Enhancing Mechanical Metamodels With a Generative Model-Based Augmented Training Dataset

Hiba Kobeissi[2]
Department of Mechanical Engineering,
Boston University,
Boston, MA 02215
e-mail: hibakob@bu.edu

Saeed Mohammadzadeh[2]
Department of Systems Engineering,
Boston University,
Boston, MA 02215
e-mail: saeedmhz@bu.edu

Emma Lejeune[1]
Department of Mechanical Engineering,
Boston University,
Boston, MA 02215
e-mail: elejeune@bu.edu

Modeling biological soft tissue is complex in part due to material heterogeneity. Microstructural patterns, which play a major role in defining the mechanical behavior of these tissues, are both challenging to characterize and difficult to simulate. Recently, machine learning (ML)-based methods to predict the mechanical behavior of heterogeneous materials have made it possible to more thoroughly explore the massive input parameter space associated with heterogeneous blocks of material. Specifically, we can train ML models to closely approximate computationally expensive heterogeneous material simulations where the ML model is trained on datasets of simulations with relevant spatial heterogeneity. However, when it comes to applying these techniques to tissue, there is a major limitation: the number of useful examples available to characterize the input domain under study is often limited. In this work, we investigate the efficacy of both ML-based generative models and procedural methods as tools for augmenting limited input pattern datasets. We find that a style-based generative adversarial network with an adaptive discriminator augmentation mechanism is able to successfully leverage just 1000 example patterns to create authentic generated patterns. In addition, we find that diverse generated patterns with adequate resemblance to real patterns can be used as inputs to finite element simulations to meaningfully augment the training dataset. To enable this methodological contribution, we have created an open access finite element analysis simulation dataset based on Cahn–Hilliard patterns. We anticipate that future researchers will be able to leverage this dataset and build on the work presented here.
[DOI: 10.1115/1.4054898]

Keywords: machine learning mechanics surrogate modeling heterogeneous materials

1 Introduction

Establishing models that realistically capture the biomechanical behavior of soft tissue is a challenging yet crucial endeavor [1,2]. High fidelity mechanical models are needed for tasks such as surgical simulation [3–5], patient-specific procedure planning [6,7], modeling of in vivo biological mechanisms [8,9], and inverse material characterization [10,11]. Capturing the mechanical behavior of soft tissue is challenging because soft tissues are often highly nonlinear and anisotropic, they can exhibit a nonlinear stiffening response, they often undergo large deformations, and they have a complex hierarchical structure [1,12–14]. For example, at the microstructural level, soft tissue may contain components such as fibers with a preferred direction, which give rise to highly anisotropic material behavior on the macroscale [14]. In addition to complex constitutive behavior, biological materials are also challenging to model because they tend to be highly heterogeneous [13,15]. As such, developing faithful mechanical models of soft tissues and numerically implementing them (e.g., in the finite element setting [16]) are both challenging and typically quite computationally expensive [2,10,14,17–19]. Notably, both the exact values of the mechanical properties of biological tissue and their heterogeneous distribution in space are often uncertain [20,21]. Therefore, in order to get a true picture of tissue behavior, it is necessary to run multiple simulations that capture the range of relevant input parameters [10]. In this context, there has been substantial recent interest in reducing the computational cost of these numerical simulations at the cost of marginal decrease in the simulation accuracy [22].

Markedly, there has been recent interest in using machine learning tools to create computationally inexpensive data-driven models of soft biological tissue in particular [23], and for various biomedical applications in general [24–27]. In previous work by our group and others [28–35], metamodels, or surrogate models [36], developed with supervised machine learning algorithms and multifidelity mechanical datasets have been used successfully to predict the mechanical behavior of heterogeneous materials via single and full-field quantities of interest (QoIs) (e.g., strain energy, displacement/strain fields, damage fields). For example, Tonutti et al. [22] used the results of finite element analysis (FEA) simulations in conjunction with artificial neural networks and support vector regression to develop computationally inexpensive patient-specific deformation models for brain pathologies. In addition, Salehi et al. [37] trained graph neural networks with FEA simulation results to speed-up the approximation of soft tissue deformation with acceptable loss of accuracy for neurosurgical applications. In Tac et al. [23], fully connected neural networks were trained with high-fidelity biaxial test data and low-fidelity analytical approximations to derive a data-driven anisotropic constitutive model of porcine and murine skin. Notably, due to the limited availability of both experimental data and high fidelity simulation data, methods that rely on multiple data fidelities (i.e., multifidelity models) have been shown to be more effective than single fidelity schemes given a small number of high fidelity data [23,28,38,39]. This is particularly true for methods that rely on deep learning where training datasets must be large for successful model implementation [40–42]. Though multifidelity methods can address the scenario where there are limited high-fidelity

[1]Corresponding author.
[2]Co-first authors.
Manuscript received March 8, 2022; final manuscript received June 23, 2022; published online xx xx, xxxx. Assoc. Editor: Adrian Buganza Tepole.

simulations results, they are not necessarily equipped to address the scenario where there is limited information about what the training dataset should contain. For example, it is unlikely that researchers will have tens of thousands of accurate examples of the heterogeneous material property distribution of a given soft tissue of interest. In this work, our goal is to systematically answer the question: is it possible to create a meaningful training dataset that ultimately improves the performance of a deep learning-based metamodel of heterogeneous material given only a small number of representative examples of the relevant material property distribution input pattern?

To address this question, we first define a benchmark problem to evaluate our proposed machine learning approach. This is important because, at present, there are only a small number of existing open access benchmark datasets related to problems in solid mechanics [43–46]. Furthermore, of the available datasets, few contain a good representation of the heterogeneous material properties most relevant to soft tissue modeling. Our benchmark dataset, the "mechanical MNIST Cahn–Hilliard" dataset, is a contribution to our previously initiated "mechanical MNIST" project where we provide simulation results for heterogeneous materials undergoing large deformation. The full dataset contains 104,813 Cahn–Hilliard patterns and associated equibiaxial extension simulations, and it is straightforward to train a deep learning-based metamodel to predict QoI from these simulations (e.g., change in strain energy $\Delta\Psi$). However, if we constrain ourselves to only a small subset of these example input patterns, e.g., if we limit our knowledge to just 1000 example patterns, it becomes much more challenging to effectively train a deep learning-based metamodel. With this benchmark dataset and imposed limitation, we are able to test both the efficacy of machine learning (ML)-based generative models, models that learn the data distribution and generate plausible examples from the distribution [47], and procedural methods at augmenting a constrained version of the available training dataset. By comparing the results of metamodels that rely on generated patterns to metamodels that are trained on true input patterns, we are able to systematically evaluate the efficacy of our proposed size-limited data augmentation approaches. We note that this premise follows from recent work in the literature where generative models have been used to augment small materials characterization datasets [48,49]. Ultimately, we are able to clearly demonstrate that leveraging the capabilities of our selected data generation models is an effective tool for augmenting small datasets of material property distributions in biological tissue for the purpose of creating training datasets for ML-based metamodels.

The remainder of the paper is organized as follows. In Sec. 2, we begin by introducing our mechanical MNIST Cahn–Hilliard dataset. Then, we describe our approach to training a metamodel to approximate the mechanical behavior of the simulations, and our approach to generating synthetic input patterns to augment the training dataset. In Sec. 3, we show the performance of our generative models and the performance of our metamodel with ML-based and procedural augmented training dataset. We conclude in Sec. 4. Finally, we note briefly that links to the code and dataset required to reproduce our work are given in Sec. 5.

2 Methods

Here, we begin in Sec. 2.1 with an introduction to our mechanical MNIST Cahn–Hilliard dataset. Then, in Sec. 2.2, we describe our metamodeling approach where a ML-based metamodel is used to predict a single quantity of interest (in this case change in strain energy $\Delta\Psi$) from an array-based representation of the input pattern. Then, in Sec. 2.3, we detail our three different approaches to ML-based generative modeling of the input pattern distribution. In Sec. 2.4, we introduce two additional procedural methods for generating synthetic input patterns. In Sec. 2.5, we present the evaluation metrics that we considered to compare the performance of the different methods that we have implemented to generate synthetic patterns. Finally, in Sec. 2.6, we define our procedure for standard rotation-based augmentation. We briefly note that in order to stay consistent with the literature, the Greek letters λ and μ refer to different constants in Secs. 2.1 and 2.5. In both cases, we provide a brief definition of each term when it is introduced.

2.1 The Mechanical MNIST Cahn–Hilliard Dataset.

In conjunction with our previous publications [28–30], we introduced the mechanical MNIST dataset of heterogeneous materials undergoing large deformation. In previous iterations of the dataset, heterogeneous input domain patterns were defined by the MNIST [50] and fashion MNIST [51] bitmap patterns. For this paper, we extend our mechanical MNIST dataset collection to include additional patterns from a different input domain distribution that is more relevant to heterogeneous biological materials. The input patterns for the mechanical MNIST Cahn–Hilliard dataset are generated based on Alan Turing's model of morphogenesis [52]—a common motif during biological development manifested in many different animal and plant patterns such as the pigmentation of animal skins, the branching of trees and other skeletal structures, and the distinct patterns on leaves and petals [53,54]. We obtain these patterns by solving a nonlinear spatio-temporal fourth-order partial differential equation referred to as the Cahn–Hilliard equation that was originally proposed to describe the process of phase separation in isotropic binary alloys [55–57].

Our new dataset, mechanical MNIST Cahn–Hilliard, contains not only Cahn–Hilliard based two-dimensional heterogeneous input patterns but also the results of finite element simulations of these material domains subjected to equibiaxial extension. Here, we will summarize the process of creating this dataset. Briefly, the Cahn–Hilliard equation, which is a fourth-order partial differential equation that governs the evolution of a binary mixture, can first be reduced to a pair of second-order equations [59,60]. This mixed formulation can be expressed in the weak form for the two unknown fields, c, the concentration of one of the components of the binary mixture, and μ, the chemical potential of a uniform solution:

$$\int_\Omega \frac{c_{n+1}-c_n}{t_{n+1}-t_n} q\,dx + \int_\Omega M\nabla\mu_{n+\theta}\cdot\nabla q\,dx = 0 \qquad \forall q \in V \tag{1}$$

$$\int_\Omega \mu_{n+1} v\,dx - \int_\Omega \frac{df_{n+1}}{dc} v\,dx - \int_\Omega \lambda\nabla c_{n+1}\cdot\nabla v\,dx = 0 \qquad \forall v \in V \tag{2}$$

where M is the mobility parameter, λ is a positive scalar that describes the thickness of the interfaces between the phases of the mixture, f is the chemical free-energy function, and q and v are test functions [59,60].

