

Data-Driven Nonlinear Deformation Design of 3D-Printable Shells

Samuel Silverman¹ Kelsey L. Snapp² Keith A. Brown^{2,3,4} Emily Whiting¹

Abstract

Designing and fabricating structures with specific mechanical properties requires understanding the intricate relationship between design parameters and performance. Understanding the design-performance relationship becomes increasingly complicated for nonlinear deformations. Though successful at modeling elastic deformations, simulation-based techniques struggle to model large elastoplastic deformations exhibiting plasticity and densification. We propose a neural network trained on experimental data to learn the design-performance relationship between 3D-printable shells and their compressive force-displacement behavior. Trained on thousands of physical experiments, our network aids in both forward and inverse design to generate shells exhibiting desired elastoplastic and hyperelastic deformations. We validate a subset of generated designs through fabrication and testing. Furthermore, we demonstrate the network's inverse design efficacy in generating custom shells for several applications.

Keywords: additive manufacturing, neural networks, inverse design, elastoplasticity, hyperelasticity

1 Introduction

Additive manufacturing has unlocked the ability to create structures with complex geometries and customized mechanical properties. These fabricated structures can be designed to exhibit unique stiffness variations[1, 2] and energy-absorbing capabilities[3, 4]. However, achieving specific mechanical behaviors, especially for large deformations involving significant plasticity and densification, demands a deep comprehension of the intricate relationship between design parameters and performance. Gaining insights through manual iterative design and testing often proves impractical and leads to costly and time-consuming design cycles. Researchers have employed self-driving labs[4, 5, 6] to explore design spaces autonomously. However, these systems are constrained by cost, complexity, and converging time.

Simulation techniques like the finite element method (FEM) and homogenization excel at modeling elasticity[1, 2, 7, 8, 9] and fracture[10, 11]. However, such strategies often lose accuracy when representing large plastic deformations, impeding the design of structures with targeted elastoplastic behaviors. Researchers have also developed plasticity simulations to achieve highly complex deformation behavior [12, 13, 14]. However, further testing is needed to evaluate these methods' ability to model the compressive behavior of thin shell structures as used in this study. Consequently, we propose a neural network trained on experimental data to learn the design-performance relationship between 3D-printable shells and their compressive deformation behavior.

Forward design presents users with predicted performance, allowing them to manipulate designs to achieve desired behavior. Data-driven approaches to predict mechanical behavior from material geometries have been applied in various fields, from composites[10, 15, 16] to material microstructures[17, 18]. However, iterative design loops with forward design are often ineffective due to the vastness of design spaces and the complexity of how individual design parameters affect performance.

On the contrary, inverse design directly identifies the designs that achieve a target performance goal. Inverse design is inherently complex; one performance is likely achievable by numerous designs, making learning algorithm convergence difficult. This one-to-many challenge mirrors complexities from other disciplines, like inverse scattering[19] and inverse kinematics problems[20]. Despite this increased complexity, inverse design empowers users to explore and generate designs with desired mechanical properties.

¹Department of Computer Science, Boston University, Boston, MA, USA.

²Department of Mechanical Engineering, Boston University, Boston, MA, USA.

³Division of Materials Science & Engineering, Boston University, Boston, MA, USA.

⁴Physics Department, Boston University, Boston, MA, USA.

We propose a tandem neural network (TNN)[21] for the forward and inverse design of a parametric family of cylindrical shells chosen for their ease of fabrication (Figure 1a). The TNN combines two sequential neural networks: an inverse design network and a forward design network, structured like an autoencoder. Researchers have used this architecture for the inverse design of nanophotonic devices[21, 22] and metamaterials[23, 24, 25]. Notably, machine learning-based inverse design extends beyond the TNN[26, 27] with techniques ranging from convolutional neural networks[11] to reinforcement learning[28]. Previous work generally focuses on mechanical properties that are easily modeled with simulation. These include properties arising from reversible elastic deformations or fracture propagation from an initial predetermined fracture site.

In this paper, we leverage an extensive experimental dataset comprising over 12,000 shells exhibiting nonlinear response to compression, as observed in their force-displacement curves, capturing a range of elastoplastic and hyperelastic deformations. We verify our TNN’s performance through experimentation on generated designs, compare the TNN to alternative methodologies, and demonstrate the TNN’s effectiveness in generating designs with optimized nonlinear deformations through several applications, such as impact absorption.

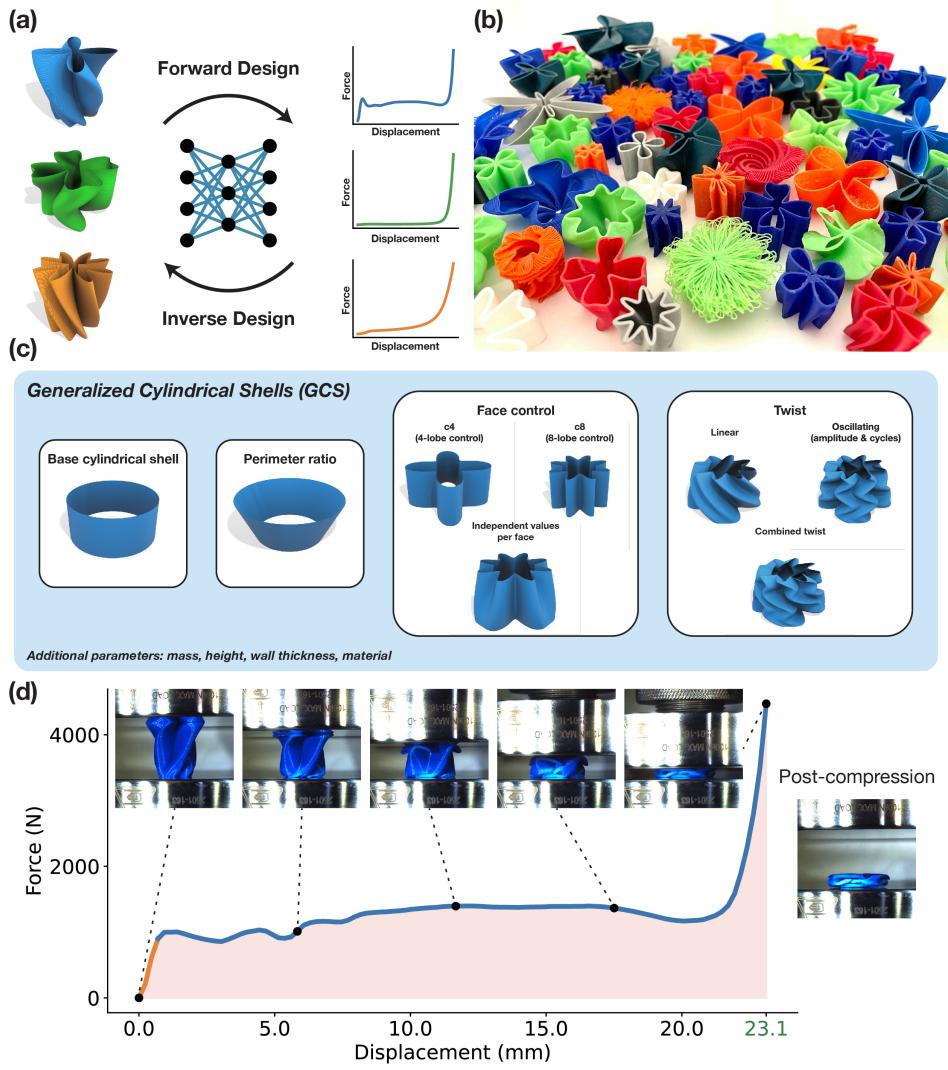


Figure 1: Overview. (a) We explore using a tandem neural network (TNN) for the forward and inverse design of generalized cylindrical shells (GCS). (b) GCS are fabricated with fused deposition modeling (FDM) 3D printers. (c) GCS geometry emerges from parameters that control a series of operations applied to cylindrical shells. The result of these operations is a diverse family of structures. (d) Compression tests applied to GCS yield force-displacement curves[6]. Fabricated with PLA, this GCS exhibits elastoplastic deformation. Highlighted metrics are the linear elastic region used to calculate stiffness (orange line), work performed (red area under curve), and maximum displacement (green value).

