SIMP method interpolation ($E = E_{\text{material}}(\rho^3)$, where ρ is the density) is used for obtaining the Young's modulus E at each of the randomly sampled domain points in each displacement network iteration. Then, with randomly sampled coordinates, a forward pass is performed through the density network and the displacement network to get the current topology and current compliance (Internal Strain Energy) respectively, which are passed to the density network loss function. For the volume fraction constraint violation term in this loss function, we use a constant grid of domain points instead of randomly sampled points in each iteration to avoid fluctuations in the volume fraction in each iteration and thus helping stabilize the optimization process. The density network loss function is defined as follows:

$$L_{\text{den}} = \frac{c}{c_0} + \alpha\left(\frac{v}{V^*} - 1\right)^2 \tag{6}$$

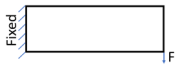
where,

c = compliance
c_0 = initial compliance
v = volume fraction
V^* = target volume fraction
α = penalty constant

The compliance (c) is a function of the densities (ρ) and displacements (u), where the displacements are also dependent on the densities. As shown in Zehnder et al. [5], the total gradient of the compliance with respect to the densities is given by $\frac{dC}{d\rho} = \frac{\partial C}{\partial \rho} + \frac{\partial C}{\partial u}\frac{du}{d\rho} = -\frac{\partial C}{\partial \rho}$, which needs to be incorporated while connecting the two neural networks and backpropagating the loss to the density network weights.

As shown in Algorithm 1, during each topology optimization iteration, the density network weights are updated once with a gradient descent step in Adam. Before starting the topology optimization, the displacement network is run

Table 1 Comparison for 2D cantilever beam problem with target volume fraction = 0.5

			
Boundary conditions	SIMP	NTopo	DMF-TONN
Convergence compliance	2.86×10^{-4}	2.54×10^{-4}	2.73×10^{-4}
Volume fraction achieved	0.50	0.50	0.50
Binary structure compliance	2.66×10^{-4}	2.40×10^{-4}	2.52×10^{-4}
Time	69 s	1465s	277 s

with the initial domain as the topology, for $n_{\text{dispinit}} = 1000$ iterations for converging to static equilibrium displacements. Then in each topology optimization iteration, utilizing the concept of transfer learning as the topology does not change too drastically, we run the displacement network only for $n_{\text{disp}} = 20$ iterations. We determined the values of the n_{dispinit} and n_{disp} variables empirically to give the best results in the cases presented in the paper. Though we have to increase the topology optimization iterations, this reduction of 50 times in displacement network iterations significantly reduces the computational time without compromising the compliance of the results.

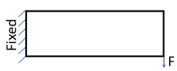
We run all of our experiments on a machine with 12th Gen Intel(R) Core(TM) i7-12700 2.10 GHz processor, 16 GB of RAM and Nvidia GeForce RTX 3060 GPU.

4 Results

We first compare our results with a conventional SIMP topology optimization (Andreassen et al. [34]) and NTopo (Zehnder et al. [5]) for a 2D cantilever beam problem. Then, for a 3D cantilever problem, we present the initial convergence history of the displacement network, and the convergence history of the compliance. Subsequently, we perform a case study for the 3D cantilever beam problem over varying volume fractions and load locations, comparing our results with a conventional 3D SIMP topology optimizer (Liu and Tovar [35]), and with a method using the Fourier Features neural network for density representation and FEA for compliance calculation (similar to Chandrasekhar and Suresh [21]). Then, we present an example showcasing the trade-offs in DMF-TONN and SIMP in terms of the compliance and computational time. Lastly, we validate our approach on some additional examples. We also present some examples using the same constant grid as SIMP for DMF-TONN instead of domain points that are randomly sampled in each displacement network iteration and density network iteration for compliance calculation using the trained displacement network.

The output of our network represents an implicit function of the spatial densities. We use the marching cubes algorithm for generating the renders of the results. It should be noted that with the DMF-TONN, one can sample infinitely many points within the domain, as a continuous and differentiable function has been learned by the density network. In this paper, we use 20 times the number of samples of the FEA grid size in each direction for the DMF-TONN figures for each example. On all solutions obtained by our approach, we run FEA for calculating the final compliance and use this to compare against the compliance obtained by SIMP (using the

Table 2 Comparison for 2D cantilever beam problem with target volume fraction = 0.3

			
Boundary conditions	SIMP	NTopo	DMF-TONN
Convergence compliance	4.94×10^{-4}	4.23×10^{-4}	4.76×10^{-4}
Volume fraction achieved	0.30	0.30	0.30
Binary structure compliance	4.55×10^{-4}	3.79×10^{-4}	4.01×10^{-4}
Time	124 s	1461s	275 s

