12-1-2021

Conditional Wasserstein generative adversarial networks applied to acoustic metamaterial design

Peter Lai
San Jose State University

Feruza Amirkulova
San Jose State University, feruza.amirkulova@sjsu.edu

Peter Gerstoft
Marine Physical Laboratory

Recommended Citation

Peter Lai, Feruza Amirkulova, and Peter Gerstoft. "Conditional Wasserstein generative adversarial networks applied to acoustic metamaterial design" *Journal of the Acoustical Society of America* (2021): 4362-4374. https://doi.org/10.1121/10.0008929

DECEMBER 16 2021

Conditional Wasserstein generative adversarial networks applied to acoustic metamaterial design

Peter Lai; Feruza Amirkulova; Peter Gerstoft

J Acoust Soc Am 150, 4362–4374 (2021)
https://doi.org/10.1121/10.0008929

JASA ARTICLE

Conditional Wasserstein generative adversarial networks applied to acoustic metamaterial design

Peter Lai,[1] Feruza Amirkulova,[1,a)] and Peter Gerstoft[2,b)]
[1]*Mechanical Engineering Department, San Jose State University, San Jose, California 95192, USA*
[2]*Marine Physical Laboratory, Scripps Institution of Oceanography, UCSD, San Diego, California 92037, USA*

ABSTRACT:
This work presents a method for the reduction of the total scattering cross section (TSCS) for a planar configuration of cylinders by means of generative modeling and deep learning. Currently, the minimization of TSCS requires repeated forward modelling at considerable computer resources, whereas deep learning can do this more efficiently. The conditional Wasserstein generative adversarial networks (cWGANs) model is proposed for minimization of TSCS in two dimensions by combining Wasserstein generative adversarial networks with convolutional neural networks to simulate TSCS of configuration of rigid scatterers. The proposed cWGAN model is enhanced by adding to it a coordinate convolution (CoordConv) layer. For a given number of cylinders, the cWGAN model generates images of 2D configurations of cylinders that minimize the TSCS. The proposed generative model is illustrated with examples for planar uniform configurations of rigid cylinders. © *2021 Acoustical Society of America*.
https://doi.org/10.1121/10.0008929
(Received 20 May 2021; revised 11 October 2021; accepted 11 November 2021; published online 16 December 2021)
[Editor: Michael R Haberman] Pages: 4362–4374

I. INTRODUCTION

Acoustic metamaterials, unlike conventional materials, can be specifically structured to achieve unnatural values of mechanical properties, such as (negative) effective mass density, bulk modulus, and refractive index.[1–4] This unique property enables more complex acoustic engineering applications, such as wave steering,[5] cloaking,[6,7] and focusing.[3,8,9] Although optimization and computational methods in direct metamaterial design have shown great performance, these methods are not capable of inverse design of complex structure. An emerging category of inverse design is based on data-driven approaches. With an increased interest of the research community in machine learning (ML), design of metamaterials and physical devices has been no exception.

Due to broader access to computational power, we are now able to design deep neural networks (NN) and train them with massive amounts of data. While the most mature works are related to computer vision,[10–12] including areas such as object detection, segmentation, motion estimation, object tracking, image classification, and captioning, we now observe the application of innovative data-driven models in various domains to solve complex problems.

ML and artificial intelligence (AI) are driving innovations in genomics,[13] natural language processing,[14] quantum mechanics, material design, marketing personalization, recommendation engines, smart cities, biometric monitoring, augmented reality, manufacturing, and self-driving cars, to name a few. Deep learning (DL) has been used in the inverse design of photonic devices,[15,16] waveguides[17] and metastructures,[18–23] and in shape reconstructions,[24] mass transport cloaking,[25] multiscale analysis of composite materials,[26] source localization,[27,28] and molecular simulations[31] with superior performance.

Bianco *et al.*[41] provided a review of a recent application of ML in acoustics, including the applications of support vector machine, K-means techniques, dictionary learning, autoencoders, and DL in acoustics. The review papers[29,41] show that ML was recently employed in solving problems of bioacoustics, speech modeling, and signal processing[29] including the automatic speech recognition, source localization in ocean acoustics,[27,28] speaker localization and tracking, and source separation and sound/audio enhancement.[30]

Recent review papers[32–36] suggest that DL, reinforcement leaning, and generative modeling assisted inverse design models can by far exceed human capability. Fan *et al.*[39] explored acoustic scattering by a single scatterer as a 2D image-to-image regression problem using CNN where the inputs were the images of convex prism objects and the outputs were the loudness fields. Gao and Zhu[22] proposed an inverse design method for acoustic metamaterials. Meng *et al.*[24] explored the inverse acoustic scattering problem that reconstructs the obstacle shape with far-field information using NN. Fan *et al.*[40] studied an inverse scattering problem using CNN which predicts the object given the total acoustic field. Kumar *et al.*[38] proposed a data-driven inverse design of spinodoid metamaterials. DL models for both forward and inverse designs work well on simple geometries described with a smaller number of parameters but they

a)Electronic mail: feruza.amirkulova@sjsu.edu, ORCID: 0000-0002-6348-4941.
b)ORCID: 0000-0002-0471-062X.

https://doi.org/10.1121/10.0008929

perform poorer as degree of freedom of parameters in design space increase, making the scaling such complex models impractical. An alternative method is to use generative modeling which will be discussed in the remaining part of this section.

Butler *et al.*[32] reviewed recent progress in ML for the chemical sciences. Sanchez-Lengeling and Aspuru-Guzik[42] discussed recent advances in inverse molecular design reviewed methods for achieving inverse design, which aims to discover tailored materials from the starting point of a particular desired functionality. Among these methods, deep generative models have been applied to numerous classes of materials, such as rational design of prospective drugs, synthetic routes to organic compounds, and optimization of photovoltaics and redox flow batteries, as well as a variety of other solid-state materials. A literature review shows the potential of ML and specifically generative modeling to learn features in high performance devices.[36,43] A properly trained network can produce high performance beyond the parameter given in the training dataset.

Generative adversarial networks (GANs)[44–49] and variational autoencoders (VAE)[50] have had a huge success since they were introduced in 2014. GANs were described by Yann LeCun, Director of AI Research at Facebook, as "the most interesting idea in the last 10 years in ML."[89] The conditional GAN[46] was introduced which can learn single-modal and multi-modal models. The Wasserstein GAN (WGAN)[47,51] was presented to improve the stability of learning and to avoid mode collapse. Mode collapse occurs when the generative model fixates on a single solution and continues to reproduce that solution, regardless of the initial noise presented to it.

Boget[49] studied a variety of adversarial regression models and made their comparison. Gretton *et al.*[52] proposed a kernel method to determine if two samples are from different distributions. Lee *et al.*[53] created the Collaborative GAN (CollaGAN) framework for missing image data imputation. Liu *et al.*[54] introduced the CoordConv model and showed that using the coordinate convolution layer, CoordConv, in a GAN produced less mode collapse. Gerstoft *et al.*[55] used a WGAN model in which the trained generator implements a parametric bootstrap.

Deep generative models have been applied in inverse design of molecular components,[32,42,56,57] metasurfaces,[18,58–61] acoustic metamaterials,[23] optical cloak[62] and filters,[63] power splitters,[64] material microstructure,[65] protein solubility,[66] rational design of prospective drugs, solid-state materials, and of photonic devices.[67–69] Liu *et al.*[67] used GAN in the network model for the inverse design of metasurfaces using electromagnetic scatterers. Jiang *et al.*[60] showed that conditional GAN[46] can learn from a small set of optimized metasurfaces to produce a large number of devices. Ma *et al.*[68] proposed and developed a deep generative model for inverse design of metamaterials using GAN.

An inverse design scheme based on the Global Optimization Network (GLOnet) was proposed[18,58,59] that reframes the optimization as the iterative training of generative NN. Wen *et al.*[61] showed that by coupling progressive growth of the network and training set with the GAN framework, generative model can output robust free-form metasurface devices. Tan *et al.*[65] developed a deep convolutional generative adversarial network (DCGAN)[70] based inverse design of microstructural materials. The optimization of an optical cloak was presented in Ref. 62 by using DCGAN[70] architecture. An *et al.*[71] designed metasurfaces for electromagnetic wave manipulation using conditional WGAN. More recently, Gurbuz *et al.*[23] proposed the design of broadband acoustic metamaterials using conditional GAN for sound insulation tasks combining GAN with finite element simulations.

Liu *et al.*[72] developed a variational autoencoder (VAE) model and modified evolution strategy (ES) to design nanophotonic metasurfaces with subwavelength features without prior knowledge of the geometry of the candidate patterns; they included the NN simulator in their scheme to expedite the searching speed. Tang *et al.*[64] proposed a conditional VAE enhanced with adversarial sensoring for the generation of power splitters. Han *et al.*[63] proposed an inverse design of metasurface optical filters using deep NN and generative modeling. More recently Ahmed *et al.*[37] and Tran *et al.*[73] proposed the inverse design models of broadband acoustic cloak using VAE and "autoencoder–like" networks.

Hybrid models were introduced combining VAE and GAN for different applications in molecular design and drug discoveries.[73,75,76] Larsen *et al.*[73] combined VAE and GAN into an unsupervised hybrid generative model that simultaneously learns to encode, generate, and compare dataset samples at the same time; they showed that the hybrid generative models, VAE/GAN, trained with learned similarity measures, produce better image samples than models trained with element-wise error measures. Brock *et al.*[77] introduced the introspective adversarial network, a hybridization of the VAE and GAN, to tackle the challenge of achieving accurate reconstructions without loss of feature quality. Kadurin *et al.*[76] developed an advanced adversarial autoencoder (AAE) model for molecular feature extraction problems, and demonstrated its advantages compared to the VAE. Blaschke *et al.*[75] studied the potential use of generative autoencoders for *de novo* molecular design considering various generative autoencoders including VAE and AAE. Zhang *et al.*[78] proposed a conditional AAE to learn the face manifold to achieve age progression and regression simultaneously by controlling the age attribute. Other hybrid models, such as VQ-VAE,[79] VEEGAN,[80] CVAE-GAN,[81] and VAE-WGAN,[82] were introduced to reduce mode collapse and lack of diversity.