2 Materials and Methods

This section explains our TNN pipeline. We introduce the experimental dataset and preprocessing steps. Furthermore, we describe the network architecture, learning objectives, and training process.

2.1 Modeling Performance With Force-Displacement Curves

We modeled performance with force-displacement curves (Figure 1d) and used the following derived metrics:

- **Stiffness** (N/mm): Measures the resistance to initial deformations reflected in the slope of the linear elastic zone (orange region in Figure 1d).
- **Work** (J): Measures the energy absorption during deformation reflected in the area under the curve (red region in Figure 1d).
- **Maximum displacement** (mm): Denotes the furthest point of deformation reached during compression testing (green region in Figure 1d). It is influenced by material properties and experimental constraints. This value ensures a realistic scaling of displacements in predicted curves.

These metrics serve as high-level descriptors for evaluating model performance and selecting desired structures. However, to ensure broad applicability, we maintain the entire force-displacement curve as the underlying performance representation, allowing users to identify metrics most pertinent to their unique design challenges.

2.2 Generalized Cylindrical Shell Dataset

In a comprehensive prior study, we conducted compression testing on 3D structures known as generalized cylindrical shells (GCS) (Figure 1b) to explore their energy-absorbing capabilities[6]. These tests generated force-displacement curves, which, along with their associated designs, constitute a substantial GCS dataset¹. This dataset holds particular value due to its wide range of measured elastoplastic and hyperelastic deformations.

Here, we provide an overview of the GCS parametric family but direct readers to the Methods section in our previous work [6] for a complete description. The radius r for an azimuthal angle ϕ defines a GCS's top and base faces,

$$r(\phi) = r_0(1 + c_4 \cos(4\phi) + c_8 \cos(8\phi)), \quad (1)$$

where c_4 and c_8 control the shape and size of 4-lobe and 8-lobe features, respectively. Interpolating between the two faces forms a cohesive 3D shell. Parameters enable *linear* and *oscillating* twisting, enhancing geometry intricacy. Moreover, adjustments to the *perimeter ratio* between the top and base faces, *mass*, *height*, and *wall thickness* allow further customization. The r_0 term is a scale factor for perimeter size. Figure 1c shows how a sequence of operations on cylindrical shells defines a GCS.

With a *material* choice from one of six thermoplastics (two elastoplastic, three hyperelastic, one intermediate) the complete GCS specification requires 12 parameters (Table 1). we used the MakerGear M3 and Ultimaker S5, two fused deposition modeling (FDM) 3D printers, to fabricate GCS by printing the design in vase mode.

2.3 Data Processing

This section discusses our steps to extract the design and performance data from the GCS dataset, resulting in 12,706 design-performance pairs.

2.3.1 Performance Dimensionality Reduction

Force-displacement curves in the GCS dataset, which include thousands of points with varying spacings and displacement ranges, are impractical for prediction tasks. We processed the curve data to 100 evenly spaced displacement values. Inspired by Yang et al. [10], we used Principal Component Analysis (PCA) to condense the corresponding 100 force values into ten principal components. Our performance vectors $\mathbf{p} \in \mathbb{R}^{11}$ combine these coefficients with the maximum displacement value. Refer to supplementary §1 for details on curve compression and analysis of PCA quality.

¹<https://hdl.handle.net/2144/46687>

Table 1: **GCS design parameters.** Twelve parameters define a GCS. We manually restricted the mass, height, and perimeter ratio values so that all parameters have well-defined ranges.

Parameter	Description	Dataset Values
c_4 (base & top)	The parameter controlling the size and shape of the 4-lobe feature.	$[0, 1.2]$
c_8 (base & top)	The parameter controlling the size and shape of the 8-lobe feature.	$[-1, 1]$
linear twist	The rotation (rad) of the top. This creates a linear twist between the base and top.	$[0, 2\pi]$
oscillating twist amplitude	The amplitude (rad) of the oscillating twist between the base and top.	$[0, \pi]$
oscillating twist cycles	The number of cycles of the oscillating twist between the base and top.	$[0, 3]$
perimeter ratio	The ratio between the top and base perimeters.	$[1, 3]$
mass	The mass (g).	$[1, 5]$
height	The height (mm).	$[10, 30]$
thickness	The wall thickness (mm).	$[0.4, 1]$
material	Elastoplastic PETG PLA Hyperelastic TPE (<i>Chinchilla 75A</i>) TPU (<i>Cheetah 95A</i> , <i>NinjaFlex 85A</i>) Intermediate TPU (<i>Armadillo 75D</i>)	Material choice

2.3.2 Design Parameter Normalization

Our investigation restricted the material parameter to experiments with at least 500 data points to ensure data quality. For the GCS design parameters, we one-hot encoded the material parameter and applied min-max normalization to all non-material parameters to normalize their values. However, the mass, height, and perimeter ratio parameters lack a clear range. We manually capped these parameters' values to [1 g, 5 g] for mass, [10 mm, 30 mm] for height, and [1, 3] for perimeter ratio.

Using one-hot-encoded materials provided a straightforward way to ensure that the materials in generated designs conform to realistic values in inverse design. Although we could have parameterized the material using a subset of continuous variables (e.g., Young's modulus and Poisson's ratio), we would have needed additional mechanisms to ensure realism in the generated values during inverse design.

These normalization operations collectively result in design vectors $\mathbf{d} \in \mathbb{R}^{17}$, comprising 11 geometric parameters and six values from the one-hot encoded material parameter.

2.4 Model Architecture

Inverse design poses complexity due to the potential one-to-many relationship between different designs and similar performances, hindering conventional learning algorithm convergence [21]. Figure 2 depicts this challenge, illustrating two distinct GCS designs with nearly identical force-displacement curves. To address this, the TNN framework, initially proposed by Liu et al. [21], has emerged as a promising solution. We leveraged this framework to generate diverse GCS designs aligned with desired performances. Figure 3 shows our network architecture.

The forward design neural network $\mathcal{F} : \mathbb{R}^{17} \rightarrow \mathbb{R}^{11}$ learns the mapping from design vectors \mathbf{d} to performance vectors \mathbf{p} . The network architecture consists of six hidden layers: A linear layer followed by a ReLU activation, repeated three times. All hidden layers maintain a uniform width of 64 units, contributing 10,190 trainable parameters.

