TOMAS: Topology Optimization of Multiscale Fluid-Flow Devices using Variational Autoencoders and Super-Shapes

Rahul Kumar Padhy[1], Krishnan Suresh[1*], Aaditya Chandrasekhar[2]

[1]Department of Mechanical Engineering, University of Wisconsin-Madison, 1513 University Ave, Madison, 53706, WI, USA.

[1*]Department of Mechanical Engineering, University of Wisconsin-Madison, 1513 University Ave, Madison, 53706, WI, USA.

[2]Advanced Photon Source, Argonne National Laboratory, 9700 S Cass Ave, Lemont, 60439, IL, USA.

*Corresponding author(s). E-mail(s): ksuresh@wisc.edu;
Contributing authors: rkpadhy@wisc.edu; cs.aaditya@gmail.com;

Abstract

In this paper, we present a framework for multiscale topology optimization of fluid-flow devices. The objective is to minimize dissipated power, subject to a desired contact-area. The proposed strategy is to design optimal microstructures in individual finite element cells, while simultaneously optimizing the overall fluid flow. In particular, parameterized super-shapes are chosen here to represent microstructures since they exhibit a wide range of permeability and contact area. To avoid repeated homogenization, a finite set of these super-shapes are analyzed *a priori*, and a variational autoencoder (VAE) is trained on their fluid constitutive properties (permeability), contact area, and shape parameters. The resulting differentiable latent space is integrated with a coordinate neural network to carry out a global multi-scale fluid flow optimization. The latent space enables the use of new microstructures that were not present in the original data-set. The proposed method is illustrated using numerous examples in 2D.

Keywords: Topology Optimization, Multiscale, Stokes Flow, Variational Auto-encoders, Super-shapes

1 Introduction

In fluid-flow based topology optimization, the typical objective is to determine the path of least resistance, i.e., least dissipation, within a design domain; see fig. 1(a). When no other constraint is imposed, the path of least resistance is a single connected path [1] as illustrated in fig. 1(b). However, when additional constraints are introduced, the optimal flow-path is typically more complex, and not necessarily a single connected path. One such constraint is the desired fluid-solid contact area, which plays a crucial role in various applications such as bio-sensors for detecting tumor cells [2], microfluidic devices for cell sorting [3–7], micro-channel heat sinks [8–10], and other microfluidic devices involving heat transfer and mass transportation/mixing mechanisms [11, 12]. In these applications, a minimum fluid-solid contact area is critical for achieving desired performance and functionality. For instance, in bio-sensors for detecting tumor cells, an increased contact area between the fluid and the sensor surface enhances the sensitivity and accuracy of the detection process. Similarly, in microfluidic devices for cell sorting, the efficiency of cell capture and separation depends on the contact area between the cells and the solid surfaces. One approach for enhancing the contact area is to employ arrays of micro-pillars, as suggested in [13, 14], but this can result in a substantial increase in dissipated power [15, 16]. A more powerful approach is to use multi-scale structures, illustrated in fig. 1(c), and the main focus of this paper.

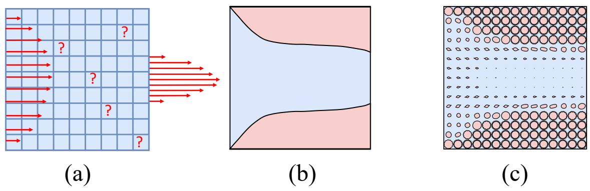

Fig. 1: (a) Fluid-flow problem. (b) Single-scale design. (c) Multi-scale design.

1.1 Single-Scale Fluid Topology Optimization

In this section, we briefly review prior research on fluid-flow topology optimization (TO). While various approaches have been proposed [1], the emphasis here is on density-based methods.

The field of fluid flow TO was initiated by the seminal work of Borrvall and Petersson [17]. In their pioneering research, they presented an optimal flow layout that minimizes pressure drop by employing the Stokes equation along with the Brinkman-Darcy equations under low Reynolds number conditions. Gersborg-Hansen et al. [18] continued this work by presenting applications with low Reynolds numbers for microfluidic problems and micro-electro-mechanical devices. Guest and Prévost [19] introduced the method of Darcy–Stokes finite elements to optimize creeping fluid flows, producing 0–1 (void-solid) topologies without artificial material regions.

A novel density-based approach for topology optimization of Stokes flow was proposed in [20] which addresses convergence issues using fractional-order Sobolev spaces for density. Anisotropic mesh adaptation is explored in [21] to improve the description of solid domains in topology optimization of flow problems. Furthermore, to address the bottlenecks of body-fitted mesh evolution method, reaction–diffusion equation-based fluid topology optimization is explored in [22]. Additionally, Wiker et al. [23] explored the use of viscosity as a dependent parameter, providing examples of channels in a tree-shaped structure for pure Darcy problems and mixed Stokes–Darcy flow. The field of fluid flow TO has also been extended to three-phase interpolation models, considering fluid permeability through porous media and impenetrable inner walls using the solid isotropic material with penalization (SIMP) interpolation functions [24]. A Matlab implementation is presented in [25], demonstrating stable low-order discretization of Stokes equations using polygonal finite elements. Parallel computations have been employed for large-scale 2D and 3D Stokes flow problems [26]. In [27], a marker-and-cell method is introduced for large-scale optimization on GPU, utilizing a geometric multigrid preconditioner. A performance-driven optimization of fluidic devices, utilizing parametric boundary descriptions and a differentiable Stokes flow solver is proposed in [28]. Furthermore, an anisotropic, differentiable constitutive model integrating different phases and boundary conditions within a Stokes model is presented in [29]. Lastly, a phase field approach is introduced in [30] for shape and topology optimization in Stokes flow, providing a well-posed problem in a diffuse interface setting.

1.2 Multiscale Topology Optimization

The methods discussed above result in a single-scale design where all design features are of the same length scale. As discussed earlier, for problems with additional constraints, one must resort to multi-scale TO (MTO), where one designs optimal microstructures in each finite-element cell while simultaneously solving the global flow problem [31]. Several MTO techniques have been proposed for structural and thermal problems. In [32, 33], the authors introduced techniques aimed at finding designs with optimal constitutive properties under structural and thermal loads. However, these theoretically optimal designs often lead to extremely small length scales, posing manufacturing challenges [34–36]. Additionally, their applicability to fluid problems remains unexplored. A more common approach is to compute optimal microstructures in each cell [31, 37, 38]. While offering broad design freedom and applicability to various physical phenomena, including fluid flow [31], these methods tend to be computationally expensive [35] since one must carry out homogenization of the evolving microstructures during each step of the optimization process [39].

Homogenization-based optimization can yield disconnected multiscale designs [35]. The process of creating connected multiscale structures from these designs is referred to as de-homogenization [40]. For example, [41, 42] utilized an inverse design methodology to design microchannel fluid flow networks featuring numerous outlets, employing Turing pattern dehomogenization. In [43], dehomogenization based on bioinspired diffusion-generated patterns converted orientation fields into explicit fluid flow channels. Furthermore, in [44], dehomogenization has been used for the inverse design of fiber-reinforced composite with a composite microstructure orientation approach [45]. Other examples of dehomogenization in multiscale topology optimization include the design of shell-infill structures [46], fluid microchannels [47] and inverse design of microreactors [48].

1.3 Variations of MTO

To tackle the computational challenges of classic MTO, other techniques have been proposed. For example, graded-MTO (GMTO) [49–56] employs graded variations of pre-selected microstructures. This allows for pre-computation of microstructural properties through offline homogenization before optimization [57, 58]. Machine learning has been utilized to learn the local mechanical properties of multiscale configurations through offline computation [59, 60]. GMTO has been used with both single [61–63] and multiple microstructures [57, 64]. One major limitation of GMTO is that the microstructural shape must be pre-selected prior to optimization. Thus, new microstructure shapes cannot be discovered during optimization. Moreover, these pre-selected shapes are typically graded using a single parameter, which further restricts the variety of microstructures that are generated. To address the aforementioned challenge, a microstructure blending-based multiscale approach has been proposed in [65], which can generate new classes of microstructures. However, the approach requires supplementary parameters beyond the conventional shape parameters. Moreover, the blending process requires additional steps to impose bounds on the blending operation, to prevent any distorted or invalid shapes in the resulting microstructures. In [64], a set of microstructures were pre-selected, and their size/orientation was optimized. While this slightly increased the design space, it is still limiting and leads to undesirable mixing of microstructures within each cell.

1.4 Contributions

In section 2, we propose an alternate and efficient MTO method that uses variational auto-encoder (VAE) in combination with super-shapes to compute fluid designs with low dissipation, and desired contact area. In section 3, we demonstrate, using numerical experiments, that this significantly increases in design space. Conclusions and future work are discussed in section 4.

2 Proposed Method

2.1 Assumptions and Strategy

Consider a design domain Ω^0 with prescribed flow boundary conditions as illustrated earlier in fig. 1(a). The objective is to compute a multiscale design that minimizes the dissipated power subject to a total contact-area (i.e., perimeter in 2D) constraint. We will assume that it is a low-Reynolds flow, and the fluid is incompressible, i.e., the fluid is governed by Stokes equation:

$$-2\nabla.[\mu\boldsymbol{\epsilon}(\boldsymbol{u})] + \boldsymbol{C}_{\boldsymbol{eff}}^{-1}.\boldsymbol{u} + \nabla p = 0 \text{ in } \Omega^0 \quad (1a)$$

$$\nabla.\boldsymbol{u} = 0 \text{ in } \Omega^0 \quad (1b)$$

$$\boldsymbol{u} = \boldsymbol{\mathcal{G}} \text{ over } \partial\Omega^0 \quad (1c)$$

where $\boldsymbol{u}$ and p are the velocity vector and pressure of the fluid, $\boldsymbol{\epsilon}(\boldsymbol{u}) = (\boldsymbol{\nabla u} + \boldsymbol{\nabla^T u})/2$ represents the rate-of-strain tensor, $\boldsymbol{C}_{\boldsymbol{eff}}^{-1}$ denotes the inverse of effective permeability tensor [66, 67] which penalizes the fluid flow in the design domain (for more details see section 2.3). The viscosity μ and mass density are assumed to be unity; $\boldsymbol{\mathcal{G}}$ is the velocity field imposed on $\partial\Omega^0$.