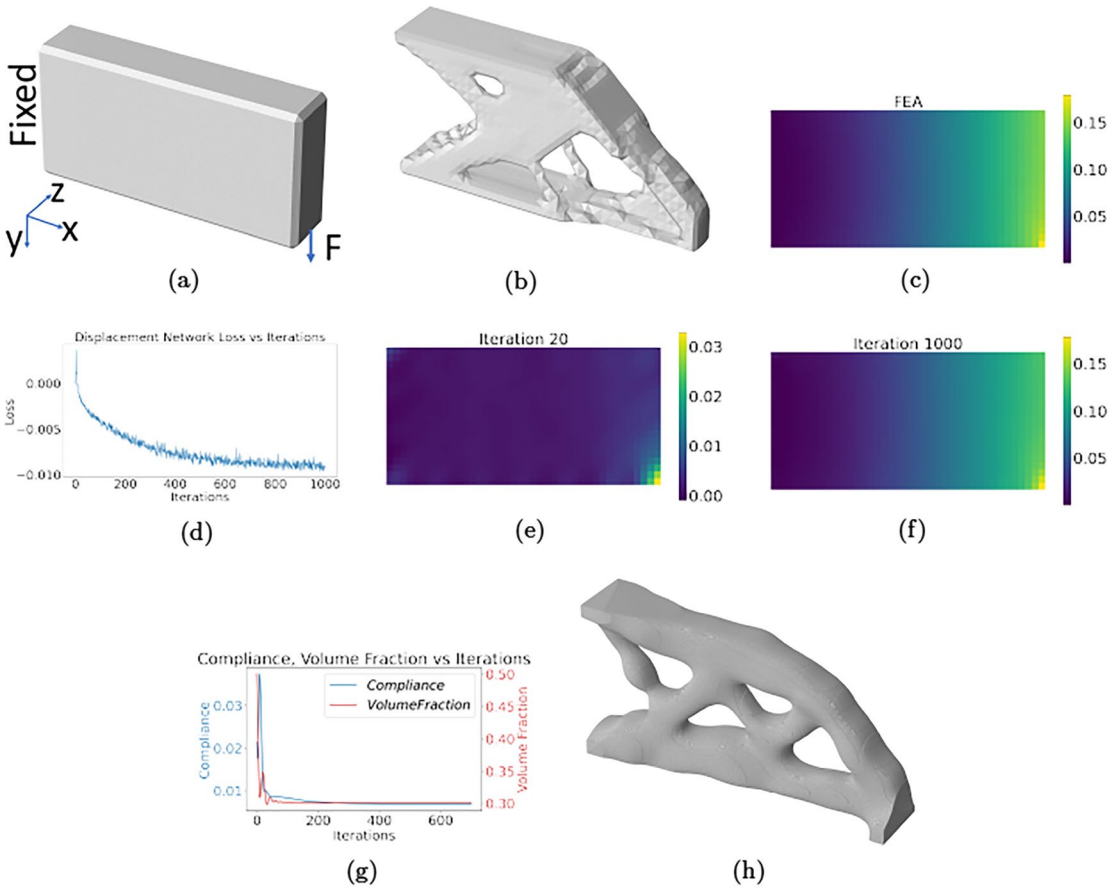


Fig. 2 Convergence history of DMF-TONN and result comparison with SIMP for 3D cantilever beam problem with target volume fraction = 0.3, Key: [compliance $\times 10^{-3}$, volume fraction]. **a** Boundary conditions, **b** Top3D (SIPM) result, [7.49, 0.30], **c** max disp = 0.179 mm, **d** initial PINN convergence, **e** max disp = 0.033 mm, **f** max disp = 0.178 mm, **g** compliance and volume fraction convergence, **h** DMF-TONN result, [6.76, 0.30]

same FE solver) for consistency. We ensure the degrees of freedom available are always more for SIMP than for DMF-TONN. Moreover, in SIMP, we use a density filter with a radius 1.5 times the mesh element size in each of the presented examples, ensuring thin features and lower compliances are possible, and there is no compromise in the SIMP results.

4.1 Comparison of DMF-TONN, SIMP and Ntopo for 2D cantilever beam example

We compare the compliances and computational times for a 2D cantilever beam example in Tables 1 and 2. We run the SIMP code (Andreassen et al. [34]) with the default convergence criterion, and run NTopo for 200 iterations as shown in (Zehnder et al. [5]) for a similar 2D cantilever beam example. We run our method for $n_{\text{opt}} = 2000$ topology optimization iterations (determined empirically for 2D problems), with $n_{\text{disp}} = 20$ iterations of the displacement network in each of these topology optimization iterations. We also compare the binary (0 and 1 density values) structure compliance (ensuring the volume fraction remains the same after thresholding) which provides the actual compliance if the optimized structures were to be used in practice. We observe that though our method is slower than SIMP, it results in a mesh-free optimization with a better compliance than SIMP and a faster computational time than NTopo.

4.2 Analysis of a 3D cantilever beam example and transfer learning

In Fig. 2, we present the convergence history of DMF-TONN and a comparison with 3D SIMP topology optimization for an example with boundary conditions shown in Fig. 2a. We use a $40 \times 20 \times 8$ grid for the SIMP FEA, which gives 6400 design variables for the topology optimization, and $6400 \times 3 = 19200$ degrees of freedom (DOF) for the FEA. We use a lesser DOF model, with the number of trainable

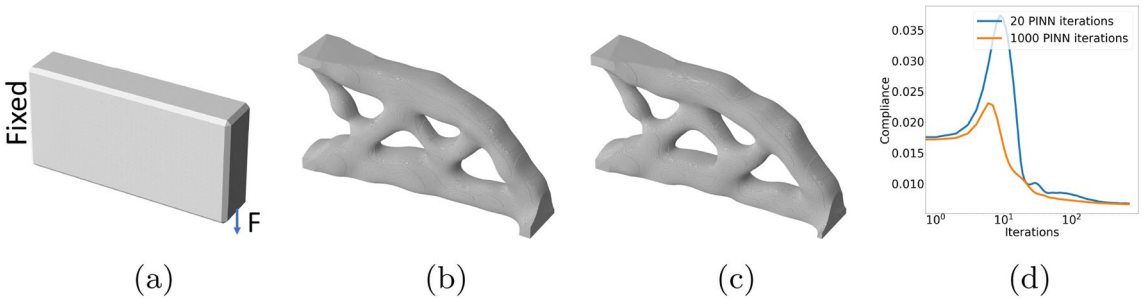


Fig. 3 Transfer learning analysis, Key: [compliance $\times 10^{-3}$, time, volume fraction]. **a** Boundary conditions, target volume fraction = 0.30, **b** 20 PINN iterations, [6.76, 588 s, 0.30], **c** 1000 PINN iteration, [6.63, 249.2 s, 0.30], **d** compliance convergence history comparison