The inverse design neural network $\mathcal{I} : \mathbb{R}^{11} \rightarrow \mathbb{R}^{17}$ learns the inverse mapping: performance vectors \mathbf{p} to design vectors \mathbf{d} , and mirrors the architecture of \mathcal{F} . The final layer of \mathcal{I} ensures appropriate values for all GCS parameters by combining softmax and sigmoid activations. The softmax activation is applied to the six parameters representing the one-hot-encoded material, assuring the predicted material has

the highest value. The sigmoid activation confines predicted values of the remaining parameters within $[0, 1]$, maintaining consistency with their normalized counterparts.

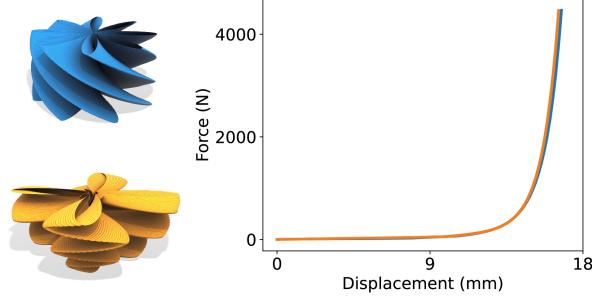


Figure 2: **One-to-many performance to design relationship.** Two GCS with distinctly different geometry share nearly identical force-displacement behavior, a common problem in inverse design.

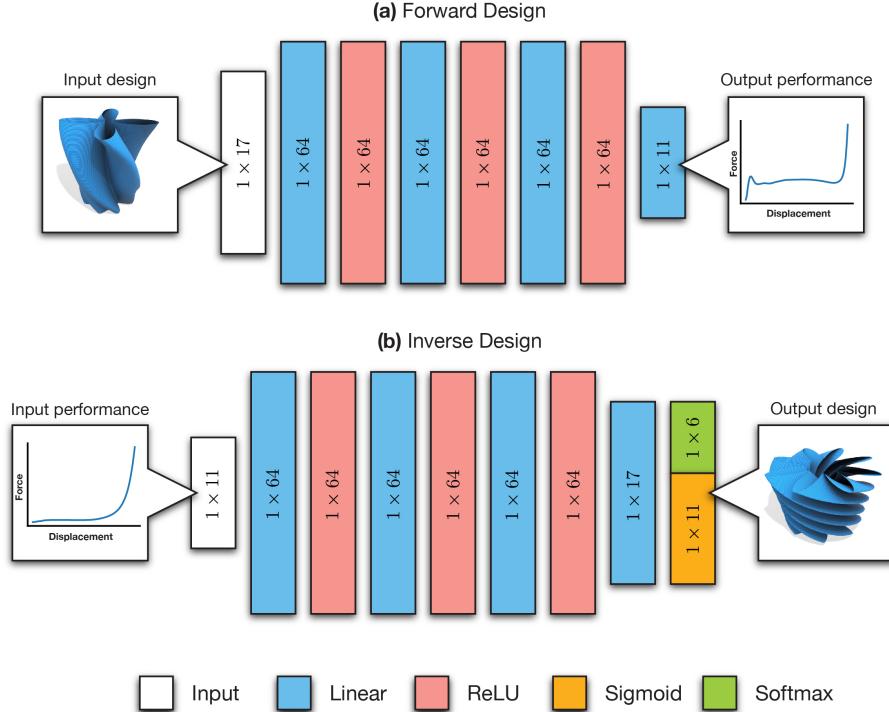


Figure 3: **TNN architecture.** (a) The forward design network (\mathcal{F}) maps GCS designs to corresponding force-displacement curves. (b) The inverse design network (\mathcal{I}) maps force-displacement curves back to GCS designs.

2.5 Objective Function

\mathcal{F} aims to minimize the error between predicted and experimental performance vectors. However, not all values in the performance vectors carry equal significance in improving the model's accuracy. Building upon insights from Yang et al. [10], we used a weighted mean squared error (MSE) loss function, $\mathcal{L}_{\mathcal{F}}$, using a weight vector $\mathbf{w} \in \mathbb{R}^{11}$:

$$\mathcal{L}_{\mathcal{F}} = \frac{1}{n} \sum_{i=1}^n \left(\mathbf{w} \cdot (\mathbf{p}^{(i)} - \mathcal{F}(\mathbf{d}^{(i)})) \right)^2, \quad (2)$$

where n is the number of samples and \mathbf{w} is defined as

$$\mathbf{w}_j = \begin{cases} \frac{\lambda_j}{\sum_{k=1}^{10} \lambda_k} & \text{if } 1 \leq j \leq 10 \\ 1 & \text{if } j = 11 \end{cases}. \quad (3)$$

By setting $\mathbf{w}_{11} = 1$, we assigned equal importance to the entries of the performance vector responsible for the displacement and force values. However, since each principal component coefficient explains a different amount of variance in the force data, as indicated by the eigenvalues $\lambda_1, \dots, \lambda_{10}$ obtained from PCA, we set the weights of individual coefficients based on their respective explained variance. By incorporating these weighted values, \mathcal{F} prioritizes the most informative principal component coefficients during training.

The objective of \mathcal{I} is twofold. First, \mathcal{I} aims to generate designs that align with desired performances. The loss function $\mathcal{L}_{\mathbf{p}}$ encourages this behavior, mirroring the form of $\mathcal{L}_{\mathcal{F}}$,

$$\mathcal{L}_{\mathbf{p}} = \frac{1}{n} \sum_{i=1}^n \left(\mathbf{w} \cdot (\mathbf{p}^{(i)} - \mathcal{F}(\mathcal{I}(\mathbf{p}^{(i)}))) \right)^2. \quad (4)$$

$\mathcal{L}_{\mathbf{p}}$ addresses the one-to-many mapping problem in inverse design by ensuring the generated GCS's predicted performance matches the target performance without directly considering the generated GCS design. Second, \mathcal{I} aims to generate printable designs. Drawing inspiration from prior work [23], we used the loss function $\mathcal{L}_{\mathbf{d}}$ to bias generated designs towards previously tested designs,

$$\mathcal{L}_{\mathbf{d}} = \frac{1}{n} \sum_{i=1}^n (\mathbf{d}^{(i)} - \mathcal{I}(\mathbf{p}^{(i)}))^2. \quad (5)$$

Specifically, $\mathcal{L}_{\mathbf{d}}$ encourages predicted designs to align with the dataset designs associated with the target performances. Such a loss is motivated by the fact that dataset designs are all known to be printable. While it might be possible to include losses that penalize parameter combinations leading to non-printable designs, we do not explore these methods here.

\mathcal{I} seeks to minimize the loss function $\mathcal{L}_{\mathcal{I}}$, which combines $\mathcal{L}_{\mathbf{d}}$ and $\mathcal{L}_{\mathbf{p}}$ in a weighted manner:

$$\mathcal{L}_{\mathcal{I}} = \mathcal{L}_{\mathbf{p}} + \alpha \mathcal{L}_{\mathbf{d}}, \quad (6)$$

where $\alpha \in \mathbb{R}$ determines the relative weight between $\mathcal{L}_{\mathbf{p}}$ and $\mathcal{L}_{\mathbf{d}}$. By fine-tuning the α value, we can control the balance between optimizing for predicted performance accuracy and maintaining proximity to dataset designs. While it may be possible to include losses that penalize parameter combinations that result in non-printable designs, we do not investigate such methods here.