We solve the Cahn–Hilliard equations using the open source finite element software FENICS [61,62] and run 2072 phase separation simulations on a unit square domain $\Omega = [0,1]$ where each simulation differs in the following: (1) the initial concentration c_0 with uniform random fluctuations of zero mean and range between –0.05 and 0.05, (2) the grid size on which the initialized concentration is allowed to spatially vary, (3) the interface thickness λ, and (4) the peak-to-valley value of the free-energy function f, a symmetric double-well function. We record the concentration parameter at multiple time steps in each simulation to obtain 105,427 spatial distribution patterns which broadly fall under two qualitative types: spotted (for $c_0 = 0.63$ and $c_0 = 0.75$), and striped (for $c_0 = 0.5$), as is expected for these types of simulations [59,63,64], and store the obtained images as 400×400 binary bitmaps. Example patterns are illustrated in Fig. 1(*a*). For further details on the underlying theory of the Cahn–Hilliard equation and our finite element implementation, we refer the reader to the supplementary document provided with the dataset (see Sec. 5). As an additional step, we visualize downsampled 64×64 vectors describing each Cahn–Hilliard pattern array in a two-dimensional space using the dimension reduction technique, uniform manifold approximation and projection (UMAP) [58], which provides us with a qualitative tool to visualize our high-dimensional dataset input parameter space. Notably, the plot in Fig. 1(*b*) clearly reveals the two distinct clusters of patterns, which is consistent with our observation that the dataset is split between the striped and spotted motifs.

From this collection of 105,427 heterogeneous input patterns, we perform a second set of finite element simulations where we use the input patterns to inform the heterogeneous material property distribution of the domain and subject it to equibiaxial extension. To accomplish this, we first convert the binary bitmap patterns into meshed domains of two different materials. Briefly, we detect the contours of the image features and extract their coordinates using the OPENCV library [65]. We then translate these coordinates into a mesh with two different subdomains, background and pattern, using PYGMSH 6.1.1 [66], a Python implementation of GMSH 4.6.0 [67]. We note briefly that from our initial collection of 105,427 images, 614 images could not be processed because they exhibited either pattern features that were too small to be detected as area domains, features that were in very close proximity to each other, or complex hierarchical contours that our pipeline was not able to detect and process. Thus, our final dataset contains 104,813 simulation results. Based on a mesh refinement study, we chose quadratic triangular elements with a characteristic length of 0.01. This led to approximately 41,000 elements in a typical domain.

Once the material domain was meshed, we performed equibiaxial extension simulations in FENICS [61,62]. Here, we chose a compressible Neo-Hookean material model defined by strain energy Ψ as:

$$\Psi = \frac{1}{2}\mu[\mathbf{F}:\mathbf{F} - 3 - 2\ln(\det\mathbf{F})] + \frac{1}{2}\lambda\left[\frac{1}{2}\left[(\det\mathbf{F})^2 - 1\right] - \ln(\det\mathbf{F})\right] \quad (3)$$

where $\mathbf{F}$ is the deformation gradient, and μ and λ are the Lamé parameters equivalent to Young's modulus E and Poisson's ratio ν as $E = \mu(3\lambda + 2\mu)/(\lambda + \mu)$ and $\nu = \lambda/(2(\lambda + \mu))$. We define the Poisson's ratio as a constant ($\nu = 0.3$), and we specify a Young's modulus E for the background domain that is 10 times lower than the Young's modulus for the "stiffer" spotted and striped patterns ($E = [1, 10]$). We set up each finite element simulation for equibiaxial deformation so that every external edge of the domain is extended by half of the value of given applied displacement in the direction of the outward normal to the surface (Fig. 1(*c*)). The set of fixed displacements $\mathbf{d}$ go up to 50% of the initial domain size as

$$\mathbf{d} = [0.0, 0.001, 0.1, 0.2, 0.3, 0.4, 0.5] \quad (4)$$

The output of each of the 104,813 large deformation simulations consisted of data on the total change in strain energy $\Delta\Psi$, total reaction force in the x and y directions, and full field domain displacement collected on a downsampled 64×64 grid (Fig. 1(*c*)). We chose the size of the grid to be the smallest possible size that could be reached without the loss of important image features. In this context, we consider the borders of the white/dark patterns to be important features that should not be distorted much by any operation to avoid misclassifiying the cells along the edges into the wrong subdomain. We note that all code to implement these

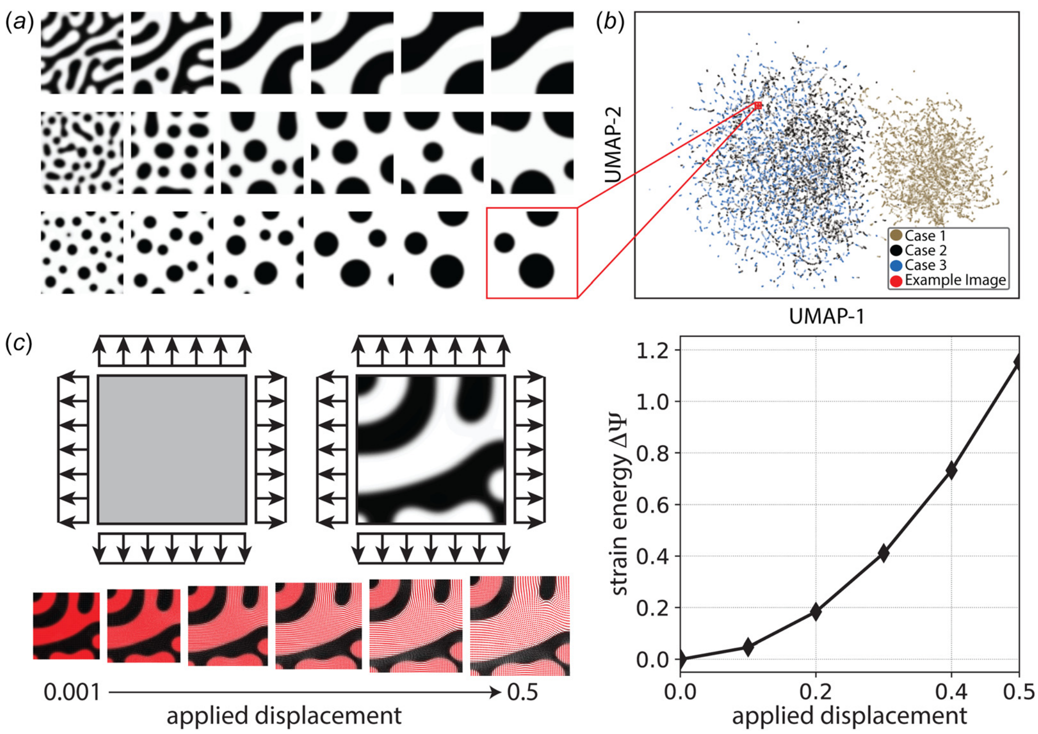

Fig. 1 (*a*) Illustration of the spatial patterns obtained from our Cahn–Hilliard simulations where each row corresponds to the time evolution in a single simulation for $c_0 = 0.5$ (case 1), $c_0 = 0.63$ (case 2), and $c_0 = 0.75$ (case 3) shown in the first, second, and the third rows, respectively. (*b*) A UMAP visualization [58] of a representative proportion of our Cahn–Hilliard patterns using `random_state = 42, n_neighbors = 30, min_dist = 0.1` as training parameters. (*c*) A schematic illustration of displacement driven equibiaxial extension applied to a heterogeneous domain dictated by the Cahn–Hilliard patterns. Here, we show an example from case 1: $c_0 = 0.5$ and plot the deformed state at the six magnitudes of applied displacement. From these finite element simulations, we obtain multiple outputs including the total change in strain energy $\Delta\Psi$ at each load step.

simulations is shared on GITHUB with access details given in Sec. 5.

2.2 Metamodel Design and Implementation. In this section, we summarize our approach to creating metamodels for predicting the change in strain energy $\Delta\Psi$ from the input Cahn–Hilliard patterns. In Secs. 2.3, 2.4, and 2.6, we describe the details of our generative model-based, procedural-based, and standard rotation-based approaches that we implement to augment the training dataset.

2.2.1 Feedforward Convolutional Neural Network. In this paper, we are focused on using machine learning techniques for predicting single quantities of interest ($\Delta\Psi$) from input arrays (Cahn–Hilliard patterns). This goal is illustrated schematically in Fig. 2(*a*). To accomplish this, we implemented a basic feedforward convolutional neural network (CNN) consisting of a total of nine convolutional layers each followed by batch normalization and rectified linear unit (ReLu) activation except for the last (ninth) layer. For downsampling input images, we used max pooling after the first three convolutional layers with *same* padding while *valid* padding is used for the rest of the convolutional layers. Our network has a total of 3,734,625 trainable parameters. We trained the network using the PYTORCH library [68] with a batch size of 64 for 100 epochs. We employ an Adam optimizer [69] with learning rate $\alpha = 0.01$ reduced to 0.001 after 50 epochs and exponential decay rates $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The output of the CNN is a single quantity of interest (QoI) for a 64×64 array input describing the simulation input pattern. We validated our model performance through a five-fold cross-validation approach based on mean-squared-error (MSE). In Sec. 3.2, we report the performance of our model on test data.

2.2.2 Transfer Learning. Our original mechanical MNIST Cahn–Hilliard dataset took approximately 5240 CPU hours to generate. Rather than expending a similar level of resources to run simulations based on generated input patterns, we decided to employ a transfer learning approach where we leverage low fidelity simulation data [70]. Specifically, we followed the approach outlined in our recent publication [28] to create low fidelity simulation versions of our dataset that are run on a coarse mesh (64×64 grid, 8192 elements) with linear elements and only subject to a perturbation displacement (0.001) rather than the full 50% extension. With these parameters, it took approximately 4.2 CPU hours to generate a low fidelity dataset of 72,000 patterns and the corresponding strain energy values only for a perturbation displacement. Notably, this is 0.08% of the time it would take to generate the equivalent number of high fidelity simulations described in Sec. 2.1. Of course, this speed up comes at the price of introducing numerical error that must be subsequently dealt with through transfer learning.

Our implementation of transfer learning is a straightforward model pretraining approach illustrated schematically in Fig. 2(*b*) and described in detail in our previous publication [28]. Part of the appeal of this approach is that it is quite straightforward to implement. First, we train the metamodel (in our case the CNN defined in Sec. 2.2.1) on the low fidelity dataset. Then, we use this pretrained metamodel as the weight initialization for additional training with the high fidelity dataset. In our case, the low fidelity dataset will contain data from up to 16,000 simulations while the high fidelity dataset will contain data from only 1000 simulations. The ideal outcome from this approach is to end up with a metamodel that is trained on predominantly low-fidelity data yet performs comparably to a metamodel trained on the target high fidelity dataset. In Sec. 3.2, we first report the metamodel performance on the low fidelity dataset (Fig. 5), and then in Sec. 3.3, we report the performance of the low fidelity models transferred to the high fidelity dataset via additional training with 1000 high fidelity real samples (Table 1).