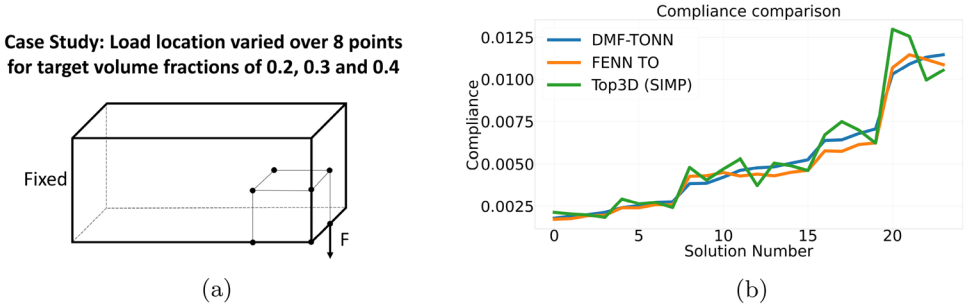
weights in both our density and displacement networks being 4096 each. We use an initial topology consisting of uniform densities of 0.5 ($\rho_{\text{init}(40\times20\times8)} = \mathbf{0.5}$) as an input for initial training of the displacement network (PINN) and Fig. 2d shows the convergence history. Figures 2e, 2f show the displacement field in the y direction at the cross-section of the domain where the force is applied and the maximum displacement value at different iterations of the initial run of the displacement network. Figure 2c shows the FEA displacement field at this cross section. We see that at the 1000^{th} iteration, the displacement network can learn a very good approximation of the FEA displacement field.

In each of the $n_{\text{opt}} = 700$ topology optimization iterations (determined empirically for 3D problems), we use only $n_{\text{disp}} = 20$ displacement network iterations and show that this is adequate for achieving results (2h) similar to SIMP (2b). Figure 2g shows the convergence history for the topology optimization, with FEA compliance plotted for consistency with SIMP. Maximum decrease occurs up till 300 iterations, but we continue up till 700 iterations for a more refined and better final topology. We observe similar trends in all other examples and hence use $n_{\text{opt}} = 700$ for each of the presented examples in this paper. Our approach takes 588 s compared to 227 s for SIMP, but achieves mesh-free topology optimization and a better compliance than SIMP for this example.

The concept of transfer learning can be utilized with the displacement network, noting the fact that the change in topologies in each topology optimization iteration is not drastic and thus the neural network weights learned in the prior topology optimization iteration can be a good initialization for learning the weights corresponding to the static equilibrium displacement values in the next topology optimization iteration. In Fig. 3, we compare the results when using $n_{\text{disp}} = 20$ and $n_{\text{disp}} = 1000$ displacement network iterations in each topology optimization iteration. We observe that once the displacement network is trained initially for $n_{\text{initdisp}} = 1000$ iterations before starting the topology optimization, $n_{\text{disp}} = 20$ gives similar compliance results to when using $n_{\text{disp}} = 1000$, but with a very low computational time, thus showcasing the utility of a good initialization.

Even though Fig. 3d shows that a higher n_{disp} tends to give a better objective value in lesser n_{dens} (the orange curve is below the blue curve), considering the computational time n_{disp} that is lower (up to a limit of 20) is better and we use $n_{\text{disp}} = 20$ for all the examples presented in this paper. If n_{disp} is chosen to be very low (less than 20), then good convergence does not occur, as the displacement network has

Fig. 4 Case study of 3D cantilever beam problem. **a** Case study boundary conditions, **b** comparison of compliance of DMF-TONN, FENN TO and Top3D (SIMP)

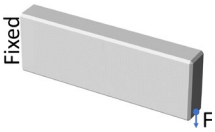
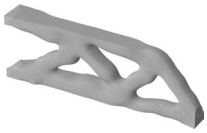
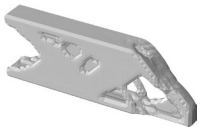


not learned much about the current topology and hence the optimization loss gradients are very inaccurate.

4.3 Case study of 3D cantilever beam problem

In Fig. 4 we compare our fully mesh-free approach with a Finite Element Analysis based Neural Network Topology Optimization (FENN TO) approach (i.e. using a Fourier Features neural network for representing density field and FEA for compliance calculation) and with the conventional SIMP approach. We vary the load location and target volume fraction for the 3D cantilever beam problem. All approaches are run for 700 iterations. We show the plots of the compliance of all three approaches for all these boundary conditions in Fig. 4b where the x-axis contains the discrete boundary conditions and y-axis represents the compliance for each of these boundary conditions. Our fully mesh-free approach achieves similar compliance values to the existing SIMP and FENN TO approaches for all the boundary conditions in this

Table 3 Long cantilever beam with bottom load with target volume fraction = 0.3

 Boundary conditions	 DMF-TONN	 SIMP ($60 \times 20 \times 8$ grid)	 SIMP ($90 \times 30 \times 12$ grid)
Displacement calculation degrees of freedom	3456	28800	97200
No. of optimization design variables	3456	9600	32400
Convergence compliance	2.40×10^{-2}	2.65×10^{-2}	2.38×10^{-2}
Volume fraction achieved	0.30	0.30	0.30
Computational time	517 s	99 s	807 s