2.6 Training

We divided our processed GCS dataset into training, validation, and test sets, following an 80-10-10% split. Our training process involved two stages. We trained \mathcal{F} in the initial stage. Upon completion, we froze \mathcal{F} and appended the untrained \mathcal{I} model. The second stage involved training \mathcal{I} using \mathcal{F} to aid in convergence.

Trained in this order, \mathcal{I} learns to generate GCS designs whose performance predicted by \mathcal{F} aligns with the desired performance. Had we trained \mathcal{I} independently, we could only use $\mathcal{L}_{\mathbf{d}}$ as $\mathcal{L}_{\mathbf{p}}$ depends on \mathcal{F} . Alone, $\mathcal{L}_{\mathbf{d}}$ exposes an ill-posed learning problem, as multiple GCS designs could yield similar performances. Penalizing deviations between generated and actual designs would impede convergence, rendering the learning process ineffective. While it is likely possible to train both networks simultaneously, similar generator and discriminator training in generative adversarial networks [29], we left this to future investigations.

We use the Adam optimizer [30] for training, with a learning rate of 0.001, a weight decay of 1, and a batch size of 16. We used early stopping to prevent overfitting, terminating after 500 epochs in each stage. Our TNN was implemented using PyTorch [31] and trained on an Apple MacBook Pro (M1 Max). Training both networks required less than an hour.

3 Results

We evaluated our TNN's forward and inverse design accuracy on the test set. In our evaluation, we repeated training ten times using different random splits of the data to report test outcomes with 95%

confidence intervals. We performed physical experimentation on a sample of generated GCS designs and compared the TNN performance to other methodologies. Finally, we generated GCS with tailored mechanical properties for two applications to demonstrate our TNN’s inverse design capabilities.

3.1 Forward Design Performance

We evaluated \mathcal{F} on the test set by comparing the predicted and experimental force-displacement curves. We compared the metrics of the predicted and experimental curves because using high-level metrics for evaluation provides interpretable units for measures such as mean absolute error (MAE). Notably, the TNN did not directly learn metrics like work or stiffness; they are computed based on the learned performance vectors.

Table 2 presents the MAE and R^2 for stiffness, work, and maximum displacement. Each metric’s MAE is $< 5\%$ of their respective ranges: 3.2% for stiffness, 2.8% for work, and 2.2% for maximum displacement. Furthermore, the small confidence intervals demonstrate that our TNN has training stability, with minimal reliance on model initialization or data splitting. Figure 4 shows \mathcal{F} ’s performance for eight test set GCS designs, demonstrating the network’s capability to predict the complete nonlinear force-displacement behavior of designs made with elastoplastic and hyperelastic materials. Refer to the supplementary material for the design parameters.

Table 2: Forward design performance. The test set results show the work, stiffness, and maximum displacement of predicted force-displacement curves by \mathcal{F} . We report each metric’s R^2 and mean absolute error (MAE) with 95% confidence intervals. For reference, we provide the mean and range of each metric in the dataset. The test set loss is $\mathcal{L}_{\mathcal{F}} = 0.21 \pm 0.01$.

	R^2	MAE	Mean	Range
Stiffness (N/mm)	0.66 ± 0.03	250 ± 20	815	7732
Work (J)	0.96 ± 0.002	1.3 ± 0.04	15.4	46.7
Max. displacement (mm)	0.94 ± 0.007	0.50 ± 0.03	18.6	22.6

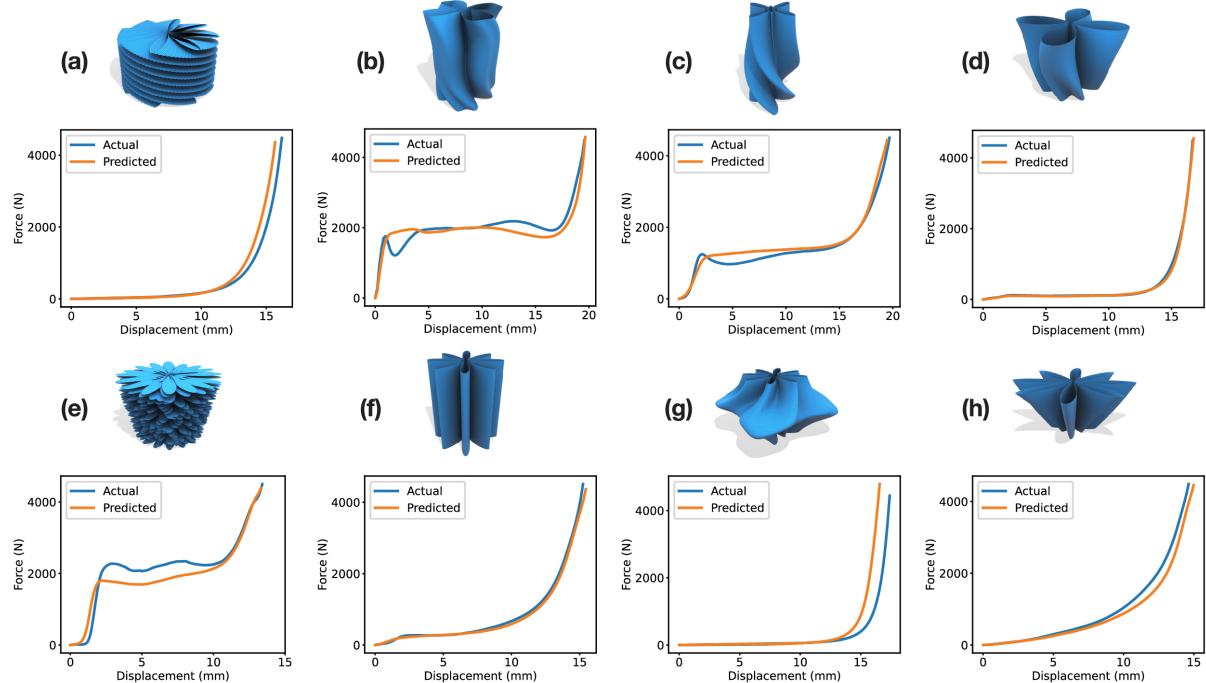


Figure 4: Forward design results. Eight randomly selected results from the test set. The GCS designs (blue) serve as input to \mathcal{F} , which predicts force-displacement curves (orange). For reference, we show the actual experimental force-displacement curves from the test set (blue). \mathcal{F} can predict nonlinear deformation behavior for elastoplastic (b, c, e) and hyperelastic (a, d, f, g, h) GCS.

3.2 Inverse Design Performance

We trained \mathcal{I} with $\alpha \in \{0, 0.01, 0.1, 1\}$, the parameter weighing the importance of generating previously tested designs. To evaluate the inverse design performance, we compare predicted designs' predicted force-displacement curves $\mathcal{F}(\mathcal{I}(\mathbf{p}))$ to the target force-displacement curves \mathbf{p} . Table 3 presents the MAE and R^2 for stiffness, work, and maximum displacement for \mathcal{I} trained with each α value.

