Supplementary material: Time-reversal differentiation of FDTD for photonic inverse design

Rui Jie Tang,[*,†,‖] Soon Wei Daniel Lim,[*,‡,‖] Marcus Ossiander,[‡,¶] Xinghui Yin,[§] and Federico Capasso[‡]

†*University of Toronto, 27 King's College Circle, Toronto, Ontario M5S 1A1, Canada*

‡*Harvard John A. Paulson School of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138, USA*

¶*Institute of Experimental Physics, Graz University of Technology, 8010 Graz, Austria*

§*LIGO – Massachusetts Institute of Technology, Cambridge, MA 02139, USA*

‖*Contributed equally to this work*

E-mail: ruijie.tang@mail.utoronto.ca; lim982@g.harvard.edu

Number of Supplementary pages: 34

Number of Supplementary figures: 3

Number of Supplementary tables: 0

Contents

1 Residual operators in AVM

Consider the second order vector wave equations for the electric and magnetic fields (Equations S1-S2), which can be obtained by combining Maxwell's equations:

$$\nabla \times \frac{1}{\mu} \nabla \times \boldsymbol{E} + \epsilon \frac{\partial^2 \boldsymbol{E}}{\partial t^2} = -\frac{\partial \boldsymbol{J}_E}{\partial t} \tag{S1}$$

$$\nabla \times \frac{1}{\epsilon} \nabla \times \boldsymbol{H} + \mu \frac{\partial^2 \boldsymbol{H}}{\partial t^2} = \nabla \times \frac{\boldsymbol{J}_E}{\epsilon} \tag{S2}$$

The fields and spatial derivatives in the vector wave equations can be discretized separately using the Galerkin method to obtain the matrix equation in Equation S3, where the electric or magnetic fields at each node position are the rows in the field matrix $\overline{u} = \{\overline{E}, \overline{H}\}$, $\overline{M}$ and $\overline{K}$ are the system matrices for the dynamics and geometry of the system, and $\overline{F}$ is a matrix of field inputs at each node position. Generally, $\overline{M}, \overline{K}$ and $\overline{F}$ are different for the two vector wave equations.

$$\overline{M}(\boldsymbol{p}, t)\ddot{\overline{u}}(t) + \overline{K}(\boldsymbol{p}, t)\overline{u}(t) = \overline{F}(t) \tag{S3}$$

The discretized residual matrix is $\overline{R}(\overline{u}, t, \boldsymbol{p}) = \overline{M}(\boldsymbol{p}, t)\ddot{\overline{u}}(t) + \overline{K}(\boldsymbol{p}, t)\overline{u}(t) - \overline{F}(t)$, which is zero if $\overline{u}(t)$ is a solution to the system of linear equations. The residual matrix derivative $\frac{\partial \overline{R}(\overline{u}, t, \boldsymbol{p})}{\partial p_i}$ with respect to the tunable parameter p_i is thus given by the system matrix derivatives $\frac{\partial \overline{M}(\boldsymbol{p}, t)}{\partial p_i}, \frac{\partial \overline{K}(\boldsymbol{p}, t)}{\partial p_i}$:

$$\frac{\partial \overline{R}(\overline{u}, t, \boldsymbol{p})}{\partial p_i} = \frac{\partial \overline{M}(\boldsymbol{p}, t)}{\partial p_i}\ddot{\overline{u}}(t) + \frac{\partial \overline{K}(\boldsymbol{p}, t)}{\partial p_i}\overline{u}(t) - \frac{\partial \overline{F}(t)}{\partial p_i} \tag{S4}$$

The residual operator derivatives $\partial \boldsymbol{R}_E / \partial p_i$ and $\partial \boldsymbol{R}_H / \partial p_i$ should thus operate on the forward fields $\boldsymbol{E}, \boldsymbol{H}$ in the same way that the discretized residual matrices $\partial \overline{R}_E(\overline{E}, t, \boldsymbol{p}) / \partial p_i$ and $\partial \overline{R}_H(\overline{H}, t, \boldsymbol{p}) / \partial p_i$ (for the electric and magnetic vector wave equation discretizations, respectively) act on the field values $\overline{E}, \overline{H}$ at each discretized point in space and time.

2 AVM for frequency-domain objectives

When the objective function $G = \int_\Omega \int_{\Delta\omega} g[\boldsymbol{E}(\boldsymbol{x},\omega), \boldsymbol{H}(\boldsymbol{x},\omega), \boldsymbol{E}^*(\boldsymbol{x},\omega), \boldsymbol{H}^*(\boldsymbol{x},\omega)] d\omega d^3\boldsymbol{x}$ is only written in terms of frequency-domain electromagnetic fields (*i.e.,* complex fields and their complex conjugates $\boldsymbol{E}(\boldsymbol{x},\omega), \boldsymbol{H}(\boldsymbol{x},\omega), \boldsymbol{E}^*(\boldsymbol{x},\omega), \boldsymbol{H}^*(\boldsymbol{x},\omega)$) over a bandwidth $\Delta\omega$, and when the parameter vector $\boldsymbol{p}$ represents the dielectric permittivities or permeabilities over a subset of pixels, AVM reduces to a much simpler form. Such conditions are well-suited for nanophotonic inverse design, in which dielectric distributions are designed to achieve specific optical functions at well-defined frequencies and frequency bands.[1–4] The integrand derivatives are:

$$\frac{\partial g}{\partial \boldsymbol{E}(\boldsymbol{x},t)} = \int_{\Delta\omega} \left[\frac{\partial g}{\partial \boldsymbol{E}(\boldsymbol{x},\omega)} \cdot \frac{\partial \boldsymbol{E}(\boldsymbol{x},\omega)}{\partial \boldsymbol{E}(\boldsymbol{x},t)} + \frac{\partial g}{\partial \boldsymbol{E}^*(\boldsymbol{x},\omega)} \cdot \frac{\partial \boldsymbol{E}^*(\boldsymbol{x},\omega)}{\partial \boldsymbol{E}(\boldsymbol{x},t)} \right] d\omega \tag{S5}$$

$$\frac{\partial g}{\partial \boldsymbol{H}(\boldsymbol{x},t)} = \int_{\Delta\omega} \left[\frac{\partial g}{\partial \boldsymbol{H}(\boldsymbol{x},\omega)} \cdot \frac{\partial \boldsymbol{H}(\boldsymbol{x},\omega)}{\partial \boldsymbol{H}(\boldsymbol{x},t)} + \frac{\partial g}{\partial \boldsymbol{H}^*(\boldsymbol{x},\omega)} \cdot \frac{\partial \boldsymbol{H}^*(\boldsymbol{x},\omega)}{\partial \boldsymbol{H}(\boldsymbol{x},t)} \right] d\omega \tag{S6}$$

Since the frequency-domain fields are Fourier transforms of the time-domain fields in the FDTD simulation, the field derivatives can be written:

$$\boldsymbol{E}(\boldsymbol{x},\omega) = \int_T \boldsymbol{E}(\boldsymbol{x},t)\exp(i\omega t)dt \Rightarrow \frac{\partial \boldsymbol{E}(\boldsymbol{x},\omega)}{\partial \boldsymbol{E}(\boldsymbol{x},t)} = \mathbb{I}\exp(i\omega t) \tag{S7}$$

$$\boldsymbol{H}(\boldsymbol{x},\omega) = \int_T \boldsymbol{H}(\boldsymbol{x},t)\exp(i\omega t)dt \Rightarrow \frac{\partial \boldsymbol{H}(\boldsymbol{x},\omega)}{\partial \boldsymbol{H}(\boldsymbol{x},t)} = \mathbb{I}\exp(i\omega t) \tag{S8}$$

$$\boldsymbol{E}^*(\boldsymbol{x},\omega) = \int_T \boldsymbol{E}(\boldsymbol{x},t)\exp(-i\omega t)dt \Rightarrow \frac{\partial \boldsymbol{E}^*(\boldsymbol{x},\omega)}{\partial \boldsymbol{E}(\boldsymbol{x},t)} = \mathbb{I}\exp(-i\omega t) \tag{S9}$$

$$\boldsymbol{H}^*(\boldsymbol{x},\omega) = \int_T \boldsymbol{H}(\boldsymbol{x},t)\exp(-i\omega t)dt \Rightarrow \frac{\partial \boldsymbol{H}^*(\boldsymbol{x},\omega)}{\partial \boldsymbol{H}(\boldsymbol{x},t)} = \mathbb{I}\exp(-i\omega t) \tag{S10}$$