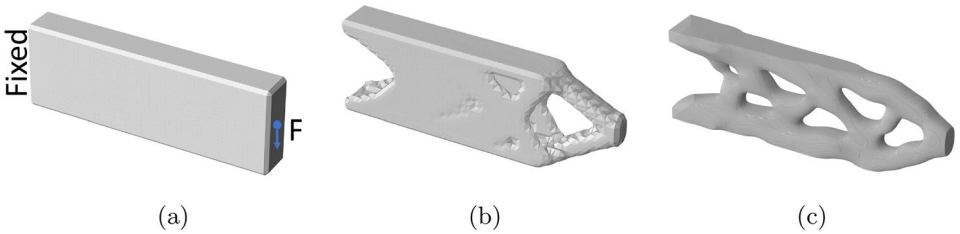


Fig. 5 Long cantilever beam with center load with target volume fraction = 0.3, Key: [compliance $\times 10^{-2}$, volume fraction]. **a** Boundary conditions, **b** Top3D (SIMP), [2.56, 0.30], **c** DMF-TONN, [2.29, 0.30]

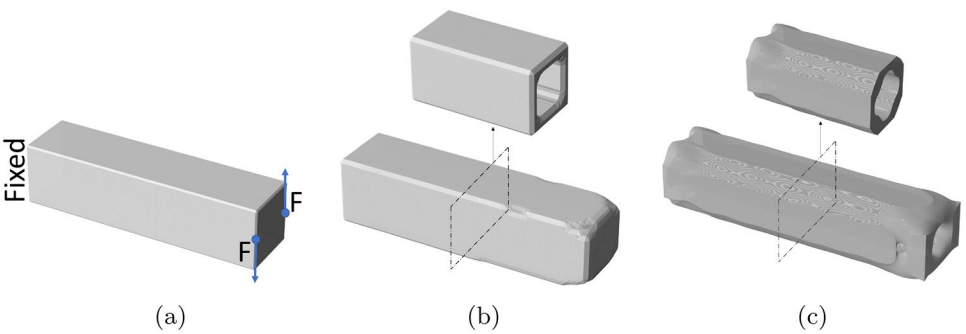


Fig. 6 Long cantilever beam with two loads with target volume fraction = 0.5, Key: [compliance $\times 10^{-3}$, volume fraction]. **a** Boundary conditions, **b** Top3D (SIMP), [8.22, 0.50], **c** DMF-TONN, [8.78, 0.50]

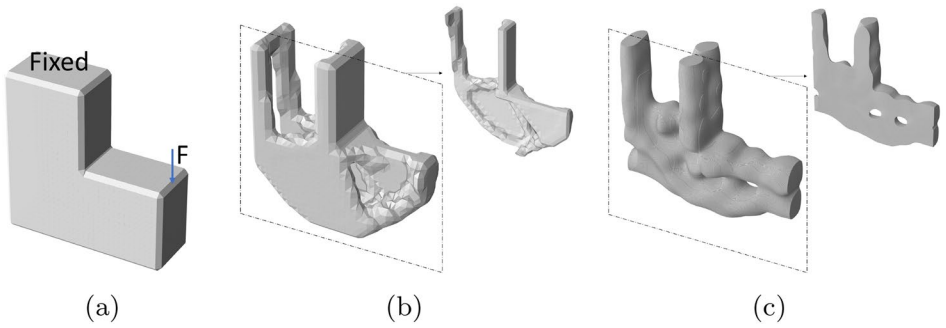


Fig. 7 L-Bracket with target volume fraction = 0.2, Key: [compliance $\times 10^{-3}$, volume fraction]. **a** Boundary conditions, **b** Top3D (SIMP), [7.47, 0.20], **c** DMF-TONN, [6.21, 0.20]

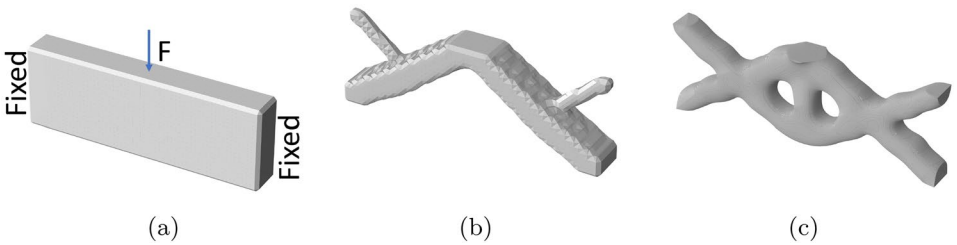


Fig. 8 Bridge with target volume fraction = 0.3, Key: [compliance $\times 10^{-3}$, volume fraction]. **a** Boundary conditions, **b** Top3D (SIMP), [1.99, 0.30], **c** DMF-TONN, [2.00, 0.30]

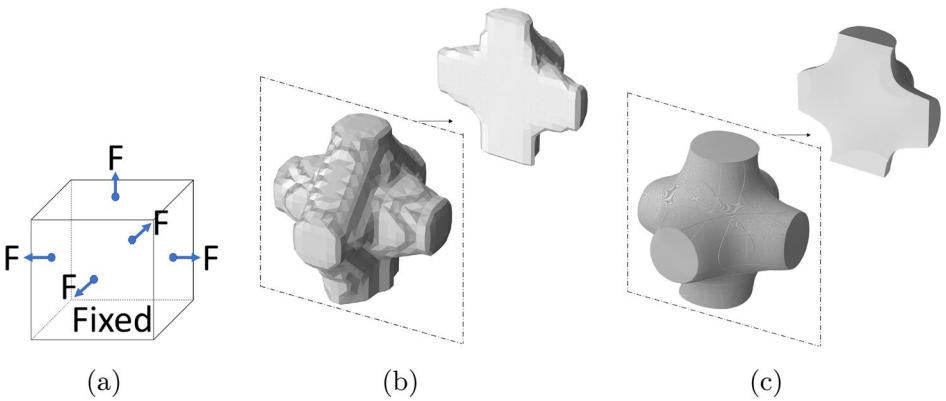


Fig. 9 5 loads with target volume fraction = 0.3, Key: [compliance $\times 10^{-3}$, volume fraction]. **a** Boundary conditions, **b** Top2D (SIMP), [1.62, 0.30], **c** DMF-TONN, [1.63, 0.30]

case study. The total computational times for running all the examples in this case study are 236 min, 96 min and 124 min for DMF-TONN, SIMP and FENN TO respectively.