2.3 Augmenting the Training Dataset With a Machine Learning-Based Generative Model. The main focus of this paper is on developing techniques to effectively train the metamodels described in Sec. 2.2 even when we have limited examples of the relevant input patterns needed for creating our training dataset. Here, we will explore methods for leveraging limited examples of input patterns by creating synthetic input patterns from a generative model. Briefly, we implement a style-based generative adversarial network using adaptive discriminator augmentation (StyleGAN2-ADA) [42], a Wasserstein generative adversarial network with weight clipping (WGAN-CP) [71], and a WGAN with gradient penalty (WGAN-GP) [72] to generate patterns that resemble the real striped and spotted Cahn–Hilliard patterns detailed in Sec. 2.1. The architectures of the generative models explored in this work are schematically shown in Figs. 2(*c*) and

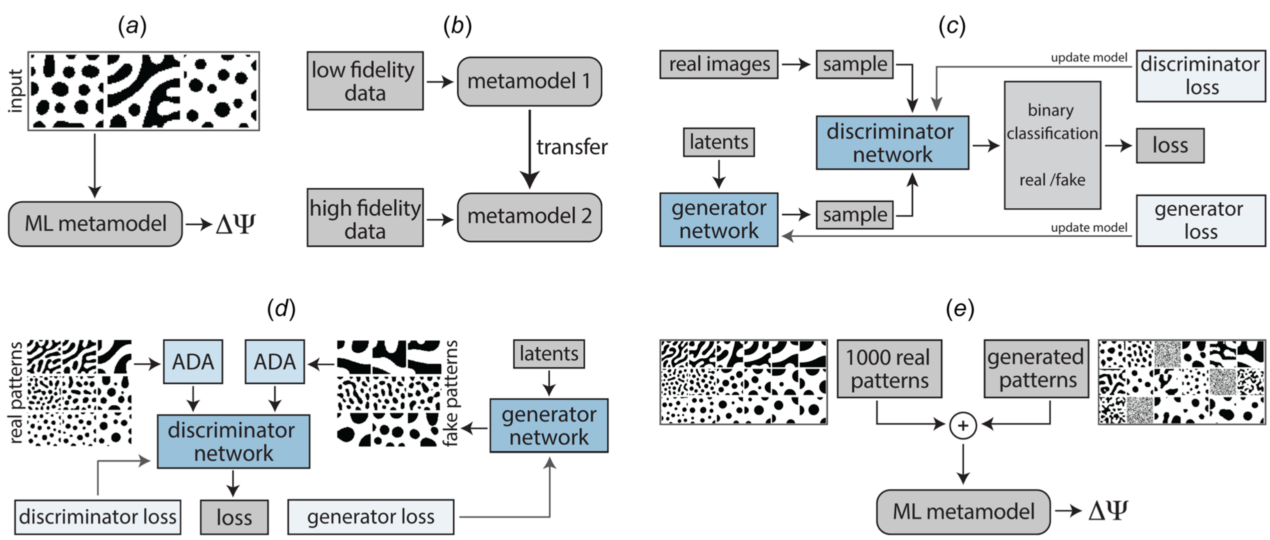

Fig. 2 (*a*) A schematic of our ML metamodels that are used to predict change in strain energy $\Delta\Psi$ at a fixed level of applied displacement from each material property distribution. (*b*) A schematic of transfer learning whereby a model trained on one dataset (in this case a low fidelity dataset) is used to make predictions on another dataset (in this case a high fidelity dataset). (*c*) Architecture of a generative adversarial network including the WGAN models trained in this paper. (*d*) An illustration of the StyleGAN2 with an ADA mechanism implemented in this work as adapted from Ref. [42]. (*e*) A schematic of combining simulations based on both generated and real patterns to create a larger training dataset.

Table 1 Results of transfer learning

Transfer learning			
	Pre-training (low fidelity)	Fine-tuning (high fidelity)	
Pattern generation method	$R2$ score (test set)	$R2$ score (test set)	MAE (test set)
Real	0.9991	0.9991	0.0046
StyleGAN2-ADA	0.9967	0.9977	0.0074
WGAN-CP	0.9973	0.9974	0.0088
WGAN-GP	0.9981	0.9974	0.0077
"Procedural"	0.9979	0.9973	0.0084
No transfer learning		0.9783	0.0198

Pre-training is performed with low fidelity data of 1000 real Cahn–Hilliard patterns and 15,000 either real or generated patterns subjected to three additional rotations of the unique domains. Fine-tuning refers to training a metamodel with 1000 high fidelity real Cahn–Hilliard patterns with the model initial weights "transferred" from the pretrained model on the corresponding row. For a metamodel that is trained on 1000 entirely real Cahn-Hilliard patterns without transfer learning, the model weights are randomly initialized. Overall, it is evident that our transfer learning approach improves the MAE by at least 55% when predicting change in strain energy. Representative plots of true strain energy versus predicted strain energy are shown in Appendix A, Fig. 6 to add additional context to these values.

2(*d*). We train all three generative adversarial network (GAN) models with a limited set of 1000 real Cahn–Hilliard patterns (the same set of real images used for training the metamodels). We then combine equibiaxial extension simulation results of both generated and real patterns to create a larger training dataset for our metamodel as shown schematically in Fig. 2(*e*). In the remainder of this section, we provide an overview of GANs, describe the specific GANs implemented in this work, and briefly present our alternative approaches for augmenting the metamodel training dataset via procedural pattern generation and standard rotation-based augmentation.

2.3.1 Generative Adversarial Networks. In the context of machine learning, generative models are models that learn data distributions such that they can then be used to output (i.e., generate) plausible new examples [73]. Building upon earlier deep generative models, generative stochastic networks [74] in particular, and inspired by the work in Refs. [75–77], Goodfellow et al. [47] developed a novel framework for generative models where the generative network is put in competition with a discriminative network that learns to distinguish between a sample obtained from the real data distribution and one that is generated from the model distribution. Known as GANs, these methods consist of training two models, a generative model G and a discriminative model D simultaneously competing in a minimax two-player game fashion [47]. In this framework, G is trained to capture the input data distribution by fooling the discriminative model D and maximizing the probability of the latter mistakenly labeling a sample synthesized by G as one from the training data.

In their original form, GANs have been applied to many domains including the MNIST dataset of handwritten digits [78,79], the Toronto face database of human faces with expressions [80], and the miscellaneous CIFAR-10 dataset [81] with promising results [47]. However, major drawbacks of the method include low resolution of the generated images, relatively low variation in the output distribution, and unstable training [82]. Furthermore, training GANs to synthesize high-quality, high-resolution output distributions typically requires at least 10^5–10^6 input images. Without a dataset of this size, the training tends to diverge as the discriminator network overfits to the small number of training data examples and can no longer provide meaningful feedback to the generator network [42]. There have been many approaches to modifying the original architecture and training formulation of GANs [47] to improve their performance. Alterations to the network structure such as the implementation of deep convolutional GANs (DCGANs) [83], where the GAN model is scaled using CNN architectures, result in more stable behavior. Other enhanced methods include WGANs and style-based generative adversarial networks (StyleGANs), which are briefly described in Secs. 2.3.2 and 2.3.3, respectively.

2.3.2 Wasserstein Generative Adversarial Networks. In contrast to modifying the GAN network structure as in DCGANs, WGANs improve the stability of GANs by replacing the bin-to-bin distance function (i.e., the *Jensen–Shannon* divergence) of the original architecture with a continuous loss function, the earth mover or the Wasserstein-1 (W) distance [71]. The shortcomings of the bin-to-bin distance functions, which generally assume an alignment between the domains of the histograms being compared, are addressed by the more robust cross-bin earth mover distance function defined as the minimal cost of a "transport plan" to transform one distribution into the other [84–86].

As proposed, the original WGAN model [71] requires that the discriminator lie within a 1-Lipschitz space so that W is continuous everywhere and differentiable almost everywhere. This Lipschitz constraint is enforced via weight clipping (WGAN-CP) whereby the weights of the discriminator are restricted to a compact space [71]. In this setting, the discriminator is no longer trained to directly label samples as "real" or "fake," but rather to learn the Lipschitz function needed to compute W. As the model training proceeds to minimize the loss function, the distance W decreases, signifying that the generated output distribution is becoming closer to the real data distribution [72]. Although more stable compared to GANs, the performance of WGAN-CPs was shown to be limited because: (1) small clipping thresholds lead to vanishing gradients while larger thresholds result in exploding gradients, and (2) the discriminator is biased to converge to simplified approximations of the Lipschitz function [72]. Improved training of WGANs was proposed by Gulrajani et al. [72], who implement a gradient penalty method (WGAN-GP) instead of weight clipping to constrain the discriminator gradient. WGAN-GP enforces the Lipschitz constraint by imposing a penalty on the gradient norm if it is not close to the theoretical value of 1.

In this work, we test the performance of WGAN-CP and WGAN-GP trained with 1000 samples from our Cahn–Hilliard dataset. Using the PYTORCH library [87], we train typical convolutional feedforward neural networks for both the generator and the discriminator networks of WGAN-CP and WGAN-GP for a total of 23,690,498 trainable parameters, 12,656,257 for the generator network and 11,034,241 for the discriminator network. We accomplish this using the code published in conjunction with Ref. [88] as a starting point. We perform no additional parameter tuning and keep all hyper-parameters at their default values.

2.3.3 Style-Based Generative Adversarial Networks. A third approach to enhancing GANs involves modifying the latent space distributions of the generator network via feature mapping, and incorporating adaptive instance normalization (AdaIN) [89]. The AdaIN operation was first implemented by Huang and Belongie [90] in style transfer algorithms [91]; transferring the style of one image to the content of another image. Specifically, AdaIN first normalizes each feature map and then scales its mean and variance according to a style input.

In these StyleGAN models, the adjustments to the traditional generator are twofold: (1) the input latent space is mapped to a much less entangled intermediate latent feature space via an eight-layer multilayer perceptron network, and (2) the generator output is controlled by AdaIN processes, which are themselves controlled by learned affine transformations that concentrate the intermediate latent space to specific styles that dictate the dominant image features at each convolution layer [89]. The StyleGAN2 architecture was later developed to remedy artifacts observed in StyleGAN generated images [92]. The StyleGAN2 using ADA [42] is an adaptation of StyleGAN2 specifically designed for small training datasets. For the simplest implementations of training GANs with augmented datasets, generated distributions are known to exhibit features that are present in the augmented dataset, but not in the original dataset [42,93,94]. Therefore, to avoid this undesirable outcome, Karras et al. [42] proposed the ADA method.

For the augmentations to be "nonleaking" (i.e., not present in the generated examples) and for the GAN model to learn the true input distribution given an augmented dataset, the set of applied distortions for augmentation are required to be differentiable and belong to an invertible transformation of a probability distribution function [42,95]. This can be achieved for a diverse set of possible augmentations when they are applied to the dataset with a probability p, with $0 < p < 0.8$ [42]. However, the target value of p is sensitive to the size of the dataset and as such, setting a fixed value for it is far from optimal. For this reason, Karras et al. [42] implemented the discriminator augmentation method in an adaptive manner where p is set to 0 initially and its value is automatically adjusted (increased or decreased) based on a metric that indicates the extent by which the discriminator is overfitting. This heuristic is obtained from the discriminator outputs for the training and validation datasets, as well as the generated images and their mean over a fixed number of consecutive minibatches. ADA can be implemented on any GAN model without modifying the network architecture or increasing training cost [42]. Notably, the StyleGAN2-ADA combination performs exceptionally well on the limited CIFAR-10 dataset [81], thus motivating our implementation of the approach in this work.

