Research article

Jiaqi Jiang and Jonathan A. Fan*

Simulator-based training of generative neural networks for the inverse design of metasurfaces

Abstract: Metasurfaces are subwavelength-structured artificial media that can shape and localize electromagnetic waves in unique ways. The inverse design of these devices is a non-convex optimization problem in a high dimensional space, making global optimization a major challenge. We present a new type of population-based global optimization algorithm for metasurfaces that is enabled by the training of a generative neural network. The loss function used for backpropagation depends on the generated pattern layouts, their efficiencies, and efficiency gradients, which are calculated by the adjoint variables method using forward and adjoint electromagnetic simulations. We observe that the distribution of devices generated by the network continuously shifts towards high performance design space regions over the course of optimization. Upon training completion, the best generated devices have efficiencies comparable to or exceeding the best devices designed using standard topology optimization. Our proposed global optimization algorithm can generally apply to other gradient-based optimization problems in optics, mechanics, and electronics.

Keywords: simulator-based training; generative networks; neural networks; adjoint variable method; global optimization.

1 Introduction

Photonic technologies serve to manipulate, guide, and filter electromagnetic waves propagating in free space and in waveguides. Due to the strong dependence of electromagnetic function on geometry, much emphasis in the field has been placed on identifying geometric designs for these devices given a desired optical response. The vast majority of existing design concepts utilize relatively simple shapes that can be described using physical intuition. For example, silicon photonic devices typically utilize adiabatic tapers and ring resonators to route and filter guided waves [1], and metasurfaces, which are diffractive optical components used for wavefront engineering, typically utilize arrays of nanowaveguides or nanoresonators comprising simple shapes [2]. While these design concepts work well for certain applications, they possess limitations, such as narrow bandwidths and sensitivity to temperature, which prevent the further advancement of these technologies.

To overcome these limitations, design methodologies based on optimization have been proposed. Among the most successful of these concepts is gradient-based topology optimization, which uses the adjoint variables method to iteratively adjust the dielectric composition of the devices and improve device functionality [3–8]. This design method, based on gradient descent, has enabled the realization of high performance, robust [9] devices with nonintuitive layouts, including new classes of on-chip photonic devices with ultrasmall footprints [10, 11], non-linear photonic switches [12], and diffractive optical components that can deflect [13–17] and focus [18, 19] electromagnetic waves with high efficiencies. While gradient-based topology optimization has great potential, it is a local optimizer and depends strongly on the initial distribution of dielectric material making up the devices [20]. The identification of a high performance device is therefore computationally expensive, as it requires the optimization of multiple random initial dielectric distributions and selecting the best device.

We present a detailed mathematical discussion of a new global optimization concept based on Global Topology Optimization Networks (GLOnets) [21], which combine adjoint variables electromagnetic calculations with the training of a generative neural network to realize high performance photonic structures. Unlike gradient-based

*Corresponding author: Jonathan A. Fan, Department of Electrical Engineering, Stanford University, 348 Via Pueblo, Stanford, CA 94305, USA, e-mail: jonfan@stanford.edu.
Jiaqi Jiang: Department of Electrical Engineering, Stanford University, 348 Via Pueblo, Stanford, CA 94305, USA.
https://orcid.org/0000-0001-7502-0872

topology optimization, which optimizes one device at a time, our approach is population-based and optimizes a distribution of devices, thereby enabling a global search of the design space. As a model system, we will apply our concept to design periodic metasurfaces, or metagratings, which selectively deflect a normally incident beam to the +1 diffraction order. In our previous work [21], we demonstrated that GLOnets conditioned on incident wavelength and deflection angle can generate ensembles of high efficiency metagratings. In this manuscript, we examine the underlying mathematical theory behind GLOnets, specifically a derivation of the objective and loss functions, discussion of the training process, interpretation of hyperparameters, and calculations of baseline performance metrics for unconditional GLOnets. We emphasize that our proposed concepts are general and apply broadly to design problems in photonics and other fields in the physical sciences in which the adjoint variables method applies.

2 Related machine learning work

In recent years, deep learning has been investigated as a tool to facilitate the inverse design of photonic devices. Many efforts have focused on using deep neural networks to learn the relationship between device geometry and optical response [22, 23], leading to trained networks serving as surrogate models mimicking electromagnetic solvers. These networks have been used together with classical optimization methods, such as simulated annealing or particle swarm algorithms, to optimize a device [24, 25]. Device geometries have also been directly optimized from a trained network by using gradients from backpropagation [26–29]. These methods work well on simple device geometries described by a few parameters. However, the model accuracy decreases as the geometric degrees of

freedom increase, making the scaling of these ideas to the inverse design of complex systems unfeasible.

An alternative approach is to utilize generative adversarial networks (GANs) [30], which have been proposed as a tool for freeform device optimization [31–33]. GANs have been of great interest in recent years and have a broad range of applications, including image generation [34, 35], image synthesis [36], image translation [37], and super resolution imaging [38]. In the context of photonics inverse design, GANs are provided images of high performance devices, and after training, they can generate high performance device patterns with geometric features mimicking the training set [32]. With this approach, devices from a trained GAN can be produced with low computational cost, but a computationally expensive training set is required. New data-driven concepts that can reduce or even eliminate the need for expensive training data would dramatically expand the scope and practicality of machine learning-enabled device optimization.