$\mathbb{I}$ is the identity matrix. Furthermore, since f is real-valued, by the Wirtinger derivative,

$$\frac{\partial g}{\partial \boldsymbol{E}(\boldsymbol{x},\omega)} = \left[\frac{\partial g}{\partial \boldsymbol{E}^*(\boldsymbol{x},\omega)} \right]^* \tag{S11}$$

$$\frac{\partial g}{\partial \boldsymbol{H}(\boldsymbol{x},\omega)} = \left[\frac{\partial g}{\partial \boldsymbol{H}^*(\boldsymbol{x},\omega)} \right]^* \tag{S12}$$

The adjoint system becomes:

$$\nabla \times \boldsymbol{E}^A(\boldsymbol{x},t) = \mu(\boldsymbol{x},t)\frac{\partial \boldsymbol{H}^A(\boldsymbol{x},t)}{\partial t} + 2Re\int_{\Delta\omega}\frac{\partial g}{\partial \boldsymbol{H}(\boldsymbol{x},\omega)}\exp(i\omega t)d\omega \tag{S13}$$

$$\nabla \times \boldsymbol{H}^A(\boldsymbol{x},t) = -\epsilon(\boldsymbol{x},t)\frac{\partial \boldsymbol{E}^A(\boldsymbol{x},t)}{\partial t} + 2Re\int_{\Delta\omega}\frac{\partial g}{\partial \boldsymbol{E}(\boldsymbol{x},\omega)}\exp(i\omega t)d\omega \tag{S14}$$

Switching to the backward time $\tau = T - t$, the adjoint system with forward solving becomes:

The adjoint system becomes:

$$\nabla \times \boldsymbol{E}^A(\boldsymbol{x},T-t) = -\mu(\boldsymbol{x},T-t)\frac{\partial \boldsymbol{H}^A(\boldsymbol{x},T-t)}{\partial t} + 2Re\int_{\Delta\omega}\frac{\partial g}{\partial \boldsymbol{H}(\boldsymbol{x},\omega)}\exp(i\omega(T-t))d\omega \tag{S15}$$

$$\nabla \times \boldsymbol{H}^A(\boldsymbol{x},T-t) = \epsilon(\boldsymbol{x},T-t)\frac{\partial \boldsymbol{E}^A(\boldsymbol{x},T-t)}{\partial t} + 2Re\int_{\Delta\omega}\frac{\partial g}{\partial \boldsymbol{E}(\boldsymbol{x},\omega)}\exp(i\omega(T-t))d\omega \tag{S16}$$

The source terms are time-harmonic and have a straightforward interpretation: one has to place a point electric dipole of amplitude proportional to $\partial g/\partial \boldsymbol{E}$ and a point magnetic dipole of amplitude proportional to $-(1/\mu)\partial g/\partial \boldsymbol{H}$ at every $\boldsymbol{x} \in \Omega$, for every discretized frequency $\omega \in \Delta\omega$ of interest.[3] Importantly, since the dipole sources are time-harmonic, one only needs the complex dipole amplitudes to determine the source behavior for all time during the adjoint simulation. This means that one does not to explicitly record the time-domain field values during the forward simulation – it will suffice to monitor the frequency-domain fields by accumulating partial Fourier sums during the FDTD to obtain the frequency-domain complex fields at the intended dipole positions.[5,6] This greatly reduces the memory requirements compared to that of the adjoint procedure for a time-domain objective function.

The derivative calculation step is simple for objective functions that depend on the pixel-wise permittivities and permeabilities. Since the tunable permittivities and permeabilities are localized to individual grid points, the system matrices in the adjoint formulation for the

electric and magnetic fields can be differentiated explicitly with respect to the permittivity ϵ_i and permeability μ_i at grid point i at position x_i. We exhibit the system matrix derivatives for objective functions that depend on the complex electric field:

$$\frac{\partial \overline{M}_E}{\partial \epsilon_i} \ddot{\overline{u}}(t) = \delta_{ii} D_{tt} u(t) \tag{S17}$$

$$\frac{\partial \overline{K}_E}{\partial \epsilon_i} = 0 \tag{S18}$$

$$\frac{\partial \overline{F}_E}{\partial \epsilon_i} = 0 \tag{S19}$$

where D_{tt} is the discretized second time derivative and δ_{ii} is a zero matrix with a 1 in the (i, i) position. The objective function gradient element is thus:

$$\frac{dG}{d\epsilon_i} = -\int_\Omega d^3\boldsymbol{x} \int_0^T dt \left[\boldsymbol{E}^A(\boldsymbol{x}, t) \cdot \frac{\partial^2}{\partial t^2} \delta(\boldsymbol{x} - \boldsymbol{x}_i) \boldsymbol{E}(\boldsymbol{x}, t) \right] \tag{S20}$$

$$= -\int_0^T dt \left[\boldsymbol{E}^A(\boldsymbol{x}_i, t) \cdot \frac{\partial^2}{\partial t^2} \boldsymbol{E}(\boldsymbol{x}_i, t) \right] \tag{S21}$$

Since the fields at the start and end of the simulation are zero, we can perform the time integral over all time.

$$\frac{dG}{d\epsilon_i} = -\int_{-\infty}^{\infty} dt \left[\boldsymbol{E}^A(\boldsymbol{x}_i, t) \cdot \frac{\partial^2}{\partial t^2} \boldsymbol{E}(\boldsymbol{x}_i, t) \right] \tag{S22}$$

$$= -\frac{1}{4\pi^2} \int_{-\infty}^{\infty} dt \left[\int_{\omega=-\infty}^{\infty} \boldsymbol{E}^A(\boldsymbol{x}_i, \omega) e^{-i\omega t} d\omega \cdot \frac{\partial^2}{\partial t^2} \int_{\omega=-\infty}^{\infty} \boldsymbol{E}(\boldsymbol{x}_i, \omega) e^{-i\omega t} d\omega \right] \tag{S23}$$

$$= -\frac{1}{4\pi^2} \int_{-\infty}^{\infty} dt \left[\int_{\omega=-\infty}^{\infty} \boldsymbol{E}^A(\boldsymbol{x}_i, \omega) e^{-i\omega t} d\omega \cdot \int_{\omega=-\infty}^{\infty} (-\omega^2) \boldsymbol{E}(\boldsymbol{x}_i, \omega) e^{-i\omega t} d\omega \right] \tag{S24}$$

$$= \frac{1}{4\pi^2} \int_{\omega=-\infty}^{\infty} \int_{\omega'=-\infty}^{\infty} \int_{t=-\infty}^{\infty} e^{-i(\omega+\omega')t} \omega'^2 dt \boldsymbol{E}^A(\boldsymbol{x}_i, \omega) \cdot \boldsymbol{E}(\boldsymbol{x}_i, \omega') d\omega d\omega' \tag{S25}$$

$$= \frac{1}{4\pi^2} \int_{\omega=-\infty}^{\infty} \int_{\omega'=-\infty}^{\infty} 2\pi \delta(\omega + \omega') \omega'^2 \boldsymbol{E}^A(\boldsymbol{x}_i, \omega) \cdot \boldsymbol{E}(\boldsymbol{x}_i, \omega') d\omega d\omega' \tag{S26}$$

$$= \frac{1}{2\pi} \int_{\omega=-\infty}^{\infty} (-\omega)^2 \boldsymbol{E}^A(\boldsymbol{x}_i, \omega) \cdot \boldsymbol{E}(\boldsymbol{x}_i, -\omega) d\omega \tag{S27}$$

$$= \frac{1}{2\pi} \int_{\omega=-\infty}^{\infty} \omega^2 \boldsymbol{E}^A(\boldsymbol{x}_i, \omega) \cdot \boldsymbol{E}^*(\boldsymbol{x}_i, \omega) d\omega \tag{S28}$$

Note that $\boldsymbol{E}(\boldsymbol{x}_i, -\omega) = \boldsymbol{E}^*(\boldsymbol{x}_i, \omega)$ since the time-domain signal is purely real. Given that the frequency-domain signal has support only over $\Delta\omega$ and $-\Delta\omega$ ($\Delta\omega$ only contains positive frequencies), we can perform the frequency integration over that domain:

$$\frac{dG}{d\epsilon_i} = \frac{1}{2\pi}\int_{\Delta\omega}\left[(-\omega)^2\boldsymbol{E}^A(\boldsymbol{x}_i,-\omega)\cdot\boldsymbol{E}^*(\boldsymbol{x}_i,-\omega)+\omega^2\boldsymbol{E}^A(\boldsymbol{x}_i,\omega)\cdot\boldsymbol{E}^*(\boldsymbol{x}_i,\omega)\right]d\omega \tag{S29}$$

$$= \frac{1}{2\pi}\int_{\Delta\omega}\left[\omega^2\boldsymbol{E}^{A*}(\boldsymbol{x}_i,\omega)\cdot\boldsymbol{E}(\boldsymbol{x}_i,\omega)+\omega^2\boldsymbol{E}^A(\boldsymbol{x}_i,\omega)\cdot\boldsymbol{E}^*(\boldsymbol{x}_i,\omega)\right]d\omega \tag{S30}$$

$$= \frac{1}{\pi}Re\int_{\Delta\omega}\omega^2\boldsymbol{E}^A(\boldsymbol{x}_i,\omega)\cdot\boldsymbol{E}^*(\boldsymbol{x}_i,\omega)d\omega \tag{S31}$$

For a single frequency objective function, the gradient is proportional to $Re[\boldsymbol{E}^A(\boldsymbol{x}_i)\cdot\boldsymbol{E}^*(\boldsymbol{x}_i)]$ evaluated at the grid position $\boldsymbol{x}_i$, the form that is commonly employed in nanophotonic inverse design. This gradient can also be derived by exploiting the symmetry of Lorentz reciprocity between time-harmonic current sources and their fields[2] for this special case.