4.4 Trade-off analysis of DMF-TONN and SIMP

In Table 3, we present the results for a right bottom end loaded cantilever beam with the ratio of 3 for the lengths of sides in the x and y directions (the orientation of the axes is the same as in Fig. 2a). For the Top3D (SIMP) method, we use their stated convergence criteria of 200 maximum iterations and 0.01 as tolerance of change in topology. Though the degrees of freedom are lesser for DMF-TONN, it still achieves a better compliance than SIMP with a grid of $60 \times 20 \times 8$. However, the computational time is much higher for DMF-TONN. Now, we increase the grid size of SIMP to $90 \times 30 \times 12$ for achieving a better compliance than DMF-TONN. However, as seen in Table 3, due to the 1.5 times increase in grid size in each direction, there was an exponential increase in computational time for the SIMP method and the computational time of DMF-TONN is lesser than the fine mesh SIMP. This showcases one of the advantages of the mesh-free nature of DMF-TONN, presenting interesting opportunities for tradeoffs to be explored in future research.

4.5 Additional examples

In Fig. 5, the load acts at the center of the right side of a long cantilever beam. The compliance values obtained by DMF-TONN for these boundary conditions are better than those obtained by Top3D (SIMP) ($60 \times 20 \times 8$ grid). In Fig. 6, the boundary conditions include two loads twisting a beam fixed on its one side. The ideal topology should be a hollowed-out beam, and DMF-TONN correctly outputs a similar topology. For this example, we observe that the compliance of the SIMP ($60 \times 15 \times 15$ grid) result is better than that of DMF-TONN. In Fig. 7 we present a case with a passive (non-design) region is present in the upper right quarter of the domain. This condition is enforced using an additional load acting at the center of each of its remaining sides. In Fig. 10, there is a passive region (sphere) at the center of the cube, and hence the topology has to form around this sphere. In Fig. 11, unlike Fig. 9, the loading is unsymmetric, i.e there are 4 loads and one side does not have a load acting on it. For the boundary conditions in Figs. 9, 10 and 11, SIMP ($20 \times 20 \times 20$ grid) achieves better compliance than DMF-TONN. Internal analysis has shown that DMF-TONN relies on several hyperparameters, including the batch-size, the kernel range and size and the learning rate. Finding the optimum values of these hyperparameters, such that DMF-TONN always (for all boundary conditions and target volume fractions) gives better compliance than SIMP, is something we do not study exhaustively here, and is a

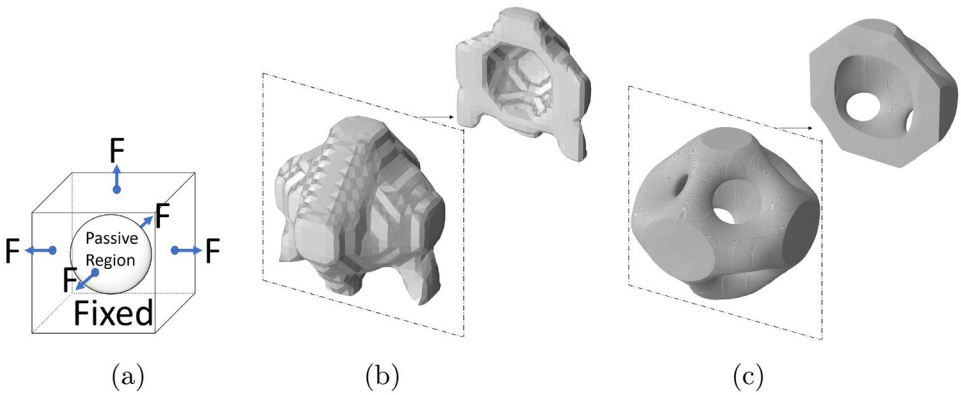


Fig. 10 5 loads with passive region and target volume fraction = 0.3, Key: [compliance $\times 10^{-3}$, volume fraction]. **a** Boundary conditions, **b** Top3D (SIMP), [1.81, 0.30], **c** DMF-TONN, [2.12, 0.30]

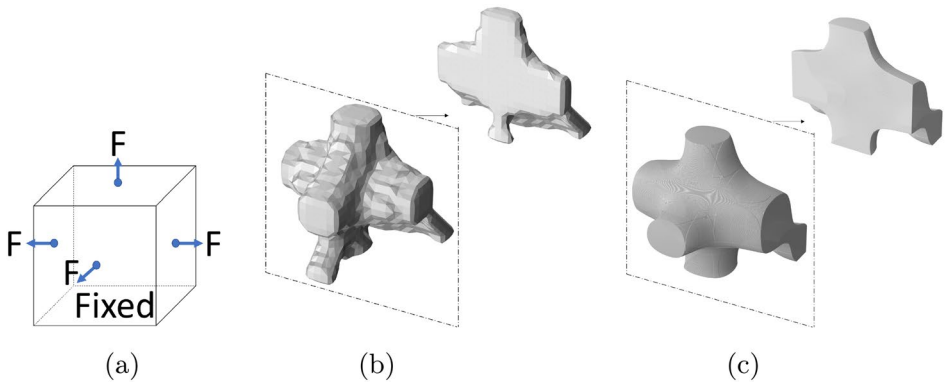


Fig. 11 4 loads (unsymmetrical) with target volume fraction = 0.3, Key: [compliance $\times 10^{-3}$, volume fraction]. **a** Boundary conditions, **b** Top3D (SIMP), [1.39, 0.30], **c** DMF-TONN, [1.47, 0.30]