In this paper, we solve the inverse design problem by means of the multiple scattering theory, DL, and generative modeling. Specifically, we attempt to inverse design an acoustic cloak,[6,7] leveraging GAN to minimize the scattering by uniform configurations of cylindrical scatterers. The generative networks[23,32–36] have a potential to produce better optimized metacluster configurations. We will demonstrate the performance of generative models for 2- and

4-scatterer configurations. Therefore, the purpose of this study is to provide a framework for incorporating DL and generative modeling techniques in inverse acoustic metacluster design. To achieve this, we develop and train a conditional Wasserstein generative adversarial network (cWGAN) to propose images of metaclusters given an expected TSCS.

The paper is organized as follows. Section II starts with a formulation of the multiple scattering problem and data generation procedure. The composition of regressor CNN model is outlined. The development of the cWGAN model and its training procedure are further described. Numerical results are presented in Sec. III considering the operating normalized wavenumber range $k_i a \in [0.35, 0.45]$ where $i = 1, \ldots, 11$ for 2-scatterer and 4-scatterer configurations. Section IV gives conclusions of this study and discusses related future work directions.

II. METHODS

A. Problem formulation and multiple scattering solution

We consider acoustic multiple scattering by an arbitrary planar configuration of M cylindrical scatterer in the context of the acoustic time harmonic wave equation.[7,83–85] The total pressure field $p(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^2$ is defined as the sum of incident p_{inc} and scattered p_{sc} pressure fields, and satisfies the Helmholtz equation

$$\nabla^2 p + k^2 p = q, \tag{1}$$

where $k = \omega/c$ is the wavenumber, c is the acoustic speed, ω is the frequency, and q represents sources. The incident field, p_{inc}, is a plane wave propagating in the positive x direction which interacts with a given configuration of scatterers. We consider these scatterers to be rigid cylinders of constant radii $a = 1$ m situated within the circular region of radius $R_{out} = 15$ m as shown in Fig. 1.

As a measure of scattering, more specifically, as a measure of how much energy is scattered by a configuration of scatterers, we use the total scattering cross section (TSCS).[86] We use the optical theorem[87] to formulate the TSCS in terms of the forward scattering amplitude:

$$\sigma = -2\mathrm{Re} f(0), \tag{2}$$

where $f(0) = f(\theta = 0)$ is the forward scattering amplitude, and the far-field amplitude form function, $f = f(\theta, \mathbf{r}_1, \ldots, \mathbf{r}_M)$, $\theta = \arg(\mathbf{x})$, is defined by the scattered pressure p_{sc} in the far-field:[7]

$$f(\theta) = \frac{2}{k} \sum_{m=1}^{M} e^{-\mathrm{i}k|\mathbf{r}_m| \cos(\theta - \arg(\mathbf{r}_m))} \sum_{n=-\infty}^{\infty} (-\mathrm{i})^n B_n^{(m)} e^{\mathrm{i}n\theta}. \tag{3}$$

Here, $\mathbf{r}_1, \ldots, \mathbf{r}_M$ are the position vectors, M is the total number of cylinders, and $B_n^{(m)}$ are the scattered pressure p_{sc} field coefficients.[7,84] For more details, see the Appendix of Ref. 83.

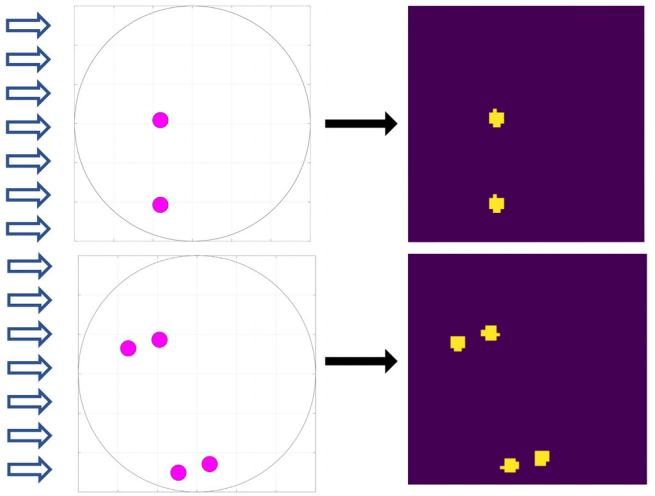


FIG. 1. (Color online) Physical scatterer configuration (left) and the corresponding scaled pixel images (right). The incident plane wave is propagating in the positive x direction indicated by blue arrows in the physical configuration image. The image is scaled to 64×64 pixel resolution size with scattering elements (yellow) and exterior fluid medium (dark magenta). Each scatterer in the physical space is altered from its cylindrical shape to fit within the discrete pixel space.

B. Data generation

To train both the CNN and cWGAN model, we generate datasets by means of the multiple scattering theory (MST)[7,85] and optical theorem Eq. (2). Specifically, we generate two datasets, one for 2-scatterer configurations and one for 4-scatterer configurations. The scatterer positions are selected randomly for each configuration and we evaluate the TSCS, σ, in MATLAB (MathWorks, Natick, MA) at discrete values of normalized wavenumber $k_i a \in [0.35, 0.45]$, $i = 1, \ldots, 11$ with an interval of 0.01 as shown in Figs. 2 and 3. We generate 60 000 samples for $M = 2$ and 4 configurations and reserve the evaluated TSCS as labels for training. This wavenumber range $ka \in [0.35, 0.45]$ was chosen for a technical convenience that takes lesser time to generate a dataset for training. At the same time, it has a physical reason: it is hard to optimize TSCS at low frequencies at which the state-of-the-art optimization algorithms struggle to find values close to the global minimum of TSCS which is zero.

Additionally, we generate a small dataset of optimal configurations with minimal TSCS. We use MATLAB's nonlinear programming solver, fmincon, with MuliStart (MathWorks, Natick, MA) to search for configurations with minimized TSCS while constrained to real physical properties. Several configurations of randomly placed scatterers, via MultiStart, are initialized. The MS solver evaluates the TSCS as well as the gradients for each configuration, and fmincon continues to perturb the scatterer positions toward minimized gradients until the optimal configurations are found. For 2-scatterer configurations and 105 data points, five unique optimal TSCS solutions are observed. However, given the larger variety of 4-scatterer configurations, the optimal dataset in not limited to only a few unique solutions.

https://doi.org/10.1121/10.0008929

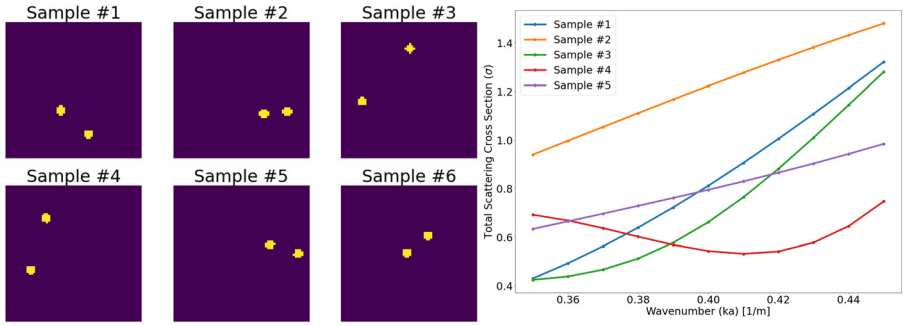


FIG. 2. (Color online) Samples of 2-scatterer configuration images selected from the $M = 2$ random configuration dataset and the corresponding TSCS.

In our initial deep learning model for acoustic cloaking, we started with fully connected neural networks (FCNN) architecture where the inputs to FCNN were the positions of each scatterer in a given planar configuration as a column vector 2M by 1 where M is the total number of scatterers. Although the dataset was large (60 000 samples), we noticed that FCNN were not able to learn and produce the average value of the TSCS in the dataset. The error during training was low but the FCNN model was performing poorly on unseen data. When we switched to binary images, both FCNN and CNN were able to approximate the TSCS. However, with an increase in the number of scatterers, the accuracy of FCNN predictions was lower in comparison to CNN results. The CNN architectures are better at generalization on unseen data as they can leverage spatial correlations in an image that would or else be lost in the flattening step for FCNN.[74]

C. Generative modeling

To solve the inverse design problem, a few components are necessary: a generative model capable of producing realistic designs, a method to enable the model to generate images with specified responses, and an evaluator that can confirm the validity of generated images. A popular generative model is the GAN.[44,45] The GAN use an adversarial process, in which two models, a generator G and critic C are trained simultaneously. GANs, however, are difficult to train; mode collapse and convergence failure are typical to the training process. As a foundation to our generative model, we chose the Wasserstein GAN[47,51] (WGAN) with penalized norm of gradient of C with respect to its input.

In the WGAN, generator G learns to generate new data with the same statistics while critic C scores the realness of an image based on the Wasserstein distance between the dataset distribution and the generated image distribution. The Wasserstein distance measures the distance between two probability densities and is informally called earth mover's distance. When the two probability distributions are described as piles of earth, or dirt, the earth mover's distance represents the minimum cost in turning one pile to the other, providing a quantifiable metric for describing two different probability densities. This Wasserstein distance is a continuous function, which provides a linear gradient at every step of the training process. The standard GAN relies on the zero sum-game between the discriminator and generator. However, once the discriminator is trained, it may fail to provide useful information in updating the generator. This results in a network that requires close attention to the model architecture and parameters to maintain a stable training process. The Wasserstein distance, however, provides smooth, linear gradients throughout the training process resulting in a more stable and robust model.