For $\alpha \in \{0, 0.01, 0.1\}$, we observed minimal change in accuracy for the work and maximum displacement. However, for $\alpha = 1$, we saw the MAE increase for maximum displacement and decrease for work. For stiffness, the accuracy improved as we increased α . We found that different metrics derived from force-displacement curves exhibit varying sensitivity levels to the effect of \mathcal{L}_d . These findings underscore a complex trade-off between generating high-performing designs and printable designs.

Table 3: Inverse design performance. The test set results show the work, stiffness, and maximum displacement of predicted force-displacement curves by \mathcal{I} trained on $\alpha \in \{0, 0.01, 0.1, 1\}$. We compare predicted designs' predicted force-displacement curves $\mathcal{F}(\mathcal{I}(\mathbf{p}))$ to the target force-displacement curves \mathbf{p} . We report each metric's R^2 and MAE with 95% confidence intervals. The test set loss is $\mathcal{L}_{\mathcal{I}} = 0.0043 \pm 0.0005$ ($\alpha = 0$), $\mathcal{L}_{\mathcal{I}} = 0.0049 \pm 0.0005$ ($\alpha = 0.01$), $\mathcal{L}_{\mathcal{I}} = 0.0094 \pm 0.0003$ ($\alpha = 0.1$), and $\mathcal{L}_{\mathcal{I}} = 0.046 \pm 0.002$ ($\alpha = 1$).

		R²	MAE
$\alpha = 0$	Stiffness (N/mm)	0.56 ± 0.08	310 ± 30
	Work (J)	0.94 ± 0.02	1.8 ± 0.3
	Max. displacement (mm)	0.99 ± 0.002	0.12 ± 0.01
$\alpha = 0.01$	Stiffness (N/mm)	0.60 ± 0.06	300 ± 30
	Work (J)	0.92 ± 0.04	2.1 ± 0.5
	Max. displacement (mm)	0.99 ± 0.002	0.12 ± 0.01
$\alpha = 0.1$	Stiffness (N/mm)	0.69 ± 0.05	250 ± 30
	Work (J)	0.95 ± 0.01	1.7 ± 0.2
	Max. displacement (mm)	0.99 ± 0.002	0.12 ± 0.008
$\alpha = 1$	Stiffness (N/mm)	0.68 ± 0.04	240 ± 20
	Work (J)	0.98 ± 0.004	0.91 ± 0.08
	Max. displacement (mm)	0.99 ± 0.001	0.23 ± 0.01

3.2.1 Physical Validation

We randomly selected eight GCS designs generated from \mathcal{I} trained with $\alpha = 1$ and experimentally obtained their force-displacement curves to assess the accuracy of predicted performance (Figure 6). We fabricated the samples on a MakerGear M3 and performed compression testing with an Instron 5965. Given a nonlinear force-displacement curve, \mathcal{I} can create GCS designs that conform to the specified deformations. Refer to supplementary material for the design parameters.

We evaluated the printability of generated designs using the two criteria established in our previous work [6]: The base perimeter should be at least 30 mm to provide a substantial contact area with the print bed, and the shell must maintain a minimum distance of 0.01 mm from its center axis to accommodate material deposition. In Figure 5, we calculated the percentage of printable predicted designs within the test set for \mathcal{I} trained on different levels of α and observe a positive correlation. This trend would suggest that further increasing α would continue to improve printability. However, as α increases, \mathcal{L}_d dominates \mathcal{L}_p , reverting the training process to the one-to-many mapping problem outlined in Section 2.6. We did not directly investigate the value of α for which this behavior begins.

3.3 Comparison to Alternative Approaches

We evaluate the TNN against two alternative methods, k -nearest neighbors (k NN) and FEM, assessing their performance and practicality.

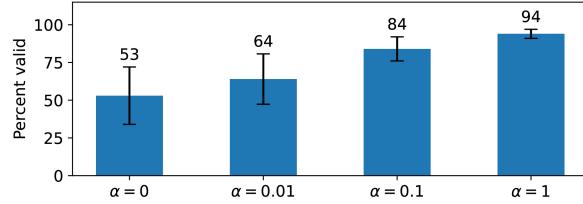


Figure 5: **GCS printability.** The percentage of GCS generated by \mathcal{I} that passes all printability checks when trained with different α values. The error bars depict 95% confidence intervals.

3.3.1 Nearest Neighbors

We trained a k NN model with $k = 1$ for forward and inverse design and evaluated its performance (Table 4). Compared to the TNN, k NN is more accurate for stiffness but less accurate for work and maximum displacement, signifying no clear best method concerning these metrics. However, k NN presents several significant limitations.

First, k NN has poor scalability. While the GCS parameterization benefits from a relatively small parameter set, the curse of dimensionality quickly becomes an issue as the parameterization becomes more complex or more materials are added. From a storage scalability perspective, k NN requires storing the entire training dataset, leading to increased model size and longer inference times.

Second, when $k > 1$, k NN interpolates between the closest designs, lacking a straightforward mechanism to ensure printability. For instance, interpolating between different one-hot-encoded materials will always produce invalid results. We have not validated that the nearest design in the normalized parameter space is the most similar. This leads to a broader question of accurately assessing “nearness” in non-Euclidean spaces, such as sparse one-hot encoded spaces or other normalized parameter spaces, which warrants its own investigation.

Third, k NN cannot transfer knowledge to structures with different parameterizations or performance representations. Transfer learning reduces experimental data requirements, making it crucial for expanding to other structures and practical applications.

Table 4: **Comparison to nearest neighbors.** We train a k -nearest neighbors (k NN) model with $k = 1$ and compare its forward and inverse design performance to the TNN. For inverse design, we display the results for the \mathcal{I} trained with $\alpha = 1$. The metrics with higher accuracy for each model are bolded.

		R^2	MAE
Forward Design			
TNN	Stiffness (N/mm)	0.66 ± 0.03	250 ± 20
	Work (J)	0.96 ± 0.002	1.3 ± 0.04
	Max. displacement (mm)	0.94 ± 0.007	0.50 ± 0.03
k NN	Stiffness (N/mm)	0.82 ± 0.02	172 ± 5
	Work (J)	0.92 ± 0.004	1.7 ± 0.04
	Max. displacement (mm)	0.90 ± 0.007	0.60 ± 0.01
Inverse Design			
TNN	Stiffness (N/mm)	0.68 ± 0.04	240 ± 20
	Work (J)	0.98 ± 0.0004	0.91 ± 0.08
	Max. displacement (mm)	0.99 ± 0.001	0.23 ± 0.01
k NN	Stiffness (N/mm)	0.92 ± 0.01	110 ± 6
	Work (J)	0.99 ± 0.001	0.62 ± 0.01
	Max. displacement (mm)	0.95 ± 0.003	0.49 ± 0.01

3.3.2 Finite Element Method

We compared the accuracy and speed of \mathcal{F} to those produced by FEM. We used Abaqus to numerically derive the compressive force-displacement curve for a GCS design. Refer to supplementary §2 for details