The overall strategy is to discretize the domain into finite element cells, and dynamically create optimal microstructures in each cell from a single family of parameterized super-shapes, discussed next.

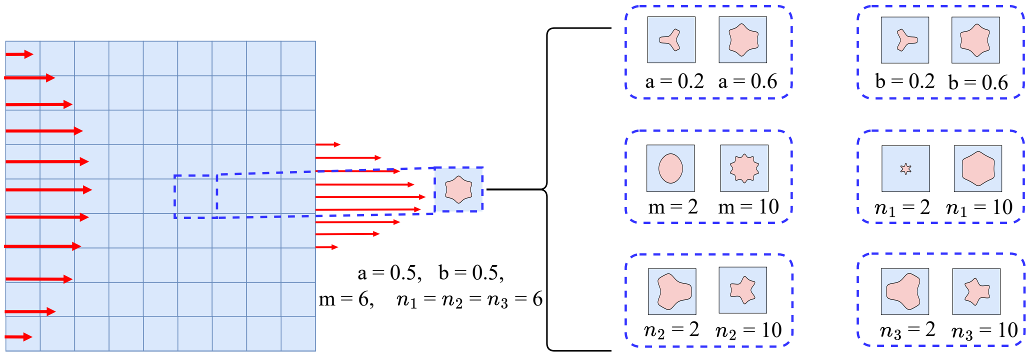

Fig. 2: A variety of microstructure generated using supershape parameters.

2.2 Super-Shapes

Super-shapes, also known as Gielis curves, were introduced by Gielis in 2003 [68] as an extension of super-quadrics. Supershapes have been extended to 3D, the approach involves the spherical product of two super-shapes [69], similar to the method employed for super-quadrics as elaborated in [70]. Unlike super-quadrics that utilize only two parameters, super-shapes incorporate six parameters namely, size: a and b, order of rotational symmetry: m, and curvature: n_1, n_2, and n_3, where all numbers are assumed to be positive real. Using these six parameters, the super-shape boundary is defined by the set of points:

$$(x, y) = \big(r(\alpha)\cos(\alpha), r(\alpha)\sin(\alpha)\big) \; ; \; 0 \le \alpha \le 2\pi \tag{2}$$

where,

$$r(\alpha) = \left[\left|\frac{1}{a}\cos\left(\frac{m\alpha}{4}\right)\right|^{n_2} + \left|\frac{1}{b}\sin\left(\frac{m\alpha}{4}\right)\right|^{n_3}\right]^{-\frac{1}{n_1}} \tag{3}$$

By varying these six parameters, a variety of shapes can be obtained as illustrated in fig. 2.

In addition to varying these parameters, we allow the shapes be oriented with respect to the x-axis, using an orientation parameter θ; see fig. 3.

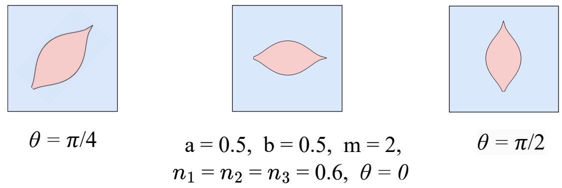

Fig. 3: A fish-shaped microstructure oriented at various angles.

The proposed strategy is to find optimal super-shape parameters and orientation, within each cell that minimize the overall dissipated power, subject to a contact-area. A naive approach would entail computing the homogenized constitutive tensors of evolving super-shapes in each cell, during each step of the optimization process. This is once again computationally intractable. Instead, we propose an off-line strategy where a finite set of super-shapes are analyzed, and their characteristics are captured using a VAE [71] (see section 2.4). The resulting decoder (and latent space) is then used for efficient multi-scale optimization. In the remaining sections, the proposed strategy is discussed in detail.

2.3 Offline Computation

We now describe the process for computing the permeability of any super-shape microstructure. Consider a generic super-shape with a given set of parameters $\boldsymbol{M} = \{a, b, m, n_1, n_2, n_3\}$ within a unit cell (i.e., $l = 1$). The contact-area (i.e., perimeter) and volume fraction are first computed by discretizing the boundary and generating the super-shape density field using the *shapely* library [72], as illustrated in fig. 4(b) and fig. 4(c) respectively.

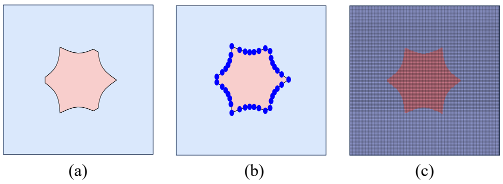

Fig. 4: Generation of super-shape density field using shapely library

To compute the 2×2 permeability tensor $\boldsymbol{C}$, the domain is discretized into a mesh of 150×150 elements. Then, two Stokes flow problems are solved, subject to unit body forces $f_x = 1$ and $f_y = 1$, as illustrated in fig. 5. The boundary conditions involve coupling boundaries 1 and 3 through periodic conditions for velocity and pressure, as well as coupling boundaries 2 and 4 similarly; see [73]. The velocities obtained from solving the problem with $f_x = 1$ are denoted as $u_0(x, y)$ and $v_0(x, y)$, while those obtained from solving the problem with $f_y = 1$ are denoted as $u_1(x, y)$ and $v_1(x, y)$. Since $u_1(x, y)$ and $v_0(x, y)$ are nearly orthogonal to the bulk flow directions, the off-diagonal terms of the permeability tensor are three orders of magnitude smaller than the diagonal terms and can be neglected [74]. Thus, the permeability tensor $\boldsymbol{C}$ is computed as follows [73, 75, 76]:

$$\boldsymbol{C} = \begin{bmatrix} C_{00} & 0 \\ 0 & C_{11} \end{bmatrix} = \frac{1}{|V|} \begin{bmatrix} \int_V u_0 dV & 0 \\ 0 & \int_V v_1 dV \end{bmatrix} \tag{4}$$

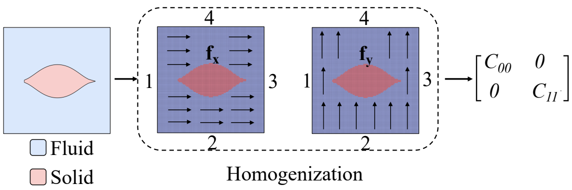

Fig. 5: Offline homogenization.

where the volume V of the unit cell is unity. Note that the orientation is taken into account through the following tensor operation to determine the transformed permeability tensor [75]:

$$\boldsymbol{C}_{trans} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} C_{00} & 0 \\ 0 & C_{11} \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}^T \tag{5}$$

As the effective permeability $\boldsymbol{C}_{eff}$ in equation eq. (1) is scale-dependent, one can scale $\boldsymbol{C}_{trans}$ according to the unit-cell size l of the problem, expressed as $\boldsymbol{C}_{eff} = l^2 \boldsymbol{C}_{trans}$.

For numerical homogenization, a Brinkman penalization of zero is applied for the fluid phase and 10^6 for the solid phase. For further details on the numerical homogenization methodology employed in this study, please see [73]. A random set of 7000 samples of super-shapes are analyzed using the above process, where parameter instances are generated using a uniform random distribution [77] as follows: $0.05 \leq a, b \leq 0.75$, $1 \leq m \leq 22$ and $0.5 \leq n_1, n_2, n_3 \leq 10$. The results from the offline computation are then analyzed using variational autoencoders, discussed next.

2.4 Variational Auto-Encoders

Variational auto-encoders (VAEs) are a type of generative model that leverages probabilistic encoding and decoding techniques to compress input data into a lower-dimensional latent space [71, 78, 79]. One of the key advantages of VAEs, as opposed to other encoding methods, is their ability to generate new samples that resemble the original input data [79]. For instance, VAEs have been successfully employed to generate novel microstructures from image databases [80]. Another important feature of VAEs is the creation of continuous and differentiable latent space. This allows for gradient-based optimization, enabling efficient exploration of the latent space. This is particularly valuable in applications such as reliability-based TO [81]. Finally, unlike linear dimensionality reduction techniques such as principal component analysis (PCA), VAEs can learn complex non-linear relationships between the input and the reduced dimensional space [82].

The proposed VAE architecture, depicted in fig. 6, consists of several essential components:

1. Firstly, the input $\boldsymbol{\Psi}$ is ten-dimensional representing six shape parameters $\boldsymbol{M}$, two permeability components (C_{00} and C_{11}), contact area Γ and volume fraction v_f.

2. The encoder E, following along [83], is a fully-connected network consisting of two hidden layers, each containing 600 neurons with a LeakyReLU activation functions [84].
3. The proposed VAE uses a two-dimensional latent space (z_1 and z_2).
4. Additionally, a decoder D is constructed with two hidden layers, each containing 600 neurons.
5. The output consists of the same ten properties: shape parameters $\hat{\boldsymbol{M}}$, permeability components $\hat{\boldsymbol{C}} \equiv (\hat{C}_{00}$ and $\hat{C}_{11})$, contact area $\hat{\Gamma}$ and volume fraction $\hat{v}_f$. They can be combined as $\hat{\boldsymbol{\Psi}} \equiv (\hat{a}, \hat{b}, \hat{m}, \hat{n}_1, \hat{n}_2, \hat{n}_3, \hat{C}_{00}, \hat{C}_{11}, \hat{\Gamma}, \hat{v}_f)$.