Here, we train the StyleGAN2-ADA model using the PYTORCH library [87] with the code provided in Ref. [42] on a small subset (1000 samples) of our Cahn–Hilliard patterns. Of the set of transformations tested in Ref. [42], we apply the ones that contextually fit the Cahn–Hilliard dataset—geometric and color transformations. Geometric distortions include pixel blitting, isotropic and anisotropic scaling, fractional translation, and less frequently arbitrary rotation. We briefly note at this point that these distortions are implemented during the generation of synthetic patterns only and are not related to the equibiaxial loading conditions of the finite element simulations performed later once the generated data patterns are obtained. For color transformations, the image brightness, contrast, and saturation were adjusted, the luma axis was flipped, and the hue axis was rotated arbitrarily. We perform no parameter tuning and keep all hyper-parameters at their default values. In total, the generator network has 22,238,990 trainable parameters, and the discriminator network has 23,406,849 trainable parameters.

2.4 Augmenting the Training Dataset With Procedural and "Bernoulli" Randomly Generated Patterns.

As discussed in Sec. 2.3 and later depicted in Figs. 3 and 4, the three different ML-based generative models, WGAN-CP, WGAN-GP, and StyleGAN2-ADA are able to generate synthetic patterns relevant to the real Cahn–Hilliard patterns without being explicitly programed to do so. However, there is a rich history of implementing procedural algorithms for material microstructure pattern generation [96–101]. For example, many researchers have created explicitly programed algorithm that draws from experimentally obtained probability distributions for creating and placing microstructural features within a domain [102,103]. These algorithms range from quite simple (e.g., Voronoi tessellation [104,105]) to quite complex (e.g., feature shape and placement based on energy minimization [106]). In this paper, we implement two additional pattern generation algorithms to compare to the ML-based generative models. First, we implement a straightforward procedural algorithm where we create synthetic patterns with spatial correlations. In Sec. 3, we refer to these patterns as procedural patterns. Second, we create random patterns following a Bernoulli distribution without spatial correlation. In Sec. 3, we refer to these patterns as Bernoulli patterns.

For the procedural patterns, we begin with a low resolution grid, a 4×4, an 8×8, or a 16×16 grid, and assign each of the grid pixels an independent and identically distributed random value drawn from a uniform distribution $\mathcal{U}\,[0,1]$. Using the multi-dimensional image processing package in SCIPY "scipy.ndimage" [107], we then increase the resolution of the resulting grayscale random image to the desired size of 64×64 and convert the upscaled image to a binary pattern by setting a brightness threshold. For the Bernoulli patterns, we obtain binary images by simply creating a 64×64 grid of zeros, and then replacing the zeros with ones based on a probability threshold $p = 0.6594$. For both types of patterns, the value of the threshold was chosen so that the light-to-dark ratio present in the real patterns is preserved. Notably, the procedural patterns lead to spatially correlated features while the Bernoulli patterns do not.

2.5 Evaluation Metrics.

For evaluating and comparing the performance of the implemented GANs and the procedural methods at creating generated examples, we considered three indicators. First, we compute the Fréchet inception distance (FID) score, a quantitative metric to compare the resemblance between the distributions of the generated and real images [108]. The FID, also known as Wasserstein-2 distance, is computed between the 2048 dimensional feature vectors, taken as the output of the last pooling layer of the pretrained Inception network, of real and generated images by [108]:

$$\mathrm{FID} = \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2^2 + Tr[\boldsymbol{C}_1 + \boldsymbol{C}_2 - 2(\boldsymbol{C}_1\boldsymbol{C}_2)^{1/2}] \tag{5}$$

where $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ and $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ are the means and covariance matrices of the real and generated feature vectors, respectively. The lower the FID score, the higher the similarity between the generated and the real images, with a FID $= 0$ indicating that the two sets are identical. Second, we perform visual inspection of the generated patterns to check for the presence of any artifacts in the generated images and confirm their resemblance to real patterns. Finally, we perform an assessment of the diversity of the generated patterns by comparing the change in strain energy ($\Delta\Psi$) obtained from finite element simulations performed on the generated patterns to the same quantity obtained from simulations performed on real patterns from the Cahn–Hilliard dataset. The performance of our generative approaches is reported in Sec. 3.1.

2.6 Note on Standard Rotation-Based Augmentation.

In addition to augmenting our training dataset with generated patterns, we further augment the training dataset of the metamodel by performing direct transformations on both real and generated input patterns. This type of straightforward data augmentation occurs after the real and generated input patterns have been used to run finite element simulations. Because we are considering an

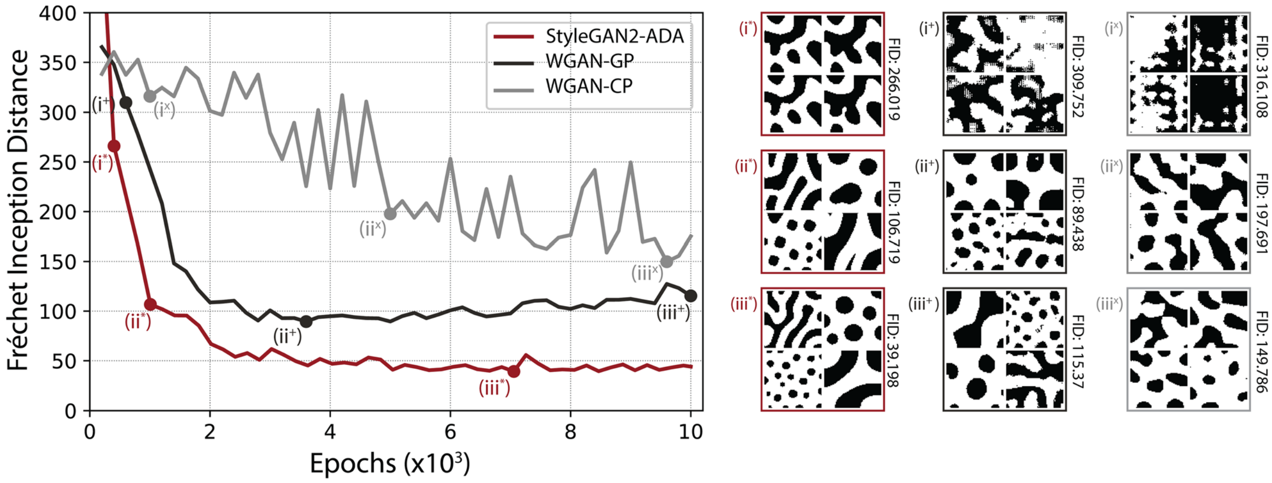

Fig. 3 FID with respect to the number of epochs for the StyleGAN2-ADA, WGAN-CP, and WGAN-GP ML-based generative models. In the right panel, we include examples of output patterns as model training proceeds to visualize the relationship between a lower FID value and improved resemblance to the real input pattern. We note that all ML-based generative models are trained with just 1000 examples.

equibiaxial extension load case in this work, we can increase the size of the training dataset by a factor of 4 by applying a set of predefined rotations (0°, 90°, 180°, 270°) on the input images. For all four rotated scenarios, the FEA simulation output $\Delta\Psi$ is identical, thus we can gain four data points per pattern. We report the significance of this standard augmentation on the metamodel performance in Sec. 3.3.

3 Results and Discussion

In this section, we report the results of employing the methods described in Secs. 2.2, 2.3, 2.4, and 2.6 to augment a small dataset of input patterns and train a convolutional neural network to predict the change in strain energy $\Delta\Psi$ for a given magnitude of applied equibiaxial extension. We begin in Sec. 3.1 by describing the performance of the generative models when trained with just 1000 examples of real Cahn–Hilliard patterns. Then, in Sec. 3.2, we demonstrate the performance of a metamodel where the training set contains simulations based on both real and generated input patterns. Finally, in Sec. 3.3, we summarize the results of our transfer learning approach and the effect of standard rotation-based augmentations on metamodel performance.

3.1 Generative Model Performance. As stated previously, we have tested three different GAN models, WGAN-CP, WGAN-GP, and SyleGAN2-ADA, with the aim of generating input patterns from a small training dataset of 1000 real Cahn–Hilliard patterns. In this section, we show the performance of these methods and demonstrate that the StyleGAN2-ADA approach performs best at capturing the Cahn–Hilliard dataset. In Fig. 3, we illustrate the performance by plotting the FID between 1000 real and 1000 generated patterns with respect to the number of epochs used for training. This plot shows that the StyleGAN2-ADA approach consistently has the lowest FID and is thus producing patterns that are a better match to the real dataset. We note that as expected, the calculated FID on real versus real patterns converged to zero as we increased the size of the comparison datasets of patterns from 1000 (FID $\approx$13.3) to 10,000 (FID $\approx$1.7). In addition, we have annotated the plot in Fig. 3 with illustrated examples of generated patterns from the three generative models. These illustrations not only confirm the intuition that as the FID decreases the patterns in the generated images more closely resemble those in the real dataset, but also show that for a converged model performance, the generated patterns look quite qualitatively realistic. Based on the higher FID for the WGAN-CP and WGAN-GP models, and the fact that FID begins to increase as the number of epochs increases, we conclude that both are inferior approaches when the goal is to generate realistic patterns that closely match the original dataset. However, we note that in terms of model training time, the StyleGAN2-ADA network is significantly more expensive to train with the training process taking approximately 7.5 h on 4 NVIDIA Tesla V100 GPUs. In comparison, it took approximately 0.5 h to train each of the WGAN-CP and WGAN-GP models on NVIDIA GeForce RTX 3060 Ti.

In Fig. 4, we plot the percentage frequency distribution of the change in strain energy $\Delta\Psi$ for 15,000 low fidelity real and generated patterns subjected to small displacement ($d = 0.001$) with equibiaxial extension finite element simulations. From comparing the distributions of $\Delta\Psi$, it appears that the StyleGAN2-ADA output distribution bears the most similarity to the real dataset. However, even though the WGAN and procedural patterns are less authentic than StyleGAN2-ADA patterns, they are more divergent from the original 1000 example real dataset while still maintaining overlap with the real distribution of $\Delta\Psi$. Finally, the Bernoulli patterns appear only weakly relevant to the real dataset. From performing these simulations, we now have multiple datasets of low fidelity finite element simulations based on both real and generated input patterns that we can use to augment our ML model training datasets.