constraint violation objective for the density network, where if the density network outputs density values close to 1 in the passive region, the loss is penalized. For this L-Bracket example, the compliance of the result obtained by DMF-TONN is better than that obtained by SIMP ($30 \times 30 \times 10$ grid). In Fig. 8, both left and right sides are fixed and the load acts at the center of the beam. The compliances of the results obtained by SIMP ($60 \times 20 \times 8$ grid) and DMF-TONN are similar for this example. In Fig. 9, the boundary conditions consist of a cube fixed at the bottom and a topic for future work.

We use a Young's Modulus (E) = $1000\,N/mm^2$, Poisson's ratio of 0.3, Force = $0.1\,N$ and we normalize the domain with the longest side = $1\,mm$ for all examples. We use 6000 randomly sampled domain points as the batch in each iteration of the displacement network (making sure the force vector location points and fixed boundary condition points are included) and during each iteration of density network for compliance calculation. For the volume fraction constraint violation calculation, we use a constant

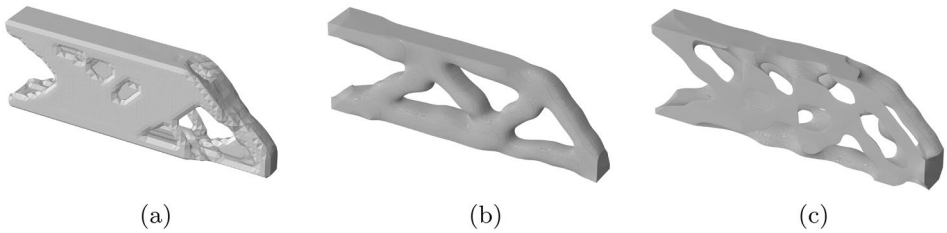


Fig. 12 Long cantilever beam with target volume fraction = 0.3, Key: [compliance $\times 10^{-2}$, volume fraction, seconds per topology optimization iteration]. **a** Top3D (SIMP), [2.65, 0.30, 0.540] **b** DMF-TONN, [2.40, 0.30, 0.739], **c** DMF-TONN with SIMP grid, [2.34, 0.30, 1.131]

grid same as the SIMP method grid used in each example (to avoid fluctuations in volume fraction). We use a kernel grid size of 16 in all three directions and set upper and lower bounds of the kernel values to 35 and -35 respectively. We find this hyper-parameters setting works well for most problems, unless when the domain has a markedly skewed size ratio, such as in the long beam and bridge examples. In those cases, one has to accordingly adjust the kernel size in the different directions. In these examples, we use 24 kernel grid points in the longest direction, and 12 kernel grid points in the other directions, and an upper and lower bounds of the kernel values of 45 and -45 respectively, for the results presented. For the examples in Figs. 9, 10 and 11, we use a smaller value of lower and upper bound (10) of kernel values for the results presented, as higher values tend to give unnecessary artifacts in the final topology. The computational time for DMF-TONN for each of the presented examples is less than 800 s. For each of the presented examples, the degrees of freedom are always more for SIMP than for DMF-TONN.

4.6 Using same grid points as SIMP for DMF-TONN

We showcase a few examples using the same grid points as SIMP for input to DMF-TONN instead of domain points that are randomly sampled in each displacement network iteration and density network iteration for compliance calculation using the trained displacement network. For the long cantilever beam example (Fig. 12), where the SIMP grid has 9600 points, we get a better compliance (Fig. 12c) than when using the randomly sampled points for DMF-TONN with batch size equal to 6000 for compliance calculation (as used

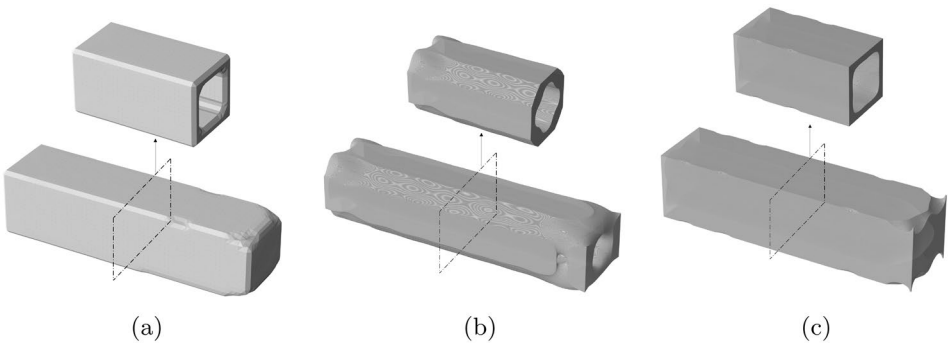


Fig. 13 Long cantilever beam with two loads with target volume fraction = 0.5, Key: [compliance $\times 10^{-3}$, volume fraction, seconds per topology optimization iteration]. **a** Top3D (SIMP), [8.22, 0.50, 1.325], **b** DMF-TONN, [8.78, 0.50, 0.744], **c** DMF-TONN with SIMP grid [7.77, 0.50, 1.543]