Note that the reconstruction will not be exact. One objective is to minimize the difference between the output and input [85]. This involves optimizing the weights associated with the encoder and decoder. Additionally, the latent space is constrained to approximate a Gaussian distribution $\mathcal{N}(0, I)$ through a KL divergence loss term expressed as $\mathrm{KL}(z||\mathcal{N})$ [71]. This ensures that similar shapes are clustered together in the latent space. Thus, the overall VAE loss function can be formulated as:

$$L_v = ||\boldsymbol{\Psi} - \hat{\boldsymbol{\Psi}}||_2 + \beta \mathrm{KL}(z||\mathcal{N}) \tag{6}$$

Here, β is set to 10^{-7} [85]. To achieve a stable convergence, the geometric parameters, contact area, and volume fraction are normalized linearly between 0 and 1, while the permeability components are scaled logarithmically due to significant variation in magnitude.

2.5 Latent Space

Once the latent space has been constructed, the trained decoder D^* can be used to generate super-shape parameters and properties via $\hat{\Psi} = D^*(z_1, z_2)$ for all points within the latent space. The generated latent space has the following features:

1. **Generation of new microstructures**: Although the data set used to train the decoder is discrete, the resulting latent space is continuous. This continuous representation facilitates a meaningful exploration of microstructure configurations throughout the latent space. For example in fig. 7, while A, B, E, and F depict points present in the data set, points C, D, G, and H are generated by the trained decoder, with corresponding microstructures. The percentage error between the actual microstructure properties obtained from numerical homogenization and the reconstructed data for points both within (A, B, E, and F) and outside (C, D, G, and H) the training dataset are summarized in table 1. We found that the percentage errors for both sets of points are comparable.
2. **Differentiable Latent Space**: The latent space is differentiable in that derivatives such as $\frac{\partial \hat{\Psi}}{\partial \boldsymbol{z_1}}$, can be computed analytically using back-propagation. This enables gradient-based optimization.

Points	$\Delta C_{00}\%$	$\Delta C_{11}\%$	$\Delta v_f\%$	$\Delta\Gamma\%$
A	0.42	3.70	0.232	1.20
B	1.84	3.63	0.291	3.30
C	3.50	2.80	0.20	0.76
D	1.80	3.60	1.60	2.60
E	0.43	3.70	0.60	2.03
F	0.122	4.05	0.947	1.24
G	1.83	3.63	1.64	2.65
H	2.05	0.322	2.38	2.63

Table 1: Prediction accuracy for points in fig. 7

2.6 Global Fluid Flow Analysis

We are now ready to address global fluid flow analysis. Here, a quadrilateral Q2-Q1 (quadratic velocity/linear pressure) element belonging to the class of the Taylor-Hood elements is used. The elemental stiffness matrix $\boldsymbol{K_e}$ and degrees of freedom vector $\boldsymbol{S_e}$ for the governing equation (see section 2.1) are given by (see [25] for details):

$$\boldsymbol{K_e} = \begin{bmatrix} \boldsymbol{A_e} & \boldsymbol{B_e} & \boldsymbol{0} \\ \boldsymbol{B_e^T} & \boldsymbol{0} & \boldsymbol{h_e} \\ \boldsymbol{0} & \boldsymbol{h_e^T} & \boldsymbol{0} \end{bmatrix}, \ \boldsymbol{S_e} = \begin{bmatrix} \boldsymbol{U_e} \\ \boldsymbol{P_e} \\ \boldsymbol{\lambda} \end{bmatrix} \tag{7}$$

where

$$\boldsymbol{A_e} = \boldsymbol{A_e^\mu} + \boldsymbol{C_{eff,e}^{-1}} \boldsymbol{A_e^\alpha} \tag{8a}$$

$$[\boldsymbol{A_e^\mu}]_{ij} = \int_{\Omega_e} 2\mu \boldsymbol{\epsilon}(\boldsymbol{N_i}) : \boldsymbol{\epsilon}(\boldsymbol{N_j}) d\Omega \tag{8b}$$

$$[\boldsymbol{A_e^\alpha}]_{ij} = \int_{\Omega_e} \boldsymbol{N_i} \boldsymbol{N_j} d\Omega \tag{8c}$$

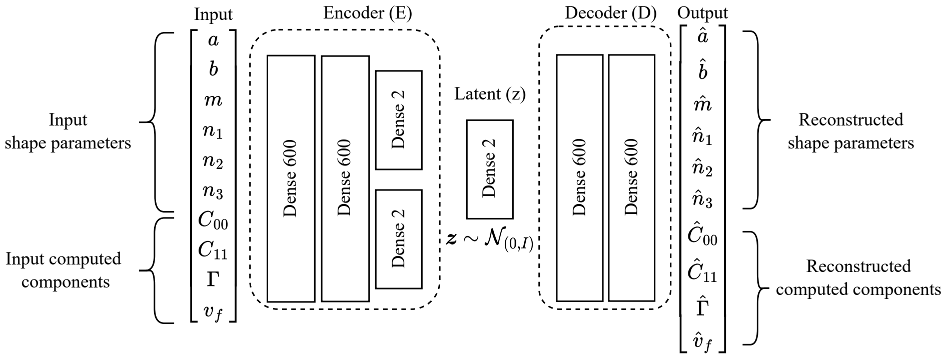

Fig. 6: Proposed VAE network.

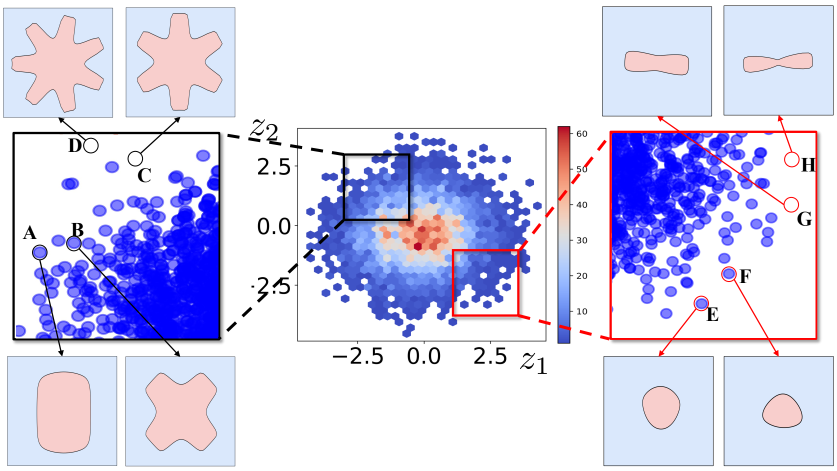

Fig. 7: The latent space density distribution and scattered plots in the insets reveal both microstructures existing in the dataset and new microstructures generated by the VAE that are not originally present in the dataset.

$$[\boldsymbol{B_e}]_{ij} = \int_{\Omega_e} \boldsymbol{L_j} \nabla . \boldsymbol{N_i} d\Omega \tag{8d}$$

$$[\boldsymbol{h_e}]_i = \int_{\Omega_e} \boldsymbol{L_i} d\Omega \tag{8e}$$

Here, $\boldsymbol{N_i}$ and $\boldsymbol{L_i}$ are the velocity and pressure basis functions, $\boldsymbol{U_e}$ and $\boldsymbol{P_e}$ represent elemental velocity and pressure degrees of freedom respectively and $\boldsymbol{C_{eff,e}}$ is the design dependent effective element permeability matrix (section 2.3). and $\boldsymbol{\epsilon}(\cdot)$ is defined in section 2.1. To ensure a unique definition of the pressure field, a zero mean condition ($\boldsymbol{h_e^T} P = 0$) is imposed by incorporating a Lagrange multiplier, denoted as λ (for details see [25, 86]). The individual elemental $\boldsymbol{K_e}$ and $\boldsymbol{S_e}$ matrices are assembled to construct the global stiffness matrix $\boldsymbol{K}$ and degrees of vector $\boldsymbol{S}$ respectively. We then solve the equation $\boldsymbol{KS} = \boldsymbol{f}$, wherein the vector $\boldsymbol{f}$ represents the boundary conditions applied. This solution determines the unknown degrees of freedom in $\boldsymbol{S}$.

2.7 Design Variables, Objective and Constraints

Finally, the optimization framework comprises of the following:

Design Variables: The design variables associated with each element are denoted by $\boldsymbol{\zeta_e} = \{z_{1,e}, z_{2,e}, \theta_e\}$, where $z_{1,e}$ and $z_{2,e}$ are the two latent space variables, and θ_e is the orientation of the super-shape. The values z_1 and z_2 are constrained to lie within $[-3, 3]$, and the orientation parameter is constrained as $0 \leq \theta \leq 2\pi$. The entire set of design variables is denoted by $\overline{\boldsymbol{\zeta}} = \{\boldsymbol{\zeta_1}, \boldsymbol{\zeta_2}, ..., \boldsymbol{\zeta_{N_e}}\}$.