3 Problem setup

The metagratings consist of silicon nanoridges and deflect normally incident light to the +1 diffraction order (Figure 1). The thickness of the gratings is fixed to be 325 nm, and the incident light is transverse magnetic (TM) polarized. The refractive index of silicon is taken from Ref. [39], and only the real part of the index is used to simplify the design problem. For each period, the metagrating is subdivided into $N = 256$ segments, each possessing a refractive index value between silicon and air during the optimization process. These refractive index values are the design variable in our problem and are specified as $\mathbf{x}$ (a $1 \times N$ vector). Deflection efficiency is defined as the intensity of light deflected to the desired
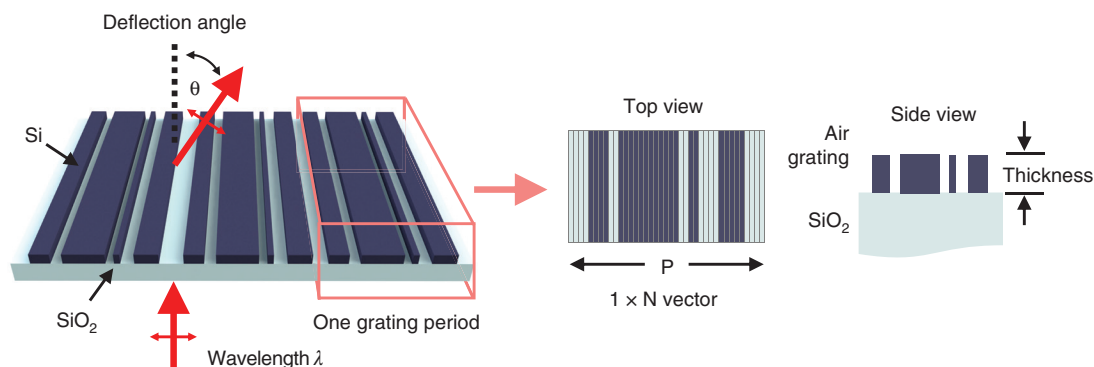
Figure 1: Schematic of a silicon metagrating that deflects normally incident TM-polarized light of wavelength λ to an outgoing angle θ. The optimization objective is to search for the metagrating pattern that maximizes deflection efficiency.

direction, defined by angle θ, normalized to the incident light intensity. The deflection efficiency is a nonlinear function of index profile $Eff = Eff(\mathbf{x})$ and is governed by Maxwell's equations. This quantity, together with the electric field profiles within a device, can be accurately solved using electromagnetic solvers.

Our optimization objective is to maximize the deflection efficiency of the metagrating at a specific operating wavelength λ and outgoing angle θ:

$$\mathbf{x}^{\star} := \underset{\mathbf{x} \in \{-1, 1\}^N}{\text{argmax}} \; Eff(\mathbf{x}) \tag{1}$$

The term $\mathbf{x}^{\star}$ represents the globally optimized device pattern, and it has an efficiency of Eff_{max}. We are interested in physical devices that possess binary index values in the vector $\mathbf{x} \in \{-1, 1\}^N$, where –1 represents air and +1 represents silicon.

4 Methods

Our proposed inverse design scheme is shown in Figure 2 and involves the training of a generative neural network to optimize a population of devices. Uniquely, our scheme does not require any training set. The input of the generator is a random noise vector $\mathbf{z} \in \mathcal{U}^N(-1, 1)$ and it has the same dimension as the output device index profile $\mathbf{x} \in [-1, 1]^N$. The generator is parameterized by ϕ, which relates $\mathbf{z}$ to $\mathbf{x}$ through a nonlinear mapping: $\mathbf{x} = G_\phi(\mathbf{z})$. In other words, the generator maps a uniform distribution of noise vectors to a device distribution $G_\phi : \mathcal{U}^N(-1, 1) \mapsto P_\phi$, where $P_\phi(\mathbf{x})$ defines the probability of generating $\mathbf{x}$ in the device space $\mathcal{S} = [-1, 1]^N$. We frame the objective of the optimization as maximizing the probability of generating the globally optimized device in $\mathcal{S}$:

$$\phi^{\star} := \underset{\phi}{\text{argmax}} \int_{\mathcal{S}} \delta(Eff(\mathbf{x}) - Eff_{max}) \cdot P_\phi(\mathbf{x}) d\mathbf{x} \tag{2}$$

4.1 Loss function formulation

While our objective function above is rigorous, it cannot be directly used for network training due to two reasons. The first is that the derivative of the δ function is nearly always 0. To circumvent this issue, we express the δ function as the following:

$$\delta(Eff(\mathbf{x}) - Eff_{max}) = \lim_{\sigma \to 0} \frac{1}{\sqrt{\pi}\sigma} \exp\left[-\left(\frac{Eff(\mathbf{x}) - Eff_{max}}{\sigma}\right)^2\right] \tag{3}$$

By substituting the δ function with this Gaussian form and leaving σ as a tunable parameter, we relax Eq. (2), and it becomes

$$\phi^{\star} := \underset{\phi}{\text{argmax}} \int_{\mathcal{S}} \exp\left[-\left(\frac{Eff(\mathbf{x}) - Eff_{max}}{\sigma}\right)^2\right] \cdot P_\phi(\mathbf{x}) d\mathbf{x} \tag{4}$$

As we will see later, the inclusion of σ as a tunable hyperparameter turns out to be important for stabilizing the network training process in the limit of training with a finite batch size.

The second reason is that the objective function depends on Eff_{max}, which is unknown. To address this problem, we approximate Eq. (4) with a different function, namely, the exponential function:

$$\phi^{\star} := \underset{\phi}{\text{argmax}} \int_{\mathcal{S}} \exp\left(\frac{Eff(\mathbf{x}) - Eff_{max}}{\sigma}\right) \cdot P_\phi(\mathbf{x}) d\mathbf{x} \tag{5}$$

This approximation is valid because $P_\phi(\mathbf{x} \,|\, Eff(\mathbf{x}) > Eff_{max}) = 0$ and our new function only needs to approximate that in Eq. (4) for efficiency values less than Eff_{max}. With this
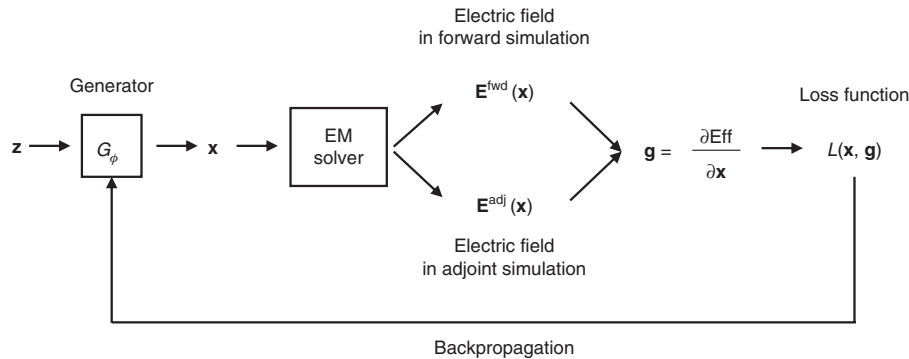

Figure 2: Schematic of generative neural network-based optimization.

approximation, we can remove $\exp(-\mathrm{Eff}_{max}/\sigma)$ from the integral:

$$\phi^{\star} := \underset{\phi}{\operatorname{argmax}} \, A \int_{\mathcal{S}} \exp\left(\frac{\mathrm{Eff}(\mathbf{x})}{\sigma}\right) \cdot P_{\phi}(\mathbf{x}) d\mathbf{x} \qquad (6)$$

$A = \exp(-\mathrm{Eff}_{max}/\sigma)$ now becomes a normalization constant and does not require explicit evaluation. Alternatives to the exponential function can be considered and tailored depending on the specific optimization problem. For this study, we will use Eq. (6).