3 Reverse mode automatic differentiation

Here, we provide an intuitive explanation of the behavior and scaling performance of reverse mode automatic differentiation (RM AD). There are two passes in RM AD. The forward pass traverses the computational tree from the inputs to the outputs and stores all the intermediate values obtained. The backward pass performs the chain rule for differentiation from the outputs back towards the inputs, drawing upon the stored intermediate values. The backward pass is also known as backpropagation, which is the foundation of modern machine learning, as it provides the objective function gradients with respect to many tunable parameters (*e.g.,* weights and biases in neural networks) for iterative model training and refinement. These forward and backward pass features are best envisioned with an exemplar objective function G consisting of N scalar functions, F_1 to F_N, which operate sequentially

on an input x:

$$G(x) = (F_N \circ F_{N-1} \circ \cdots \circ F_1)(x) = F_N(F_{N-1}(\cdots F_1(x) \cdots)) \tag{S32}$$

This computation tree is a single long line of operations from x through $F_1, F_2, \ldots$ to G. Each operation is a continuous elementary function of the previous variable with a stored functional derivative form. For example, F_i is a function of F_{i-1} such that the functional derivative $\partial F_i / \partial F_{i-1}$ is well-defined in terms of F_{i-1}, and this information is stored in the respective tree nodes. We show the objective function and intermediate functions are shown as single-valued for simplicity, but the tree is generalizable to multi-valued functions and multi-valued inputs. During backpropagation, we seek to compute the sensitivity, dG/dP, where $P \in \{G, F_N, F_{N-1}, \ldots, F_1, x\}$ represents every node in the tree. The sensitivity is the derivative of the final objective function with respect to the value of a node. Notice that the sensitivity is written as a total derivative and not a partial derivative; the parameter P may not be explicitly represented in the objective function G but G is indirectly affected by P through the other intermediate parameters. The final sensitivity value to be calculated is dG/dx, the gradient of the objective function for descent optimization.

The first adjoint term is $dG/dG = 1$ by definition. If we compute the adjoints in reverse order from G towards x, we can re-use (backpropagate) the adjoint value from the previous step to compute the subsequent adjoint value:

$$\begin{aligned}
\frac{dG}{dF_N} &= \frac{dG}{dG}\frac{\partial G}{\partial F_N}\bigg|_{F_N} \\
\frac{dG}{dF_{N-1}} &= \frac{dG}{dF_N}\frac{\partial F_N}{\partial F_{N-1}}\bigg|_{F_{N-1}} \\
&\vdots \\
\frac{dG}{dF_1} &= \frac{dG}{dF_2}\frac{\partial F_2}{\partial F_1}\bigg|_{F_1} \\
\frac{dG}{dx} &= \frac{dG}{dF_1}\frac{\partial F_1}{\partial x}\bigg|_{x}
\end{aligned} \tag{S33}$$

At every calculation step, the backpropagating sensitivity term is multiplied into the derivative $\partial F_i/\partial F_{i-1}$ evaluated at F_{i-1}. The stored node values $\{F_i, x\}_{i=1,\ldots,N}$ during the forward pass thus provide the necessary information to calculate the derivatives during backpropagation.

This simple example also demonstrates how the computation complexity of RM AD is independent of N_{input}. Suppose that $x \in \mathbb{R}^{N_{input}}$ is an N_{input}-dimensional vector ($N_{input} \ll N$ so there are many more mathematical operations than tunable parameters, as would be typical) and $F_1 : \mathbb{R}^{N_{input}} \mapsto \mathbb{R}$. We keep all other operations $F_2, \ldots, F_N, G$ the same so that the backpropagation algorithm yields dG/dF_1 with the same number of steps as described previously. Then the multi-valued gradient $\nabla_x G$ can be easily obtained from dG/dF_1 and the functional form of F_1 through N_{input} additional elementary steps:

$$\nabla_x G = \left\{ \frac{dG}{dx_i} \right\}_{i=1,\ldots,N_{input}} = \left\{ \frac{dG}{dF_1} \frac{\partial F_1}{\partial x_i} \right\}_{i=1,\ldots,N_{input}} \tag{S34}$$

This gradient calculation thus does not scale with N_{input} since the number of mathematical operations $N \gg N_{input}$ is large and dominates the computational cost. Instead of running the full computation tree once for each of the N_{input} degrees of freedom, and performing N operations each time, as a single-sided finite difference technique would require, backpropagation eliminates redundant calculations and attains favorable scaling with respect to N_{input}. As mentioned earlier, RM AD is guaranteed to provide the multivariate gradient in no more than five times the number of operations in the objective function (*i.e.,* within five times the runtime of the forward pass), independent of the degrees of freedom.[7] This scaling performance is comparable to the adjoint method, which also yields the multivariate gradient in a constant multiple of simulation runtimes (*i.e.,* two runs).

4 Time and memory scaling derivations for gradient calculation methods

We derive here the time and memory scaling relations for gradient calculation in different schemes.

4.1 Complexity of one FDTD simulation

Consider an FDTD simulation with N_V pixels and over N_T timesteps. We consider a common use of the FDTD to yield the steady-state frequency-domain response of a time-independent dielectric geometry. This involves illuminating the dielectric geometry with an electromagnetic pulse and recording the time-domain field at the positions of interest. The pulsed source is a superposition of temporal frequencies, allowing the steady-state frequency domain field at each pixel position to be obtained by Fourier transformation of the recorded time-domain field, such as in the case below for angular frequency ω:

$$E(\omega) = \int_0^\infty E(t)e^{-i\omega t}dt \tag{S35}$$

We do not need to perform the integration all the way to infinity because $E(t)$ decays over a finite time interval, allowing the FDTD simulation to be halted at a maximum time step N_T and the Fourier transform approximated by a discrete form:

$$E(\omega) = \sum_{n=0}^{N_T} E(n\Delta t)e^{-i\omega n\Delta t}\Delta t \tag{S36}$$

Frequency-domain field values thus depend on the field values over the simulation duration. This requires that the time-domain field values be stored over the length of the simulation, which will require significant memory storage that scales as $O(N_V N_T)$ if the frequency-domain response of all FDTD pixels is desired. FDTD users typically select a subset of the pixels for frequency-domain evaluation by introducing a monitor object, but

this large storage requirement is typically alleviated by only evaluating the Fourier transform over a small number of frequency points $N_f \ll N_T$ that have been specified in advance. For each frequency of interest ω, the FDTD program records an additional value $P(\omega, n)$ at each pixel of interest. $P(\omega, n)$ is the partial sum at timestep n of the discrete Fourier transform at that frequency.[6]

$$P(\omega, n) = \sum_{k=0}^{n} E(k\Delta t) e^{-i\omega k \Delta t} \Delta t \tag{S37}$$

The partial sum value can be updated at every timestep and converges to the discrete Fourier transform value by the end of the FDTD simulation. The memory required for this partial sum accumulation (assuming all N_V pixels are treated in this manner) thus scales as $O(N_V N_f)$, which is much better than the $O(N_V N_T)$ memory scaling required for storing all the time-domain information for Fourier transformation after simulation since $N_f \ll N_T$ typically.

The time complexity for a single FDTD run scales as $O(N_V N_T N_f)$ since the N_V pixels need to be updated for N_T timesteps, and there are N_f partial sums to be updated for each field value in each pixel.

4.2 Finite Difference performance scaling

The single-sided finite difference evaluates the objective function at the position of interest, then evaluates the effect of N_{input} small perturbations, one in each of the degrees of freedom. Thus, it takes $N_{input} + 1$ function calls to make N_{input} first order gradient approximations. Each function call is equivalent to an FDTD simulation run that has $O(N_V N_T N_f)$ operations and a peak simulation memory usage of $O(N_V N_f)$. $O(N_{input})$ simulation calls result in a $O(N_{input} N_V N_T N_f)$ runtime complexity. The peak simulation memory usage scales with N_{input} if the simulations are run in parallel.