Objective: The objective is to minimize the dissipated power given by [17, 25]:

$$J(\overline{\boldsymbol{\zeta}}) = \sum_{e=1}^{N_e} \frac{1}{2} \boldsymbol{U_e}^T [\boldsymbol{A_e^\mu} + \boldsymbol{A_e^\alpha} \boldsymbol{C_{eff,e}^{-1}}] \boldsymbol{U_e} \tag{9}$$

Contact-Area Constraint: The contact area $\hat{\Gamma}_e$ of each microstructure is reconstructed using the decoder, and the following global constraint is imposed:

$$g_\Gamma(\overline{\boldsymbol{\zeta}}) \equiv 1 - \frac{\sum_{e=1}^{N_e} \hat{\Gamma}_e}{\Gamma_{min}} \leq 0 \tag{10}$$

where Γ_{min} is the lower bound on the contact area and N_e represents the number of elements in the design domain.

Volume Constraint: Instead of imposing a contact area constraint, one can impose a volume constraint:

$$g_V(\overline{\boldsymbol{\zeta}}) \equiv \frac{\sum_{e=1}^{N_e} \hat{v}_{f,e}}{N_e v_{max}} - 1 \leq 0 \tag{11}$$

where, $\hat{v}_{f,e}$ is the fluid volume fraction, and v_{max} is the upper bound on the volume fraction.

2.8 Multiscale Optimization Problem

Consequently, one can pose the multiscale problem in a finite-element setting as:

$$\begin{aligned}
&\min_{\overline{\boldsymbol{\zeta}} = \{\boldsymbol{\zeta_1}, \boldsymbol{\zeta_2}, \ldots \boldsymbol{\zeta_{N_e}}\}} && J(\overline{\boldsymbol{\zeta}}) && \text{(12a)}\\
&\text{subject to} && \boldsymbol{K}(\overline{\boldsymbol{\zeta}})\boldsymbol{S} = \boldsymbol{f} && \text{(12b)}\\
& && g_\Gamma(\overline{\boldsymbol{\zeta}}) \leq 0 && \text{(12c)}\\
&\text{(or)} && g_V(\overline{\boldsymbol{\zeta}}) \leq 0 && \text{(12d)}\\
&\text{with} && -3 \leq z_{e,0}, z_{e,1} \leq 3 \ , \ \forall e && \text{(12e)}\\
& && 0 \leq \theta_e \leq 2\pi \ , \ \forall e && \text{(12f)}
\end{aligned}$$

To solve the above optimization problem, optimization techniques such as the method of moving asymptotes [87] or optimality criteria [88] can be used. However, we use neural networks (NN) for optimization [89] due to several advantages. Most NN implementations support automatic differentiation [90], which enables seamless gradient calculations. They can also capture highly nonlinear behavior with very few design variables.

2.9 Optimization using a Neural Network

The proposed neural-network (NN) architecture for global optimization is illustrated in fig. 8, and it consists of the following entities:

1. **Input Layer**: The input to the NN are points $\boldsymbol{x} \in \mathbf{R}^2$ within the domain Ω^0. Although these points can be arbitrary, they correspond here to the center of the elements.
2. **Fourier Projection**: The sampled points from the Euclidean domain are directed through a frequency space, associated with a frequency range $\boldsymbol{F}$. Prior research [91, 92] indicates that implicit coordinate-based neural networks are biased to lower frequency components of the target signal. To address this issue and speed up convergence, a Fourier projection layer is integrated before the standard activation layers [93].
3. **Hidden Layers**: The hidden layers consist of a series of fully connected LeakyReLU activated neurons, LeakyReLU is a differentiable function, as opposed to ReLU, and is therefore preferred in this work [94]. In particular, the neural network used here consists of two hidden layers, each activated with the LeakyReLU function and each layer has 20 neurons.
4. **Output Layer**: The output layer consists of 3 neurons corresponding the design variables for each element $\boldsymbol{\zeta_e} = \{z_1(\boldsymbol{x}), z_2(\boldsymbol{x}), \theta(\boldsymbol{x})\}$. The output neurons are activated by a Sigmoid

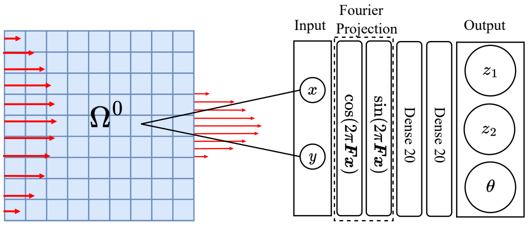

Fig. 8: Topology optimization network.

function $\sigma(\cdot)$. The neurons associated with the latent space variables are scaled as $z_i \leftarrow -3 + 6\sigma(z_i)$ to retrieve values in the range of a standard Gaussian Normal distribution. Further, the output neuron associated with the orientation is scaled as $\theta \leftarrow 2\pi\sigma(\theta)$. Thus, the box constraints in eq. (12e) and eq. (12f) are not needed.

5. **NN Design Variables**: The weights and bias associated with the NN, denoted by the $\boldsymbol{w}$, now become the primary design variables, i.e., we have $z_1(\boldsymbol{x};\boldsymbol{w})$, $z_2(\boldsymbol{x};\boldsymbol{w})$ and $\theta(\boldsymbol{x};\boldsymbol{w})$. The weights in the network are initialized using Xavier weight initialization [95] with a seed value of 77.
6. **Optimizer:** The Adam optimizer is used with a learning rate of $4 \cdot 10^{-3}$. The optimization process is set to run for a maximum of 300 iterations (epochs). To ensure convergence, the optimization monitors the change in loss (ΔL_c^*) (see eq. (14)) with a threshold of 10^{-5}.

Thus, eq. (12) reduces to:

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad J(\boldsymbol{w}) \tag{13a}$$

$$\text{subject to} \quad \boldsymbol{K}(\boldsymbol{w})\boldsymbol{S} = \boldsymbol{f} \tag{13b}$$

$$g_\Gamma(\boldsymbol{w}) \equiv 1 - \frac{\sum_{e=1}^{N_e} \hat{\Gamma}_e(\boldsymbol{w})}{\Gamma_{min}} \leq 0 \tag{13c}$$

$$\text{(or)} \quad g_V(\boldsymbol{w}) \equiv \frac{\sum_{e=1}^{N_e} (\hat{v}_{f,e}(\boldsymbol{w}))}{N_e v_{max}} - 1 \leq 0 \tag{13d}$$

Since neural networks are designed to minimize an unconstrained loss function, we convert the constrained minimization problem into a loss function minimization by employing the penalty scheme [96]. Specifically, the loss function is defined as:

$$L_T(\boldsymbol{w}) = \frac{J(\boldsymbol{w})}{J^0} + \gamma g(\boldsymbol{w})^2 \tag{14}$$

where the parameter γ is updated during each iteration, making the enforcement of the constraint (g) stricter as the optimization progresses. The constraint penalty in the current framework starts with an initial value of $\gamma = 1.$ and is incremented by $\Delta\gamma = 0.1$ after every epoch. J^0 represents the initial iteration's objective value, which serves as a scaling factor for the objective function.

Thus, the overall framework is illustrated in fig. 9.

2.10 Sensitivity Analysis

A critical ingredient in gradient-based optimization is the sensitivity, i.e., derivative, of the objective and constraint(s) with respect to the optimization parameters. Typically the sensitivity analysis is carried out manually. For example, the derivatives of the objective function are typically expressed as follows where each term is computed manually:

$$\frac{\partial L_T}{\partial \boldsymbol{w}} = \Big[\frac{\partial L_T}{\partial J}\frac{\partial J}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{K}}\Big(\frac{\partial \boldsymbol{K}}{\partial \hat{\boldsymbol{C}}}\frac{\partial \hat{\boldsymbol{C}}}{\partial \boldsymbol{w}_{\boldsymbol{D}}^*}\frac{\partial \boldsymbol{w}_{\boldsymbol{D}}^*}{\partial \boldsymbol{z}}\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{w}} + \frac{\partial \boldsymbol{K}}{\partial \boldsymbol{\theta}}\frac{\partial \boldsymbol{\theta}}{\partial \boldsymbol{w}}\Big) + \frac{\partial L_T}{\partial g_\Gamma}\frac{\partial g_\Gamma}{\partial \hat{\boldsymbol{\Gamma}}}\frac{\partial \hat{\boldsymbol{\Gamma}}}{\partial \boldsymbol{w}_{\boldsymbol{D}}^*}\frac{\partial \boldsymbol{w}_{\boldsymbol{D}}^*}{\partial \boldsymbol{z}}\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{w}}\Big] \tag{15}$$

This can be laborious and error-prone, especially for non-trivial objectives. Here, by expressing all our computations including computing the permeability tensors, stiffness matrix, FEA, objectives, and constraints in PyTorch [90], we use the NN's automatic differentiation (AD) capabilities to completely automate this step [97]. In other words, only the forward expressions need to be defined, and all required derivatives are computed to machine precision by PyTorch computing library. PyTorch automatically handles computing the gradients by constructing a computational graph, with the weights of the NN serving as the leaf tensors [98]. This graph encompasses all the operations depicted in the flow chart in fig. 9. By invoking the gradient function with respect to the loss, PyTorch effectively calculates the sensitivities eq. (15), including the adjoint computation $\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{K}}$.

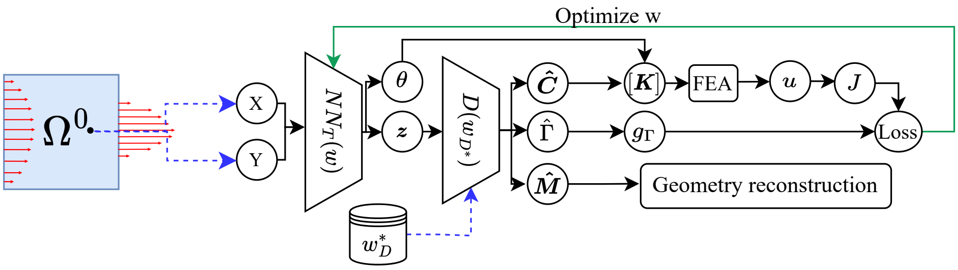

Fig. 9: Optimization flowchart.