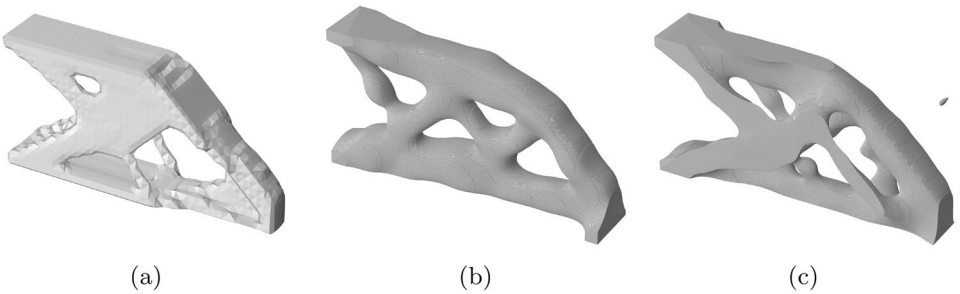


Fig. 14 Cantilever beam with target volume fraction = 0.3, Key: [compliance $\times 10^{-3}$, volume fraction, seconds per topology optimization iteration]. **a** Top3D (SIMP), [7.49, 0.30, 0.324], **b** DMF-TONN, [6.76, 0.30, 0.840], **c** DMF-TONN with SIMP grid, [6.57, 0.30, 0.901]

in all previous examples up till now) (Fig. 12b). Note that the randomly sampled DMF-TONN compliance is already better than the SIMP compliance (Fig. 12a). For the long cantilever beam with two loads example (Fig. 13), the DMF-TONN result had worse compliance than SIMP, but now after using the SIMP grid in DMF-TONN, it achieves a far better compliance than SIMP. This better compliance when using the same grid points as SIMP in DMF-TONN can be attributed to the increased number of points (9600 instead of 6000 in the long cantilever beam problem and 13,500 instead of 6000 in long cantilever beam with two loads problem). However, it should be noted as the batch size of DMF-TONN now increased, the computation time increased by 205 s and 571 s respectively for the two examples. Also, sometimes we do not get a better result with a SIMP grid even if the number of points is higher, as seen in the cantilever beam example (Fig. 14), which has better compliance than DMF-TONN but a dangling feature is present. We can attribute this dangling feature to the fact that random sampling in each topology optimization iteration as well as displacement network iteration is not present for the compliance calculation and all the hyperparameters of DMF-TONN (such as the neural network kernel size and frequency) are tuned for random sampling with a batch size of 6000.

In each of the examples presented in this paper, we use a SIMP grid that always has a higher number of points than the DMF-TONN batch size of 6000, to satisfy the ratio of lengths of sides while making sure SIMP never has any disadvantage due to lower number of points. From the preliminary analysis presented in this section we conclude that using the SIMP grid as input to DMF-TONN, we may get a better result than the original DMF-TONN in terms of compliance, but there is an increase in the computation time and there may be issues such as dangling features. The domain coordinate sampling method used in DMF-TONN plays a crucial role in the results and detailed analysis and experimentation of different sampling methods is a good direction for future work.

5 Conclusion

We show that using directly connected displacement field estimation and density field estimation neural networks is indeed an effective approach for mesh-free topology optimization. We verify through various examples that DMF-TONN, which involves using just one gradient descent step of the density network in each topology optimization epoch without any sensitivity filtering or density filtering leads to comparable results to conventional topology optimization. We significantly reduce the computational time compared to prior related works and also explore the trade-offs between DMF-TONN and SIMP for a cantilever beam example, showcasing the advantage of the mesh-free nature of DMF-TONN.

There are several limitations observed currently with DMF-TONN. The first one concerns the kernels used in the density and displacement neural network. The kernel grid has to be scaled according to the domain size if there exist sides with ratio of lengths greater than or equal to 3. Moreover, we observed that for low target volume fractions (less than 0.2), the kernel is not able to capture the required features and the optimization does not converge. Also, there are multiple hyperparameters involved in DMF-TONN, and an exhaustive study for finding their optimum values is a topic for future work.

To remove the meshing process in the complex inverse design problem of topology optimization, we devise DMF-TONN and show that it works well for various 3D problems and opens up avenues taking advantage of the mesh-free nature for seamless integration with post-processing software, or performing optimizations where FEA may be unsuitable. Future work involves improving the robustness of the kernel, extending the approach to complex problems, comparisons with state-of-the-art irregular grid topology optimization software, and experimenting with and analyzing the effect and advantages of different domain coordinate sampling methods.

Funding Open Access funding provided by Carnegie Mellon University. No funding was received to assist with the preparation of this manuscript.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Bendsøe MP (1989) Optimal shape design as a material distribution problem. Struct Optim 1:193–202