1. cWGAN model architecture

A method to enable image generation with specified inputs is seen in a WGAN variant: the conditional GAN,[46] here cWGAN. This model injects the condition, or expected response, early into the training process to allow the GAN to learn the correlation between designs and responses. We provide the TSCS as labels for each real scatterer configuration to the WGAN, and allow the WGAN to further generate configurations targeting the specified TSCS.

Our generative model, see Fig. 4, consists of the standard WGAN architecture with the aforementioned modifications to improve the capability of the standard model. By introducing both the conditional functionality[46,66] and the CoordConv[54] layer to the standard WGAN,[51] we enable the model to generate design images targeting specified TSCS

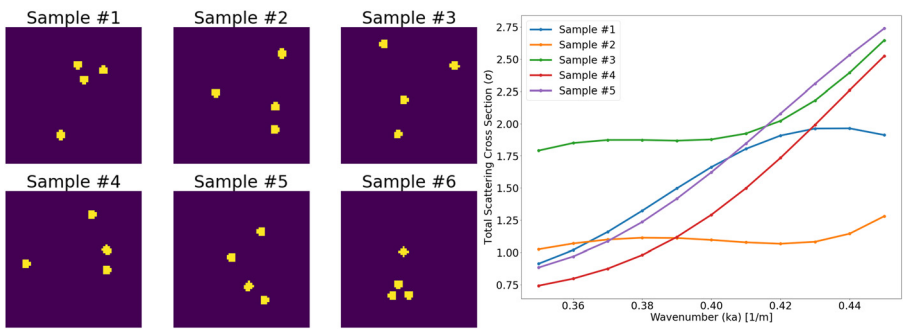


FIG. 3. (Color online) Four-scatterer configuration image samples selected from the $M = 4$ random configuration dataset and the corresponding TSCS.

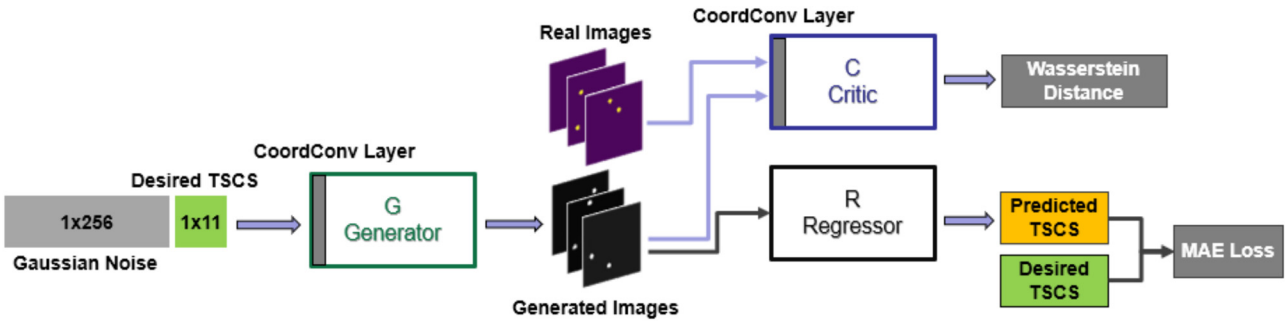


FIG. 4. (Color online) Schematic of cWGAN model. The cWGAN model comprises a CoordConv implemented generator and critic, as well as the pre-trained regressor. The regressor is trained on the same random configuration dataset as the cWGAN. A Gaussian distributed noise vector of size 256 is selected arbitrarily which we take as a multiple of 64, concatenated with the desired TSCS label, and is fed into the generator. The generator produces an image and passes it to both the critic and the regressor. The critic determines the Wasserstein distance between the real image and the generated image. The regressor predicts the TSCS value from the generated image and the MAE between predicted and desired TSCS is evaluated. Both the Wasserstein loss and the MAE loss are then used to update the generator's weights through backpropagation of the total loss function's derivatives with respect to the weights.

responses while improving the metacluster accuracy and image quality. Liu *et al.*[54] introduced the coordinate convolution layer (CoordConv) that performs a coordinate transformation between coordinates in Cartesian space and pixel space much faster with fewer parameters than convolution and improves CNN performance. The CoordConv layer is an extension of the standard convolution layer; additional hard-coded channels, with constant values for coordinate information, is concatenated to the standard convolution layer which allows it to better recognize positions of objects in images. They showed that using CoordConv in a GAN reduced mode collapse as the transform between high-level spatial latent variables and pixels was easier learned. We adapted the CoordConv[54] layer and included it in both G and C of cWGAN.

The generator G consists of an input layer, a CoordConv layer,[54] and multiple convolution layers. In the input layer, the generator G receives the latent noise vector of size 256, which is selected arbitrarily and taken as a multiple of 64. The latent vector, a realization of a (pseudo-)random vector $\mathbf{g} \in \mathbb{R}^1$, and the target TSCS is concatenated and passed to the generative model. The model generates an approximated realization of binary images of scatterer configurations where $\mathbf{g} \sim \mathcal{N}(0,1)$ which is chosen for convenience. The input layer, of 1×1 size, is tiled to a 64×64 size by replicating the input multiple times to match the target image resolution, and the channel is expanded by a size of 2 when processed by the CoordConv layer; this introduces Cartesian coordinate information to the model. The output is processed through a series of convolutions with decreasing channel size, resulting in an unchanged final image resolution size, but a condensed channel size. We use the rectified linear unit (ReLU) activation for all convolution layers.

The critic C model follows closely with the standard WGAN critic architecture. The input layer, which accepts 64×64 images proposed by the generator, is passed to the CoordConv layer to expand the channel size. This output is processed through five convolution layers with a Leaky ReLU activation of $\alpha = 0.2$ and a dropout of 0.25 after each convolution. As suggested,[47] we omit batch normalization in the critic C. The resulting output is flattened and passed into a single unit dense layer. Table I details the WGAN generator G architecture whereas Table II describes the WGAN critic C architecture. The cWGAN model is developed using Tensorflow (Mountain View, CA) and Keras (Mountain View, CA).

D. TSCS CNN regressor *R* model

In order to influence the generative model to produce designs with TSCS corresponding to the dataset distribution, we use a CNN model, the regressor R, to predict the TSCS from images of 2- or 4-scatterer configurations, see Table III. Including the regressor R provides an additional avenue to improve the model's selective design generation capability. While the critic determine the realness or fakeness of the images, the regressor R validates the incoming image's TSCS. When the generator G proposes a design to the critic C for evaluation, the same design is processed through the R regressor for TSCS prediction. The mean absolute error (MAE) between the predicted and expected TSCS is incorporated in the cWGAN loss function. The CNN model is developed using Tensorflow and Keras.

TABLE I. WGAN Generator G architecture. The input layer consists of the noise (1×256) and TSCS input (1×11), is tiled into a 64×64 output size where two additional channels are added through the CoordConv[54] procedure. The generator has 4 50 433 trainable parameters. Each convolution layer uses the ReLU activation function.

Layer	Size	Kernel	Padding	Output
Input	256	—	—	$1 \times 1 \times 267$
CoordConv	—	—	—	$64 \times 64 \times 269$
Conv2D	512	1×1	Valid	$64 \times 64 \times 512$
Conv2D	256	1×1	Valid	$64 \times 64 \times 256$
Conv2D	256	1×1	Valid	$64 \times 64 \times 256$
Conv2D	128	1×1	Same	$64 \times 64 \times 128$
Conv2D	64	1×1	Same	$64 \times 64 \times 64$
Conv2D	64	3×3	Same	$64 \times 64 \times 64$
Conv2D	64	3×3	Same	$64 \times 64 \times 64$
Conv2D	1	1×1	Same	$64 \times 64 \times 1$

https://doi.org/10.1121/10.0008929

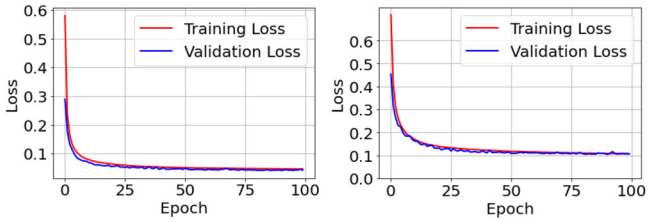
TABLE II. WGAN Critic C model architecture. CoordConv[54] is applied to the input layer to expand the channel size to include Cartesian coordinate information when examining the generated image. Subsequent convolution layers, a flattening layer, and a dense layer reduce the output size to 1 while incorporating dropouts prevent overfitting. All convolutional 2D (Conv2D) layers used a 3×3 kernel with stride of 2 and Leaky ReLU.