2.11 Algorithms

The three algorithms used here are summarized in algorithm 1, algorithm 2, and algorithm 3. In the first algorithm, the primary objective is to generate a set of microstructures and their properties Ψ.

Algorithm 1 Generate Microstructure properties data-set

---

procedure DATAGEN
  $\boldsymbol{M} \rightarrow \Gamma, v_f, C_{00}, C_{11}$ ▷ Offline Computation section 2.3
  $\boldsymbol{M}, \Gamma, v_f, C_{00}, C_{11} \rightarrow \Psi$ ▷ Data-set creation
end procedure ▷ Output: Microstructure data-set

In algorithm 2, using Ψ, the VAE is trained to produce a lower-dimensional latent space.

Algorithm 2 Encode Microstructure properties

---

procedure MSTRENCODE(Ψ, E, D) ▷ Input: Training data, encoder and decoder
  epoch $= 0$ ▷ iteration counter
  repeat ▷ VAE training
    $E(\Psi) \rightarrow \boldsymbol{z}$ ▷ Forward prop.encoder
    $D(\boldsymbol{z}) \rightarrow \hat{\Psi}$ ▷ Forward prop.decoder
    $\{\Psi, \hat{\Psi}, \boldsymbol{z}\} \rightarrow L_V$ ▷ VAE loss eq. (6)
    $\boldsymbol{w}, \nabla L_V \rightarrow \boldsymbol{w}$ ▷ Adam optimizer step; update weights
    epoch $++$
  until $||\Delta L_V|| < \Delta \hat{L}_V$ or epoch < max_epoch ▷ check for convergence
end procedure ▷ Output: Trained Decoder

Once the training is complete, the encoder E is discarded, and the decoder D is retained. The main optimization algorithm is summarized in algorithm 3. First, the domain Ω^0 is discretized for finite element analysis, and the stiffness matrix components are computed (line 3). The mesh is sampled at the center of each element (line 4); these serve as inputs to the NN. The penalty parameter γ and NN weights $\boldsymbol{w}$ are initialized [95] (line 5).

In the main iteration, the design variables $\overline{\boldsymbol{\zeta}}$ are computed using the NN (line 7). Then the latent space variables serve as input to the trained decoder D^*, followed by the computation of the microstructural geometric properties $\hat{\boldsymbol{M}}$, permeability components $\hat{\boldsymbol{C}}$ and contact area $\hat{\Gamma}$ (or, alternately, the volume fraction) for each element (line 8). The reconstructed permeability components from the decoder along with the orientation from the NN are used to calculate the effective permeability tensor (line 9). The effective permeability is used to construct the stiffness matrix and to solve for the velocity and pressure (lines 10-11). Then the objective and contact area (or volume) constraint are computed (lines 12 - 13), leading to the loss function (line 14). The sensitivities are computed in an automated fashion (line 15). The weights $\boldsymbol{w}$ are then updated using the Adam optimization scheme (line 16). Finally, the penalty parameters are updated (line 17). The process is repeated until termination, i.e., until the relative change in loss is below a certain threshold or the iterations exceed a maximum value.

3 Numerical Experiments

In this section, we conduct several experiments to demonstrate the proposed framework. All experiments were conducted on a MacBook M2 Air, using the PyTorch library [90] in Python. Although some hyperparameter tuning was necessary initially (specifically, learning rate, neural network

Algorithm 3 Fluid Topology Optimization

procedure TopOpt(Ω^0, BC, Γ_{min}, D^*) ▷ Input: Design domain, boundary conditions, area constraint, and trained decoder
  $\Omega^0 \rightarrow \Omega_h^0$ section 2.6
  $\Omega_h^0 \rightarrow \boldsymbol{A^\mu}, \boldsymbol{A^\alpha}, \boldsymbol{B}, \boldsymbol{h}$ eq. (8)
  $\boldsymbol{x} = \{x_e, y_e\}_{e \in \Omega_h^0}$ $\boldsymbol{x} \in \mathbb{R}^{n_e \times 2}$
  epoch $= 0$; $\gamma = \gamma_0$; $\boldsymbol{w} = \boldsymbol{w}_0$ ▷ initialization
  repeat ▷ optimization (Training)
    $NN(\boldsymbol{x}; \boldsymbol{w}) \rightarrow \overline{\boldsymbol{z}}(\boldsymbol{x}), \overline{\boldsymbol{\theta}}(\boldsymbol{x})$ ▷ section 2.9
    $D^*(\overline{\boldsymbol{z}}(\boldsymbol{x}; \boldsymbol{w})) \rightarrow \hat{\boldsymbol{M}}(\boldsymbol{x}), \hat{\boldsymbol{C}}$ ▷ fwd prop
    $\hat{C}_{00}(\boldsymbol{x}), \hat{C}_{11}(\boldsymbol{x}), \theta(\boldsymbol{x}) \rightarrow \boldsymbol{C}_{eff}(\boldsymbol{x})$ ▷ eq. (5)
    $\boldsymbol{C}_{eff}(\boldsymbol{x}) \rightarrow \boldsymbol{K}, \boldsymbol{f}$ eq. (7)
    $\boldsymbol{K}, \boldsymbol{f} \rightarrow \boldsymbol{S}$ ▷ solve eq. (13b)
    $\boldsymbol{K}, \boldsymbol{S} \rightarrow J$ ▷ Objective, eq. (13a)
    $\hat{\Gamma}, \Gamma_{min} \rightarrow g_\Gamma$ eq. (10)
    $J, g_\Gamma \rightarrow L$ ▷ loss from eq. (14)
    $AD(L, \boldsymbol{w}) \rightarrow \nabla L$ ▷ sensitivity analysis
    $\boldsymbol{w}, \nabla L \rightarrow \boldsymbol{w}$ ▷ Adam optimizer step
    $\gamma + \Delta\gamma \rightarrow \gamma$ ▷ increment penalty
    epoch $++$
  until $||\Delta L|| < \Delta L_c^*$ or epoch $<$ max_epoch ▷ check for convergence
end procedure

configuration, and update schemes), we employed the same values across all experiments.

3.1 Ideal Microstructure Selection

In this experiment, our objective is to identify a microstructure with a solid volume fraction of approximately 0.25, with the highest permeability. Towards this end, the latent space is uniformly sampled at 200×200 points using the decoder. Microstructures with a solid volume fraction within the range 0.25 ± 0.001 are then identified; see fig. 10. Among these, the microstructure with the highest value of the trace of permeability tensor, i.e., highest $\hat{C}_{00} + \hat{C}_{11}$, is selected [99]. The chosen microstructure has the following shape parameters $\boldsymbol{M^*} = \{a = 0.7158, b = 0.3757, m = 0.6039, n_1 = 1.4787, n_2 = 0.4349, n_3 = 0.5857\}$. As one can observe in fig. 10, it exhibits a fish-like shape. This particular microstructure will be used in the next numerical experiment.

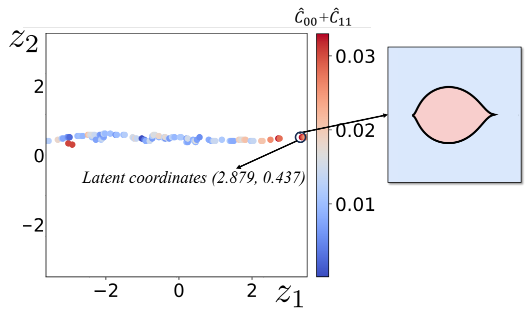

Fig. 10: Latent space coordinates of microstructures with solid volume fraction of approximately 0.25.

3.2 Bent-Pipe

For the remainder of the paper, we set the inlet velocity to 1, the mass density to 1, and viscosity to 1 [25, 100].

We now consider the bent-pipe problem proposed in [31], and illustrated in fig. 11(a). The inlet and outlet boundaries are subject to parabolic velocity conditions, of unit magnitude, and the domain is discretized into 20×60 elements. In [31], a two-scale topology optimization was carried out to minimize the dissipated power, with a constraint that the optimal microstructure must occupy exactly 25 percent of each unit cell. The reported topology is illustrated in fig. 11(b); the final dissipated power was not reported. However, as noted in [31], the computed microstructures resemble the fish-body. In [64] a GMTO approach was employed with pre-defined microstructures to achieve a similar design as depicted in fig. 11(c); the dissipated power was reported to be 16.6. Here we use the microstructure selected in the previous experiment to occupy each unit cell. Only the orientation of the microstructure in each cell is optimized. The resulting design is illustrated in fig. 11(d) with the final dissipated power of 15.1. This experiment highlights that super-shapes sampled via the decoder can generate high-performing microstructural designs.