In practice, it is not possible to evaluate Eq. (6) over the entire design space $\mathcal{S}$. We instead sample a batch of devices $\{\mathbf{x}^{(m)}\}_{m=1}^{M}$ from P_{ϕ}, which leads to further approximation of the objective function:

$$\phi^{\star} := \underset{\phi}{\operatorname{argmax}} \, \underset{\mathbf{x} \sim P_{\phi}}{\mathbb{E}} \exp\left(\frac{\mathrm{Eff}(\mathbf{x})}{\sigma}\right) \qquad (7)$$

$$\approx \underset{\phi}{\operatorname{argmax}} \frac{1}{M} \sum_{m=1}^{M} \exp\left(\frac{\mathrm{Eff}(\mathbf{x}^{(m)})}{\sigma}\right) \qquad (8)$$

We note that the deflection efficiency of device $\mathbf{x}$ is calculated using an electromagnetic solver, such that $\mathrm{Eff}(\mathbf{x})$ is not directly differentiable for backpropagation. To bypass this problem, we use the adjoint variables method to compute the efficiency gradient with respect to the refractive indices for device $\mathbf{x}$: $\mathbf{g} = \dfrac{\partial \mathrm{Eff}}{\partial \mathbf{x}}$ (Figure 2). Details pertaining to these gradient calculations can be found in other inverse design papers [11–13]. To summarize, electric field profiles within the device layer are calculated using two different electromagnetic excitation conditions. The first is the forward simulation, in which $\mathbf{E}^{\mathrm{fwd}}$ are calculated by propagating a normally incident electromagnetic wave from the substrate to the device, as shown in Figure 1. The second is the adjoint simulation, in which $\mathbf{E}^{\mathrm{adj}}$ are calculated by propagating an electromagnetic wave in the direction opposite of the desired outgoing direction. The efficiency gradient $\mathbf{g}$ is calculated by integrating the overlap of those electric field terms:

$$\mathbf{g} = \frac{\partial \mathrm{Eff}(\mathbf{x})}{\partial \mathbf{x}} \propto \mathrm{Re}(\mathbf{E}^{\mathrm{fwd}} \cdot \mathbf{E}^{\mathrm{adj}}) \qquad (9)$$

Finally, we use our adjoint gradients and objective function to define the loss function $L = L(\mathbf{x}, \mathbf{g}, \mathrm{Eff})$. Our goal is to define L such that minimizing L is equivalent to maximizing the objective function $\dfrac{1}{M} \sum_{m=1}^{M} \exp\left(\dfrac{\mathrm{Eff}(\mathbf{x}^{(m)})}{\sigma}\right)$ during generator training. With this definition, L must satisfy $-\dfrac{\partial L}{\partial \mathbf{x}^{(m)}} = \dfrac{1}{M} \dfrac{\partial}{\partial \mathbf{x}^{(m)}} \exp\left(\dfrac{\mathrm{Eff}(\mathbf{x}^{(m)})}{\sigma}\right)$ and is defined as:

$$L(\mathbf{x}, \mathbf{g}, \mathrm{Eff}) = -\frac{1}{M} \sum_{m=1}^{M} \frac{1}{\sigma} \exp\left(\frac{\mathrm{Eff}^{(m)}}{\sigma}\right) \mathbf{x}^{(m)} \cdot \mathbf{g}^{(m)} \qquad (10)$$

$\mathrm{Eff}^{(m)}$ and $\mathbf{g}^{(m)}$ are treated as independent variables calculated from electromagnetic simulations and have no dependence on $\mathbf{x}^{(m)}$. Finally, we add a regularization term $-|\mathbf{x}| \cdot (2 - |\mathbf{x}|)$ to L to ensure that the generated patterns are binary. This term reaches a minimum when the generated patterns are fully binarized. A coefficient γ is introduced to balance binarization with efficiency enhancement, and we have as our final loss function:

$$L(\mathbf{x}, \mathbf{g}, \mathrm{Eff}) =$$
$$-\frac{1}{M} \sum_{m=1}^{M} \frac{1}{\sigma} \exp\left(\frac{\mathrm{Eff}^{(m)}}{\sigma}\right) \mathbf{x}^{(m)} \cdot \mathbf{g}^{(m)} - \gamma \cdot \frac{1}{M} \sum_{m=1}^{M} |\mathbf{x}^{(m)}| \cdot (2 - |\mathbf{x}^{(m)}|)$$
$$(11)$$

4.2 Network architecture

The architecture of the generative neural network is adapted from DCGAN [40], which comprises two fully connected layers, four transposed convolution layers, and a Gaussian filter at the end to eliminate small features. LeakyReLU is used for all activation functions except for the last layer, which uses a tanh activation function. We also add dropout layers and batchnorm layers to enhance the diversity of the generated patterns. Periodic paddings are used to account for the fact that the devices are periodic structures.