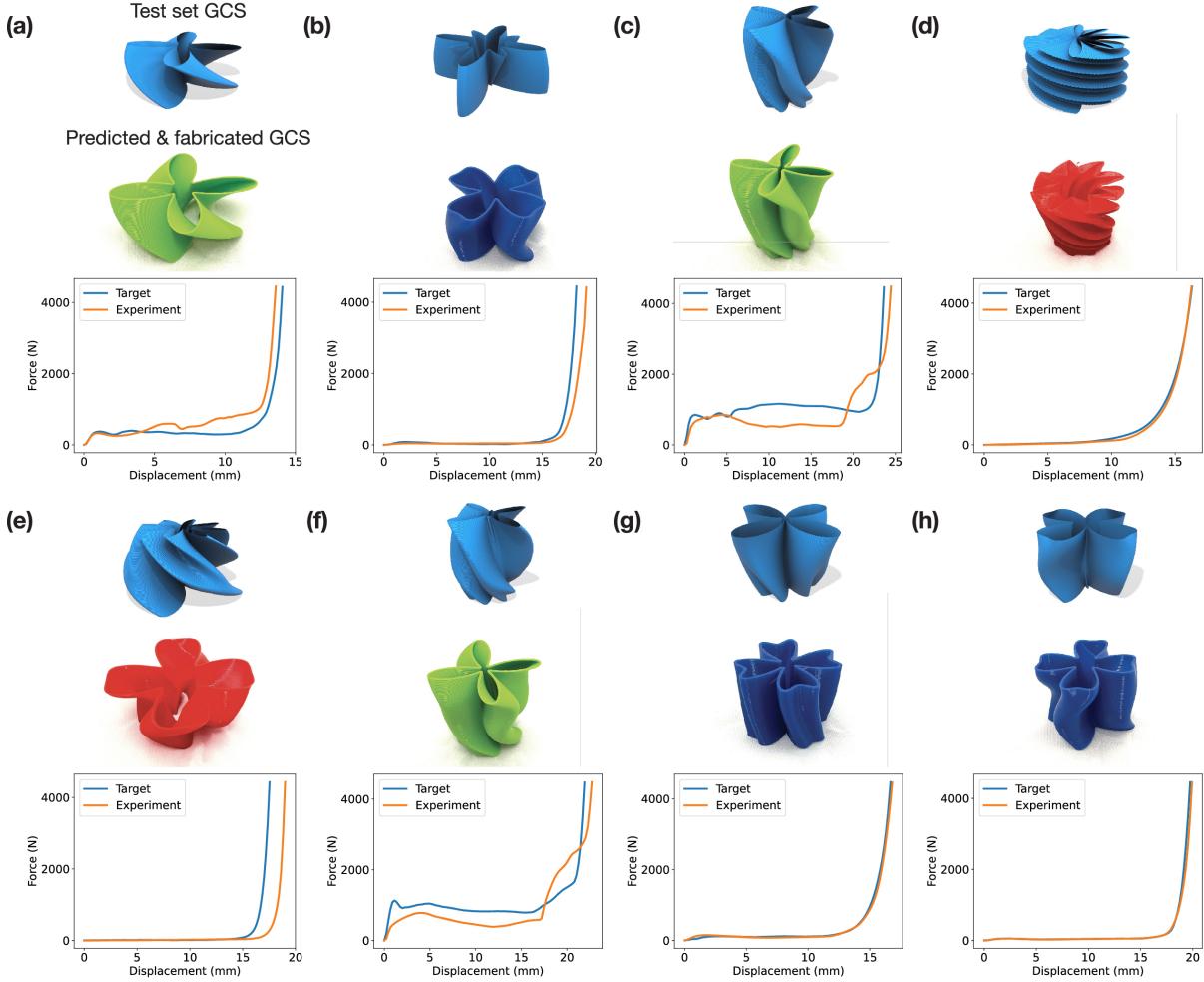


Figure 6: Inverse design results. Eight randomly selected results from the test set. The force-displacement curves (blue) serve as input to \mathcal{I} , which predicts GCS designs. We fabricated and performed compression testing on the predicted GCS designs to obtain experimental force-displacement curves (orange). \mathcal{I} can generate GCS designs that exhibit nonlinear elastoplastic (**a**, **c**, **f**) and hyperelastic (**b**, **d**, **e**, **g**, **h**) deformations. For reference, we include the test set GCS designs (blue) associated with the inputted curves to illustrate the one-to-many relationship between performance and designs. Generated designs differ, sometimes significantly, from their test set counterparts.

on the simulation setup.

Figure 7 presents the force-displacement curves obtained from Abaqus and \mathcal{F} . FEM accurately portrayed the linear elastic force-displacement relationship, but the computation began to lose accuracy for the nonlinear plastic deformations. The FEM simulation ultimately terminated prematurely and failed to converge beyond a fraction of the total experimental displacement. In this plastic region of compression, the numerous self-collisions and tearings presented computational challenges for FEM that were not easily addressed.

Experimental results for a GCS design can be obtained in 25 minutes (10 minutes for fabrication and 15 minutes for compression testing). Our TNN has inference times of < 20 ms with under one hour to train. In comparison, the compression test simulation time was 74 minutes on an 8-core CPU with 32 GB of RAM.

We used experimental data instead of simulated data to explore predictive capabilities, as a publicly available dataset capturing the mechanical behavior of interest exists. However, if such data is not accessible, using synthetic datasets from simulation to train machine learning models is a common strategy, provided that the simulation methods accurately model the behavior of interest.

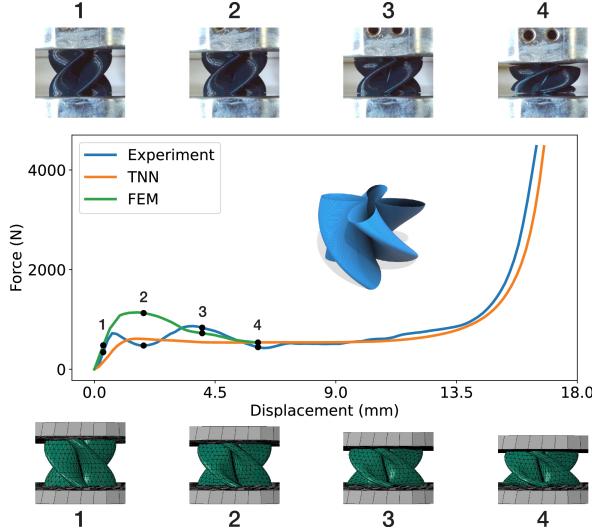


Figure 7: **Comparison to FEM.** We display the force-displacement curves for a GCS obtained through experimentation (blue), predicted by \mathcal{F} (orange), and simulated via FEM (green). Four points along the curves indicate the deformations from experimentation (top) and FEM (bottom).

3.4 Applications

We used \mathcal{I} to generate GCS with mechanical behavior tailored to two applications: impact absorption and material emulation. Refer to supplementary §3 for more details on each application.

3.4.1 Impact Absorption

An impact-absorbing structure must absorb the total impact energy while containing peak forces within specified limits to prevent damage or injury. Given a force threshold F , we optimized for a force-displacement curve meeting (or exceeding) a target energy absorption E to use as input for \mathcal{I} ,

$$\arg \min_{\text{valid } \mathbf{p}} E - E_F(\mathbf{p}), \quad (7)$$

where $E_F(\mathbf{p})$ denotes the calculated energy absorption (work) before exceeding F (Figure 8a).