4.3 Forward Mode Automatic Differentiation performance scaling

In the forward mode AD calculation, the chain rule calculations are performed in the same order as in the forward computation of the objective function. The forward mode calculation uses the computation tree as a backbone. Each node or intermediate variable in the computation tree is augmented to store the derivative of the intermediate variable with respect to every degree of freedom, of which there are N_{input}. For example, for intermediate variable y, and input degrees of freedom $\{x_i\}_{i=1,\ldots,N_{input}}$, the node for y also contains the N_{input} derivative values $\{dG/dx_i\}_{i=1,\ldots,N_{input}}$. These derivatives are propagated forward as the computation progresses in the forward direction, until the computation reaches the objective function value G, at which time the algorithm yields the derivative values $\{dG/dx_i\}_{i=1,\ldots,N_{input}}$, the desired gradient.

Importantly, during the forward mode AD process, the derivative values stored at previous nodes are not re-used. This means that once the partial derivatives at each node are propagated to the next layer in the FDTD computation tree, which corresponds to an update time step, the memory associated with the partial derivatives can be freed. Effectively, the forward mode AD just needs to keep track of the field values and its derivatives at one timestep. Thus, the peak memory complexity of forward mode AD in FDTD scales as $O(N_{input}N_V N_f)$.

Since the elementary mathematical operations at each node are duplicated by N_{input} times, the time complexity of forward mode AD in FDTD is N_{input} times that of the single run FDTD, yielding a time complexity of $O(N_{input}N_V N_T N_f)$.

4.4 Reverse Mode Automatic Differentiation performance scaling

The reverse mode algorithm requires the entire computation tree and intermediate values to be stored for the backpropagation step. Thus, the peak memory complexity scales as $O(N_{output}N_V N_T N_f)$. For each objective function, the gradient calculation time complexity scales as an integer multiplied by the time complexity of a single run, independent of N_{input}.

Thus, the time complexity of reverse mode AD in FDTD is $O(N_{output} N_V N_T N_f)$.

4.5 Adjoint variable method performance scaling

Adjoint optimization in nanophotonics typically requires two FDTD simulations: the forward and adjoint simulation. Both simulations take place on the same simulation volume and over the same spectral range. Thus, the time complexity for the adjoint method scales as that of an FDTD run that is performed N_{output} times, one for each objective function: $O(N_{output} N_V N_T N_f)$. The peak memory complexity scales as that of a single FDTD run, $O(N_{output} N_V N_f)$, assuming that the adjoint simulation for different objective functions are performed in parallel and the objective function is either in the frequency-domain or uses checkpointing to reduce backpropagation memory requirements.

4.6 Direct Differentiation performance scaling

As described in the main text, the peak memory requirements for DD come from two contributions: storage of the field derivative values (with respect to each of the N_{output} objective functions) for one timestep of the simulation and storage of the time-domain field values at the recording boundary. The memory required to store the fields at one timestep is $O(N_V N_f)$ and is thus $O(N_{output} N_V N_f)$ for N_{output} field derivative values. For the recording boundary, the peak memory required to store the time-domain field values is proportional to the number of timesteps multiplied by the number of pixels along the recording boundary ∂N_V. Thus, the peak memory scaling for DD is $O(N_{output} N_V N_f + N_T \partial N_V)$.

The time complexity of DD is the same as that of reverse mode AD, less $O(N_V N_T N_f)$ memory storage operations (no need to store the forward pass intermediate values) and with an additional $O(N_V N_T N_f)$ operations for the backward time-stepping FDTD. Thus, the time complexity of DD is also $O(N_{output} N_V N_T N_f)$.

5 Direct Differentiation Key Equations

5.1 FDTD forward update equations

The FDTD update equations solve Maxwell Equations on the Yee grid. Consider a system discretized into cubic grids of equal side length $\Delta = \Delta x = \Delta y = \Delta z$. We label each grid element and the field components within the grid by an integer index tuple (x, y, z) so that one corner is located at $(x \cdot \Delta, y \cdot \Delta, z \cdot \Delta)$. Each grid contains six field components $D_x, D_y, D_z, H_x, H_y, H_z$ that are defined at different locations inside the grid. D_l is stored at $(x \cdot \Delta, y \cdot \Delta, z \cdot \Delta) + (1/2)\boldsymbol{e}_l\Delta$ and H_l is stored at $((x + 1/2) \cdot \Delta, (y + 1/2) \cdot \Delta, (z + 1/2) \cdot \Delta) - (1/2)\boldsymbol{e}_l\Delta$ for $l = x, y, z$ and unit vectors $\boldsymbol{e}_l$. Each grid cube is associated with a single uniform dielectric permittivity $\epsilon(x, y, z)$ for the model considered here, although more complex dielectric tensors can be incorporated. The simulation time is discretized into N_T time steps, each of duration Δt. In each full timestep, the $\boldsymbol{D}$ and $\boldsymbol{H}$ fields are updated in alternating order so that the time interval between $\boldsymbol{D}$ and $\boldsymbol{H}$ field updates is $(1/2)\Delta t$. In the following equations, the four-tuple (x, y, z, t) comprising integers and half-integers corresponds to the real spacetime location $(x \cdot \Delta, y \cdot \Delta, z \cdot \Delta, t \cdot \Delta t)$.

Considering Maxwell's equations in matter with SI units with no free currents and magnetization:

$$\frac{\partial \boldsymbol{D}}{\partial t} = \nabla \times \boldsymbol{H} \tag{S38}$$

$$\frac{\partial \boldsymbol{H}}{\partial t} = -\frac{1}{\mu}\nabla \times \boldsymbol{E} \tag{S39}$$

It is generally useful to express Maxwell's equations in Gaussian units (setting $\mu = \mu_0$) so that $\boldsymbol{D}, \boldsymbol{H}$ and $\boldsymbol{E}$ are all on the same order of numerical magnitude:

$$\frac{\partial \boldsymbol{D}}{\partial t} = c\nabla \times \boldsymbol{H} \tag{S40}$$

$$\frac{\partial \boldsymbol{H}}{\partial t} = -c\nabla \times \boldsymbol{E} \tag{S41}$$

The rest of this Supplementary section will use Gaussian units. Discretizing the derivatives using finite differences and isolating the time-advanced terms, we obtain the FDTD forward update equations:

$$\begin{aligned} D_x\left(x+\frac{1}{2},y,z,t\right) &= D_x\left(x+\frac{1}{2},y,z,t-1\right) \\ &+ \frac{c\Delta t}{\Delta}\left[H_z\left(x+\frac{1}{2},y+\frac{1}{2},z,t-\frac{1}{2}\right) - H_z\left(x+\frac{1}{2},y-\frac{1}{2},z,t-\frac{1}{2}\right)\right. \\ &\left. - H_y\left(x+\frac{1}{2},y,z+\frac{1}{2},t-\frac{1}{2}\right) + H_y\left(x+\frac{1}{2},y,z-\frac{1}{2},t-\frac{1}{2}\right)\right] \end{aligned} \tag{S42}$$

$$\begin{aligned} D_y\left(x,y+\frac{1}{2},z,t\right) &= D_y\left(x,y+\frac{1}{2},z,t-1\right) \\ &+ \frac{c\Delta t}{\Delta}\left[H_x\left(x,y+\frac{1}{2},z+\frac{1}{2},t-\frac{1}{2}\right) - H_x\left(x,y+\frac{1}{2},z-\frac{1}{2},t-\frac{1}{2}\right)\right. \\ &\left. - H_z\left(x+\frac{1}{2},y+\frac{1}{2},z,t-\frac{1}{2}\right) + H_z\left(x-\frac{1}{2},y+\frac{1}{2},z,t-\frac{1}{2}\right)\right] \end{aligned} \tag{S43}$$

$$\begin{aligned} D_z\left(x,y,z+\frac{1}{2},t\right) &= D_z\left(x,y,z+\frac{1}{2},t-1\right) \\ &+ \frac{c\Delta t}{\Delta}\left[H_y\left(x+\frac{1}{2},y,z+\frac{1}{2},t-\frac{1}{2}\right) - H_y\left(x-\frac{1}{2},y,z+\frac{1}{2},t-\frac{1}{2}\right)\right. \\ &\left. - H_x\left(x,y+\frac{1}{2},z+\frac{1}{2},t-\frac{1}{2}\right) + H_x\left(x,y-\frac{1}{2},z+\frac{1}{2},t-\frac{1}{2}\right)\right] \end{aligned} \tag{S44}$$

$$H_x\left(x,y+\frac{1}{2},z+\frac{1}{2},t+\frac{1}{2}\right)=H_x\left(x,y+\frac{1}{2},z+\frac{1}{2},t-\frac{1}{2}\right)+\frac{c\Delta t}{\Delta}\left[E_y\left(x,y+\frac{1}{2},z+1,t\right)-E_y\left(x,y+\frac{1}{2},z,t\right)-E_z\left(x,y+1,z+\frac{1}{2},t\right)+E_z\left(x,y,z+\frac{1}{2},t\right)\right] \tag{S45}$$