4.3 Training procedure

Algorithm 1: Generative neural network-based optimization

Parameters: M, batch size. σ, loss function coefficient. α, learning rate. β_1 and β_2, momentum coefficients used in Adam. γ, binarization coefficient.

initialization;

while i < *Total iterations* **do**

    Sample $\{\mathbf{z}^{(m)}\}_{m=1}^{M} \sim \mathcal{U}^{N}(-1, 1)$;

    $\{\mathbf{x}^{(m)} = G_{\phi}(\mathbf{z}^{(m)})\}_{m=1}^{M}$, device samples;

    $\{\mathbf{g}^{(m)}\}_{m=1}^{M}$, $\{\mathrm{Eff}^{(m)}\}_{m=1}^{M} \leftarrow$ forward and adjoint simulations;

    $g_{\phi} \leftarrow$
    $\nabla_{\phi}\left[\frac{1}{M} \sum_{m=1}^{M} \frac{1}{\sigma} \exp\left(\frac{\mathrm{Eff}^{(m)}}{\sigma}\right) \mathbf{x}^{(m)} \cdot \mathbf{g}^{(m)} + \gamma \cdot \frac{1}{M} \sum_{m=1}^{M} |\mathbf{x}^{(m)}| \cdot (2 - |\mathbf{x}^{(m)}|)\right]$;

    $\phi \leftarrow \phi + \alpha \cdot \mathrm{Adam}(\phi, g_{\phi})$;

end

$\mathbf{x}^{\star} \leftarrow \operatorname{argmax}_{\mathbf{x} \in \{\mathbf{x}^{(m)} | \mathbf{x}^{(m)} \sim P_{\phi^{\star}}\}_{m=1}^{M}} \mathrm{Eff}(\mathbf{x})$

The training procedure is shown in Algorithm 1. The Adaptive Moment Estimation (Adam) algorithm, which is a variation of gradient descent, is used to optimize the network parameters ϕ. β_1 and β_2 are two hyperparameters used in Adam [41]. Initially, with the use of an identity shortcut [42], the device distribution P_ϕ is approximately a uniform distribution over the whole device space $\mathcal{S}$. During the training process, P_ϕ is continuously refined and maps more prominently to high-efficiency device subspaces. When the generator is done training, the devices produced from the generator have a high probability to be highly efficient. The final optimal device design is determined by generating a batch of devices from the fully trained generator $\{\mathbf{x}^{(m)} | \mathbf{x}^{(m)} \sim P_{\phi^*}\}_{m=1}^M$, simulating each of those devices, and selecting the best one.

4.4 Comparison with gradient-based topology optimization

In gradient-based topology optimization, a large number of local optimizations are used to search for the global optimum. For each run, device patterns are randomly initialized, and a local search in the design space is performed using gradient descent. The highest efficiency device among those optimized devices is taken as the final design. With this approach, many devices get trapped in local optima or saddle points in $\mathcal{S}$, and the computational resources used to optimize those devices do not contribute to finding or refining the globally optimal device. Additionally, finding the global optimum in a very high dimensional space can require an exceedingly large number of individual optimization runs. GLOnets are qualitatively different, as they optimize a distribution of devices to perform global optimization. As indicated in Eq. (11), each device sample $\mathbf{x}^{(m)}$ is weighted by the term $\exp(\text{Eff}^{(m)}/\sigma)$, which biases generator training towards higher efficiency devices and pushes P_ϕ towards more favorable design subspaces. In this manner, computational resources are not wasted optimizing devices within subspaces possessing low-efficiency local optima.

5 Numerical experiments

5.1 A toy model

We first perform GLOnet-based optimization on a simple test case, where the input $\mathbf{z}$ and output $\mathbf{x}$ are two dimensional. The "efficiency" function $\text{Eff}(\mathbf{x})$ is defined as:

$$\text{Eff}(x_1, x_2) = \exp(-2x_1^2)\cos(9x_1) + \exp(-2x_2^2)\cos(9x_2) \quad (12)$$

This function is non-convex and has many local optima and one global optimum at (0, 0). We use Algorithm 1 to search for the global optimum. Hyperparameters are chosen to be $\alpha = 1e-3$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $a = 30$, and $\sigma = 0.5$, and the batch size $M = 100$ is fixed throughout network training. The generator is trained for 150 iterations, and the generated samples over the course of training are shown as red dots in Figure 3. Initially, the generated "device" distribution is spread out over the $\mathbf{x}$ space, and it then gradually converges to a cluster located at the global optimum. In the training run shown, no device is trapped in any local optima. Upon training 100 distinct GLOnets, 96 of them successfully produced the globally optimized device.

5.2 Inverse design of metagratings

We next apply our algorithm to the inverse design of 63 different types of metagratings, each with differing operating wavelengths and deflection angles. The wavelengths λ range from 800 nm to 1200 nm, in increments of 50 nm, and the deflection angles θ range from 40° to 70°, in increments of 5°. Unlike our conditional GLOnet in Ref. [21], where
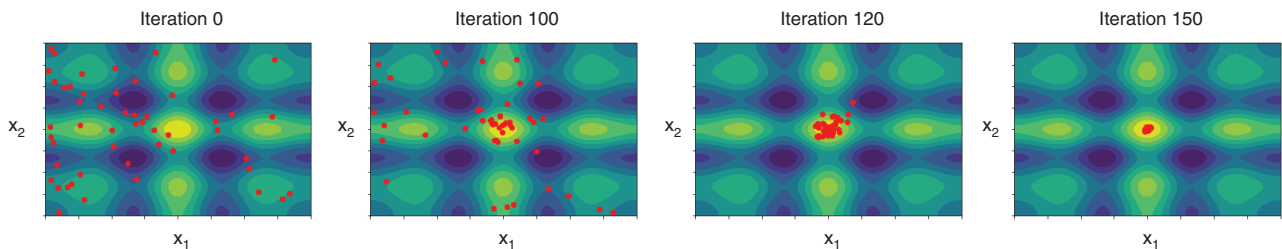
Figure 3: Results from a toy model test.
Samples generated from the generator, shown as red dots, evolve in the $[-1, 1]^2$ space over the course of training.

many different types of metagratings are simultaneously designed using a single network, we use distinct unconditional GLOnets to design each device type operating for specific wavelength and deflection angle parameters.

5.2.1 Implementation details

The hyperparameters we use are $\alpha = 0.05$, $\beta_1 = 0.9$, $\beta_2 = 0.99$, $\sigma = 0.5$, and $\gamma = 0.2$. The batch size is 100. To prevent vanishing gradients when the generated patterns are binarized as $\mathbf{x} \in \{-1, 1\}^N$, we specify the last activation function to be 1.05*tanh.

For each combination of wavelength and angle, we train the generator for 1000 iterations. Upon completion of network training, 500 different values of **z** are used to generate 500 different devices. All devices are simulated, and the highest efficiency device is taken as the final design.

The network is implemented using the pytorch-1.0.0 package. The forward and adjoint simulations are performed using the Reticolo RCWA [43] electromagnetic solver in MATLAB. The network is trained on an Nvidia Titan V GPU and four CPUs, and it takes 10 min for one

device optimization. Our code implementation can be found at https://github.com/jiaqi65/GLOnet.git.