We identified GCS optimized for impact absorption in the context of the egg drop test (Figure 8b). This test involves constructing padding to protect an egg from breaking during a substantial fall. In our experimental setup, we dropped eggs from 50 cm onto a pad containing four GCS parts. We set $F = 10$ N and $E = 0.0735$ J for the target force-displacement curve. Using \mathcal{I} , we optimized for a GCS design that absorbs the impact energy of the drop without breaking the egg. We tested three setups, each with five eggs: the optimized pad, an unoptimized pad, and no pad. We observed a 100% survival rate for the optimized pad, while the unoptimized pad and no pad showed significantly lower survival rates of 20% and 0%, respectively.

3.4.2 Material Emulation

Our TNN enables the creation of GCS that emulate the mechanical behaviors of different materials. By mimicking the behavior of other materials, one can optimize for non-mechanical properties like weight, cost, and fabrication time. We designed GCS parts that replicate the behavior of polyurethane (PUR) foam (Figure 8c), a material commonly employed in packaging.

4 Discussion

Discrepancies between predicted and actual curves can come from model prediction errors and lost information from PCA compression. However, we did not examine which source of error contributes to poor predictions. In the future, we plan to extend our investigation to look at the performance of PCA compression and explore nonlinear compression methods such as autoencoders.

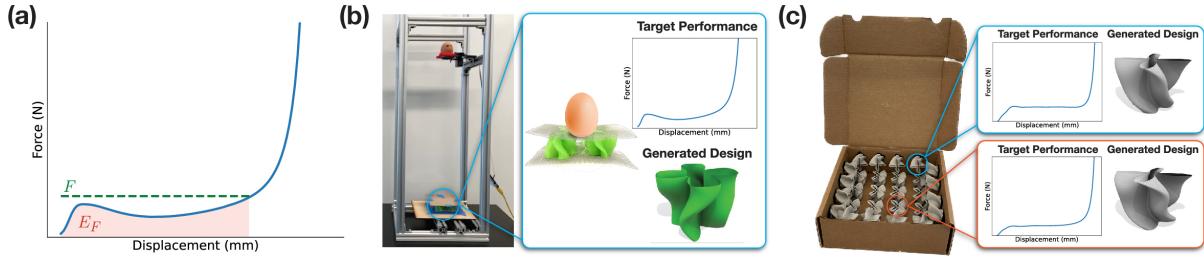


Figure 8: **Applications.** (a) Given a force threshold F , E_F is the energy absorbed before exceeding F . (b) Custom GCS create padding for the egg drop test. A pad of four GCS absorbs the impact energy of an egg dropped from 50 cm without breaking it. (c) GCS emulating the mechanical properties of polyurethane foam (PUR), a common material in packaging.

Real-world constraints often restrict parameters such as height, mass, or material. However, our current TNN architecture does not allow for user-defined values of generated design parameters, suggesting a clear direction for future enhancements. One potential strategy is to explore conditioning techniques employed in other neural network architectures [32] to grant users fine-grained control over the generated design parameters.

Exploring transfer learning techniques for our TNN presents an exciting avenue for extending its capabilities to diverse 3D printable structures. Fabricated structures span various parameterizations, yielding structures like lattices [2, 25, 33], crossed barrels [4], and foams [9]. Transferring the acquired design-performance knowledge to different structures, especially those with limited empirical data, holds significant promise for future research.

Finally, understanding how simulation and experimentation can be used in unison to predict high-deformation mechanical properties is essential for future work. Such approaches offer viable alternatives to collecting extensive experimental datasets, a process typically reliant on access to self-driving labs. We hope to learn how experimental data can improve simulated outcomes and how much experimental data is needed for this purpose.

5 Conclusion

We explore using a Tandem Neural Network (TNN) for the forward and inverse design of FDM 3D printed shells, representing a diverse and versatile class of structures. Trained on a comprehensive experimental dataset, our TNN reveals the intricate design-performance relationship between shell parameters and compressive behaviors. By utilizing entire force-displacement curves as performance representations, the network captures a range of nonlinear elastoplastic and hyperelastic deformations. In forward design, our TNN predicts these nonlinear force-displacement curves based on shell design parameters. Conversely, in inverse design, the network generates shell designs that exhibit specific desired compressive deformations. We validate generated shell designs through fabrication and testing. Additionally, we generate shells with tailored mechanical properties for several applications. To encourage further exploration, we make our code and processed dataset publicly available².

6 Acknowledgements

The authors thank Adedire Adesiji for brainstorming, assistance in constructing applications, and photography; Helena Gill, Xingjian Han, and Abinit Sati for their work on constructing and running the impact absorption application; and Peter Yichen Chen for his discussions and input.

7 Author contributions

Samuel Silverman: Conceptualization, Methodology, Software, Investigation, Writing-Original Draft. Kelsey L. Snapp: Conceptualization, Investigation, Validation, Writing-Review & Editing. Keith A.

²<https://github.com/samsilverman/nonlinear-deformation-design>

Brown: Conceptualization, Methodology, Writing-Review & Editing, Supervision. Emily Whiting: Conceptualization, Methodology, Writing-Original Draft, Writing-Review & Editing, Supervision.

8 Author Disclosure Statement

No competing financial interests exist.

9 Funding Information

This work was supported by the US Army CCDC Soldier Center (contract W911QY2020002).

References

- [1] Panetta J, Zhou Q, Malomo L, Pietroni N, Cignoni P, Zorin D. Elastic Textures for Additive Fabrication. *ACM Trans Graph*. 2015 jul;34(4). Available from: <https://doi.org/10.1145/2766937>.
- [2] Martínez J, Skouras M, Schumacher C, Hornus S, Lefebvre S, Thomaszewski B. Star-Shaped Metrics for Mechanical Metamaterial Design. *ACM Trans Graph*. 2019 jul;38(4). Available from: <https://doi.org/10.1145/3306346.3322989>.
- [3] Bates SRG, Farrow IR, Trask RS. 3D printed polyurethane honeycombs for repeated tailored energy absorption. *Materials & Design*. 2016;112:172-83. Available from: <https://www.sciencedirect.com/science/article/pii/S026412751631125X>.
- [4] Gongora AE, Xu B, Perry W, Okoye C, Riley P, Reyes KG, et al. A Bayesian experimental autonomous researcher for mechanical design. *Science Advances*. 2020;6(15):eaaz1708. Available from: <https://www.science.org/doi/abs/10.1126/sciadv.aaz1708>.
- [5] Erps T, Foshey M, Luković MK, Shou W, Goetzke HH, Dietsch H, et al. Accelerated discovery of 3D printing materials using data-driven multiobjective optimization. *Science Advances*. 2021;7(42):eabf7435. Available from: <https://www.science.org/doi/abs/10.1126/sciadv.abf7435>.
- [6] Snapp KL, Verdier B, Gongora AE, Silverman S, Adesiji AD, Morgan EF, et al. Superlative mechanical energy absorbing efficiency discovered through self-driving lab-human partnership. *Nature Communications*. 2024 May;15(1):4290. Available from: <https://doi.org/10.1038/s41467-024-48534-4>.
- [7] Bickel B, Bächer M, Otaduy MA, Lee HR, Pfister H, Gross M, et al. Design and fabrication of materials with desired deformation behavior. *ACM Trans Graph*. 2010 jul;29(4). Available from: <https://doi.org/10.1145/1778765.1778800>.
- [8] Schumacher C, Bickel B, Rys J, Marschner S, Daraio C, Gross M. Microstructures to Control Elasticity in 3D Printing. *ACM Trans Graph*. 2015 jul;34(4). Available from: <https://doi.org/10.1145/2766926>.
- [9] Martínez J, Dumas J, Lefebvre S. Procedural voronoi foams for additive manufacturing. *ACM Trans Graph*. 2016 jul;35(4). Available from: <https://doi.org/10.1145/2897824.2925922>.
- [10] Yang C, Kim Y, Ryu S, Gu GX. Prediction of composite microstructure stress-strain curves using convolutional neural networks. *Materials & Design*. 2020;189:108509. Available from: <https://www.sciencedirect.com/science/article/pii/S0264127520300423>.
- [11] Li B, Deng B, Shou W, Oh TH, Hu Y, Luo Y, et al. Computational discovery of microstructured composites with optimal stiffness-toughness trade-offs. *Science Advances*. 2024;10(5):eadk4284. Available from: <https://www.science.org/doi/abs/10.1126/sciadv.adk4284>.
- [12] Li X, Li M, Jiang C. Energetically consistent inelasticity for optimization time integration. *ACM Trans Graph*. 2022 jul;41(4). Available from: <https://doi.org/10.1145/3528223.3530072>.