3.3 Microstructure Variation

We continue with the previous experiment, but we will now allow the shape and size of microstructures to vary across the domain. A global volume constraint of 0.75 is imposed instead of a unit-cell volume constraint. The resulting design is

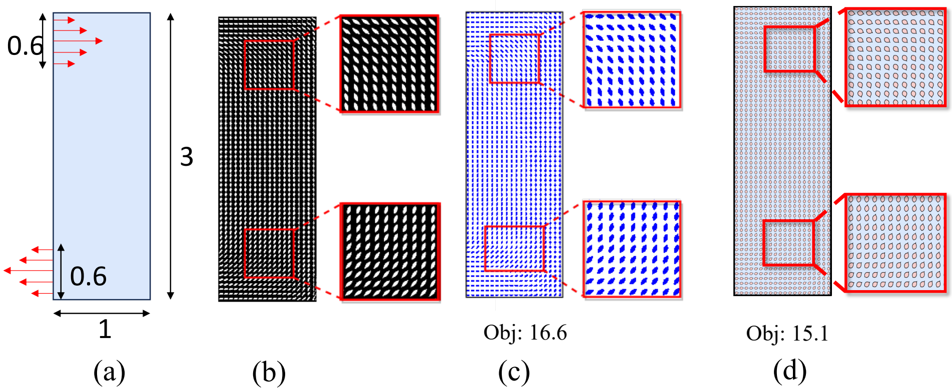

Fig. 11: Validation: (a) Problem definition. (b) Solution reported in [31]. (c) Solution reported in [64]. (d) Topology generated via the proposed method.

illustrated in fig. 12a with a dissipated power of 9.61, i.e., the performance improves with increased design space, as expected. The contact area for this particular design happens to be 75.69.

Finally, instead of imposing a volume constraint, we impose a contact area constraint of 75.69, and optimize the design. The final design is illustrated in fig. 12b, with a dissipated power of 7.56, i.e., further improvement in performance is achieved for the same contact area. The presence of small solid islands can be attributed to the absence of penalization in our algorithm.

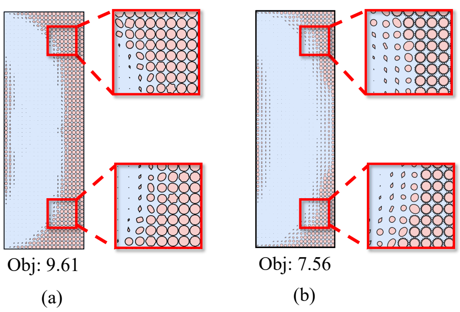

Fig. 12: Optimized design with microstructure variation with: (a) volume constraint, and (b) contact area constraint.

Note that the dissipated power of 7.56 and contact area of 75.69 for the design in fig. 12b are computed using the decoder. For validation, we re-computed the true values using a global FEA/homogenization of the final design. The dissipated power was found to be 7.87 and the contact area was 78.49, i.e., the decoder-reconstruction errors are relatively small.

3.4 Convergence

In this experiment, we demonstrate the typical convergence of the proposed algorithm using a diffuser problem, as shown in Figure fig. 13(a). The maximum inlet velocity imposed is 1 unit, and the outlet velocity is 3 units. The desired contact area was set to 60. The convergence of the dissipated power, contact area, and the evolving topologies are illustrated in fig. 13(b). We observed a stable convergence using the simple penalty formulation and Adam optimizer. Similar convergence behavior was observed for other examples as well.

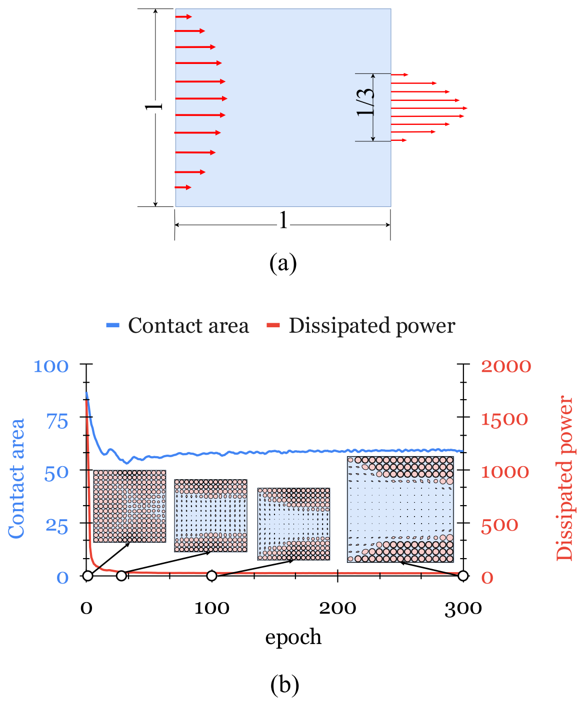

Fig. 13: Convergence of dissipated power and topologies for a diffuser problem. Topologies are illustrated at the 0th, 20th, 100th, and 300th (final) iterations.

3.5 Pareto trade-off

Understanding the trade-off between the objective (dissipated power) and constraint (contact area) through exploration of the Pareto-front is crucial in making informed design choices. In this study, we considered the diffuser problem in fig. 13(a), using the entire design space of microstructures. We computed the optimal topologies for different contact area constraints. Figure 14 illustrates that dissipated power increases with increasing contact area, as expected.

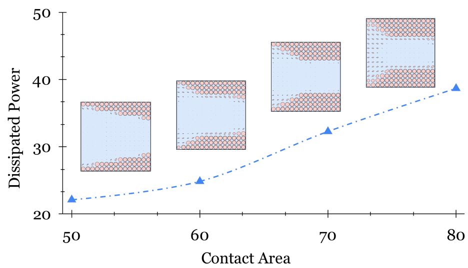

Fig. 14: Pareto curve of dissipated power against contact area.

Once again, to determine the accuracy of decoder-reconstruction, we considered the design at the left-bottom corner in fig. 14. For this design, and using the values predicted by the decoder we obtain a dissipated power of 22.13 and a contact area of 50. Further, we reconstructed the microstructures using the shape parameters predicted by the decoder. This is then used to compute the *actual* homogenized matrices. Performing an analysis using these values, we obtain a dissipated power of 22.72 and a contact area of 51.08. This corresponds to an error of 2.6% and 2.9% for the dissipated power and contact area respectively. In addition, for a contact area constraint 60, we compare the velocity magnitude obtained through a full-scale fluid flow simulation using Ansys [101] (see fig. 15(a)), versus the proposed framework (see fig. 15(b)). The maximum velocity magnitude obtained using Ansys is 2.87, whereas using the proposed method results in a value of 2.81.

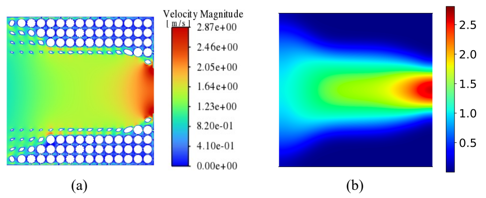

Fig. 15: Velocity magnitude plot of diffuser problem defined in fig. 13(a): (a) using Ansys and (b) proposed homogenization approach.

3.6 Computational cost

One of the central hypotheses of this paper is that the proposed offline decoder-based framework offers a significant computational advantage over concurrent homogenization-based optimization. Here, we report the computational costs to validate this claim.

The offline homogenization and data-generation of 7000 microstructures (algorithm 1) required 164 minutes, while the VAE training (algorithm 2) required 90 minutes, i.e., the total one-time overhead is around 250 minutes. The optimization of the bent-pipe (using a grid size of 20×60) took 32 minutes (300 iterations), while the optimization of the diffuser (using a grid size of 15×15) took 1.5 minutes (300 iterations).

We now consider a hypothetical scenario of concurrent homogenization. From the above data, observe that the time required for each homogenization is 164/7000, i.e., 1.4 seconds. Now for the concurrent homogenization of the bent-pipe, one must carry out homogenization over each cell within the grid of 20×60 over 300 iterations, the expected optimization time is at least $1.4 \times 1200 \times 300/60$, i.e., 8400 minutes. Similarly, for the diffuser, the expected optimization time is at least $1.4 \times 225 \times 300/60$, i.e., 1575 minutes. Thus, the proposed offline decoder-based method is computationally far superior.

3.7 Fabrication

To demonstrate the manufacturability of the designs produced by our framework, we consider a design domain with boundary conditions, as depicted in fig. 16 (a). In this example, we maintain a $1:1$ ratio and a $3:1$ ratio between the magnitude of the outlet velocity profiles and the inlet velocity profile. Additionally, we enforce a contact area constraint of 70, resulting in the design showcased in fig. 16 (b). To ensure that these designs can be manufactured successfully, we impose a minimum area constraint on each microstructure. One can also post-process the obtained design to remove microstructures with small volume fractions (as can be seen in fig. 13(b)). The final 3D printed part is illustrated in fig. 16 (c).

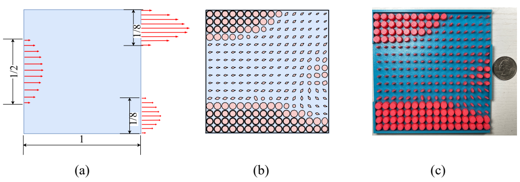

Fig. 16: (a) Design domain with boundary conditions, (b) optimized design, (c) 3d printed design

4 Conclusion

In this paper, we presented a novel multi-scale fluid flow topology optimization framework using super-shape microstructures. An offline homogenization, along with the training of a VAE was used to generate a continuous and differentiable latent space of microstructural properties. This was followed by global optimization, where the dissipated power was minimized subject to contact area (or volume) constraint.

The numerical results demonstrate that the proposed method is computationally far superior to concurrent homogenization, with minimal loss in accuracy. Furthermore, super-shapes increase the design space, yielding superior design compared to pre-defined microstructures.

Future research includes extending the framework to (1) high Reynolds flow, (2) thermo-fluid applications, where the contact area is determined indirectly via heat transfer, and (3) structural applications where microstructures with a genus greater than zero are desirable, and connectivity is also critical. Experimental validation, extension to 3D, and imposition of additional manufacturing constraints are also desirable.

Acknowledgments and Funding Information

The University of Wisconsin, Madison Graduate School supported this work. The authors extend their thanks to Subodh Subedi for his assistance in the 3D printing process.