3.2 Metamodel Performance With an Augmented Training Dataset. With our trained ML-based generative models and procedural algorithm-based generative models, we are able to generate synthetic input patterns and use them as inputs to finite element simulations where the results are used to augment our metamodel training datasets. In Fig. 5, we show the test performance of the CNN-based metamodel defined in Sec. 2.2.1 trained on these data. We report the $R2$ score computed on held out test data with respect to dataset size for five different types of training dataset. The first training dataset type is composed of real patterns only. The rest of the training dataset types contain a fixed number of real data points (1000), and the size of these datasets is increased by adding simulation results obtained from patterns generated using WGAN-CP, WGAN-GP, StyleGAN2-ADA, procedural, or Bernoulli methods, respectively. For all training dataset types, we consider sample sizes of 1000, 2000, 4000, and 16,000 patterns. For reference, training our CNN-based metamodel for 100 epochs with 16,000 samples took $\approx$2 min on a single Nvidia Tesla V100 GPU. The results presented in Fig. 5 reveal that metamodels trained with WGAN-GP and procedural patterns perform nearly equivalently with $R2$ scores of 0.9975 and exhibit only a slightly inferior performance to a metamodel trained entirely on

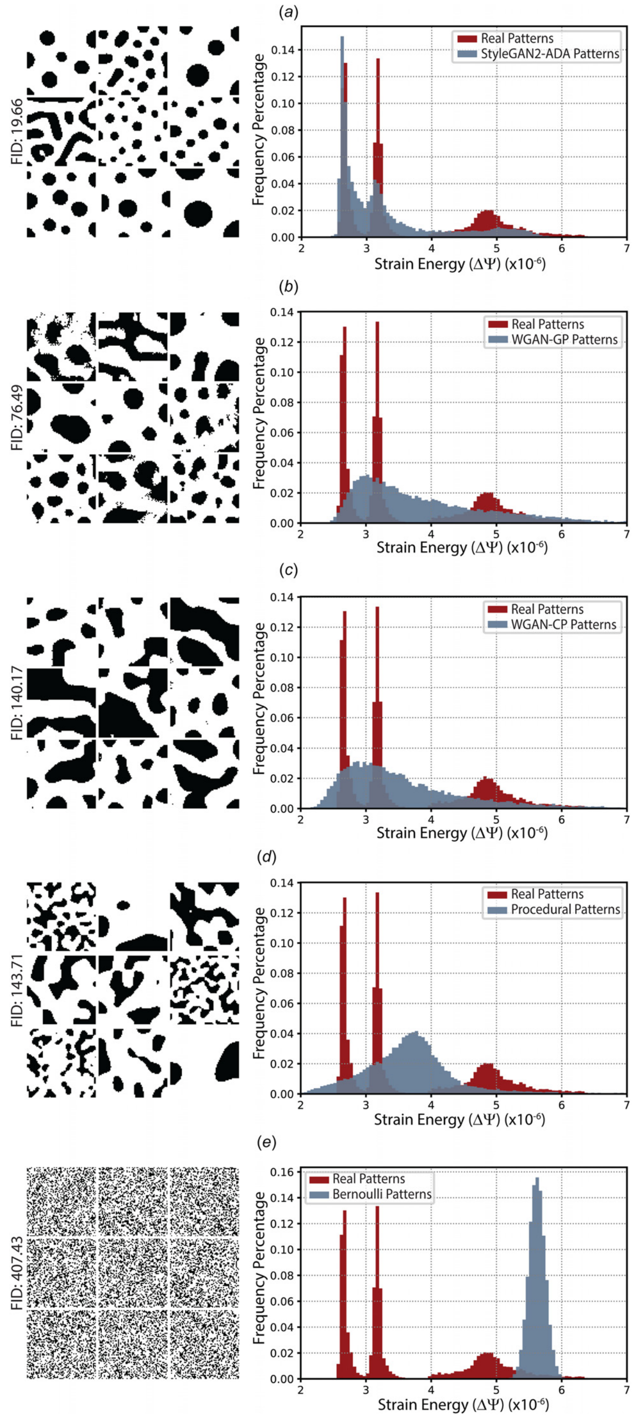

Fig. 4 Visualization of the ML-based and procedural generative model results in order of increasing FID. For each pattern type, we show a comparison of strain energy $\Delta\Psi$ at $d = 0.001$ for real and generated patterns with low fidelity data for: (*a*) StyleGAN2-ADA patterns, (*b*) WGAN-GP patterns, (*c*) WGAN-CP patterns, (*d*) procedural patterns, and (*e*) Bernoulli patterns. Overall, patterns produced with StyleGAN2-ADA have the highest similarity to the real dataset. We note that all ML-based generative models were trained with just 1000 examples, whereas the procedural and Bernoulli patterns rely on no training data, only a knowledge of the average number of bright and dark pixels in the target dataset.

real patterns ($R2 = 0.9992$) for a dataset size of 16, 000. Notably, in all cases, the addition of the generated input patterns improves the performance of the metamodel except for 1000 and 3000 random Bernoulli patterns, which decreased the metamodel

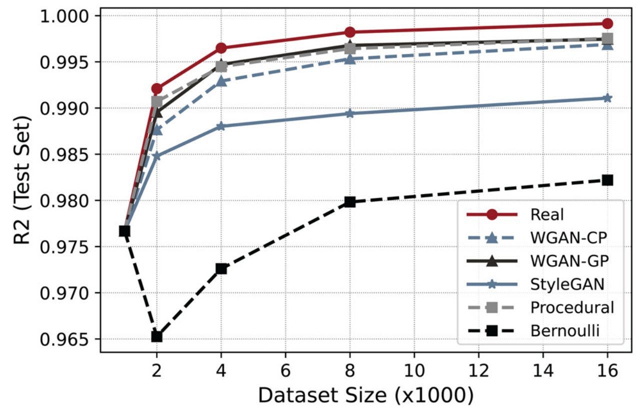

Fig. 5 Metamodel performance with respect to the size of the training dataset. Note that "dataset size" refers to the combined number of unique real and generated synthetic patterns. For a dataset of 16, 000 real patterns, *R*2 is 0.9992. For a dataset of 1000 real and 15, 000 synthetic patterns, the metamodel performance with procedural and WGAN-GP patterns is almost identical with *R*2 values of 0.9975. For augmentations with WGAN-CP, StyleGAN2-ADA, and Bernoulli patterns, the corresponding *R*2 values are 0.9969, 0.9911, and 0.9822, respectively.

performance. This is anticipated because these patterns are not spatially correlated the way real Cahn–Hilliard patterns are, as depicted in Fig. 4. In fact, we find the very slight improvement in the performance of the metamodels when the dataset is augmented with more than 8000 of this type of synthetic patterns to be counterintuitive. Comparing the metamodel performance for dataset augmentations with StyleGAN2-ADA patterns versus WGAN-GP and procedural patterns, we anticipate that the diversity of the WGAN-GP and procedural synthetic patterns proves to be more important than the authenticity of the StyleGAN2-ADA patterns for enhancing metamodel performance. Namely, even though the StyleGAN2-ADA patterns were closer to real patterns than the WGAN-GP patterns, they were perhaps less diverse or even too similar to the real patterns used for training and thus less beneficial when training the predictive model.

3.3 Metamodel Performance With Transfer Learning.

After training the metamodels on datasets based on low fidelity simulation data, we evaluate the efficacy of our straightforward transfer learning approach described in Sec. 2.2.2 to make predictions on the corresponding high fidelity simulation dataset. We begin with our metamodel pretrained using the weights obtained from our low fidelity dataset metamodel trained with 1000 real and 15, 000 generated patterns with rotation-based augmentation as described in Sec. 2.6. With this additional rotation-based augmentation, a dataset size of N corresponds to $4N$ training points. Then, we perform additional training with 1000 real pattern-based high fidelity simulations. As shown in Table 1, this transfer learning-based training process predicts the change in strain energy $\Delta\Psi$ at the final displacement for test data with $R2$ score of 0.9977 and corresponding mean absolute error (MAE) of 0.0074 when the weights are initialized with the best performing metamodel trained with a dataset augmented with StyleGAN2-ADA patterns in addition to the rotation-based methods. We note that although the performance of metamodels trained with datasets augmented with WGAN-CP, WGAN-GP and procedural patterns (with or without additional rotations), is better than equivalent metamodels trained based on StyleGAN2-ADA augmented datasets (see Fig. 5 and Table 1 "pre-training" column), the StyleGAN2-ADA augmented model performs best after transfer learning. Alternatively, training a metamodel initialized with random weights predicts $\Delta\Psi$ for the same high fidelity dataset with an $R2$ of 0.9783 and corresponding MAE of 0.0198. We note that the real patterns used in the training and test sets in the low fidelity metamodel training match the patterns used in the high fidelity metamodel training, and that the same 1000 real patterns are used as training data for our metamodels and the generative models. Overall, this demonstrates that synthetic pattern-based and rotation-based data augmentation strategies can be combined with our previously explored transfer learning approach [28] to create meaningful training datasets that rely on only a small number of representative input pattern images and are computationally cheap to generate. Based on our investigations, we find that procedural patterns, when possible to generate, can not only be an effective choice, but also may be a better choice than ML-based generative models in some circumstances. When it is not possible to generate procedural patterns, our results indicate that both WGAN-GP and StyleGAN2-ADA are good choices for ML-based generative models.

4 Conclusion

In this paper, we extend our previous work on using machine learning-based metamodels to predict mechanical quantities of interest in heterogeneous materials [28–30] to include a method for working with size-limited datasets. Specifically, we are interested in developing tools for making smaller datasets (with as few as 1000 example input patterns) amenable to deep learning approaches. To accomplish this, we first create a new dataset of spatially heterogeneous domains undergoing large deformation with material property patterns based on the Cahn–Hilliard equation, the mechanical MNIST Cahn–Hilliard dataset. In contrast to our previous work [43,44], these input patterns are more relevant to heterogeneous biological materials. In this paper, we present a brief overview of the underlying theory behind the Cahn–Hilliard equations and describe the procedure for generating the dataset. Then, with this dataset, we test the efficacy of different generative adversarial network (GAN) models at generating new Cahn–Hilliard patterns from a limited training dataset of 1000 example patterns. Of the approaches that we explored, we found that the StyleGAN2-ADA model performed best at generating synthetic Cahn–Hilliard patterns (FID $= 39.2$). In addition to GAN-based synthetic patterns, we explored two procedural approaches and created two additional types of synthetic Cahn–Hilliard patterns, procedural patterns and spatially uncorrelated Bernoulli patterns. With ML-based and procedural-based generated patterns, we then created low fidelity (i.e., computationally cheap through coarse mesh and perturbation displacements) finite element simulation datasets comprised of 1000 simulations based on real input patterns and $15,000$ simulations based on generated patterns. We then compared the performance of metamodels trained on these hybrid real and generated input pattern datasets to a metamodel trained entirely on real patterns and found that our data augmentation approaches were highly effective ($R2$ of 0.9975 for procedural and WGAN-GP augmentation-based datasets and $R2$ of 0.9992 for the dataset based entirely on real patterns). In addition, we built on our previous work in using transfer learning to leverage low fidelity simulation datasets [28], and demonstrated that with just 1000 high fidelity (i.e., refined mesh, full applied displacement) finite element simulations, we could transfer a low fidelity metamodel to the high fidelity dataset and obtain an $R2$ score of 0.9976 and corresponding MAE of 0.0074 for predicting change in strain energy. This final result was obtained with 1000 unique real input patterns, 1000 real pattern low fidelity simulations, 1000 real pattern high fidelity simulations, and $15,000$ low fidelity simulations with StyleGAN2-ADA generated input patterns.