Layer	Size	Output
Input	(64×64)	$64 \times 64 \times 1$
CoordConv	—	$64 \times 64 \times 3$
Conv2D	64	$32 \times 32 \times 64$
ZeroPadding	—	$33 \times 33 \times 64$
Conv2D	128	$17 \times 17 \times 127$
Dropout	0.25	—
Conv2D	256	$9 \times 9 \times 256$
Dropout	0.25	—
Conv2D	512	$5 \times 5 \times 512$
Conv2D	1024	$3 \times 3 \times 1024$
Dropout	0.25	—
Flattening layer	—	9216
Dense	1	—

We train the regressor R with the random configuration dataset, for either 2-scatterer or 4-scatterer configurations, for 100 epochs using a batch size of 50. We use the RMSprop[88] optimizer with a small learning rate of 0.0001 and a mean absolute error (MAE) loss. To validate the training we use a tenfold cross-validation method. The dataset is first separated into 50 000 training samples and 10 000 test samples. Of the 50 000 training samples, we use a tenfold cross-validation method where the training set is further split into 10 different combinations of 45 000 train and 5000 validation samples. The regressor model is trained on each combination and the average mean absolute error is evaluated. The average MAE across the tenfold cross-validation training for 2-scatterer configurations is 0.0352 and the average standard deviation is 0.0280. For 4-scatterer configurations, the average MAE is 0.0861 and the average standard deviation is 0.0581. The average mean absolute percentage error (MAPE) is 5.56% for the 2-scatterer model and 7.28% for the 4-scatterer model; both models make σ predictions with $>90\%$ accuracy. Figure 5 shows a sample loss plot of onefold in the cross-validation training. The converging training and validation loss indicates that the model is not overfitting to the data.

TABLE III. The CNN regressor R architecture for TSCS prediction. The input layer consists of the 64×64 pixel image of 2-scatterer or 4-scatterer configurations. The output dense layer contains 11 units for TSCS evaluated at normalized wavenumber $k_i a \in [0.35, 0.45]$ with an interval of 0.01. Each convolution layer uses the ReLU activation function.

Layer	Units	Kernel	Stride	Padding
Conv2D	32	3×3	1×1	Same
Batch Norm	—	—	—	—
MaxPool2D	—	2×2	—	—
Conv2D	8	3×3	1×1	Same
Batch Norm	—	—	—	—
MaxPool2D	—	2×2	—	—
Dense	256	—	—	—
Dense	11	—	—	—



(a) 2-scatterer (b) 4-scatterer

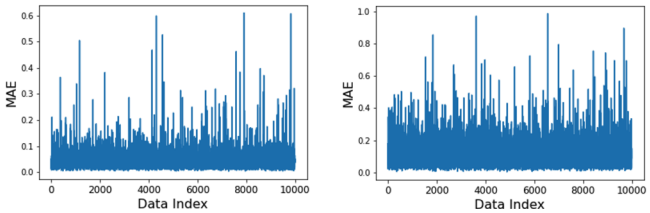
FIG. 5. (Color online) TSCS CNN regressor R training and validation loss for (a) 2-scatterer, and (b) 4-scatterer metacluster designs after 100 epochs. Both training and validation losses follow the same trend and converge to a similar value. Two-scatterer images converge to 0.045, and 4-scatterer images to 0.106.

After training the regressor R, the remaining 10 000 test samples are used to further confirm its performance on unseen data. Resulting MAE evaluated is shown in Fig. 6. The average MAE across the entire test set is 0.044 for 2-scatterer images and 0.107 for 4-scatterer images. Figure 7 illustrates the CNN TSCS prediction compared to the analytically evaluated TSCS by MST for randomly selected 2-scatterer and 4-scatterer test samples. The performance of R regressor model on the test set reflects the training performance, highlighting the model's ability to predict TSCS from unseen data. Additionally, we test the regressor on the optimal configuration dataset. We present images of σ minimized scatterer configurations to the regressor and allow the regressor to predict the σ. The 2-scatterer trained regressor predicted the σ with an average MAE of 0.0352 with standard deviation of 0.0179 while the 4-scatterer regressor resulted in an average MAE of 0.0897 with an average standard deviation of 0.0472. These results confirm that the 2-scatterer and 4-scatterer regressor trained on random configurations are robust and can predict σ from unseen configurations.

III. NUMERICAL RESULTS

A. cWGAN training procedure

To train the cWGAN, we use the dataset described in Sec. II B, consisting of 64×64 pixel images of random 2-scatterer and 4-scatterer configurations with corresponding TSCS σ evaluated at 11 discrete values of wavenumber $ka \in [0.35, 0.45]$. In the cWGAN model, a batch of random



(a) 2-scatterer (b) 4-scatterer

FIG. 6. (Color online) MAE for all test dataset samples for (a) 2-scatterer. and (b) 4-scatterer configurations. Average MAE across all 10 000 test points is 0.044 for 2-scatterer and 0.107 for 4-scatterer configurations.

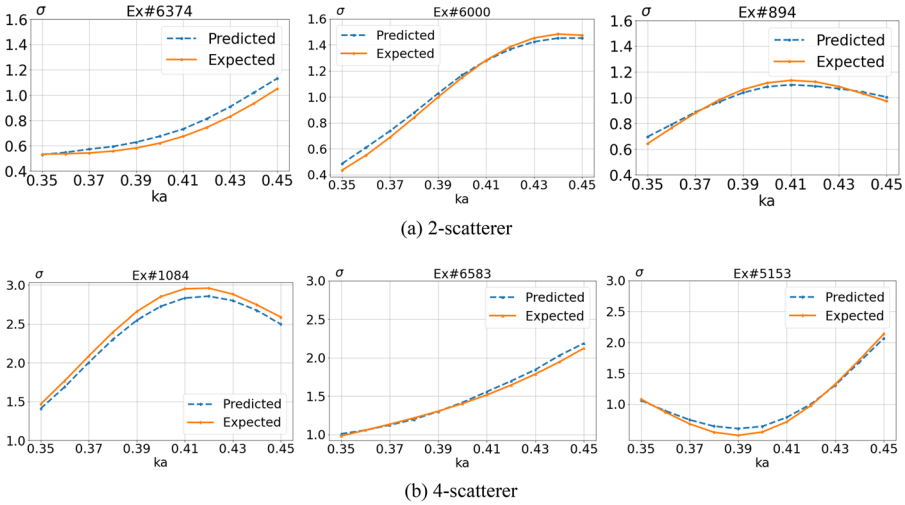


FIG. 7. (Color online) CNN TSCS prediction versus the expected σ with randomly selected (a) 2-scatterer, and (b) 4-scatterer test samples.

Gaussian distributed noise vectors of length 256 is first generated. To apply a condition to the training, a randomly selected σ response from the batch is appended to each noise vector. The generator produces an image, from the combined noise and σ vector, and proposes the image to the critic. The critic then calculates the Wasserstein distance between the generated image and its trained perception of a real image. Instead of weight clipping, as proposed in the original WGAN paper,[47] we use the gradient-penalty method,[51] to satisfy the Lipschitz constraint.

The loss functions are defined as follows. The generator loss function L_G is generally defined as

$$L_G = L_W + \alpha L_{reg}, \tag{4}$$

where L_W is the Wasserstein distance between the generated set and real dataset evaluated by the critic C, L_{reg} is the MAE loss supplied by the regressor R, α is the coefficient that determines the strength of the regressor loss in comparison to the Wasserstein loss. During our tests, we determined that $\alpha = 10$ achieved reasonable results. The critic loss is defined as

$$L_C = L_W + L_{GP}, \tag{5}$$

where L_{GP} is the loss supplied by the gradient penalty.

The expanded form for L_W is[51]

$$L_W = \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[C(\tilde{x})] - \mathbb{E}_{x\sim\mathbb{P}_r}[C(x)], \tag{6}$$

where $\mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[C(\tilde{x})]$ is the expected value of critic evaluated scores of generated image $\tilde{x}$ from the generated dataset $\mathbb{P}_g$ and $\mathbb{E}_{x\sim\mathbb{P}_r}[C(x)]$ is the expected score from the real images x in the dataset $\mathbb{P}_r$.

To implement the gradient penalty to the critic loss L_C, an interpolated image $\hat{x}$ between the real image x and the generated image $\tilde{x}$ is determined. The difference between the gradient norm of the critic score for interpolated images, $C(\hat{x})$, and the target norm 1 is used to penalize model. The gradient penalty loss is defined as

$$L_{GP} = \lambda\, \mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}(||\nabla_{\tilde{x}} C(\hat{x})||_2 - 1)^2, \tag{7}$$

where the coefficient λ is empirically selected as 10.

The regressor loss, L_{reg}, is the MAE loss between the evaluated TSCS from generated samples $\tilde{x}$ and real samples x:

$$L_{reg} = E_{\tilde{x}\sim P_g, \tilde{y}\sim P_r}||R(\tilde{x}) - \tilde{y}||, \tag{8}$$

where $R(\tilde{x})$ is the CNN TSCS prediction of generated sample $\tilde{x}$ and $\tilde{y}$ is the expected TSCS from the dataset.

Additionally, the generated images are passed to the auxiliary CNN regressor model developed in Sec. II D which predicts the TSCS σ values given the scatterer configuration. A mean squared error is calculated between the predicted and actual values of σ, defined by Eq. (8), and is combined with the calculated Wasserstein distance of the critic to form the full loss function for the generator by Eq. (4).

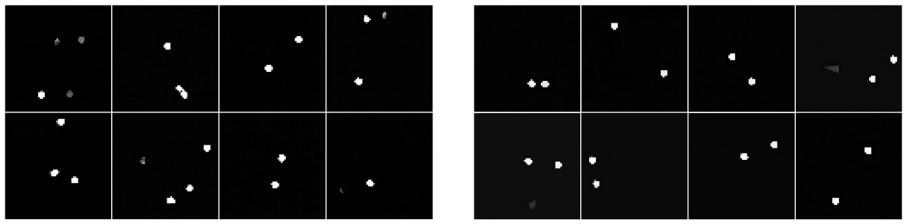
Following the standard WGAN process, a random batch is sampled and the critic is trained five times before the generator is trained once. The cWGAN is trained for 1 50 000 iterations using a batch size of 60 and a small learning rate of 5e-5.