2. Zhou M, Rozvany G (1991) The coc algorithm, part II: topological, geometrical and generalized shape optimization. Comput Methods Appl Mech Eng 89(1–3):309–336
3. Chandrasekhar A, Suresh K (2021) Tounn: topology optimization using neural networks. Struct Multidiscip Optim. https://doi.org/10.1007/s00158-020-02748-4
4. Samaniego E, Anitescu C, Goswami S, Nguyen-Thanh VM, Guo H, Hamdia K, Zhuang X, Rabczuk T (2020) An energy approach to the solution of partial differential equations in computational mechanics via machine learning: concepts, implementation and applications. Comput Methods Appl Mech Eng 362:112790
5. Zehnder J, Li Y, Coros S, Thomaszewski B (2021) Ntopo: mesh-free topology optimization using implicit neural representations. Adv Neural Inf Process Syst 34:10368–10381
6. Bendsøe MP, Kikuchi N (1988) Generating optimal topologies in structural design using a homogenization method. Comput Methods Appl Mech Eng 71(2):197–224
7. Allaire G, Jouve F, Toader A-M (2002) A level-set method for shape optimization. CR Math 334(12):1125–1130
8. Wang MY, Wang X, Guo D (2003) A level set method for structural topology optimization. Comput Methods Appl Mech Eng 192(1–2):227–246
9. Woldseth RV, Aage N, Bærentzen JA, Sigmund O (2022) On the use of artificial neural networks in topology optimisation. Struct Multidiscip Optim 65(10):294
10. Raissi M, Perdikaris P, Karniadakis GE (2019) Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys 378:686–707
11. Nguyen-Thanh VM, Zhuang X, Rabczuk T (2020) A deep energy method for finite deformation hyperelasticity. Eur J Mech-A/Solids 80:103874
12. Sitzmann V, Martel J, Bergman A, Lindell D, Wetzstein G (2020) Implicit neural representations with periodic activation functions. Adv Neural Inf Process Syst 33:7462–7473
13. Tancik M, Srinivasan P, Mildenhall B, Fridovich-Keil S, Raghavan N, Singhal U, Ramamoorthi R, Barron J, Ng R (2020) Fourier features let networks learn high frequency functions in low dimensional domains. Adv Neural Inf Process Syst 33:7537–7547
14. Banga S, Gehani H, Bhilare S, Patel S, Kara L (2018) 3d topology optimization using convolutional neural networks. arXiv preprint arXiv:1808.07440. Accessed 1 Jun 2023
15. Yu Y, Hur T, Jung J, Jang IG (2019) Deep learning for determining a near-optimal topological design without any iteration. Struct Multidiscip Optim 59(3):787–799
16. Nakamura K, Suzuki Y(2020) Deep learning-based topological optimization for representing a user-specified design area. arXiv preprint arXiv:2004.05461. Accessed 1 Jun 2023
17. Nie Z, Lin T, Jiang H, Kara LB (2021) Topologygan: topology optimization using generative adversarial networks based on physical fields over the initial domain. J Mech Des 143(3):031715
18. Behzadi MM, Ilieş HT (2021) Real-time topology optimization in 3d via deep transfer learning. Comput Aided Des 135:103014
19. Mazé F, Ahmed F (2023) Diffusion Models Beat GANs on Topology Optimization. Proc AAAI Conf Artif Intell 37(8):9108–9116. https://doi.org/10.1609/aaai.v37i8.26093
20. White DA, Arrighi WJ, Kudo J, Watts SE (2019) Multiscale topology optimization using neural network surrogate models. Comput Methods Appl Mech Eng 346:1118–1135
21. Chandrasekhar A, Suresh K (2021) Length scale control in topology optimization using fourier enhanced neural networks. CoRR arXiv: abs/2109.01861. Accessed 1 Jun 2023
22. Chandrasekhar A, Suresh K (2021) Multi-material topology optimization using neural networks. CAD Comput Aided Des. https://doi.org/10.1016/j.cad.2021.103017
23. Deng H, To AC (2020) Topology optimization based on deep representation learning (drl) for compliance and stress-constrained design. Comput Mech 66(2):449–469
24. Zhang Z, Yao W, Li Y, Zhou W, Chen X (2023) Topology optimization via implicit neural representations. Comput Methods Appl Mech Eng 411:116052
25. Hoyer S, Sohl-Dickstein J, Greydanus S (2019) Neural reparameterization improves structural optimization. arXiv preprint arXiv:1909.04240. Accessed 1 Jun 2023
26. Chen H, Joglekar A, Whitefoot KS, Kara LB (2023) Concurrent build direction, part segmentation, and topology optimization for additive manufacturing using neural networks. J Mech Des 145(9):091702
27. Chen H, Joglekar A, Kara LB(2023) Topology optimization using neural networks with conditioning field initialization for improved efficiency. In: ASME 2023 International Design Engineering Technical Conferences and Computers and Information in Engineering and Conference. American Society of Mechanical Engineers
28. He J, Chadha C, Kushwaha S, Koric S, Abueidda D, Jasiuk I (2022) Deep energy method in topology optimization applications. Acta Mechanica 234:1365
29. Jeong H, Bai J, Batuwatta-Gamage C, Rathnayaka C, Zhou Y, Gu Y (2023) A physics-informed neural network-based topology optimization (pinnto) framework for structural optimization. Eng Struct 278:115484
30. Lu L, Pestourie R, Yao W, Wang Z, Verdugo F, Johnson SG (2021) Physics-informed neural networks with hard constraints for inverse design. SIAM J Sci Comput 43(6):1105–1132
31. Mai HT, Mai DD, Kang J, Lee J, Lee J (2023) Physics-informed neural energy-force network: a unified solver-free numerical simulation for structural optimization. Engineering with Computers, 1–24. https://doi.org/10.1007/s00366-022-01760-0
32. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980. Accessed 1 Jun 2023
33. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M, Ghemawat S, Goodfellow I, Harp A, Irving G, Isard M, Jia Y, Jozefowicz R, Kaiser L, Kudlur M, Levenberg J, Mane D, Monga, R, Moore S, Murray D, Olah C, Schuster M, Shlens J, Steiner B, Sutskever I, Talwar K, Tucker P, Vanhoucke V, Vasudevan V, Viegas F, Vinyals O, Warden P, Wattenberg M, Wicke M, Yu Y, Zheng X (2016) Tensorflow: large-scale machine learning on heterogeneous distributed systems. In: Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation (OSDI'16). USENIX Association, USA, 265–283.
34. Andreassen E, Clausen A, Schevenels M, Lazarov BS, Sigmund O (2011) Efficient topology optimization in matlab using 88 lines of code. Struct Multidiscip Optim 43:1–16
35. Liu K, Tovar A (2014) An efficient 3d topology optimization code written in matlab. Struct Multidiscip Optim 50:1175–1196

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.