Broadly speaking, we anticipate that the work presented in this paper will motivate multiple future research directions. To this end, we have made both our mechanical MNIST Cahn–Hilliard dataset and our metamodel implementation readily available for other research groups to build on under open-source licenses (see Sec. 5). In the future, we anticipate that others may implement alternative approaches to this problem that exceed the baseline performance established in this paper. Here, we established baseline performance for three problems: (1) training generative models with just 1000 example patterns, (2) demonstrating the effectiveness of simple procedural data generation and augmentation approaches, and (3) training a metamodel based on a finite element simulation dataset where the relevant material property distribution is defined by just 1000 example patterns. However, because our dataset is published under an open source license, others are free to formulate different challenges and attempt the same problem with an entirely different metamodeling approach. In particular, we anticipate future work in developing more sophisticated approaches for representing the input pattern space, future work in predicting full field quantities of interest in addition to the single quantity of interest predicted here, and future work in accounting for more aspects that render modeling soft tissue very challenging, such as material anisotropy and the broad uncertainty in material properties. In addition, we plan to extend the mechanical MNIST Cahn–Hilliard dataset to include additional constitutive parameters, a more diverse set of constitutive models, and additional loading scenarios in the future. In addition, we note that there should be further future investigation into the minimum number of data points required to train a GAN model for this type of problem. In this work, we relied entirely on a pragmatic selection of 1000 data points simply because 100 would likely be insufficient for training a GAN, and $10,000$ would no longer be resolutely in the size-limited datasets regime. Looking forward, we hope that the findings in this work will make deep learning-based metamodels much more accessible for researchers working with limited examples of their input pattern spaces of interest.

5 Additional Information

The mechanical MNIST Cahn–Hilliard dataset is available through the OpenBU Institutional Repository at following link[3] [109]. We provide with this dataset a supplementary document that includes more details on the theory of the Cahn–Hilliard equation and our finite element implementation. The codes for generating the Cahn–Hilliard patterns and for performing the finite element equibiaxial extension simulations using FENICS computing platform[4] are available on GITHUB at following link.[5] The codes for implementing the metamodel pipeline including the convolutional neural network model for a single quantity of interest prediction and the GAN model for data synthesis are also made available at the following link.[6]

Acknowledgment

We would like to thank the staff of the Boston University research computing services and the OpenBU Institutional Repository (in particular Eleni Castro) for their invaluable assistance with generating and disseminating the mechanical MNIST Cahn–Hilliard dataset. This work was made possible through start up funds from the Boston University Department of Mechanical Engineering, the David R. Dalton Career Development Professorship, the Hariri Institute Junior Faculty Fellowship, the National Science Foundation Engineering Research Center CELL-MET NSF EEC-1647837, and the Office of Naval Research Award N00014-22-1-2066.

Funding Data

- National Science Foundation (Grant No. CELL-MET NSF EEC-1647837; Funder ID: 10.13039/100000001).

[3]https://open.bu.edu/handle/2144/43971
[4]https://fenicsproject.org
[5]https://github.com/elejeune11/Mechanical-MNIST-Cahn-Hilliard
[6]https://github.com/saeedmhz/cahn-hilliard

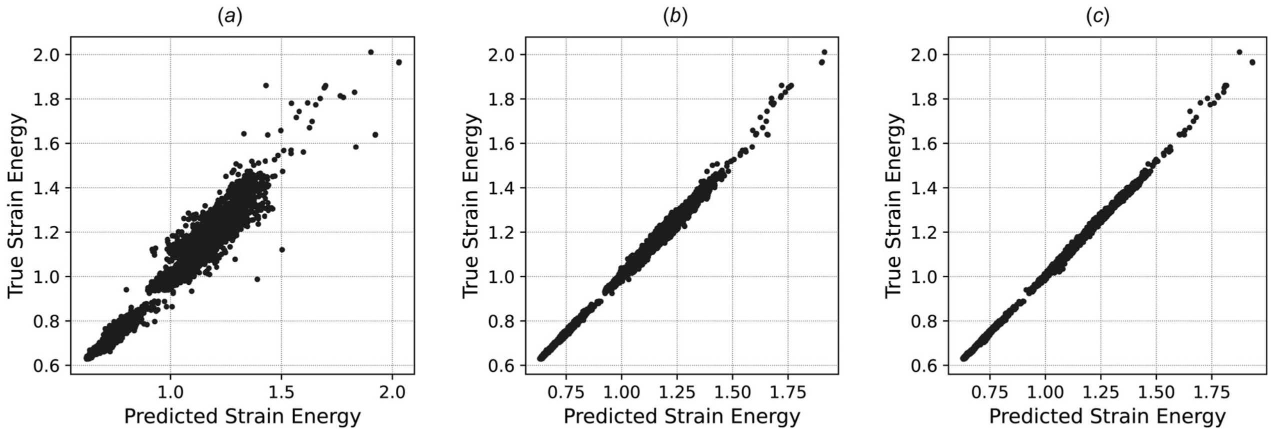

Fig. 6 Qualitative interpretation of *R*2 scores for transfer learning evaluation. True versus predicted strain energy values of high fidelity test data are plotted for three different metamodels trained with 1000 high fidelity real data points. (*a*) Metamodel weights are initialized randomly (i.e., no transfer learning is performed). (*b*) Metamodel weights are transferred from a model trained on 1000 low fidelity real samples and 15,000 generated samples from StyleGAN2-ADA with extra rotation-based augmentations. (*c*) Metamodel weights are initialized by transferring weights of a model trained on 16,000 low fidelity real data with extra rotation-based augmentations.

- Office of Naval Research (Grant No. N00014-22-1-2066; Funder ID: 10.13039/100000006).

Appendix A: Qualitative Visualization of Metamodel Performance

In this Appendix, we provide additional information to support how synthetic data augmentation combined with a simple transfer learning approach can help improve the performance of our metamodel. As shown in Sec. 3.3, initializing the weights of our metamodel with the weights of a model trained on low fidelity real data augmented with the proper set of generated data can increase the $R2$ score of predicted high fidelity strain energy values from 0.9783 to 0.9977. In order to qualitatively interpret the benefits of this improvement, we plotted true versus predicted strain energy values for all samples in the test set of the high fidelity dataset for three different models (Fig. 6). Figure 6(*a*) shows the results where no transfer learning is performed, whereas Fig. 6(*b*) shows the case where the weights are transferred from a model trained on 1000 low fidelity real samples and 15,000 synthetic samples generated from StyleGAN2-ADA with extra rotation-based augmentations. In Fig. 6(*c*) the initial weights are transferred from a low fidelity model trained on 16,000 real data with rotation-based augmentations. Overall, this figure further supports our findings from Table 1 and Sec. 3.3 where we state the performance of our metamodels in terms of $R2$ score.

References

[1] Nikpasand, M., Mahutga, R. R., Bersie-Larson, L. M., Gacek, E., and Barocas, V. H., 2021, "A Hybrid Microstructural-Continuum Multiscale Approach for Modeling Hyperelastic Fibrous Soft Tissue," J. Elast., **145**(1–2), pp. 295–319.

[2] Fan, R., and Sacks, M. S., 2014, "Simulation of Planar Soft Tissues Using a Structural Constitutive Model: Finite Element Implementation and Validation," J. Biomech., **47**(9), pp. 2043–2054.

[3] Berkley, J., Turkiyyah, G., Berg, D., Ganter, M., and Weghorst, S., 2004, "Real-Time Finite Element Modeling for Surgery Simulation: An Application to Virtual Suturing," IEEE Trans. Visual. Comput. Graph., **10**(3), pp. 314–325.

[4] Joldes, G. R., Wittek, A., and Miller, K., 2009, "Suite of Finite Element Algorithms for Accurate Computation of Soft Tissue Deformation for Surgical Simulation," Med. Image Anal., **13**(6), pp. 912–919.

[5] Zhang, J., Zhong, Y., and Gu, C., 2018, "Deformable Models for Surgical Simulation: A Survey," IEEE Rev. Biomed. Eng., **11**, pp. 143–164.

[6] Joldes, G., Bourantas, G., Zwick, B., Chowdhury, H., Wittek, A., Agrawal, S., Mountris, K., Hyde, D., Warfield, S. K., and Miller, K., 2019, "Suite of Meshless Algorithms for Accurate Computation of Soft Tissue Deformation for Surgical Simulation," Med. Image Anal., **56**, pp. 152–171.

[7] Sahli-Costabal, F., Seo, K., Ashley, E., and Kuhl, E., 2020, "Classifying Drugs by Their Arrhythmogenic Risk Using Machine Learning," Biophys. J., **118**(5), pp. 1165–1176.

[8] Sree, V. D., Rausch, M. K., and Tepole, A. B., 2019, "Linking Microvascular Collapse to Tissue Hypoxia in a Multiscale Model of Pressure Ulcer Initiation," Biomech. Model. Mechanobiol., **18**(6), pp. 1947–1964.

[9] Kong, F., Pham, T., Martin, C., McKay, R., Primiano, C., Hashim, S., Kodali, S., and Sun, W., 2018, "Finite Element Analysis of Tricuspid Valve Deformation From Multi-Slice Computed Tomography Images," Ann. Biomed. Eng., **46**(8), pp. 1112–1127.

[10] Kakaletsis, S., Meador, W. D., Mathur, M., Sugerman, G. P., Jazwiec, T., Malinowski, M., Lejeune, E., Timek, T. A., and Rausch, M. K., 2021, "Right Ventricular Myocardial Mechanics: Multi-Modal Deformation, Microstructure, Modeling, and Comparison to the Left Ventricle," Acta Biomater., **123**, pp. 154–166.

[11] Avazmohammadi, R., Li, D. S., Leahy, T., Shih, E., Soares, J. S., Gorman, J. H., Gorman, R. C., and Sacks, M. S., 2018, "An Integrated Inverse Model-Experimental Approach to Determine Soft Tissue Three-Dimensional Constitutive Parameters: Application to Post-Infarcted Myocardium," Biomech. Model. Mechanobiol., **17**(1), pp. 31–53.

[12] Ogden, R. W., 2017, *Nonlinear Continuum Mechanics and Modeling the Elasticity of Soft Biological Tissues With a Focus on Artery Walls*, Springer International Publishing, Cham, Switzerland, pp. 83–156.

[13] Ateshian, G. A., 2017, *Mixture Theory for Modeling Biological Tissues: Illustrations From Articular Cartilage, Pages*, Springer International Publishing, Cham, Switzerland, pp. 1–51.

[14] Holzapfel, G., 2001, *Biomechanics of Soft Tissue*, Academic Press, San Diego, CA, pp. 1049–1063.

[15] Humphrey, J. D., 2013, "Multiscale Modeling of Arterial Adaptations: Incorporating Molecular Mechanisms Within Continuum Biomechanical Models," *Computer Models in Biomechanics, Year 2013*, Gerhard A. Holzapfel, and Ellen Kuhl, eds., Springer Netherlands, Dordrecht, pp. 119–127.