Statements and Declarations

Compliance with ethical standards The authors declare that they have no conflict of interest.

Replication of Results The Python code is available at github.com/UW-ERSL/TOMAS. An implementation can also be found in the supplementary section of the paper.

References

[1] Alexandersen, J. & Andreasen, C. S. A review of topology optimisation for fluid-based problems. *Fluids* **5**, 29 (2020).

[2] Nagrath, S. *et al.* Isolation of rare circulating tumour cells in cancer patients by microchip technology. *Nature* **450**, 1235–1239 (2007).

[3] Fan, Z. H. *et al.* Dynamic dna hybridization on a chip using paramagnetic beads. *Analytical chemistry* **71**, 4851–4859 (1999).

[4] Hayes, M. A., Polson, N. A., Phayre, A. N. & Garcia, A. A. Flow-based microimmunoassay. *Analytical chemistry* **73**, 5896–5902 (2001).

[5] Jiang, G. & Harrison, D. J. mrna isolation in a microfluidic device for eventual integration of cdna library construction. *Analyst* **125**, 2176–2179 (2000).

[6] Liu, Y.-J. *et al.* A micropillar-integrated smart microfluidic device for specific capture and sorting of cells. *Electrophoresis* **28**, 4713–4722 (2007).

[7] Choi, J.-W. *et al.* An integrated microfluidic biochemical detection system for protein analysis with magnetic bead-based sampling capabilities. *Lab on a Chip* **2**, 27–30 (2002).

[8] Zhu, Y. *et al.* Prediction and characterization of dry-out heat flux in micropillar wick structures. *Langmuir* **32**, 1920–1927 (2016).

[9] Guo, D. *et al.* Multiphysics modeling of a micro-scale stirling refrigeration system. *International journal of thermal sciences* **74**, 44–52 (2013).

[10] Moran, M., Wesolek, D., Berhane, B. & Rebello, K. *Microsystem cooler development*, 5611 (2004).

[11] Bixler, G. D. & Bhushan, B. Bioinspired rice leaf and butterfly wing surface structures combining shark skin and lotus effects. *Soft matter* **8**, 11271–11284 (2012).

[12] Bixler, G. D. & Bhushan, B. Fluid drag reduction and efficient self-cleaning with rice leaf and butterfly wing bioinspired surfaces. *Nanoscale* **5**, 7685–7710 (2013).

[13] Huang, X. *et al.* Review on optofluidic microreactors for artificial photosynthesis. *Beilstein journal of nanotechnology* **9**, 30–41 (2018).

[14] Li, L. *et al.* High surface area optofluidic microreactor for redox mediated photocatalytic water splitting. *International journal of hydrogen energy* **39**, 19270–19276 (2014).

[15] Lauder, G. V. *et al.* Structure, biomimetics, and fluid dynamics of fish skin surfaces. *Physical Review Fluids* **1**, 060502 (2016).

[16] Bocanegra Evans, H., Gorumlu, S., Aksak, B., Castillo, L. & Sheng, J. Holographic microscopy and microfluidics platform for measuring wall stress and 3d flow over surfaces textured by micro-pillars. *Scientific reports* **6**, 1–12 (2016).

[17] Borrvall, T. & Petersson, J. Topology optimization of fluids in stokes flow. *International journal for numerical methods in fluids* **41**, 77–107 (2003).

[18] Gersborg-Hansen, A., Sigmund, O. & Haber, R. B. Topology optimization of channel flow problems. *Structural and multidisciplinary optimization* **30**, 181–192 (2005).

[19] Guest, J. K. & Prévost, J. H. Topology optimization of creeping fluid flows using a darcy–stokes finite element. *International Journal for Numerical Methods in Engineering* **66**, 461–484 (2006).

[20] Haubner, J., Neumann, F. & Ulbrich, M. A novel density based approach for topology optimization of stokes flow. *SIAM Journal on Scientific Computing* **45**, A338–A368 (2023).

[21] Jensen, K. E. Topology optimization of stokes flow on dynamic meshes using simple optimizers. *Computers & Fluids* **174**, 66–77 (2018).

[22] Li, H. *et al.* Topology optimization for lift–drag problems incorporated with distributed unstructured mesh adaptation. *Structural and Multidisciplinary Optimization* **65**, 222 (2022).

[23] Wiker, N., Klarbring, A. & Borrvall, T. Topology optimization of regions of darcy and stokes flow. *International journal for numerical methods in engineering* **69**, 1374–1404 (2007).

[24] Shen, C., Hou, L., Zhang, E. & Lin, J. Topology optimization of three-phase interpolation models in darcy-stokes flow. *Structural and Multidisciplinary Optimization* **57**, 1663–1677 (2018).

[25] Pereira, A., Talischi, C., Paulino, G. H., M Menezes, I. F. & Carvalho, M. S. Fluid flow topology optimization in polytop: stability and computational implementation. *Structural and Multidisciplinary Optimization* **54**, 1345–1364 (2016).

[26] Aage, N., Poulsen, T. H., Gersborg-Hansen, A. & Sigmund, O. Topology optimization of large scale stokes flow problems. *Structural and Multidisciplinary Optimization* **35**, 175–180 (2008).

[27] Liu, J. *et al.* A marker-and-cell method for large-scale flow-based topology optimization on gpu. *Structural and Multidisciplinary Optimization* **65**, 125 (2022).

[28] Du, T. *et al.* Functional optimization of fluidic devices with differentiable stokes flow. *ACM Transactions on Graphics (TOG)* **39**, 1–15 (2020).

[29] Li, Y. *et al.* Fluidic topology optimization with an anisotropic mixture model. *ACM Transactions on Graphics (TOG)* **41**, 1–14 (2022).

[30] Garcke, H. & Hecht, C. in *A phase field approach for shape and topology optimization in stokes flow* 103–115 (Springer, 2015).

[31] Wu, T. *Topology Optimization of Multiscale Structures Coupling Fluid, Thermal and Mechanical Analysis.* Ph.D. thesis, Purdue University Graduate School (2019).

[32] Allaire, G., Bonnetier, E., Francfort, G. & Jouve, F. Shape optimization by the homogenization method. *Numerische Mathematik* **76**, 27–68 (1997).

[33] Allaire, G. & Kohn, R. V. Optimal design for minimum weight and compliance in plane stress using extremal microstructures. *European journal of mechanics. A. Solids* **12**, 839–878 (1993).

[34] Groen, J. P. & Sigmund, O. Homogenization-based topology optimization for high-resolution manufacturable microstructures. *International Journal for Numerical Methods in Engineering* **113**, 1148–1163 (2018).

[35] Wu, J., Sigmund, O. & Groen, J. P. Topology optimization of multi-scale structures: a review. *Structural and Multidisciplinary Optimization* **63**, 1455–1480 (2021).

[36] Allaire, G., Geoffroy-Donders, P. & Pantz, O. Topology optimization of modulated and oriented periodic microstructures by the homogenization method. *Computers & Mathematics with Applications* **78**, 2197–2229 (2019).

[37] Coelho, P. G., Fernandes, P. R., Guedes, J. M. & Rodrigues, H. C. A hierarchical model for concurrent material and topology optimisation of three-dimensional structures. *Structural and Multidisciplinary Optimization* **35**, 107–115 (2008).

[38] Xia, L. & Breitkopf, P. Concurrent topology optimization design of material and structure within fe2 nonlinear multiscale analysis framework. *Computer Methods in Applied Mechanics and Engineering* **278**, 524–542 (2014).

[39] Zhou, S. & Li, Q. Design of graded two-phase microstructures for tailored elasticity gradients. *Journal of Materials Science* **43**, 5157–5167 (2008).

[40] Pantz, O. & Trabelsi, K. A post-treatment of the homogenization method for shape optimization. *SIAM Journal on Control and Optimization* **47**, 1380–1398 (2008).

[41] Dede, E. M., Zhou, Y. & Nomura, T. Inverse design of microchannel fluid flow networks using turing pattern dehomogenization. *Structural and Multidisciplinary Optimization* **62**, 2203–2210 (2020).

[42] Dede, E. M. *et al.* Measurement of low reynolds number flow emanating from a turing pattern microchannel array using a modified bernoulli equation technique. *Experimental Thermal and Fluid Science* **139**, 110722 (2022).

[43] Hankins, S. N., Zhou, Y., Lohan, D. J. & Dede, E. M. Generative design of large-scale fluid flow structures via steady-state diffusion-based dehomogenization. *Scientific Reports* **13**, 14344 (2023).

[44] Jung, T., Lee, J., Nomura, T. & Dede, E. M. Inverse design of three-dimensional fiber reinforced composites with spatially-varying fiber size and orientation using multiscale topology optimization. *Composite Structures* **279**, 114768 (2022).

[45] Nomura, T. *et al.* Inverse design of structure and fiber orientation by means of topology optimization with tensor field variables. *Composites Part B: Engineering* **176**, 107187 (2019).

[46] Lee, J. *et al.* Design of spatially-varying orthotropic infill structures using multiscale topology optimization and explicit de-homogenization. *Additive Manufacturing* **40**, 101920 (2021).

[47] Feppon, F. Multiscale topology optimization of modulated fluid microchannels based on asymptotic homogenization. *Computer Methods in Applied Mechanics and Engineering* **419**, 116646 (2024).

[48] Zhou, Y., Lohan, D. J., Zhou, F., Nomura, T. & Dede, E. M. Inverse design of microreactor flow fields through anisotropic porous media optimization and dehomogenization. *Chemical Engineering Journal* **435**, 134587 (2022).

[49] Nguyen, C. H. P. & Choi, Y. Multiscale design of functionally graded cellular structures for additive manufacturing using level-set descriptions. *Structural and Multidisciplinary Optimization* **64**, 1983–1995 (2021).