5.2.2 Baseline

We benchmark our method with gradient-based topology optimization. For each design target (λ, θ), we start with 500 random gray-scale vectors and iteratively optimize each device using efficiency gradients calculated from forward and adjoint simulations. A threshold filter binarizes the device patterns. Each initial dielectric distribution is optimized for 200 iterations, and the highest efficiency device among 500 candidates is taken as the final design. The computational budget is set to be the same used for GLOnets training to facilitate a fair comparison.

5.2.3 Results

The efficiencies of the best devices designed using gradient-based topology optimization and GLOnets are shown in Figure 4. Ninety percent of the best devices from
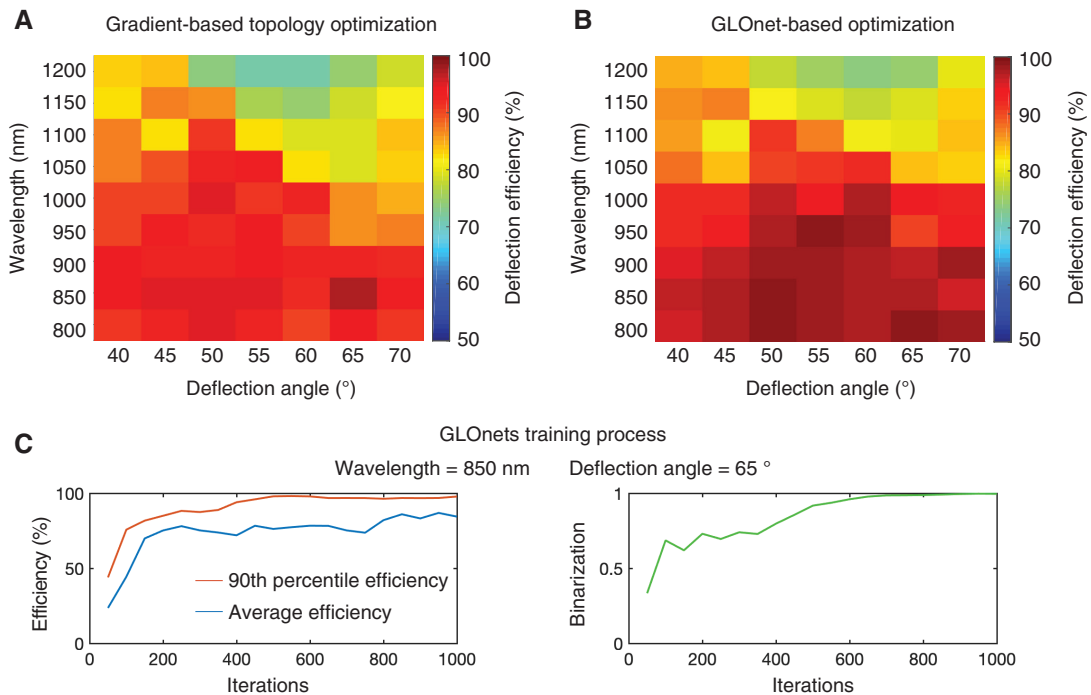

Figure 4: Summary of GLOnet performance.
(A) Plot of efficiency for devices operating with different wavelength and angle values, designed using gradient-based topology optimization. For each wavelength and angle combination, 500 individual topology optimizations are performed and the highest efficiency device is used for the plot. (B) Plot of efficiency for devices designed using GLOnet-based optimization. For each wavelength and angle combination, 500 devices are generated and the highest efficiency device is used for the plot. (C) Training process of GLOnets. The figure on the left shows the 90th percentile efficiency and average efficiency of the device batch over the course of training. The figure on the right shows the binarization degree of generated devices, which is defined as $\sum_{i=1}^{N} |\mathbf{x}_i| / N$.

GLOnets have higher or the same efficiencies compared to the best devices produced from gradient-based topology optimization; 98% of the best devices from GLOnets have efficiencies within 5% of the best devices from gradient-based topology optimization. For wavelengths and angles for which GLOnets perform worse than gradient-based topology optimization, we can perform multiple network trainings or further tune the batch size and *sigma* to get better GLOnet results. The efficiency histograms from GLOnets and gradient-based topology optimization, for select wavelength and angle pairs, are displayed in Figure 5. For most cases, efficiency histograms produced from our method have higher average efficiencies and maximal efficiencies, indicating that

low-efficiency local optima are often avoided during the training of the generator.