$$H_y\left(x+\frac{1}{2},y,z+\frac{1}{2},t+\frac{1}{2}\right)=H_y\left(x+\frac{1}{2},y,z+\frac{1}{2},t-\frac{1}{2}\right)+\frac{c\Delta t}{\Delta}\left[E_z\left(x+1,y,z+\frac{1}{2},t\right)-E_z\left(x,y,z+\frac{1}{2},t\right)-E_x\left(x+\frac{1}{2},y,z+1,t\right)+E_x\left(x+\frac{1}{2},y,z,t\right)\right] \tag{S46}$$

$$H_z\left(x+\frac{1}{2},y+\frac{1}{2},z,t+\frac{1}{2}\right)=H_z\left(x+\frac{1}{2},y+\frac{1}{2},z,t-\frac{1}{2}\right)+\frac{c\Delta t}{\Delta}\left[E_x\left(x+\frac{1}{2},y+1,z,t\right)-E_x\left(x+\frac{1}{2},y,z,t\right)-E_y\left(x+1,y+\frac{1}{2},z,t\right)+E_y\left(x,y+\frac{1}{2},z,t\right)\right] \tag{S47}$$

The $\boldsymbol{D}$ and $\boldsymbol{E}$ fields are related through the constitutive relations:

$$E_x\left(x+\frac{1}{2},y,z,t\right)=\frac{1}{\epsilon(x,y,z)}D_x\left(x+\frac{1}{2},y,z,t\right) \tag{S48}$$

$$E_y\left(x,y+\frac{1}{2},z,t\right)=\frac{1}{\epsilon(x,y,z)}D_y\left(x,y+\frac{1}{2},z,t\right) \tag{S49}$$

$$E_z\left(x,y,z+\frac{1}{2},t\right)=\frac{1}{\epsilon(x,y,z)}D_z\left(x,y,z+\frac{1}{2},t\right) \tag{S50}$$

The fields are initialized at zero. A single FDTD loop proceeds as follows:

1. $\boldsymbol{E}$ fields are used to update the $\boldsymbol{H}$ fields.

2. TFSF boundary conditions are enforced and the source $\boldsymbol{E}, \boldsymbol{H}$ fields are injected.

3. $\boldsymbol{H}$ fields are used to update the $\boldsymbol{D}$ fields.

4. Lossy boundary $\boldsymbol{D}, \boldsymbol{H}$ fields are recorded at the recording layer.

5. $\boldsymbol{D}$ fields are used to compute the $\boldsymbol{E}$ fields at the same location using the constitutive equations.

6. For frequency-domain field monitors, the $\boldsymbol{E}$ fields are used to update the running discrete Fourier transform to obtain the complex $\boldsymbol{E}$ fields at the monitor positions.

5.2 FDTD reverse update equations

By exchanging the positions of the time-advanced and time-retarded terms, we obtain the time-reversed FDTD update equations:

$$D_x\left(x+\frac{1}{2},y,z,t-1\right) = D_x\left(x+\frac{1}{2},y,z,t\right) - \frac{c\Delta t}{\Delta}\left[H_z\left(x+\frac{1}{2},y+\frac{1}{2},z,t-\frac{1}{2}\right) - H_z\left(x+\frac{1}{2},y-\frac{1}{2},z,t-\frac{1}{2}\right) - H_y\left(x+\frac{1}{2},y,z+\frac{1}{2},t-\frac{1}{2}\right) + H_y\left(x+\frac{1}{2},y,z-\frac{1}{2},t-\frac{1}{2}\right)\right] \quad \text{(S51)}$$

$$D_y\left(x,y+\frac{1}{2},z,t-1\right) = D_y\left(x,y+\frac{1}{2},z,t\right) - \frac{c\Delta t}{\Delta}\left[H_x\left(x,y+\frac{1}{2},z+\frac{1}{2},t-\frac{1}{2}\right) - H_x\left(x,y+\frac{1}{2},z-\frac{1}{2},t-\frac{1}{2}\right) - H_z\left(x+\frac{1}{2},y+\frac{1}{2},z,t-\frac{1}{2}\right) + H_z\left(x-\frac{1}{2},y+\frac{1}{2},z,t-\frac{1}{2}\right)\right] \quad \text{(S52)}$$

$$D_z\left(x,y,z+\frac{1}{2},t-1\right) = D_z\left(x,y,z+\frac{1}{2},t\right) - \frac{c\Delta t}{\Delta}\left[H_y\left(x+\frac{1}{2},y,z+\frac{1}{2},t-\frac{1}{2}\right) - H_y\left(x-\frac{1}{2},y,z+\frac{1}{2},t-\frac{1}{2}\right) - H_x\left(x,y+\frac{1}{2},z+\frac{1}{2},t-\frac{1}{2}\right) + H_x\left(x,y-\frac{1}{2},z+\frac{1}{2},t-\frac{1}{2}\right)\right] \quad \text{(S53)}$$

$$H_x\left(x,y+\frac{1}{2},z+\frac{1}{2},t-\frac{1}{2}\right) = H_x\left(x,y+\frac{1}{2},z+\frac{1}{2},t+\frac{1}{2}\right) - \frac{c\Delta t}{\Delta}\left[E_y\left(x,y+\frac{1}{2},z+1,t\right) - E_y\left(x,y+\frac{1}{2},z,t\right) - E_z\left(x,y+1,z+\frac{1}{2},t\right) + E_z\left(x,y,z+\frac{1}{2},t\right)\right] \quad \text{(S54)}$$

$$H_y\left(x+\frac{1}{2},y,z+\frac{1}{2},t-\frac{1}{2}\right) = H_y\left(x+\frac{1}{2},y,z+\frac{1}{2},t+\frac{1}{2}\right) - \frac{c\Delta t}{\Delta}\left[E_z\left(x+1,y,z+\frac{1}{2},t\right) - E_z\left(x,y,z+\frac{1}{2},t\right) - E_x\left(x+\frac{1}{2},y,z+1,t\right) + E_x\left(x+\frac{1}{2},y,z,t\right)\right] \quad \text{(S55)}$$

$$H_z\left(x+\frac{1}{2},y+\frac{1}{2},z,t-\frac{1}{2}\right) = H_z\left(x+\frac{1}{2},y+\frac{1}{2},z,t+\frac{1}{2}\right) - \frac{c\Delta t}{\Delta}\left[E_x\left(x+\frac{1}{2},y+1,z,t\right) - E_x\left(x+\frac{1}{2},y,z,t\right) - E_y\left(x+1,y+\frac{1}{2},z,t\right) + E_y\left(x,y+\frac{1}{2},z,t\right)\right] \quad \text{(S56)}$$

The fields are initialized at zero for the final timestep. A single time-reversed FDTD loop

proceeds as follows:

1. Inject stored $\boldsymbol{H}$ components into the boundary pixels at the recording layer.
2. Use $\boldsymbol{H}$ values to reverse-update $\boldsymbol{D}$.
3. Inject stored $\boldsymbol{D}$ components into the boundary pixels at the recording layer.
4. Use $\boldsymbol{D}$ values to compute $\boldsymbol{E}$.
5. Use $\boldsymbol{E}$ values to reverse-update $\boldsymbol{H}$.

5.3 Gradient calculation

In this section, we detail the equations that allow the gradient backpropagation to occur under the direct differentiation framework. Specifically, we show the calculation of the derivative $dG/d\boldsymbol{F}(t-1/2)$ from $dG/d\boldsymbol{F}(t)$ for objective function G and fields $\boldsymbol{F}(t)$ at time $t > 0$ and demonstrate how the field values $\boldsymbol{F}(t)$ from the simultaneously-running reverse simulation are incorporated into the calculation. For this section, we consider an objective function that depends only on the electric fields $G[\boldsymbol{E}(t), \boldsymbol{H}(t)]$ with analytically-defined derivatives $\partial G/\partial \boldsymbol{E}(t), \partial G/\partial \boldsymbol{H}(t)$. The tunable parameters are contained in vector $\boldsymbol{p}$ and the system is nondispersive.

At the start of the gradient calculation proceed, we initialize empty total derivative fields $dG/d\boldsymbol{E}, dG/d\boldsymbol{H}$ and empty electromagnetic fields $\boldsymbol{E}, \boldsymbol{H}$. A single gradient calculation loop proceeds as follows:

1. Use $dG/d\boldsymbol{H}$ and the analytical derivative $\partial G/\partial \boldsymbol{E}$ to compute $dG/d\boldsymbol{E}$.
2. Accumulate $dG/d\boldsymbol{p}$ using $dG/d\boldsymbol{E}$ and $dG/d\boldsymbol{H}$.
3. Perform a time-reversed FDTD update.
4. Use $dG/d\boldsymbol{E}$ to compute $dG/d\boldsymbol{D}$ for one backwards time-step.