[50] Zhao, R., Zhao, J. & Wang, C. Stress-constrained multiscale topology optimization with connectable graded microstructures using the worst-case analysis. *International Journal for Numerical Methods in Engineering* **123**, 1882–1906 (2022).

[51] Zheng, L., Kumar, S. & Kochmann, D. M. Data-driven topology optimization of spinodoid metamaterials with seamlessly tunable anisotropy. *Computer Methods in Applied Mechanics and Engineering* **383**, 113894 (2021).

[52] Wang, L., Tao, S., Zhu, P. & Chen, W. Data-driven topology optimization with multiclass microstructures using latent variable gaussian process. *Journal of Mechanical Design* **143** (2021).

[53] Wang, L. *et al.* Data-driven multiscale design of cellular composites with multiclass microstructures for natural frequency maximization. *Composite Structures* **280**, 114949 (2022).

[54] Watts, S., Arrighi, W., Kudo, J., Tortorelli, D. A. & White, D. A. Simple, accurate surrogate models of the elastic response of three-dimensional open truss micro-architectures with applications to multiscale topology design. *Structural and Multidisciplinary Optimization* **60**, 1887–1920 (2019).

[55] White, D. A., Arrighi, W. J., Kudo, J. & Watts, S. E. Multiscale topology optimization using neural network surrogate models. *Computer Methods in Applied Mechanics and Engineering* **346**, 1118–1135 (2019).

[56] Wang, Y., Xu, H. & Pasini, D. Multiscale isogeometric topology optimization for lattice materials. *Computer Methods in Applied Mechanics and Engineering* **316**, 568–585 (2017).

[57] Chandrasekhar, A., Sridhara, S. & Suresh, K. Graded multiscale topology optimization using neural networks. *Advances in Engineering Software* **175**, 103359 (2023).

[58] Li, D., Dai, N., Tang, Y., Dong, G. & Zhao, Y. F. Design and optimization of graded cellular structures with triply periodic level surface-based topological shapes. *Journal of Mechanical Design* **141** (2019).

[59] Zhou, Z., Zhu, Y. & Guo, X. Machine learning based asymptotic homogenization and localization: Predictions of key local behaviors of multiscale configurations bearing microstructural varieties. *International Journal for Numerical Methods in Engineering* **124**, 639–669 (2023).

[60] Ma, C. *et al.* Compliance minimisation of smoothly varying multiscale structures using asymptotic analysis and machine learning. *Computer Methods in Applied Mechanics and Engineering* **395**, 114861 (2022).

[61] Geng, D., Wei, C., Liu, Y. & Zhou, M. Concurrent topology optimization of multiscale cooling channels with inlets and outlets. *Structural and Multidisciplinary Optimization* **65**, 234 (2022).

[62] Takezawa, A., Zhang, X., Kato, M. & Kitamura, M. Method to optimize an additively-manufactured functionally-graded lattice structure for effective liquid cooling. *Additive Manufacturing* **28**, 285–298 (2019).

[63] Takezawa, A., Zhang, X. & Kitamura, M. Optimization of an additively manufactured functionally graded lattice structure with liquid cooling considering structural performances. *International Journal of Heat and Mass Transfer* **143**, 118564 (2019).

[64] Padhy, R. K., Chandrasekhar, A. & Suresh, K. Fluto: Graded multi-scale topology optimization of large contact area fluid-flow devices using neural networks. *Engineering with Computers* 1–17 (2023).

[65] Chan, Y.-C., Da, D., Wang, L. & Chen, W. Remixing functionally graded structures: data-driven topology optimization with multiclass shape blending. *Structural and Multidisciplinary Optimization* **65**, 135 (2022).

[66] Andreasen, C. S. *Multiscale topology optimization of solid and fluid structures* (DTU Mechanical Engineering, 2011).

[67] Sanchez-Palencia, E. Fluid flow in porous media. *Non-homogeneous media and vibration theory* 129–157 (1980).

[68] Gielis, J. A generic geometric transformation that unifies a wide range of natural and abstract shapes. *American journal of botany* **90**, 333–338 (2003).

[69] Fougerolle, Y. D., Gribok, A., Foufou, S., Truchetet, F. & Abidi, M. A. Boolean operations with implicit and parametric representation of primitives using r-functions. *IEEE Transactions on Visualization and Computer Graphics* **11**, 529–539 (2005).

[70] Barr, A. H. Global and local deformations of solid primitives. *ACM Siggraph Computer Graphics* **18**, 21–30 (1984).

[71] Kingma, D. P., Welling, M. *et al.* An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning* **12**, 307–392 (2019).

[72] Gillies, S. *et al.* Shapely (2022). URL https://doi.org/10.5281/zenodo.7428463.

[73] Andreassen, E. & Andreasen, C. S. How to determine composite material properties

using numerical homogenization. *Computational Materials Science* **83**, 488–495 (2014).

[74] Wang, Y., Sun, S. & Yu, B. On full-tensor permeabilities of porous media from numerical solutions of the navier-stokes equation. *Advances in Mechanical Engineering* **5**, 137086 (2013).

[75] Lang, P., Paluszny, A. & Zimmerman, R. Permeability tensor of three-dimensional fractured porous rock and a comparison to trace map predictions. *Journal of Geophysical Research: Solid Earth* **119**, 6288–6307 (2014).

[76] Vianna, R. S., Cunha, A. M., Azeredo, R. B., Leiderman, R. & Pereira, A. Computing effective permeability of porous media with fem and micro-ct: An educational approach. *Fluids* **5**, 16 (2020).

[77] Oliphant, T. E. *et al.* *Guide to numpy* Vol. 1 (Trelgol Publishing USA, 2006).

[78] Kingma, D. P. & Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[79] Doersch, C. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908* (2016).

[80] Wang, L. *et al.* Deep generative modeling for mechanistic-based learning and design of metamaterial systems. *Computer Methods in Applied Mechanics and Engineering* **372**, 113377 (2020).

[81] Gladstone, R. J., Nabian, M. A., Keshavarzzadeh, V. & Meidani, H. Robust topology optimization using variational autoencoders. *arXiv preprint arXiv:2107.10661* (2021).

[82] Heaton, J. Ian goodfellow, yoshua bengio, and aaron courville: Deep learning: The mit press, 2016, 800 pp, isbn: 0262035618. *Genetic Programming and Evolvable Machines* **19**, 305–307 (2018).

[83] Higgins, I. *et al.* *beta-vae: Learning basic visual concepts with a constrained variational framework* (2016).

[84] Schmidhuber, J. Deep learning in neural networks: An overview. *Neural networks* **61**, 85–117 (2015).

[85] Rautela, M., Senthilnath, J., Huber, A. & Gopalakrishnan, S. Towards deep generation of guided wave representations for composite materials. *IEEE Transactions on Artificial Intelligence* (2022).

[86] Bochev, P. & Lehoucq, R. B. On the finite element solution of the pure neumann problem. *SIAM review* **47**, 50–66 (2005).

[87] Svanberg, K. The method of moving asymptotes,Äîa new method for structural optimization. *International journal for numerical methods in engineering* **24**, 359–373 (1987).

[88] Rozvany, G. I. *Structural design via optimality criteria: the Prager approach to structural optimization* Vol. 8 (Springer Science & Business Media, 2012).

[89] Chandrasekhar, A. & Suresh, K. Tounn: Topology optimization using neural networks. *Structural and Multidisciplinary Optimization* **63**, 1135–1149 (2021).

[90] Paszke, A. *et al.* in *Pytorch: An imperative style, high-performance deep learning library* (eds Wallach, H. *et al.*) *Advances in Neural Information Processing Systems 32* 8024–8035 (Curran Associates, Inc., 2019). URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-lear pdf.

[91] Rahaman, N. *et al.* *On the spectral bias of neural networks*, 5301–5310 (PMLR, 2019).

[92] Tancik, M. *et al.* Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems* **33**, 7537–7547 (2020).

[93] Chandrasekhar, A. & Suresh, K. Approximate length scale filter in topology optimization using fourier enhanced neural networks. *Computer-Aided Design* **150**, 103277 (2022).

[94] Maas, A. L., Hannun, A. Y., Ng, A. Y. *et al.* *Rectifier nonlinearities improve neural network acoustic models*, Vol. 30, 3 (Atlanta, Georgia, USA, 2013).

[95] Glorot, X. & Bengio, Y. *Understanding the difficulty of training deep feedforward neural networks*, 249–256 (JMLR Workshop and Conference Proceedings, 2010).

[96] Wright, S. J. *Numerical optimization* (2006).

[97] Chandrasekhar, A., Sridhara, S. & Suresh, K. Auto: a framework for automatic differentiation in topology optimization. *Structural and Multidisciplinary Optimization* **64**, 4355–4365 (2021).

[98] Vasilev, I., Slater, D., Spacagna, G., Roelants, P. & Zocca, V. *Python Deep Learning: Exploring deep learning techniques and neural network architectures with Pytorch, Keras, and TensorFlow* (Packt Publishing Ltd, 2019).

[99] Liakopoulos, A. C. Darcy's coefficient of permeability as symmetric tensor of second rank. *Hydrological Sciences Journal* **10**, 41–48 (1965).

[100] Alexandersen, J. A detailed introduction to density-based topology optimisation of fluid flow problems with implementation in matlab. *Structural and Multidisciplinary Optimization* **66**, 12 (2023).

[101] DeSalvo, G. J. & Swanson, J. A. *ANSYS Engineering Analysis System: User's Manual* (Swanson Analysis Systems, 1979).