5.2.4 GLOnet stability

To validate the stability of GLOnet-based optimization, we train eight unconditional GLOnets independently for the same wavelength ($\lambda = 850$ nm) and deflection angle ($\theta = 65°$). For each trained GLOnet, we generate 500 devices and visualize the top 20 devices in a 2D plane using principle component analysis (PCA) (Figure 6). The principle basis is the same for all eight figures and is calculated using the top 20 devices of each GLOnet for a total of 160 devices. Six of the
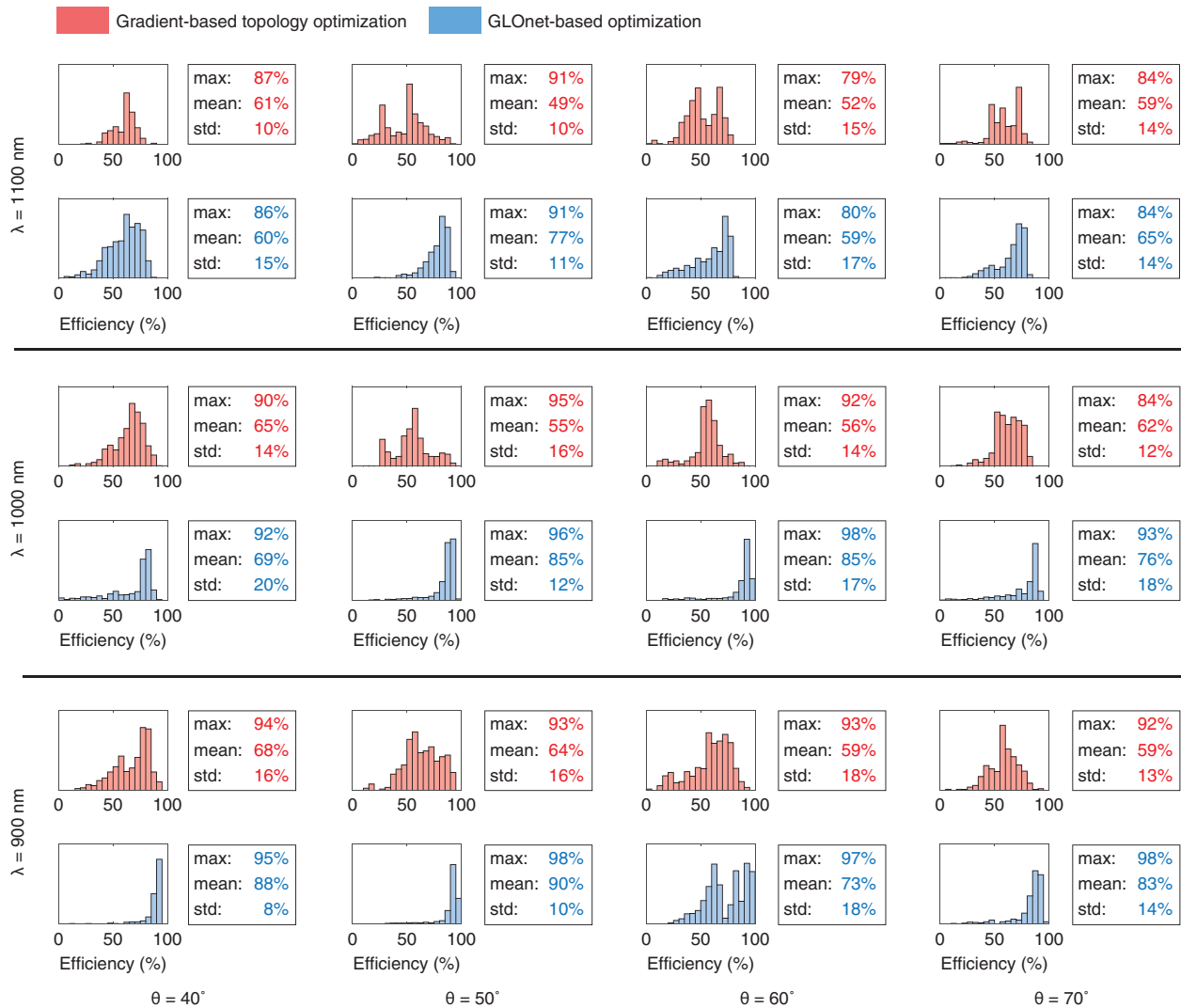
Figure 5: Efficiency histograms of 500 devices designed using gradient-based topology optimization (red) and GLOnet-based optimization (blue).
The statistics of device efficiencies in each histogram are also displayed. For most wavelength and angle values, the efficiency distributions from GLOnets are narrower and have higher maximum values compared to those from gradient-based topology optimization.

eight GLOnets converge to the same optimum, which is a device with 97% efficiency, while one GLOnet converges to a nearby optimum, which is a device with 96% efficiency. While we cannot prove that the device with 97% efficiency is globally optimal, the consistent convergence of GLOnet to this single optimum is suggestive that the network is finding the global optimum. At the very least, this demonstration shows that GLOnets have the potential to consistently generate exceptionally high performance devices.

5.2.5 Discussion of hyperparameter σ and batch size

In principle, σ approaching 0 could be used if the entire design space could be sampled to train the neural network. In this case, the globally optimized structure would be sampled and be the only device that contributes to neural network training, pushing the response of the network towards our desired objective response. However, the design space is immense and infeasible to probe in its entirety. Furthermore, this scenario would lead to the direct readout of the globally optimized device, negating the need to perform an optimization.

In practice, we can only realistically process small batches of devices that comprise a very small fraction of the total design space during network training. For many of these iterations, the best device within each batch will only be in locally optimal regions of the design space. To prevent the network from getting trapped within these local optima, we specify σ to be finite, which adds noise to the training process. In our loss function, this noise manifests in the form $\exp(-\text{Eff}/\sigma)$. This exponential expression has a Boltzmann form, and σ can therefore be treated as an effective temperature. In a manner analogous to the process of simulated annealing, σ can be modulated throughout the training process.

The impact of batch size and σ on GLOnet performance for $\lambda = 850$ nm and $\theta = 65°$ is summarized in Figure 7. In Figure 7A, σ is fixed to be 0.5, and the batch size is varied from 10 to 1000 devices per iteration. When the batch size is too small, the design space is undersampled, which increases the difficulty of finding the global optimum. As the batch size is increased, the performance of the GLOnet starts to saturate such that the design space is oversampled, leading to a waste of computational resources. For this particular GLOnet, a proper batch size that balances optimization capability with resource management is 100.

Figure 7B summarizes the impact of σ on GLOnet training, given a fixed batch size of 100 devices. The plot indicates that a proper range of σ that produces the highest efficiency devices is between 0.5 and 1.0. When σ is less than 0.5, there is insufficient noise in the training process and the network gets more easily trapped within local optima, particularly early in the training process. When σ becomes larger than 1, the performance of the GLOnet begins to deteriorate because low efficiency devices contribute more significantly in the training process, leading to excess noise.