B. cWGAN generated designs

After training the cWGAN, we check the model's performance by generating an image and verifying the σ response. We select 8 σ samples evaluated at 11 discrete values of ka from the test set and combine the σ vector with a randomly generated Gaussian distributed noise vector. This concatenated vector is passed to the generator as a design condition to generate an image. To reiterate the improvement by the addition of CoordConv layers in the cWGAN model, Fig. 8 shows generated images without CoordConv (a) and with CoordConv (b). The generated images with CoordConv implemented shows higher consistency than the generated images without CoordConv.

To evaluate the model's performance in design generation, we tasked the generator to repeatedly generate designs for each expected σ. Given the variety of configurations proposed, we exclude unfavorable results by limiting the generator to produce designs with a regressor predicted σ within a MAE of 0.05 for 2-scatterer images and 0.10 for 4-scatterer

https://doi.org/10.1121/10.0008929



(a) cWGAN without CoordConv (b) cWGAN with CoordConv

FIG. 8. Images generated by cWGAN models (a) without. and (b) with CoordConv (Ref. 54) layer. 8 TSCS samples are selected first within the test set as the conditions to generate 2-scatterer. Both models are trained for 150 000 iterations with the same hyperparameters, a low learning rate 5e-5 and batch size 60. The generated images with CoordConv implemented shows higher consistency.

images from the real σ value at ka = 0.35, 0.40, and 0.45. We use a PYTHON-developed function to record pixel coordinates of the scatterers and rescale these values to Cartesian coordinates. These Cartesian coordinates are passed into our multiple scattering solver on MATLAB to evaluate the σ, TSCS at different values of normalized wavenumber ka.

Figure 9 displays the generated (grayscale) and real (colored) images of 2-scatterer configurations. Similar spacing between scatterers is observed for both configurations which shows that the model was able to learn the dependency of TSCS on the position vector $\mathbf{r}_{jm}$, i.e., $\sigma = \sigma(\mathbf{r}_{jm}ka)$.[7] The position vector is defined as: $\mathbf{r}_{mj} = \mathbf{r}_m - \mathbf{r}_j \ (= -\mathbf{r}_{jm})$ where $m,j = 1,\ldots,M$ and M is the total number of cylinders. Figure 10 illustrates the dependency of TSCS σ on normalized wavenumber ka for the real and generated configurations of $M=2$ scatterers from samples of configurations of $M=2$ scatterers from Fig. 9. The red continuous lines correspond to TSCS produced by generated configurations and blue dashed lines correspond to TSCS by real configurations. Note that the cWGAN is trained on images with a wavenumber $ka \in [0.35, 0.45]$.

Following our image generation procedure, 100 randomly selected σ labels from the 2-scatterer configuration test set are passed, as the design condition, to the generator. After generating the images, we retain all generated configurations with a regressor predicted MAE loss less than 0.05.

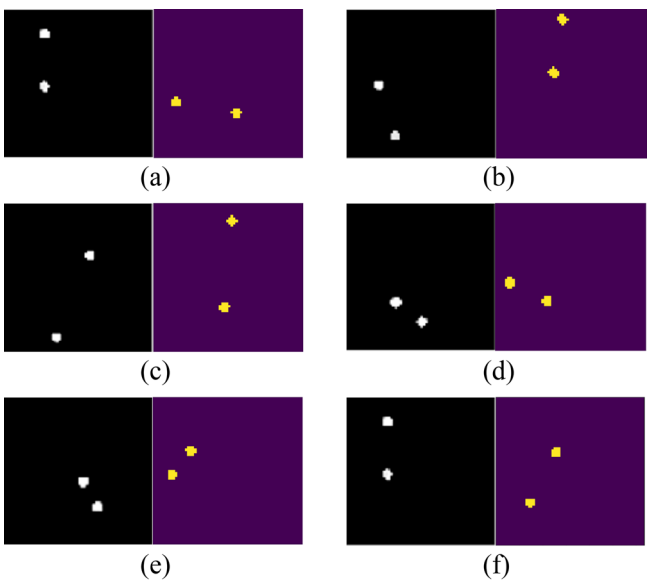


FIG. 9. (Color online) Generated (grayscale) and real (colored) images of 2-scatterer configurations. Similar spacing between scatterers is observed for the generated and real configurations.

The σ of the generated configurations are evaluated by the MS solver. The average MAE of all generated configurations for the 100 sample labels is 0.0771 with a standard deviation of 0.0685. The wide standard deviation indicates that the generator will still produce configurations with a larger, or smaller, error than expected. This error range stems from the regressor's inaccuracy in predicting σ.

The same model architecture and training procedure was used to train the cWGAN with the random configuration 4-scatterer dataset. We sample 100 σ labels from the

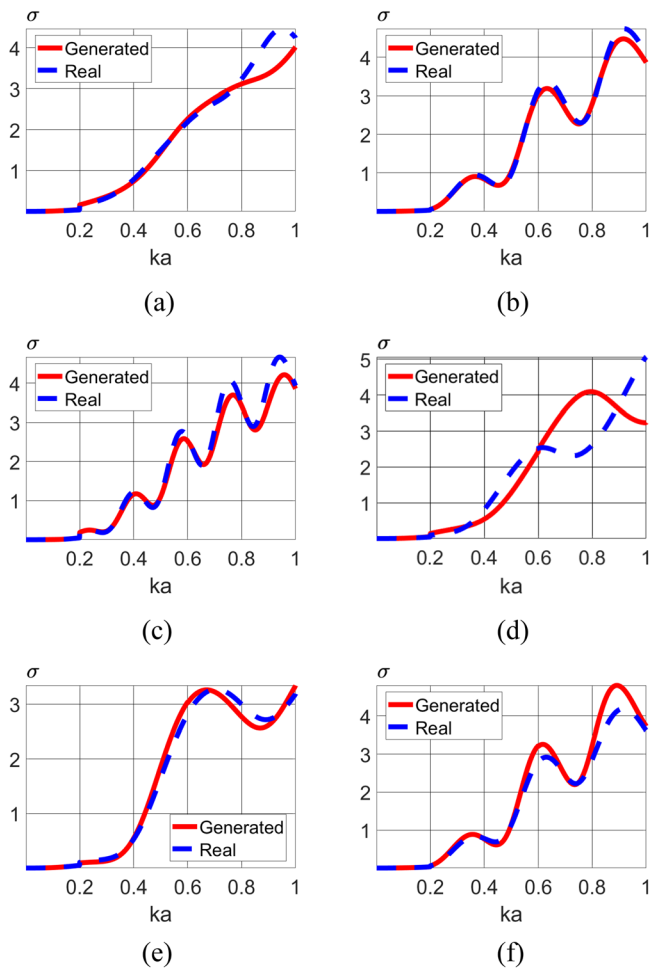


FIG. 10. (Color online) The variation of TSCS, σ with normalized wavenumber ka for the real and generated configurations from samples given in Fig. 9. The red continuous and blue dashed lines correspond to TSCS produced by real and generated configurations, respectively. Although the cWGAN is trained on images with $ka \in [0.35, 0.45]$, the σ evaluated from generated configurations compares closely to the real configuration.

test set and provide these labels to the generator to produce configuration images while retaining the images with a regressor predicted σ less than 0.10, which corresponds to the loss from the 4-scatterer regressor training. Evaluating all generated images results in a MAE of 0.1213 with a standard deviation of 0.0568. Figure 11 presents a comparison between generated (grayscale) and real (colored) images of $M = 4$ scatterer configurations. Figure 12 illustrates the variation of TSCS, σ with wavenumber ka for the configurations shown in Fig. 11, where the red continuous and blue dashed lines correspond to TSCS produced by generated and real configurations, respectively. Note that cWGAN is only trained for wavenumbers $k_i a \in [0.35, 0.45]$ $(i = \overline{1,11})$.

C. Optimal configuration generation with cWGAN

When a generative network is properly trained, it can produce optimal performance beyond the parameter given in the training dataset. Our cWGAN model can generate optimal configurations that produce near zero TSCS values. The cWGAN model, previously trained on random configurations of 2-scatterer or 4-scatterer configurations, is tested with the optimal configuration dataset. The fmincon minimized σ labels are concatenated to a noise vector and passed to the generator. The generator repeatedly generates scatterer configurations and images with a regressor predicted σ MAE of 0.07 and 0.15, for 2-scatterer and 4-scatterer configurations respectively, are saved. Six samples of cWGAN generated images are shown in Fig. 13 and the resulting TSCS comparison between real and generated images are shown in Fig. 14.

Out of the seven unique minimized σ solutions for 2-scatterer configurations, the cWGAN was able to produce designs for six of them. After generating multiple images for these six solutions, the resulting MAE between the expected

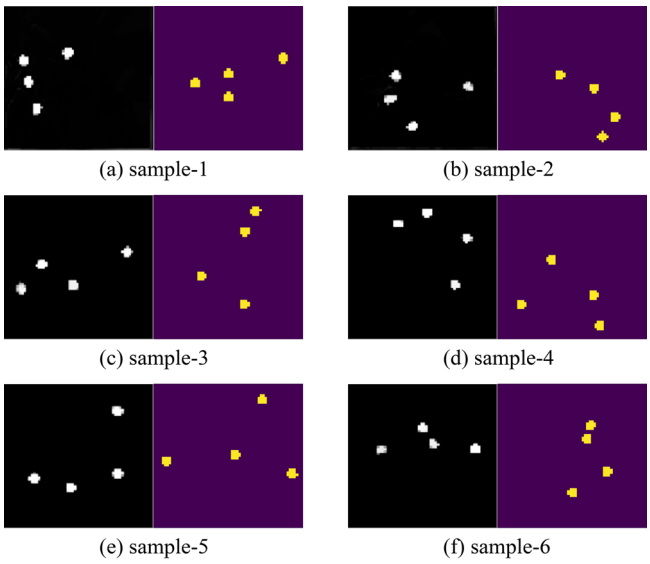


FIG. 11. (Color online) Generated (grayscale) and real (colored) images of 4-scatterer configurations. Samples are generated through the standard image generation procedure and are filtered so that configurations with better predicted σ fit are selected.