- [13] Zong Z, Li X, Li M, Chiaramonte MM, Matusik W, Grinspun E, et al. Neural Stress Fields for Reduced-order Elastoplasticity and Fracture. In: SIGGRAPH Asia 2023 Conference Papers. SA '23. New York, NY, USA: Association for Computing Machinery; 2023. Available from: <https://doi.org/10.1145/3610548.3618207>.
- [14] Cirio G, Li D, Grinspun E, Otaduy MA, Zheng C. Crumpling sound synthesis. ACM Trans Graph. 2016 dec;35(6). Available from: <https://doi.org/10.1145/2980179.2982400>.
- [15] Abueidda DW, Almasri M, Ammourah R, Ravaioli U, Jasiuk IM, Sobh NA. Prediction and optimization of mechanical properties of composites using convolutional neural networks. Composite Structures. 2019;227:111264. Available from: <https://www.sciencedirect.com/science/article/pii/S0263822319312383>.
- [16] Yang Z, Yu CH, Guo K, Buehler MJ. End-to-end deep learning method to predict complete strain and stress tensors for complex hierarchical composite microstructures. Journal of the Mechanics and Physics of Solids. 2021;154:104506. Available from: <https://www.sciencedirect.com/science/article/pii/S0022509621001721>.
- [17] Ben Chaabene W, Flah M, Nehdi ML. Machine learning prediction of mechanical properties of concrete: Critical review. Construction and Building Materials. 2020;260:119889. Available from: <https://www.sciencedirect.com/science/article/pii/S0950061820318948>.
- [18] Herriott C, Spear AD. Predicting microstructure-dependent mechanical properties in additively manufactured metals with machine- and deep-learning methods. Computational Materials Science. 2020;175:109599. Available from: <https://www.sciencedirect.com/science/article/pii/S0927025620300902>.
- [19] Colton DL, Kress R, Kress R. Inverse acoustic and electromagnetic scattering theory. vol. 93. Springer; 1998.
- [20] Kucuk S, Bingul Z. Robot kinematics: Forward and inverse kinematics. INTECH Open Access Publisher London, UK; 2006.
- [21] Liu D, Tan Y, Khoram E, Yu Z. Training Deep Neural Networks for the Inverse Design of Nanophotonic Structures. ACS Photonics. 2018;5(4):1365-9. Available from: <https://doi.org/10.1021/acspophotonics.7b01377>.
- [22] Xu X, Sun C, Li Y, Zhao J, Han J, Huang W. An improved tandem neural network for the inverse design of nanophotonics devices. Optics Communications. 2021;481:126513. Available from: <https://www.sciencedirect.com/science/article/pii/S0030401820309317>.
- [23] Ma W, Cheng F, Liu Y. Deep-Learning-Enabled On-Demand Design of Chiral Metamaterials. ACS Nano. 2018;12(6):6326-34. PMID: 29856595. Available from: <https://doi.org/10.1021/acsnano.8b03569>.
- [24] Bastek JH, Kumar S, Telgen B, Glaesener RN, Kochmann DM. Inverting the structure–property map of truss metamaterials by deep learning. Proceedings of the National Academy of Sciences. 2022;119(1):e2111505119. Available from: <https://www.pnas.org/doi/abs/10.1073/pnas.2111505119>.
- [25] Van 't Sant S, Thakolkaran P, Martínez J, Kumar S. Inverse-designed growth-based cellular metamaterials. Mechanics of Materials. 2023;182:104668. Available from: <https://www.sciencedirect.com/science/article/pii/S016766362300114X>.
- [26] Deng B, Zareei A, Ding X, Weaver JC, Rycroft CH, Bertoldi K. Inverse Design of Mechanical Metamaterials with Target Nonlinear Response via a Neural Accelerated Evolution Strategy. Advanced Materials. 2022;34(41):2206238. Available from: <https://onlinelibrary.wiley.com/doi/abs/10.1002/adma.202206238>.
- [27] Li Y, Coros S, Thomaszewski B. Neural Metamaterial Networks for Nonlinear Material Design. ACM Trans Graph. 2023 dec;42(6). Available from: <https://doi.org/10.1145/3618325>.

- [28] Gongora AE, Mysore S, Li B, Shou W, Matusik W, Morgan EF, et al. Designing Composites with Target Effective Young's Modulus using Reinforcement Learning. In: Proceedings of the 6th Annual ACM Symposium on Computational Fabrication. SCF '21. New York, NY, USA: Association for Computing Machinery; 2021. Available from: <https://doi.org/10.1145/3485114.3485123>.
- [29] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative Adversarial Nets. In: Ghahramani Z, Welling M, Cortes C, Lawrence N, Weinberger KQ, editors. Advances in Neural Information Processing Systems. vol. 27. Curran Associates, Inc.; 2014. Available from: https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.
- [30] Kingma D, Ba J. Adam: A Method for Stochastic Optimization. In: International Conference on Learning Representations (ICLR). San Diega, CA, USA; 2015. .
- [31] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: Wallach H, Larochelle H, Beygelzimer A, d'Alché-Buc F, Fox E, Garnett R, editors. Advances in Neural Information Processing Systems. vol. 32. Curran Associates, Inc.; 2019. Available from: https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.
- [32] Mirza M, Osindero S. Conditional Generative Adversarial Nets; 2014.
- [33] Tozoni DC, Dumas J, Jiang Z, Panetta J, Panozzo D, Zorin D. A Low-Parametric Rhombic Microstructure Family for Irregular Lattices. ACM Trans Graph. 2020 aug;39(4). Available from: <https://doi.org/10.1145/3386569.3392451>.