The optimal batch size and σ values are highly problem dependent and require tuning for each optimization objective. For example, proper GLOnet optimization within a design space with relatively few local optima can be achieved with relatively small batch sizes and small values of σ. The proper selection of these hyperparameters is not intuitive and requires experience and parametric sweeps.
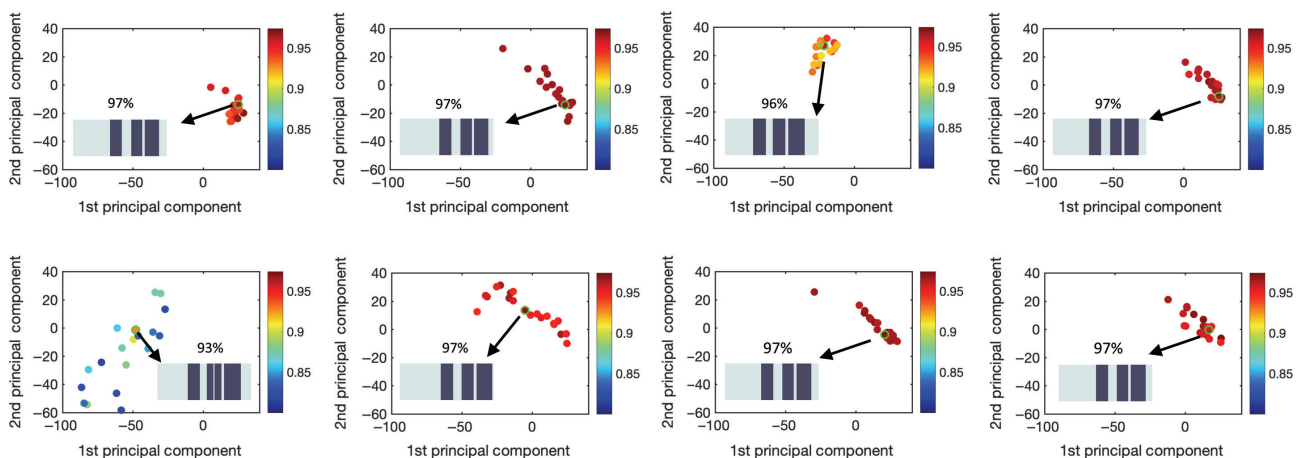
Figure 6: PCA visualization of GLOnet-optimized devices.
Eight unconditional GLOnets are trained independently, and the top 20 devices of each GLOnet are visualized. The pattern and efficiency of the best device in each plot are shown as insets.
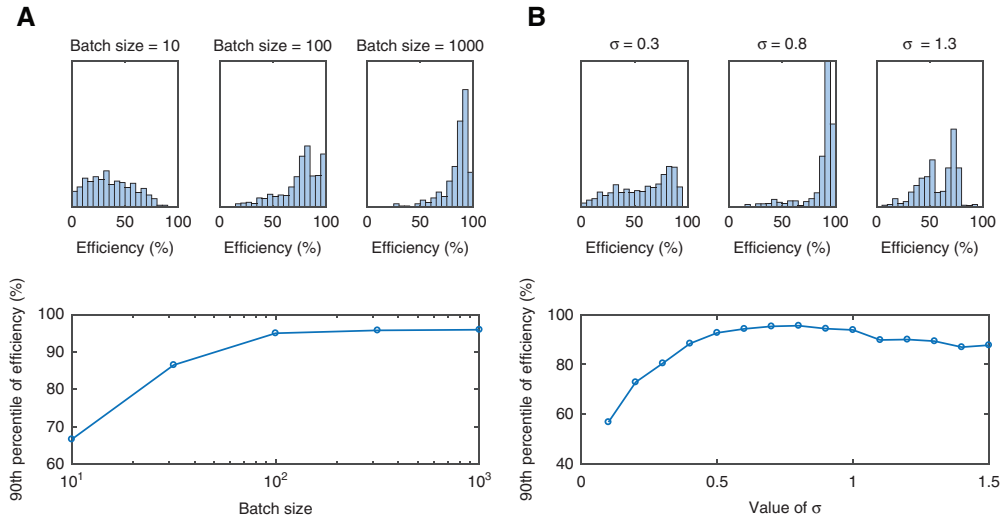
Figure 7: Performance of the unconditional GLOnet for different values of (A) batch size and (B) σ.

6 Comparison with evolution strategies

Evolutionary strategies (ES) represent classical global optimization strategies. One such algorithm is the genetic algorithm, which has been applied to many types of photonic design problems, including metasurface design [44]. Compared to our approach, genetic algorithms are not efficient and require many thousands of iterations to search for even a simple optimal device structure. The difficulty is due to the complicated relationship between optical response and device geometry, governed by Maxwell's equations. Methods like ours, which incorporate gradients, can more efficiently locate favorable regions of the design space because gradients provide clear, non-heuristic instruction on how to improve device performance.

Another ES algorithm is the Covariance Matrix Adaptation Evolution Strategy (CMA-ES), which is a probability distribution-based ES algorithm. CMA-ES assumes an explicit form of the probability distribution of the design variables (e.g. multivariate normal distribution), which is typically parameterized by several terms. Our algorithm has two main differences compared with CMA-ES. First, instead of defining an explicit probability distribution, we define an explicit generative model parameterized by the network parameters. The probability distribution in our algorithm is therefore implicit and has no assumed form. This is important as there is no reason why the probability distributions of the design variables should have a simple, explicitly defined form such as the multivariate normal distribution. Second, CMA-ES is derivative-free, but our

algorithm uses gradients and is therefore more efficient at generating device populations in the desirable parts of the design space.

7 Conclusions and future directions

In this paper, we present a generative neural network-based global optimization algorithm for metasurface design. Instead of optimizing many devices individually, which is the case for gradient-based topology optimization, we reframe the global optimization problem as the training of a generator. The efficiency gradients of all devices generated each training iteration are used to collectively improve the performance of the generator and map the noise input to favorable regions of the device subspace.

An open topic of future study is understanding how to properly select and tune the network hyperparameters dynamically during network training. We anticipate that, as the distribution of generated devices converges to a narrow range of geometries over the course of network training, the batch size can be dynamically decreased, leading to computational savings. We also hypothesize that dynamically decreasing σ can help further stabilize the GLOnet training process. These variations in batch size and σ can be predetermined prior to network training or be dynamically modified using feedback during the training process.

We are also interested in applying our algorithm to more complex systems, such as 2D or 3D metasurfaces, multi-function metasurfaces, and other photonics design

problems. A deeper understanding of loss function engineering will be necessary for multi-function metasurfaces design, which requires optimizing multiple objectives simultaneously. We envision that our algorithm has strong potential to solve inverse design problems in other domains of the physical sciences, such as mechanics and electronics.

References

[1] Jalali B, Fathpour S. Silicon photonics. J Lightwave Technol 2006;24:4600–15.

[2] Genevet P, Capasso F, Aieta F, Khorasaninejad M, Devlin R. Recent advances in planar optics: from plasmonic to dielectric metasurfaces. Optica 2017;4:139–52.

[3] Molesky S, Lin Z, Piggott AY, Jin W, Vuckovic J, Rodriguez AW. Inverse design in nanophotonics. Nat Photonics 2018;12: 659–70.

[4] Jensen JS, Sigmund O. Topology optimization for nanophotonics. Laser Photonics Rev 2011;5:308–21.