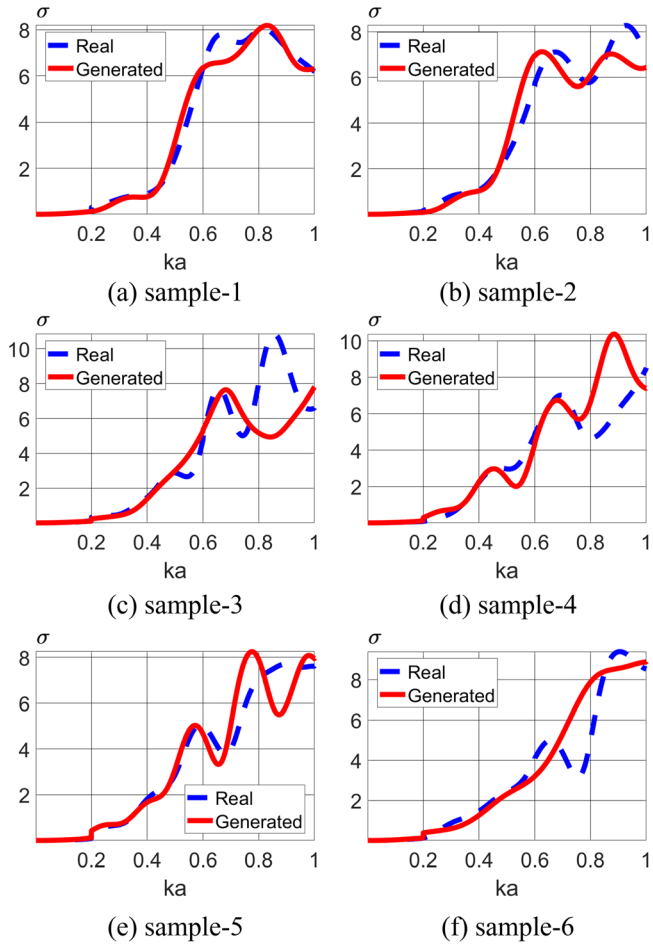


FIG. 12. (Color online) TSCS versus ka for $M = 4$ cylinder configuration from samples depicted in Fig. 11. Note that cWGAN is only trained for wavenumbers $ka \in [0.35, 0.45]$. The red continuous and blue dashed lines correspond to TSCS produced by generated and real configurations, respectively.

σ and the generated images σ is 0.1822, 0.0921, 0.1078, 0.0696, 0.0613, and 0.2467. However, further examination of the error shows a mean error of -0.0401, –0.0386, –0.0276, 0.0279, 0, and -0.0252. The negative error for generated optimal configurations shows that the configurations have, on average, lower evaluated σ than the expected value.

Given the larger variation of 4-scatterer configurations, there are many different solutions within the 105 sample optimal configuration dataset. We follow the same procedure to generate 4-scatterer configurations. The optimal configuration σ labels are combined with noise to generate new configurations. After comparing with the expected TSCS, the 4-scatterer configuration generator results in an average MAE of 0.1561. Generated 4-scatterer images and the corresponding TSCS comparison graphs are shown in Figs. 15 and 16, respectively.

IV. CONCLUSIONS

A method using generative modeling instead of optimization algorithms to determine the optimal planar configuration of rigid cylinders producing near zero TSCS is

https://doi.org/10.1121/10.0008929

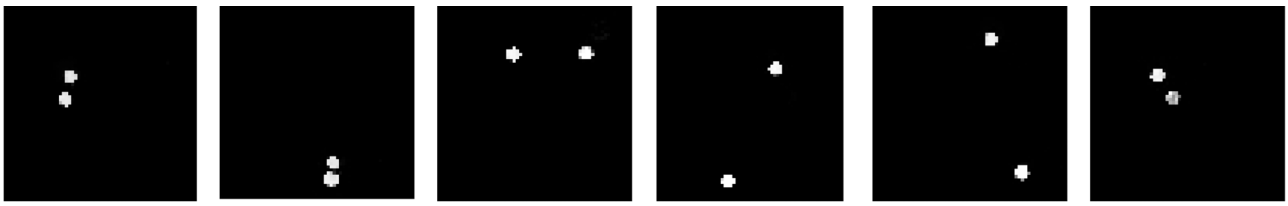


FIG. 13. WGAN generated images for optimized 2-scatterer configurations. Resulting TSCS comparison is shown in Fig. 14.

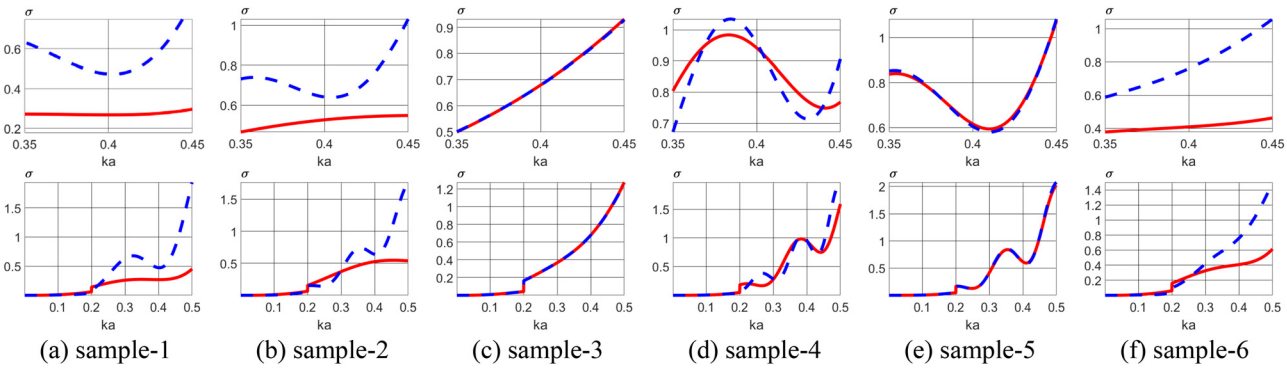


FIG. 14. (Color online) Variation of TSCS, σ, with wavenumber $ka \in [0.35, 0.45]$ (top row) and $ka \in [0, 0.50]$ (bottom row) for optimal configurations of $M = 2$ scatterers from the generated configurations (solid red line) in Fig. 13 compared to the expected response (dashed blue line). Most samples show a strong correlation between the evaluated σ from generated configurations and real configurations. The MAE for all generated images of samples 1–6 in the figure across the specified wavenumber range is 0.0266 and the average error across the samples is −0.0173. The negative average error shows that the majority of generated images have σ that is lower than the expected σ.

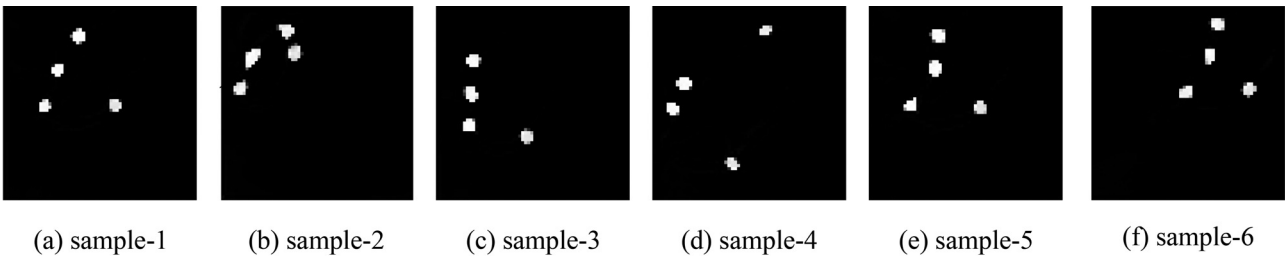


FIG. 15. Sample generated images of 4-scatterer configurations. The corresponding evaluated TSCS is shown in Fig. 16.

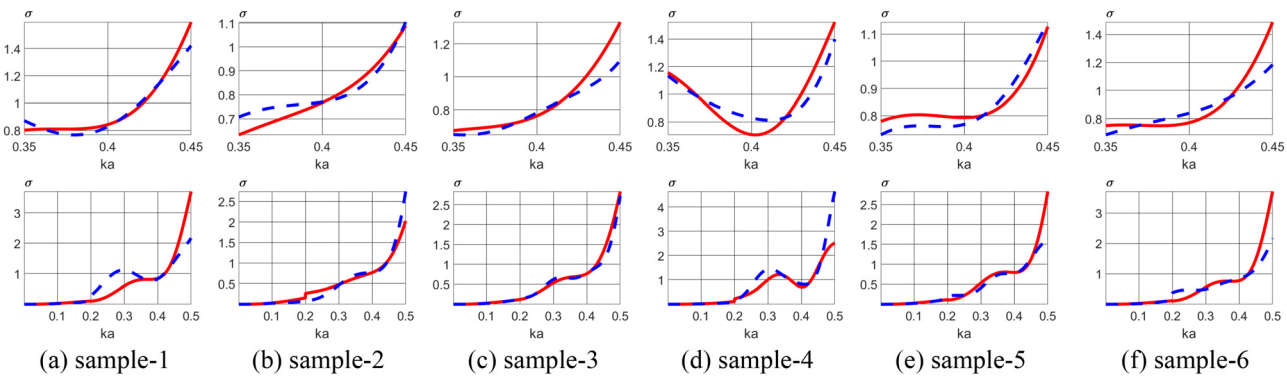


FIG. 16. (Color online) Variation of TSCS for 4-scatterer generated optimal configurations as shown in 16. Although the average MAE for all tested samples is 0.1561, we examined some of the best performing generated configurations. The average MAE for the six samples shown is 0.0494 with a standard deviation of 0.0173.

introduced. By implementing the conditional functionality to the standard WGAN[51] architecture, the model is able to learn the relation between scatterer configurations and the corresponding TSCS. Additionally, applying the CoordConv layer to the generator and critic noticeably improved the quality and accuracy of generated designs. We use the random configuration dataset to train the cWGAN and allow the model to produce TSCS optimized designs. Resulting images and evaluated TSCS show the model's capability to produce comparable metacluster designs in regards to minimal TSCS.