5. Use $dG/d\boldsymbol{D}$ and the analytical derivative $\partial G/\partial \boldsymbol{H}$ to compute $dG/d\boldsymbol{H}$ for one backwards time-step.

The explicit equations are listed below. Equation components in blue are provided by the user as they change based on the form of the objective function. Other equation components in black do not need to be changed as they reflect the internal FDTD calculation functional dependence. The derivatives can be obtained by differentiation of the FDTD update equations in Equations S42-S44. The Courant number is $S_c \equiv c\Delta t/\Delta$.

The $dG/d\boldsymbol{E}$ time-reversed update equations are:

$$\frac{dG}{dE_x\left(x+\frac{1}{2},y,z,t\right)} = S_c\left[\frac{dG}{dH_y\left(x+\frac{1}{2},y,z+\frac{1}{2},t+\frac{1}{2}\right)} - \frac{dG}{dH_y\left(x+\frac{1}{2},y,z-\frac{1}{2},t+\frac{1}{2}\right)} - \frac{dG}{dH_z\left(x+\frac{1}{2},y+\frac{1}{2},z,t+\frac{1}{2}\right)} + \frac{dG}{dH_z\left(x+\frac{1}{2},y-\frac{1}{2},z,t+\frac{1}{2}\right)}\right] + \frac{\partial G}{\partial E_x\left(x+\frac{1}{2},y,z,t\right)} \tag{S57}$$

$$\frac{dG}{dE_y\left(x,y+\frac{1}{2},z,t\right)} = S_c\left[\frac{dG}{dH_z\left(x+\frac{1}{2},y+\frac{1}{2},z,t+\frac{1}{2}\right)} - \frac{dG}{dH_z\left(x-\frac{1}{2},y+\frac{1}{2},z,t+\frac{1}{2}\right)} - \frac{dG}{dH_x\left(x,y+\frac{1}{2},z+\frac{1}{2},t+\frac{1}{2}\right)} + \frac{dG}{dH_x\left(x,y+\frac{1}{2},z-\frac{1}{2},t+\frac{1}{2}\right)}\right] + \frac{\partial G}{\partial E_y\left(x,y+\frac{1}{2},z,t\right)} \tag{S58}$$

$$\frac{dG}{dE_z\left(x,y,z+\frac{1}{2},t\right)} = S_c\left[\frac{dG}{dH_x\left(x,y+\frac{1}{2},z+\frac{1}{2},t+\frac{1}{2}\right)} - \frac{dG}{dH_x\left(x,y-\frac{1}{2},z+\frac{1}{2},t+\frac{1}{2}\right)} - \frac{dG}{dH_y\left(x+\frac{1}{2},y,z+\frac{1}{2},t+\frac{1}{2}\right)} + \frac{dG}{dH_y\left(x-\frac{1}{2},y,z+\frac{1}{2},t+\frac{1}{2}\right)}\right] + \frac{\partial G}{\partial E_z\left(x,y,z+\frac{1}{2},t\right)} \tag{S59}$$

The form of the $dG/d\boldsymbol{p}$ calculation varies based on the tunable parameter of choice. We

exhibit one such calculation here for the pixel-wise isotropic dielectric permittivity $\epsilon(x, y, z)$.

$$\frac{dG}{d\epsilon(x,y,z)} = -\frac{1}{\epsilon(x,y,z)^2}\sum_t \left[\frac{dG}{dE_x\left(x+\frac{1}{2},y,z,t\right)}D_x\left(x+\frac{1}{2},y,z,t\right) + \frac{dG}{dE_y\left(x,y+\frac{1}{2},z,t\right)}D_y\left(x,y+\frac{1}{2},z,t\right) + \frac{dG}{dE_z\left(x,y,z+\frac{1}{2},t\right)}D_z\left(x,y,z+\frac{1}{2},t\right)\right] \tag{S60}$$

Since $dG/d\epsilon(x, y, z)$ is a sum of contributions over all timesteps, this sum can be accumulated in every gradient calculation loop.

The $dG/d\boldsymbol{D}$ calculation equations are:

$$\frac{dG}{dD_x\left(x+\frac{1}{2},y,z,t\right)} = \frac{1}{\epsilon(x,y,z)}\frac{dG}{dE_x\left(x+\frac{1}{2},y,z,t\right)} + \frac{dG}{dD_x\left(x+\frac{1}{2},y,z,t+1\right)} \tag{S61}$$

$$\frac{dG}{dD_y\left(x,y+\frac{1}{2},z,t\right)} = \frac{1}{\epsilon(x,y,z)}\frac{dG}{dE_y\left(x,y+\frac{1}{2},z,t\right)} + \frac{dG}{dD_y\left(x,y+\frac{1}{2},z,t+1\right)} \tag{S62}$$

$$\frac{dG}{dD_z\left(x,y,z+\frac{1}{2},t\right)} = \frac{1}{\epsilon(x,y,z)}\frac{dG}{dE_z\left(x,y,z+\frac{1}{2},t\right)} + \frac{dG}{dD_z\left(x,y,z+\frac{1}{2},t+1\right)} \tag{S63}$$

The $dG/d\boldsymbol{H}$ time-reversed update equations are:

$$\frac{dG}{dH_x\left(x, y+\frac{1}{2}, z+\frac{1}{2}, t-\frac{1}{2}\right)} = S_c \left[\frac{dG}{dD_y\left(x, y+\frac{1}{2}, z, t\right)} - \frac{dG}{dD_y\left(x, y+\frac{1}{2}, z+1, t\right)} - \frac{dG}{dD_z\left(x, y, z+\frac{1}{2}, t\right)} + \frac{dG}{dD_z\left(x, y+1, z+\frac{1}{2}, t\right)} \right] + \frac{dG}{dH_x\left(x, y+\frac{1}{2}, z+\frac{1}{2}, t+\frac{1}{2}\right)} + \frac{\partial G}{\partial H_x\left(x, y+\frac{1}{2}, z+\frac{1}{2}, t-\frac{1}{2}\right)} \tag{S64}$$

$$\frac{dG}{dH_y\left(x+\frac{1}{2}, y, z+\frac{1}{2}, t-\frac{1}{2}\right)} = S_c \left[\frac{dG}{dD_z\left(x, y, z+\frac{1}{2}, t\right)} - \frac{dG}{dD_z\left(x+1, y, z+\frac{1}{2}, t\right)} - \frac{dG}{dD_x\left(x+\frac{1}{2}, y, z, t\right)} + \frac{dG}{dD_x\left(x+\frac{1}{2}, y, z+1, t\right)} \right] + \frac{dG}{dH_y\left(x+\frac{1}{2}, y, z+\frac{1}{2}, t+\frac{1}{2}\right)} + \frac{\partial G}{\partial H_y\left(x+\frac{1}{2}, y, z+\frac{1}{2}, t-\frac{1}{2}\right)} \tag{S65}$$

$$\frac{dG}{dH_z\left(x+\frac{1}{2}, y+\frac{1}{2}, z, t-\frac{1}{2}\right)} = S_c \left[\frac{dG}{dD_x\left(x+\frac{1}{2}, y, z, t\right)} - \frac{dG}{dD_x\left(x+\frac{1}{2}, y+1, z, t\right)} - \frac{dG}{dD_y\left(x, y+\frac{1}{2}, z, t\right)} + \frac{dG}{dD_y\left(x+1, y+\frac{1}{2}, z, t\right)} \right] + \frac{dG}{dH_z\left(x+\frac{1}{2}, y+\frac{1}{2}, z, t+\frac{1}{2}\right)} + \frac{\partial G}{\partial H_z\left(x+\frac{1}{2}, y+\frac{1}{2}, z, t-\frac{1}{2}\right)} \tag{S66}$$

6 Recording layer implementation

The recording layer serves as a simulation region boundary that records the electromagnetic field values in time during the forward pass, then plays back the field values during the time-reversal simulation. Specifically, the recording layer comprises a discrete set of connected pixels $(i, j, k) \in \partial$ that forms a closed boundary. The pixels at which the topological optimization is performed (pixels where the shape derivatives are required) must be enclosed within the interior of ∂. The electromagnetic $\boldsymbol{D}$ and $\boldsymbol{H}$ fields are stored as a function of

time during the forward sweep. Due to the half-unit displaced positions within the FDTD Yee Grid scheme, for the pixel spanning the cubic cell $[i, i+1] \times [j, j+1] \times [k, k+1]$, we store the following:

$$D_x(i+1/2, j, k, t), D_y(i, j+1/2, k, t), D_z(i, j, k+1/2, t), \tag{S67}$$

$$H_x(i, j+1/2, k+1/2, t+1/2), H_y(i+1/2, j, k+1/2, t+1/2),$$
$$H_z(i+1/2, j+1/2, k, t+1/2) \tag{S68}$$

For the time-reversal simulation that runs in parallel with the back-propagation step, the simulation region is restricted to the interior of ∂. For cases in which ∂ comprises multiple non-intersecting recording boundaries, which we did not perform in this study, one may divide the disjoint interiors into multiple independent simulation regions and time-reverse them independently and in parallel with multi-threading. There are no additional boundary conditions (*e.g.,* PMLs) required during the time-reversal simulation since the recording layer forms a closed boundary around the time-reversal simulation region.