[5] Campbell SD, Sell D, Jenkins RP, Whiting EB, Fan JA, Werner DH. Review of numerical optimization techniques for meta-device design [Invited]. Opt Mater Express 2019;9:1842–63.

[6] Sigmund O, Maute K. Topology optimization approaches. Struct Multidiscip O 2013;48:1031–55.

[7] Sigmund O. On the design of compliant mechanisms using topology optimization. J Struct Mech 1997;25: 493–524.

[8] Lalau-Keraly CM, Bhargava S, Miller OD, Yablonovitch E. Adjoint shape optimization applied to electromagnetic design. Opt Express 2013;21:21693–701.

[9] Wang EW, Sell D, Phan T, Fan JA. Robust design of topology-optimized metasurfaces. Opt Mater Express 2019;9:469–82.

[10] Jensen JS, Sigmund O. Systematic design of photonic crystal structures using topology optimization: low-loss waveguide bends. Appl Phys Lett 2004;84:2022–4.

[11] Piggott AY, Lu J, Lagoudakis KG, Petykiewicz J, Babinec TM, Vučković J. Inverse design and demonstration of a compact and broadband on-chip wavelength demultiplexer. Nat Photonics 2015;9:374–7.

[12] Hughes TW, Minkov M, Williamson IAD, Fan S. Adjoint method and inverse design for nonlinear nanophotonic devices. ACS Photonics 2018;5:4781–7.

[13] Sell D, Yang J, Doshay S, Yang R, Fan JA. Large-angle, multifunctional metagratings based on freeform multimode geometries. Nano Lett 2017;17:3752–7.

[14] Yang J, Fan JA. Analysis of material selection on dielectric metasurface performance. Opt Express 2017;25:23899–909.

[15] Sell D, Yang J, Doshay S, Fan JA. Periodic dielectric metasurfaces with high-efficiency, multiwavelength functionalities. Adv Opt Mater 2017;5:2017.

[16] Yang J, Sell D, Fan JA. Freeform metagratings based on complex light scattering dynamics for extreme, high efficiency beam steering. Ann Phys 2018;530:1700302.

[17] Sell D, Yang J, Wang EW, Phan T, Doshay S, Fan JA. Ultra-high-efficiency anomalous refraction with dielectric metasurfaces. ACS Photonics 2018;5:2402–7.

[18] Lin Z, Groever B, Capasso F, Rodriguez AW, Lončar M. Topology-optimized multilayered metaoptics. Phys Rev Appl 2018;9:044030.

[19] Phan T, Sell D, Wang EW, Doshay S, Edee K, Yang J, Fan JA. High-efficiency, large-area, topology-optimized metasurfaces. Light Sci Appl 2019;8:48.

[20] Yang J, Fan JA. Topology-optimized metasurfaces: impact of initial geometric layout. Opt Lett 2017;42:3161–4.

[21] Jiang J, Fan JA. Global optimization of dielectric metasurfaces using a physics-driven neural network. Nano Lett 2019;19:5366–72.

[22] Zhang QJ, Gupta KC, Devabhaktuni VK. Artificial neural networks for RF and microwave design – from theory to practice. IEEE Trans Microw Theory Tech 2003;51:1339–50.

[23] Malheiros-Silveira GN, Hernandez-Figueroa HE. Prediction of dispersion relation and PBGs in 2-D PCs by using artificial neural networks. IEEE Photonics Technol Lett 2012;24:1799–801.

[24] Luna DR, Vasconcelos CFL, Cruz RMS. Using natural optimization algorithms and artificial neural networks in the design of effective permittivity of metamaterials. In: 2013 SBMO/IEEE MTT-S International Microwave Optoelectronics Conference (IMOC), 2013;1–4.

[25] Silva PHdaF, Cruz RMS, d'Assuncao AG. Blending PSO and ANN for optimal design of FSS filters with Koch Island patch elements. IEEE Trans Magn 2010;46:3010–3.

[26] Peurifoy J, Shen Y, Jing L, et al. Nanophotonic particle simulation and inverse design using artificial neural networks. Sci Adv 2018;4:eaar4206.

[27] Inampudi S, Mosallaei H. Neural network based design of metagratings. Appl Phys Lett 2018;112:241102.

[28] Ma W, Cheng F, Liu Y. Deep-learning-enabled on-demand design of chiral metamaterials. ACS Nano 2018;12:6326–34.

[29] Liu D, Tan Y, Khoram E, Yu Z. Training deep neural networks for the inverse design of nanophotonic structures. ACS Photonics 2018;5:1365–9.

[30] Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets. In: Advances in Neural Information Processing Systems, 2014;2672–80.

[31] Liu Z, Zhu D, Rodrigues SP, Lee KT, Cai W. Generative model for the inverse design of metasurfaces. Nano Lett 2018;18:6570–6.

[32] Jiang J, Sell D, Hoyer S, Hickey J, Yang J, Fan JA. Free-form diffractive metagrating design based on generative adversarial networks. ACS Nano 2019;13:8872–8.

[33] So S, Rho J. Designing nanophotonic structures using conditional deep convolutional generative adversarial networks. Nanophotonics 2019;8:1255–61.

[34] Brock A, Donahue J, Simonyan K. Large scale GAN training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096, 2018.

[35] Karras T, Aila T, Laine S, Lehtinen J. Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196, 2017.

[36] Zhu JY, Zhang Z, Zhang C, et al. Visual object networks: image generation with disentangled 3D representations. In: Advances in neural information processing systems, 2018;118–29.

[37] Zhu JY, Park T, Isola P, Efros AA. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE international conference on computer vision, 2017;2223–32.

[38] Ledig C, Theis L, Huszár F, et al. Photo-realistic single image super-resolution using a generative adversarial network. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017;4681–90.

[39] Green MA. Self-consistent optical parameters of intrinsic silicon at 300k including temperature coefficients. Sol Energy Mater Sol Cells 2008;92:1305–10.

[40] Radford A, Metz L, Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434, 2015.

[41] Kingma DP, Adam JB. A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[42] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016:770–8.

[43] Hugonin JP, Lalanne P. Reticolo software for grating analysis. Palaiseau, France: Institut d'Optique, 2005.

[44] Egorov V, Eitan M, Scheuer J. Genetically optimized all-dielectric metasurfaces. Opt Express 2017;25:2583–93.