Our generative network model was implemented based on datasets generated by means of optical theorem which works at normal incidence. The generative model can be extended to minimize MS at an oblique incidence or reduce MS by different nonuniform configurations of scatterers of various shapes, radii, and/or material properties by developing new datasets for such cases and by using transfer learning[45] to speed up training and improve the performance of proposed generative model.

This work validates the inclusion of DL applications in acoustic metamaterial design. However, the limitations of this current model lie with the regressor. Given the conditional design, the regressor is necessary to influence the generator to produce configurations with a specified TSCS. With the current regressor, we see a difference in performance as indicated in the rising MAE loss between 2-scatterer and 4-scatterer models, 0.05 and 0.10, respectively. Incidentally, the regressor overfits on 10-scatterer images. As a bottleneck, the removal of the regressor for a more accurate model is imperative to proceeding with more complex structures.

Further work may improve on the current generative process by joining the cWGAN with an analytical model that can evaluate the TSCS while providing the MAE loss to the generator. By including an analytical approach, rather than a DL model, two important improvements can be seen: the error inherent to CNN training is removed and the number of tunable parameters in the full generative model is reduced resulting in a model that is more robust and easier to train. Further work will improve the capability of this model to produce configurations with a much larger number of scatterers to be compatible with more complex acoustic applications such as cloaking and wave steering. To increase the number of material properties stored in each scatterer visually, tunable color channels may be introduced where different hues of color will correspond to a normalized material property value.

ACKNOWLEDGMENTS

P.L. and F.A. acknowledge Small Grant Project grant support from San Jose State University.

[1]M. Kadic, T. Bückmann, R. Schittny, and M. Wegener, "Metamaterials beyond electromagnetism," Rep. Prog. Phys. **76**(12), 126501 (2013).
[2]G. F. Méjica and A. D. Lantada, "Comparative study of potential pentamodal metamaterials inspired by Bravais lattices," Smart Mater. Struct. **22**(11), 115013 (2013).
[3]X. Su and A. N. Norris, "Focusing, refraction, and asymmetric transmission of elastic waves in solid metamaterials with aligned parallel gaps," J. Acoust. Soc. Am. **139**(6), 3386–3394 (2016).
[4]A. Titovich and A. Norris, "Acoustic poisson-like effect in periodic structures," J. Acoust. Soc. Am. **139**(6), 3353–3356 (2016).
[5]P. Packo, A. N. Norris, and D. Torrent, "Metaclusters for the full control of mechanical waves," Phys. Rev. Appl. **15**(1), 014051 (2021).
[6]A. N. Norris, "Acoustic cloaking theory," Proc. R. Soc. A **464**, 2411–2434 (2008).
[7]F. A. Amirkulova and A. N. Norris, "The gradient of total multiple scattering cross-section and its application to acoustic cloaking," J. Theor. Comput. Acoust. **28**, 1950016 (2020).
[8]A. Titovich, A. Norris, and M. Haberman, "A high transmission broadband gradient index lens using elastic shell acoustic metamaterial elements," J. Acoust. Soc. Am. **139**, 3357–3364 (2016).
[9]L. Fahey, F. Amirkulova, and A. Norris, "Broadband acoustic metamaterial design using gradient-based optimization," J. Acoust. Soc. Am. **146**(4), 2830 (2019).
[10]E. R. Davies, *Computer Vision: Principles, Algorithms, Applications, Learning* (Academic Press Inc., Cambridge, MA, 2017).
[11]S. Holden, *Computer Vision: Advanced Techniques and Applications* (Clanrye International, New York, 2019).
[12]M. Elgendy, *Deep Learning for Vision Systems* (Manning Publications, Shelter Island, NY, 2020).
[13]W. Andreopoulos, A. Geller, M. Lucke, J. Balewski, A. Clum, N. Ivanova, and A. Levy, "Deeplasmid: Deep learning accurately separates plasmids from bacterial chromosomes," Cold Spring Harb. Laborator. (2021). https://www.biorxiv.org/
[14]A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, and L. Kaiser, "Attention is all you need," arXiv:1706.03762 (2017).
[15]J. Peurifoy, Y. Shen, L. Jing, Y. Yang, F. Cano-Renteria, B. G. DeLacy, J. Joannopoulos, M. Tegmark, and M. Soljačić, "Nanophotonic particle simulation and inverse design using artificial neural networks," Sci. Adv. **4**(6), r4206 (2018).
[16]Z. Liu, D. Zhu, S. Rodrigues, W. Lee, and K. Cai, "Training deep neural networks for the inverse design of nanophotonic structures," ACS Photonics **5**, 1365–1369 (2018).
[17]T. Zhang, J. Wang, Q. Liu, J. Zhou, J. Dai, X. Han, Y. Zhou, and K. Xu, "Efficient spectrum prediction and inverse design for plasmonic waveguide systems based on artificial neural networks," Photonics Res. **7**(3), 368–380 (2019).
[18]J. Jiang and J. Fan, "Global optimization of dielectric metasurfaces using a physics-driven neural network," Nano Lett. **19**(8), 5366–5372 (2019).
[19]S. Inampudi and H. Mosallaei, "Neural network based design of metagratings," Appl. Phys. Lett **112**(24), 241102 (2018).
[20]C. Cueto and B. A. Hadithi, "Cancelling out skull-induced aberrations: Analysis of acoustic metamaterials using neural networks," IEEE Lat. Am. Trans. **15**(10), 1948–1959 (2017).
[21]G. Oliveri and J. Overvelde, "Inverse design of mechanical metamaterials that undergo buckling," Adv. Funct. Mater. **30**(12), 1909033 (2020).
[22]H. Gao and J. Zhu, "Inverse design method for acoustic metamaterials," J. Acoust. Soc. Am. **146**(4), 2828 (2019).
[23]C. Gurbuz, F. Kronowetter, C. Dietz, M. Eser, J. Schmid, and S. Marburg, "Generative adversarial networks for the design of acoustic metamaterials," J. Acoust. Soc. Am. **149**(2), 1162–1174 (2021).
[24]P. Meng, L. Su, W. Yin, and S. Zhang, "Solving a kind of inverse scattering problem of acoustic waves based on linear sampling method and neural network," Alex. Eng. J. **59**(3), 1451–1462 (2020).
[25]R. Khodayi-mehr and M. M. Zavlanos, "Deep learning for robotic mass transport cloaking," IEEE Trans. Robot **36**(3), 967–974 (2020).
[26]S. Saha, Z. Gan, L. Cheng, J. Gao, O. Kafka, X. Xie, H. Li, M. Tajdari, H. Kim, and W. Liu, "Hierarchical deep learning neural network (HiDeNN): An artificial intelligence (AI) framework for computational science and engineering," Comput. Methods Appl. Mech. Eng. **373**, 113452 (2021).
[27]Y. Liu, H. Niu, and Z. Li, "A multi-task learning convolutional neural network for source localization in deep ocean," J. Acoust. Soc. Am. **148**(2), 873–883 (2020).
[28]Y. Liu, H. Niu, Z. Li, and M. Wang, "Deep-learning source localization using autocorrelation functions from a single hydrophone in deep ocean," JASA Express Lett. **1**(3), 036002 (2021).
[29]H. Purwins, B. Li, J. Schluter, S.-Y. Chang, and T. Sainath, "Deep learning for audio signal processing," IEEE J. Sel. Top. Sign. Process **13**(2), 206–219 (2019).