The recording boundary can enclose a smaller subset of the simulation volume than the lossy boundaries, allowing the reverse time-stepping simulation to be performed over a smaller volume and thus consuming even less memory and computation time. Although this smaller volume does not need to enclose the FDTD field sources (since the information is already encoded on the recording boundary), the recording boundary must enclose the structure vector $\boldsymbol{p}$ locations to preserve the updating of the gradient $dG/d\boldsymbol{p}$ there, as would be the case in AVM as well.

In first order FDTD, only the pixels immediately adjacent to the pixel to be updated are involved in the update equation. When pixels adjacent to the recording layer are time-stepped in reverse, they extract the requisite field values from the recording layer to be substituted in the update equation.

7 Validation against commercial FDTD software suite

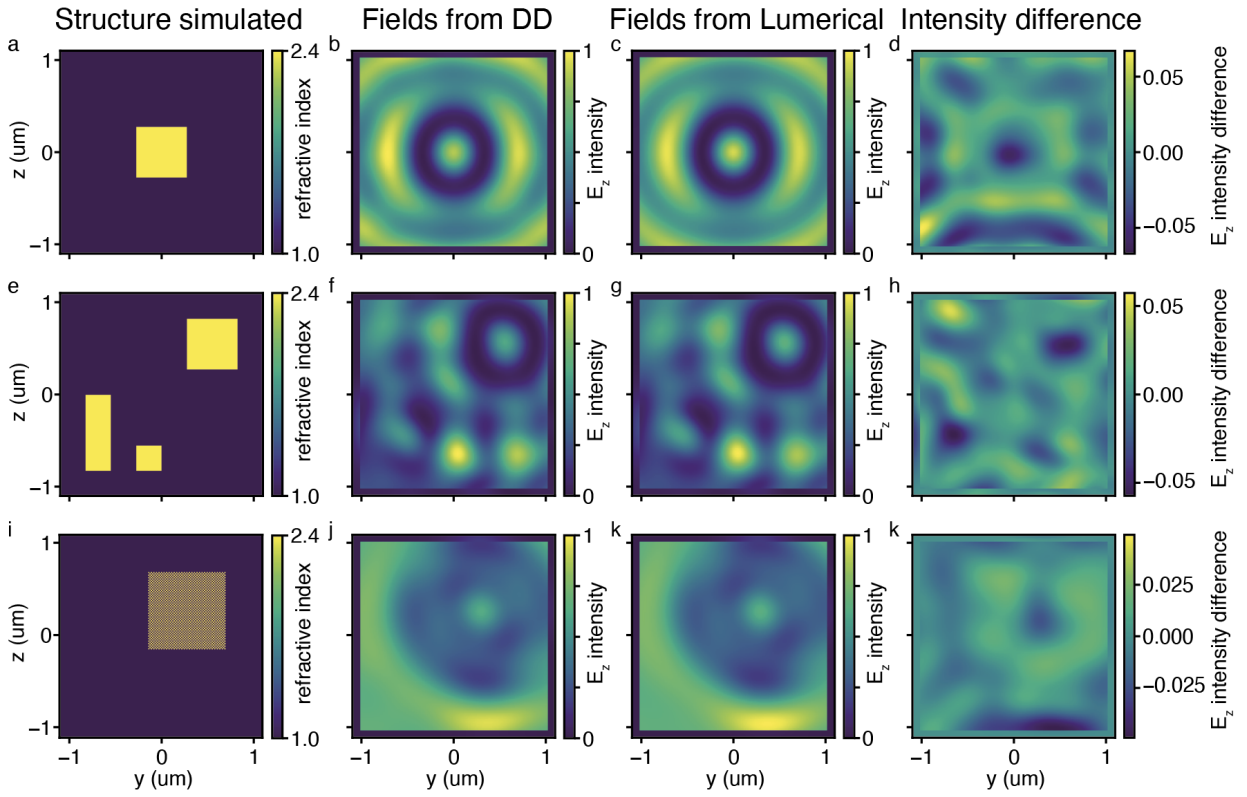

Figure S1: Validation of the accuracy of DD FDTD field calculations against that of a commercial FDTD suite (Ansys Lumerical). Three pillar structures are simulated (first column: **a**, **e**, and **i**) and the transmitted field intensities in the yz plane using the DD FDTD and commercial FDTD are plotted in the second (**b**, **f**, **j**) and third (**c**, **g**, **k**) columns. The fourth column (**d**, **h**, **k**) plots the difference in intensity obtained between the two simulation platforms.

We validated the DD FDTD platform results against those obtained from a commercial FDTD software suite, Lumerical FDTD 2021 R2 (Ansys Inc.) (Supplementary Figure S1). The commercial FDTD is used to simulate the identical geometry, pixel size, and Courant number as in Figure 2a in the main text for different pillar configurations. As in the DD FDTD, a Total Field Scattered Field (TFSF) source and boundary is employed and the FDTD boundary conditions are all PMLs. For the commercial FDTD, the simulation time is set to 50 fs with an automatic shutoff threshold of 5×10^{-12} as the energy fraction below which the simulation

does not end before the full time elapses. A staircasing mesh method is used where the permittivity mesh coincides with the spatial Cartesian mesh. The electromagnetic field intensity is recorded at the same monitor plane as in Figure 2a. Three different pillar configurations are simulated: a single square pillar, three non-identical rectangular pillars, and alternating air/material pixels in a checkerboard pattern, corresponding to each row of Supplementary Figure S1, respectively. The normalized cross-correlation between the DD FDTD intensity pattern and the commercial FDTD intensity pattern is 0.997, 0.997 and 0.999 for the three pillar configurations, respectively, indicating strong numerical agreement. For the single square pillar, the zeroth order transmission phase is -0.023 rad for DD FDTD and -0.027 rad for the commercial FDTD. For the three rectangular pillars, the transmission phase is 0.160 rad for DD FDTD and 0.167 rad for the commercial FDTD. For the checkerboard pattern, the transmission phase is 0.129 rad for DD FDTD and 0.128 rad for the commercial FDTD. Thus, the transmission phase deviation obtained from the two simulators is on the order of $10^{-3}\pi$. The slight deviation between the two simulators is likely due to the different PML implementation.

8 Adjoint method derivation for field phase

Here, we derive the adjoint equations for optimization systems in which the objective function is written in terms of the phase of field values. Consider the system represented by Figure 2a in the main text, in which the objective function $G[\boldsymbol{E}, \boldsymbol{H}] = \arg \sum_\Omega E_z$ is the phase of the average z-directed transverse electric field over a domain Ω. We derive the adjoint system using the Lorentz reciprocity approach since the objective function is written in the frequency-domain.[2,3] We consider the shift in the objective function dG due to a small change $\delta E(\boldsymbol{x}_m)$ in the z-directed electric field E at a position $\boldsymbol{x}_m \in \Omega$. All summations are

performed over Ω and the z-subscript for the electric field is suppressed for concision.

$$dG = \text{atan2}\left[Im(\Sigma E + \delta E(\boldsymbol{x}_m)), Re(\Sigma E + \delta E(\boldsymbol{x}_m))\right] - \text{atan2}[Im(\Sigma E), Re(\Sigma E)] \tag{S69}$$

$$= \text{atan2}\left[Im(\Sigma E) + Im(\delta E(\boldsymbol{x}_m)), Re(\Sigma E) + Re(\delta E(\boldsymbol{x}_m))\right] - \text{atan2}[Im(\Sigma E), Re(\Sigma E)] \tag{S70}$$

The derivatives of atan2 with respect to each of its arguments is:

$$\frac{d(\text{atan2}(y,x))}{dx} = -\frac{y}{x^2+y^2} \tag{S71}$$

$$\frac{d(\text{atan2}(y,x))}{dy} = \frac{x}{x^2+y^2} \tag{S72}$$

Thus, linearizing the objective function change in $\delta E(\boldsymbol{x}_m)$,

$$dG = \frac{Re(\Sigma E)}{[Re(\Sigma E)]^2 + [Im(\Sigma E)]^2} Im[\delta E(\boldsymbol{x}_m)] - \frac{Im(\Sigma E)}{[Re(\Sigma E)]^2 + [Im(\Sigma E)]^2} Re[\delta E(\boldsymbol{x}_m)] \tag{S73}$$

$$= \frac{Re(\Sigma E) Im[\delta E(\boldsymbol{x}_m)] - Im(\Sigma E) Re[\delta E(\boldsymbol{x}_m)]}{|\Sigma E|^2} \tag{S74}$$