[16] Hughes, T. J., 2012, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*, Courier Corporation, Mineola, NY.

[17] Korenczuk, C. E., Dhume, R. Y., Liao, K. K., and Barocas, V. H., 2019, "Ex Vivo Mechanical Tests and Multiscale Computational Modeling Highlight the Importance of Intramural Shear Stress in Ascending Thoracic Aortic Aneurysms," ASME J. Biomech. Eng., **141**(12), p. 121010.

[18] Leng, Y., de Lucio, M., and Gomez, H., 2021, "Using Poro-Elasticity to Model the Large Deformation of Tissue During Subcutaneous Injection," Comput. Methods Appl. Mech. Eng., **384**, p. 113919.

[19] Mei, Y., Liu, J., Guo, X., Zimmerman, B., Nguyen, T. D., and Avril, S., 2021, "General Finite-Element Framework of the Virtual Fields Method in Nonlinear Elasticity," J. Elast., **145**(1–2), pp. 265–294.

[20] Madireddy, S., Sista, B., and Vemaganti, K., 2015, "A Bayesian Approach to Selecting Hyperelastic Constitutive Models of Soft Tissue," Comput. Methods Appl. Mech. Eng., **291**, pp. 102–122.

[21] Jadidi, M., Sherifova, S., Sommer, G., Kamenskiy, A., and Holzapfel, G. A., 2021, "Constitutive Modeling Using Structural Information on Collagen Fiber Direction and Dispersion in Human Superficial Femoral Artery Specimens of Different Ages," Acta Biomater., **121**, pp. 461–474.

[22] Tonutti, M., Gras, G., and Yang, G.-Z., 2017, "A Machine Learning Approach for Real-Time Modelling of Tissue Deformation in Image-Guided Neurosurgery," Artif. Intell. Med., **80**, pp. 39–47.

[23] Tac, V., Sree, V. D., Rausch, M. K., and Tepole, A. B., 2021, "Data-Driven Modeling of the Mechanical Behavior of Anisotropic Soft Biological Tissue," arXiv preprint arXiv:2107.05388.

[24] Tajdari, M., Pawar, A., Li, H., Tajdari, F., Maqsood, A., Cleary, E., Saha, S., Zhang, Y. J., Sarwark, J. F., and Liu, W. K., 2021, "Image-Based Modelling for Adolescent Idiopathic Scoliosis: Mechanistic Machine Learning Analysis and Prediction," Comput. Methods Appl. Mech. Eng., **374**, p. 113590.

[25] Li, A., Farimani, A. B., and Zhang, Y. J., 2021, "Deep Learning of Material Transport in Complex Neurite Networks," Sci. Rep., **11**(1), pp. 1–13.

[26] Peng, G. C., Alber, M., Tepole, A. B., Cannon, W. R., De, S., Dura-Bernal, S., Garikipati, K., Karniadakis, G., Lytton, W. W., Perdikaris, P., Petzold, L., and Kuhl, E., 2021, "Multiscale Modeling Meets Machine Learning: What Can we Learn?," Arch. Comput. Methods Eng., **28**(3), pp. 1017–1037.

[27] Peirlinck, M., Costabal, F. S., Sack, K. L., Choy, J. S., Kassab, G. S., Guccione, J. M., De Beule, M., Segers, P., and Kuhl, E., 2019, "Using Machine Learning to Characterize Heart Failure Across the Scales," Biomech. Model. Mechanobiol., **18**(6), pp. 1987–2001.

[28] Lejeune, E., and Zhao, B., 2021, "Exploring the Potential of Transfer Learning for Metamodels of Heterogeneous Material Deformation," J. Mech. Behav. Biomed. Mater., **117**, p. 104276.

[29] Lejeune, E., 2020, "Mechanical MNIST: A Benchmark Dataset for Mechanical Metamodels," Ext. Mech. Lett., **36**, p. 100659.

[30] Mohammadzadeh, S., and Lejeune, E., 2022, "Predicting Mechanically Driven Full-Field Quantities of Interest With Deep Learning-Based Metamodels," Ext. Mech. Lett., **50**, p. 101566.

[31] Yang, Z., Yu, C.-H., Guo, K., and Buehler, M. J., 2021, "End-to-End Deep Learning Method to Predict Complete Strain and Stress Tensors for Complex Hierarchical Composite Microstructures," J. Mech. Phys. Solids, **154**, p. 104506.

[32] Mianroodi, J. R., Siboni, N. H., and Raabe, D., 2021, "Teaching Solid Mechanics to Artificial Intelligence-a Fast Solver for Heterogeneous Materials," NPJ Comput. Mater., **7**(1), pp. 1–10.

[33] Stowers, C., Lee, T., Bilionis, I., Gosain, A. K., and Tepole, A. B., 2021, "Improving Reconstructive Surgery Design Using Gaussian Process Surrogates to Capture Material Behavior Uncertainty," J. Mech. Behav. Biomed. Mater., **118**, Article No. 104340.

[34] Yang, H., Guo, X., Tang, S., and Liu, W. K., 2019, "Derivation of Heterogeneous Material Laws Via Data-Driven Principal Component Expansions," Comput. Mech., **64**(2), pp. 365–379.

[35] Liu, Z., Wu, C. T., and Koishi, M., 2019, "A Deep Material Network for Multiscale Topology Learning and Accelerated Nonlinear Modeling of Heterogeneous Materials," Comput. Methods Appl. Mech. Eng., **345**, pp. 1138–1168.

[36] Forrester, A. I., and Keane, A. J., 2009, "Recent Advances in Surrogate-Based Optimization," Prog. Aerospace Sci., **45**(1–3), pp. 50–79.

[37] Salehi, Y., and Giannacopoulos, D., 2021, "PhysGNN: A Physics-Driven Graph Neural Network Based Model for Predicting Soft Tissue Deformation in Image-Guided Neurosurgery," arXiv preprint arXiv:2109.04352.

[38] Lu, L., Dao, M., Kumar, P., Ramamurty, U., Karniadakis, G. E., and Suresh, S., 2020, "Extraction of Mechanical Properties of Materials Through Deep Learning From Instrumented Indentation," Proc. Natl. Acad. Sci., **117**(13), pp. 7052–7062.

[39] Teichert, G. H., and Garikipati, K., 2019, "Machine Learning Materials Physics: Surrogate Optimization and Multi-Fidelity Algorithms Predict Precipitate Morphology in an Alternative to Phase Field Dynamics," Comput. Methods Appl. Mech. Eng., **344**, pp. 666–693.

[40] Haghighat, E., Raissi, M., Moure, A., Gomez, H., and Juanes, R., 2021, "A Physics-Informed Deep Learning Framework for Inversion and Surrogate Modeling in Solid Mechanics," Comput. Methods Appl. Mech. Eng., **379**, p. 113741.

[41] Brock, A., Donahue, J., and Simonyan, K., 2019, "Large Scale GAN Training for High Fidelity Natural Image Synthesis," International Conference on Learning Representation, New Orleans, LA, May 6–9.

[42] Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., and Aila, T., 2020, "Training Generative Adversarial Networks with Limited Data," Advances in Neural Information Processing Systems 33, Virtual, Dec. 6–12, Curran Associates, Red Hook, NY, pp. 12104–12114.

[43] Lejeune, E., 2020, "Mechanical MNIST-Fashion," OpenBU, Boston, MA, accessed Jan. 5, 2022, https://open.bu.edu/handle/2144/41450

[44] Lejeune, E., 2019, "Mechanical MNIST-Uniaxial Extension," OpenBU, Boston, MA, accessed Jan. 5, 2022, https://open.bu.edu/handle/2144/38693

[45] Mohammadzadeh, S., and Lejeune, E., 2021, "Mechanical MNIST Crack Path," OpenBU, Boston, MA, accessed Jan. 5, 2022, https://open.bu.edu/handle/2144/42757

[46] Prachaseree, P., and Lejeune, E., 2021, "Asymmetric Buckling Columns (ABC)," OpenBU, Boston, MA, accessed Feb. 3, 2022, https://open.bu.edu/handle/2144/43730

[47] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., 2014, "Generative Adversarial Nets," Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, NIPS'14, MIT Press, Cambridge, MA, pp. 2672–2680.

[48] Jangid, D. K., Brodnik, N. R., Khan, A., Goebel, M. G., Echlin, M. P., Pollock, T. M., Daly, S. H., and Manjunath, B. S., 2022, "3D Grain Shape Generation in Polycrystals Using Generative Adversarial Networks," Integr. Mater. Manuf. Innov., **11**(1), pp. 71–84.

[49] Ma, W., Kautz, E. J., Baskaran, A., Chowdhury, A., Joshi, V., Yener, B., and Lewis, D. J., 2020, "Image-Driven Discriminative and Generative Machine Learning Algorithms for Establishing Microstructure–Processing Relationships," J. Appl. Phys., **128**(13), p. 134901.

[50] LeCun, Y., and Cortes, C., 2010, "MNIST Handwritten Digit Database," MNIST, accessed Dec. 28, 2021, http://yann.lecun.com/exdb/mnist/

[51] Xiao, H., Rasul, K., and Vollgraf, R., 2017, "Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms," arXiv preprint arXiv:1708.07747

[52] Turing, A. M., 1990, "The Chemical Basis of Morphogenesis," Bull. Math. Biol., **52**(1–2), pp. 153–197.

[53] Maini, P. K., Woolley, T. E., Baker, R. E., Gaffney, E. A., and Lee, S. S., 2012, "Turing's Model for Biological Pattern Formation and the Robustness Problem," Interface Focus, **2**(4), pp. 487–496.

[54] Garikipati, K., 2017, "Perspectives on the Mathematics of Biological Patterning and Morphogenesis," J. Mech. Phys. Solids, **99**, pp. 192–210.

[55] Grant, C. P., 1993, "Spinodal Decomposition for the Cahn–Hilliard Equation," Commun. Partial Diff. Eqs., **18**(3–4), pp. 453–490.

[56] Wang, Z., Huan, X., and Garikipati, K., 2019, "Variational System Identification of the Partial Differential Equations Governing the Physics of Pattern-Formation: Inference Under Varying Fidelity and Noise," Comput. Methods Appl. Mech. Eng., **356**, pp. 44–74.

[57] Wang, Z., Huan, X., and Garikipati, K., 2021, "Variational System Identification of the Partial Differential Equations Governing Microstructure Evolution in Materials: Inference Over Sparse and Spatially Unrelated Data," Comput. Methods Appl. Mech. Eng., **377**, p. 113706.

[58] Sainburg, T., McInnes, L., and Gentner, T. Q., 2021, "Parametric Umap Embeddings for Representation and Semisupervised Learning," Neural Comput., **33**(11), pp. 2881–2907.

[59] Wells, G. N., Kuhl, E., and Garikipati, K., 2006, "A Discontinuous Galerkin Method for the Cahn–Hilliard Equation," J. Comput. Phys., **218**(2), pp. 860–877.