[30]J. Hershey,Z. Chen, J. L. Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," Proc. - IEEE Int. Conf. Acoust. Speech Signal Process. 16021700 (2016).
[31]R. Barrett, M. Chakraborty, D. Amirkulova, H. Gandhi, and A. White, "A GPU-accelerated machine learning framework for molecular simulation: Hoomd-blue with TensorFlow," ChemRxiv (2019).
[32]K. Butler, D. Davies, H. Cartwright, O. Isayev, and A. Walsh, "Machine learning for molecular and materials science," Nature **559**(7715), 547–555 (2018).
[33]D. Elton, Z. Boukouvalas, M. Fuge, and P. Chung, "Deep learning for molecular design—a review of the state of the art," Mol. Syst. Des. Eng. **4**(4), 828–849 (2019).
[34]J. Noh, G. Gu, S. Kim, and Y. Jung, "Machine-enabled inverse design of inorganic solid materials: Promises and challenges," Chem. Sci. **11**(19), 4871–4881 (2020).
[35]S. So, T. Badloe, J. Noh, J. Rho, and J. Bravo-Abad, "Deep learning enabled inverse design in nanophotonics," Nanophotonics **9**(5), 1041–1057 (2020).
[36]S. Campbell, D. Sell, R. Jenkins, E. Whiting, J. Fan, and D. Werner, "Review of numerical optimization techniques for meta-device design," Opt. Mater. Express **9**(4), 1842–1863 (2019).
[37]W. Ahmed, M. Farhat, X. Zhang, and Y. Wu, "Deterministic and probabilistic deep learning models for inverse design of broadband acoustic cloak," Phys. Rev. Res. **3**(1), 013142 (2021).
[38]S. Kumar, S. Tan, L. Zheng, and D. Kochmann, "Inverse-designed spinodoid metamaterials," Npj Comp. Mater. **6**(1), 368–380 (2020).
[39]Z. Fan, V. Vineet, H. Gamper, and N. Raghuvanshi, "Fast acoustic scattering using convolutional neural networks," in *Icassp 2020* (IEEE, 2020).
[40]Z. Fan, V. Vineet, C. Lu, T. Wu, and K. McMullen, "Prediction of object geometry from acoustic scattering using convolutional neural networks," arXiv:2010.10691 (2021).
[41]M. J. Bianco, P. Gerstoft, J. Traer, E. Ozanich, M. Roch, S. Gannot, and C. Deledalle, "Machine learning in acoustics: Theory and applications," J. Acoust. Soc. Am. **146**(5), 3590–3628 (2019).
[42]B. Sanchez-Lengeling and A. Aspuru-Guzik, "Inverse molecular design using machine learning: Generative models for matter engineering," Science **361**(6400), 360–365 (2018).
[43]J. Jiang, D. Sell, S. Hoyer, J. Hickey, J. Yang, and J. Fan, "Data-driven metasurface discovery," arXiv:1811.12436 (2018).
[44]I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," Adv. Neural Inf. Process. Syst. 27 (2014).
[45]I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016), http://www.deeplearningbook.org (Last viewed December 5, 2021).
[46]M. Mirza and S. Osindero, "Conditional generative adversarial nets," arXiv:1411.1784v1 (2014).
[47]M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceed. of 34th ICML* (2017).
[48]A. Ghosh, V. Kulharia, V. Namboodiri, P. H. Torr, and P. K. Dokania, "Multi-agent diverse generative adversarial networks," in *2018 Ieee/Cvf* (IEEE, 2018).
[49]Y. Boget, "Adversarial regression. Generative adversarial networks for non-linear regression: Theory and assessment," Master's thesis, University of Neuchâtel, 2019.
[50]M. Kingma and D. Welling, "Auto-encoding variational bayes," arXiv:1312.6114v10 (2014).
[51]I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of Wasserstein GANs," arXiv:1704.00028 (2017).
[52]A. Gretton, K. Borgwardt, B. Rasch, M. Scholkopf, and A. Smola, "A kernel method for the two-sample-problem," Adv. Neural Inf. Process. Syst. 19 (2006).
[53]D. Lee, J. Kim, W.-J. Moon, and J. C. Ye, "CollaGAN: Collaborative GAN for missing image data imputation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* (2019), pp. 2487–2496.
[54]R. Liu, J. Lehman, P. Molino, F. P. Such, E. Frank, A. Sergeev, and J. Yosinski, "An intriguing failing of convolutional neural networks and the CoordConv solution," arXiv:1807.03247v2 (2018).
[55]P. Gerstoft, H. Groll, and C. F. Mecklenbräuker, "Parametric bootstrapping of array data with a generative adversarial network," in *2020 IEEE 11th Sensor Array and Multichannel Signal Processing Workshop (SAM)* (2020), pp. 1–5.
[56]E. Putin, A. Asadulaev, Y. Ivanenkov, V. Aladinskiy, B. Sanchez-Lengeling, A. Aspuru-Guzik, and A. Zhavoronkov, "Reinforced adversarial neural computer for de novo molecular design," J. Chem. Inf. Model **58**(6), 1194–1204 (2018).
[57]Y. Dan, Y. Zhao, X. Li, S. Li, M. Hu, and J. Hu, "Generative adversarial networks (GAN) based efficient sampling of chemical composition space for inverse design of inorganic materials," Npj Comput. Mater. **6**(1), 84 (2020).
[58]J. A. Fan, "Freeform metasurface design based on topology optimization," MRS Bull. **45**(3), 196–201 (2020).
[59]J. Jiang and J. A. Fan, "Simulator-based training of generative neural networks for the inverse design of metasurfaces," Nanophotonics **9**(5), 1059–1069 (2019).
[60]J. Jiang, D. Sell, S. Hoyer, J. Hickey, J. Yang, and J. A. Fan, "Free-form diffractive metagrating design based on generative adversarial networks," ACS Nano **13**(8), 8872–8878 (2019).
[61]F. Wen, J. Jiang, and J. Fan, "Robust freeform metasurface design based on progressively growing generative networks," ACS Photon. **7**, 2098 (2020).
[62]A.-P. Blanchard-Dionne and O. J. F. Martin, "Successive training of a generative adversarial network for the design of an optical cloak," OSA Contin. **4**(1), 87 (2021).
[63]X. Han, Z. Fan, Z. Liu, C. Li, and L. J. Guo, "Inverse design of metasurface optical filters using deep neural network with high degrees of freedom," InfoMat **3**, 432 (2021).
[64]Y. Tang, K. Kojima, T. Koike-Akino, Y. Wang, P. Wu, M. Tahersima, D. Jha, K. Parsons, and M. Qi, "Generative deep learning model for a multi-level nano-optic broadband power splitter," in *Ofc 2020* (OSA, 2020).
[65]R. K. Tan, N. L. Zhang, and W. Ye, "A deep learning–based method for the design of microstructural materials," Struct. Multidiscipl. Optim. **61**(4), 1417–1438 (2020).
[66]X. Han, L. Zhang, K. Zhou, and X. Wang, "Deep learning framework DNN with conditional WGAN for protein solubility prediction," arXiv:1811.07140v1 (2018).
[67]Z. Liu, D. Zhu, S. Rodrigues, K. Lee, and W. Cai, "Generative model for the inverse design of metasurfaces," Nano Lett. **18**, 6570–6576 (2018).
[68]W. Ma, F. Cheng, Y. Xu, Q. Wen, and Y. Liu, "Probabilistic representation and inverse design of metamaterials based on a deep generative model with semi-supervised learning strategy," Adv. Mater. **31**(35), 1901111 (2019).
[69]M. H. Tahersima, K. Kojima, T. Koike-Akino, D. Jha, B. Wang, C. Lin, and K. Parsons, "Deep neural network inverse design of integrated photonic power splitters," Sci. Rep. **9**(1), 1368 (2019).
[70]A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," arXiv:1511.06434v2 (2015).
[71]S. An, B. Zhen, H. Tang, M. Shalaginov, L. Zhou, H. Li, M. Kang, K. Richardson, T. Gu, J. Hu, C. Fowler, and H. Zhang, "Multifunctional metasurface design with a generative adversarial network," Adv. Opt. Mater **9**(5), 2001433 (2021).
[72]Z. Liu, L. Raju, D. Zhu, and W. Cai, "A hybrid strategy for the discovery and design of photonic nanostructures," arXiv:1902.02293 (2019).
[73]T. Tran, F. Amirkulova, and E. Khatami, "Acoustic cloak design via machine learning," https://arxiv.org/abs/2111.01230 (2021).
[74]A. Larsen, S. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," PMLR **48**, 1558–1566 (2016), see arxiv: https://arxiv.org/abs/1512.09300.
[75]T. Blaschke, M. Olivecrona, O. Engkvist, J. Bajorath, and H. Chen, "Application of generative autoencoder in de novo molecular design," Mol. Inform **37**(1–2), 1700123 (2018).
[76]A. Kadurin, S. Nikolenko, K. Khrabrov, A. Aliper, and A. Zhavoronkov, "druGAN: An advanced generative adversarial autoencoder model for de novo generation of new molecules with desired molecular properties in silico," Mol. Pharm **14**(9), 3098–3104 (2017).
[77]A. Brock, T. Lim, J. Ritchie, and N. Weston, "Neural photo editing with introspective adversarial networks," arXiv:1609.07093 (2017).
[78]Z. Zhang, Y. Song, and H. Qi, "Age progression/regression by conditional adversarial autoencoder," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. 5810–5818 (2017).
[79]A. Razavi, A. V. D. Oord, and O. Vinyals, "Generating diverse high-fidelity images with vq-vae-2," in *33rd Conference NeurIPS 2019*, Vancouver, Canada (2019).
[80]A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton, "VEEGAN: Reducing mode collapse in gans using implicit variational learning," arXiv:1705.07761 (2017).

[81]J. Bao, D. Chen, F. Wen, H. Li, and G. Hua, "CVAE-GAN: Fine-grained image generation through asymmetric training," arXiv:1703.10155 (2017).
[82]X. Hou, K. Sun, L. Shen, and G. Qiu, "Improving variational autoencoder with deep feature consistent and generative adversarial training," Neurocomputing **341**, 183–194 (2019).
[83]T. Shah, L. Zhou, P. Lai, A. D. L. Rosa-Moreno, F. Amirkulova, and P. Gerstoft, "Reinforcement learning applied to metamaterial design," J. Acoust. Soc. Am. **150**(1), 321–338 (2021).
[84]F. A. Amirkulova, "Acoustic and elastic multiple scattering and radiation from cylindrical structures," Ph.D. thesis, Rutgers University, Piscataway, NJ, 2014.
[85]P. Martin, *Multiple Scattering: Interaction of Time-Harmonic Waves with N Obstacles* (Cambridge University Press, New York, 2006).
[86]P. Martin, "Multiple scattering and scattering cross sections," J. Acoust. Soc. Am. **143**, 995–1002 (2018).
[87]A. Norris, "Acoustic integrated extinction," Proc. R. Soc. A **471**, 20150008 (2015).
[88]S. De, A. Mukherjee, and E. Ullah, "Convergence guarantees for RMSProp and ADAM in non-convex optimization and an empirical comparison to Nesterov acceleration," arXiv:1807.06766v3 [cs.LG] (2018).
[89]J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "Review on generative adversarial networks: Algorithms, theory, and applications," J. Latex Class Files (published online) (2021).