[60] FEniCS Project, 2016, "Cahn–Hilliard Equation," FEniCS Project, accessed Dec. 4, 2020, https://fenicsproject.org/olddocs/dolfin/1.4.0/python/demo/documented/cahn-hilliard/python/documentation.html

[61] Alnæs, M., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., Richardson, C., Ring, J., Rognes, M. E., and Wells, G. N., 2015, "The FEniCS Project Version 1.5," Arch. Numer. Software, **3**(100), pp. 9–23.

[62] Logg, A., Mardal, K.-A., and Wells, G., 2012, *Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book*, Vol. 84, Springer Science & Business Media, Heidelberg, Germany.

[63] Wodo, O., and Ganapathysubramanian, B., 2011, "Computationally Efficient Solution to the Cahn–Hilliard Equation: Adaptive Implicit Time Schemes, Mesh Sensitivity Analysis and the 3D Isoperimetric Problem," J. Comput. Phys., **230**(15), pp. 6037–6060.

[64] Gómez, H., Calo, V. M., Bazilevs, Y., and Hughes, T. J., 2008, "Isogeometric Analysis of the Cahn–Hilliard Phase-Field Model," Comput. Methods Appl. Mech. Eng., **197**(49–50), pp. 4333–4352.

[65] Bradski, G., and Kaehler, A., 2008, *Learning OpenCV: Computer Vision With the OpenCV Library*, O'Reilly Media, Sebastopol, CA.

[66] Schlömer, N., Cervone, A., McBain, G. D., Tryfon mw, van Staden, R., Gokstorp, F., Toothstone, Dokken, J. S., Anzil, Sanchez, J., Kempf, D., Bussonnier, M., Feng, Y., Awa5114, Maric, T., Chen, S., Nilswagner, Nate, Ivanmultiwave, and Fu, F., 2020, "nschloe/pygmsh v6.1.1," Zenodo, Switzerland.

[67] Geuzaine, C., and Remacle, J.-F., 2009, "GMSH: A 3D Finite Element Mesh Generator With Built-In Pre-and Post-Processing Facilities," Int. J. Numer. Methods Eng., **79**(11), pp. 1309–1331.

[68] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S., 2019, " Pytorch: An Imperative Style, High-Performance Deep Learning Library," Advances in Neural Information Processing Systems 32, Vancouver, BC, Canada, Dec. 8–14, Curran Associates, Red Hook, NY, pp. 8024–8035.

[69] Kingma, D. P., and Ba, J., 2017, "Adam: A Method for Stochastic Optimization," arXiv preprint arXiv:1412.6980.

[70] Yosinski, J., Clune, J., Bengio, Y., and Lipson, H., 2014, "How Transferable Are Features in Deep Neural Networks?" Advances in Neural Information Processing Systems 27, Montréal, QC, Canada, Dec. 8–13, Curran Associates, Red Hook, NY.

[71] Arjovsky, M., Chintala, S., and Bottou, L., 2017, "Wasserstein Generative Adversarial Networks," Proceedings of the 34th International Conference on Machine Learning, Volume 70 of Proceedings of Machine Learning Research, D. Precup, and Y. W. Teh, eds., PMLR, Sydney, Australia, Aug. 6–11, pp. 214–223.

[72] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C., 2017, "Improved Training of Wasserstein GANs," *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds., Vol. 30, Curran Associates, Red Hook, NY.

[73] Bishop, C. M., and Nasrabadi, N. M., 2006, *Pattern Recognition and Machine Learning*, Vol. 4, Springer, New York, Chap. 1, p. 43.

[74] Bengio, Y., Laufer, E., Guillaume, A., and Yosinski, J., 2014, "Deep Generative Stochastic Networks Trainable by Backprop," Proceedings of the 31st International Conference on Machine Learning, Beijing, China, June 21–26, E. P. Xing and T. Jebara, eds., **32**(2), PMLR, pp. 226–234.

[75] Gutmann, U. M., and Hyvärinen, A., 2010, "Noise-Contrastive Estimation: A New Estimation Principle for Unnormalized Statistical Models," Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS2010), JMLR Workshop and Conference Proceedings, International Conference, Sardinia, Italy, May 13–15, pp. 297–304.

[76] Schmidhuber, J., 1992, "Learning Factorial Codes by Predictability Minimization," Neural Comput., **4**(6), pp. 863–879.

[77] Tu, Z., 2007, "Learning Generative Models Via Discriminative Approaches," IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, June 17–22, pp. 1–8.

[78] Deng, L., 2012, "The Mnist Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]," IEEE Signal Process. Mag., **29**(6), pp. 141–142.
[79] Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P., 1998, "Gradient-Based Learning Applied to Document Recognition," Proc. IEEE, **86**(11), pp. 2278–2324.
[80] Susskind, J. M., Anderson, A. K., and Hinton, G. E., 2010, *The Toronto Face Database*, Department of Computer Science, University of Toronto, Toronto, ON, Canada, Report No. TR-2010-001.
[81] Krizhevsky, A., Nair, V., and Hinton, G., 2014, "The Cifar-10 Dataset," accessed Jan. 10, 2021, https://www.cs.toronto.edu/~kriz/cifar.html
[82] Karras, T., Aila, T., Laine, S., and Lehtinen, J., 2018, "Progressive Growing of GANS for Improved Quality, Stability, and Variation," International Conference on Learning Representations, Vancouver, BC, Canada, Apr. 30–May 3.
[83] Radford, A., Metz, L., and Chintala, S., 2016, "Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks," International Conference on Learning Representations, San Juan, Puerto Rico, May 2–4.
[84] Rubner, Y., Tomasi, C., and Guibas, L. J., 2000, "The Earth Mover's Distance as a Metric for Image Retrieval," Int. J. Comput. Vision, **40**(2), pp. 99–121.
[85] Rubner, Y., and Tomasi, C., 2001, *Perceptual Metrics for Image Database Navigation*, Springer Science & Business Media, Norwell, MA.
[86] Ling, H., and Okada, K., 2007, "An Efficient Earth Mover's Distance Algorithm for Robust Histogram Comparison," IEEE Trans. Pattern Anal. Mach. Intell., **29**(5), pp. 840–853.
[87] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A., 2017, "Automatic Differentiation in Pytorch," Proceeding of the 31st Conference on Neural Information Processing Systems, Long Beach, CA, Dec. 4–9.
[88] Zeleni9, 2021, "pytorch-wgan," Zeleni9, accessed Nov. 3, 2021, https://github.com/Zeleni9/pytorch-wgan
[89] Karras, T., Laine, S., and Aila, T., 2019, "A Style-Based Generator Architecture for Generative Adversarial Networks," IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, June 16–20, pp. 4396–4405.
[90] Huang, X., and Belongie, S., 2017, "Arbitrary Style Transfer in Real-Time With Adaptive Instance Normalization," Proceedings of the IEEE International Conference on Computer Vision, Oct. 22–29, Venice, Italy, pp. 1501–1510.
[91] Gatys, L. A., Ecker, A. S., and Bethge, M., 2016, "Image Style Transfer Using Convolutional Neural Networks," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, June 26–July 1, pp. 2414–2423.
[92] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T., 2020, "Analyzing and Improving the Image Quality of StyleGAN," IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, June 14–19, pp. 8110–8119.
[93] Zhang, H., Zhang, Z., Odena, A., and Lee, H., 2020, "Consistency Regularization for Generative Adversarial Networks," International Conference on Learning Representations, Virtual, Apr. 26–May 1.
[94] Zhao, Z., Singh, S., Lee, H., Zhang, Z., Odena, A., and Zhang, H., 2020, "Improved Consistency Regularization for GANS," Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, Feb. 2–9, OJS, **35**(12), pp. 11033–11041.
[95] Bora, A., Price, E., and Dimakis, A. G., 2018, "AmbientGAN: Generative Models From Lossy Measurements," ICLR, International Conference on Learning Representations, Vancouver, BC, Canada, Apr. 30–May 3.
[96] Cule, D., and Torquato, S., 1999, "Generating Random Media From Limited Microstructural Information Via Stochastic Optimization," J. Appl. Phys., **86**(6), pp. 3428–3437.
[97] Fujii, D., Chen, B. C., and Kikuchi, N., 2001, "Composite Material Design of Two-Dimensional Structures Using the Homogenization Design Method," Int. J. Numer. Methods Eng., **50**(9), pp. 2031–2051.
[98] Jiao, Y., Stillinger, F. H., and Torquato, S., 2007, "Modeling Heterogeneous Materials Via Two-Point Correlation Functions: Basic Principles," Phys. Rev. E, **76**(3), p. 031110.
[99] Redenbach, C., 2009, "Microstructure Models for Cellular Materials," Comput. Mater. Sci., **44**(4), pp. 1397–1407.
[100] Pasko, A., Fryazinov, O., Vilbrandt, T., Fayolle, P.-A., and Adzhiev, V., 2011, "Procedural Function-Based Modelling of Volumetric Microstructures," Grap. Models, **73**(5), pp. 165–181.
[101] Walters, D. J., Luscher, D. J., and Yeager, J. D., 2020, "Volumetric Analysis and Mesh Generation of Real and Artificial Microstructural Geometries," MethodsX, **7**, p. 100856.
[102] Vaughan, T. J., and McCarthy, C. T., 2010, "A Combined Experimental-Numerical Approach for Generating Statistically Equivalent Fibre Distributions for High Strength Laminated Composite Materials," Compos. Sci. Technol., **70**(2), pp. 291–297.
[103] Romanov, V., Lomov, S. V., Swolfs, Y., Orlova, S., Gorbatikh, L., and Verpoest, I., 2013, "Statistical Analysis of Real and Simulated Fibre Arrangements in Unidirectional Composites," Compos. Sci. Technol., **87**, pp. 126–134.
[104] Aurenhammer, F., 1991, "Voronoi Diagrams-a Survey of a Fundamental Geometric Data Structure," ACM Comput. Surv. (CSUR), **23**(3), pp. 345–405.
[105] Falco, S., Jiang, J., De Cola, F., and Petrinic, N., 2017, "Generation of 3D Polycrystalline Microstructures With a Conditioned Laguerre-Voronoi Tessellation Technique," Comput. Mater. Sci., **136**, pp. 20–28.
[106] Bargmann, S., Klusemann, B., Markmann, J., Schnabel, J. E., Schneider, K., Soyarslan, C., and Wilmers, J., 2018, "Generation of 3D Representative Volume Elements for Heterogeneous Materials: A Review," Prog. Mater. Sci., **96**, pp. 322–384.
[107] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., 2020, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," Nat. Methods, **17**(3), pp. 261–272.
[108] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S., 2017, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," Proceeding of the 31st Conference on Neural Information Processing Systems - Vol. 30, Long Beach, CA, Dec. 4–9, pp. 6626–6637.
[109] Kobeissi, H., and Lejeune, E., 2022, Mechanical MNIST—Cahn-Hilliard," OpenBU, Boston, MA, accessed Mar. 15, 2022, https://open.bu.edu/handle/2144/43971