The induced field at $\boldsymbol{x}_m$ due to an induced dipole moment $p^{ind}(\boldsymbol{x}') = \delta\epsilon\Delta V E(\boldsymbol{x}')$ at $\boldsymbol{x}'$ (the pixels to be optimized) is $\delta E(\boldsymbol{x}_m) = Gr(\boldsymbol{x}_m, \boldsymbol{x}') p^{ind}(\boldsymbol{x}')$, where $Gr(\boldsymbol{x}_m, \boldsymbol{x}')$ is the Green's function. Substituting,

$$dG = \frac{Re(\Sigma E) Im[Gr(\boldsymbol{x}_m, \boldsymbol{x}') p^{ind}(\boldsymbol{x}')] - Im(\Sigma E) Re[Gr(\boldsymbol{x}_m, \boldsymbol{x}') p^{ind}(\boldsymbol{x}')]}{|\Sigma E|^2} \tag{S75}$$

$$= \frac{Im[Re(\Sigma E) Gr(\boldsymbol{x}_m, \boldsymbol{x}') p^{ind}(\boldsymbol{x}')] - Re[Im(\Sigma E) Gr(\boldsymbol{x}_m, \boldsymbol{x}') p^{ind}(\boldsymbol{x}')]}{|\Sigma E|^2} \tag{S76}$$

Using the Lorentz symmetry of the Green's function, $Gr(\boldsymbol{x}_m, \boldsymbol{x}') = Gr(\boldsymbol{x}', \boldsymbol{x}_m)$,

$$dG = \frac{Im[Gr(\boldsymbol{x}', \boldsymbol{x}_m) Re(\Sigma E) p^{ind}(\boldsymbol{x}')] - Re[Gr(\boldsymbol{x}', \boldsymbol{x}_m) Im(\Sigma E) p^{ind}(\boldsymbol{x}')]}{|\Sigma E|^2} \tag{S77}$$

Defining the adjoint fields for a single dipole $E_m^{A,r} = Gr(\boldsymbol{x}', \boldsymbol{x}_m) Re(\Sigma E)/|\Sigma E|^2, E_m^{A,i} =$

$Gr(\boldsymbol{x}', \boldsymbol{x}_m) Im(\Sigma E)/|\Sigma E|^2$, we can rewrite the objective function shift as:

$$dG = Im[E_m^{A,r}(\boldsymbol{x}')p^{ind}(\boldsymbol{x}')] - Re[E_m^{A,i}(\boldsymbol{x}')p^{ind}(\boldsymbol{x}')] \tag{S78}$$

$$= Im[E_m^{A,r}(\boldsymbol{x}')p^{ind}(\boldsymbol{x}')] - Im[iE_m^{A,i}(\boldsymbol{x}')p^{ind}(\boldsymbol{x}')] \tag{S79}$$

$$= Im\{[E_m^{A,r}(\boldsymbol{x}') - iE_m^{A,i}(\boldsymbol{x}')]p^{ind}(\boldsymbol{x}')]\} \tag{S80}$$

Observe that we can add the adjoint field components as:

$$E_m^A(\boldsymbol{x}') \equiv E_m^{A,r}(\boldsymbol{x}') - iE_m^{A,i}(\boldsymbol{x}') \tag{S81}$$

$$= Gr(\boldsymbol{x}', \boldsymbol{x}_m)\left[\frac{Re(\Sigma E)}{|\Sigma E|^2} - i\frac{Im(\Sigma E)}{|\Sigma E|^2}\right] \tag{S82}$$

$$= Gr(\boldsymbol{x}', \boldsymbol{x}_m)\left[\frac{(\Sigma E)^*}{|\Sigma E|^2}\right] \tag{S83}$$

where the asterisk indicates complex conjugation. Thus, the functional change is:

$$dG = Im\left[E_m^A(\boldsymbol{x}')p^{ind}(\boldsymbol{x}')\right] \tag{S84}$$

This is the shift in the objective function just due to a single field point on the monitor. We will have one such shift for every monitor point and can run these simultaneously since their field contributions add linearly. Thus, we have:

$$dG = \sum_{m\in\Omega} Im[E_m^A(\boldsymbol{x}')p^{ind}(\boldsymbol{x}')] \tag{S85}$$

$$= Im\left\{\left[\sum_{m\in\Omega} E_m^A(\boldsymbol{x}')\right]p^{ind}(\boldsymbol{x}')\right\} \tag{S86}$$

$$= \delta\epsilon\Delta V Im[E^A(\boldsymbol{x}')E(\boldsymbol{x}')], \quad E^A(\boldsymbol{x}') \equiv \sum_{m\in\Omega} E_m^A(\boldsymbol{x}') \tag{S87}$$

summing over all points m on the monitor plane. The implementation of the adjoint calculation is as follows:

1. Simulate a plane wave incident on the nanostructures from the glass side and record the electric field at the pillars $E(\boldsymbol{x}')$ and on the monitor plane $E(\boldsymbol{x}_m)$.

2. For each point on the monitor plane, place a dipole of amplitude $(\sum E)^*/|\sum E|^2$.

3. Run the simulation again without the incident plane wave and record the total adjoint field $E^A(\boldsymbol{x}')$ at the nanostructure locations.

4. Compute the change in figure of merit using $dG = d\phi = \delta\epsilon\Delta V Im[E^A(\boldsymbol{x}')E(\boldsymbol{x}')]$.

9 Design of color sorter

The full 3D FDTD region for the passive color sorter has a geometry similar to that of Figure 2a, with an increased $x \times y \times z$ extent of 184 × 50 × 70 pixels (2530 nm × 687.5 nm × 962.5 nm) excluding the ten PML layers surrounding all six boundary surfaces. The TFSF region has an extent of 152 × 38 × 68 pixels (2090 nm × 522.5 nm × 935 nm) so that its x extents are 6 pixels away from the PML boundaries in the x direction. The substrate is glass with a fixed refractive index of 1.44. The structure to be optimized is a compact 30 × 60 array of pillars on the yz plane. Each pillar has an $x \times y \times z$ dimension of 72 × 1 × 1 pixels (990 nm × 13.75 nm × 13.75 nm) so that the full array has an extent of 72 × 30 × 60 pixels (990 nm × 412.5 nm × 825 nm). The optimization tunable parameters are the permittivities of these 30 × 60 = 1800 pillars. To avoid optimizing with parameter bounds, we choose a latent space for the normalized permittivity p to be on the full real line, then map the real line monotonically to a bounded relative permittivity ϵ_r range using the hyperbolic tangent function. To account for material dispersion, we let the relative permittivity bounds correspond to that of real materials: air and TiO_2. Thus, for a pillar with normalized permittivity p, its relative permittivities at the two wavelengths of interest

λ_1, λ_2 are:

$$\epsilon_r(p, \lambda_1) = \epsilon_{r,TiO_2}(\lambda_1)\frac{1}{2}\left(\tanh\frac{p}{2}+1\right) + \epsilon_{r,air}(\lambda_1)\left[1-\frac{1}{2}\left(\tanh\frac{p}{2}+1\right)\right] \tag{S88}$$

$$\epsilon_r(p, \lambda_2) = \epsilon_{r,TiO_2}(\lambda_2)\frac{1}{2}\left(\tanh\frac{p}{2}+1\right) + \epsilon_{r,air}(\lambda_2)\left[1-\frac{1}{2}\left(\tanh\frac{p}{2}+1\right)\right] \tag{S89}$$

The numerical values used are $\epsilon_{r,TiO_2}(488\text{ nm}) = 2.7312^2$ and $\epsilon_(r, TiO_2)(633\text{ nm}) = 2.3893^2$. This method of incorporating material dispersion is not unique and more complex versions which incorporate analytic approximations can be used as well. The device is illuminated by a z-polarized plane wave propagating in the x-direction. We define the objective function $G_{1,2}$ to be maximized for each wavelength to be the overlap between the transverse intensities $|E_z|^2(y, z)$ recorded at a 30×60 pixel (412.5 nm $\times$ 825 nm) monitor plane placed 50 pixels (687.5 nm) above the pillars and a desired intensity profile. We use $|E_z|^2$ as a proxy for the total intensity because the incident polarization is z-directed and the total transmitted intensity is dominated by the $|E_z|^2$ component. We pick the desired intensity profile for each wavelength to be gaussians with different center positions $(y_{1,2}, z_{1,2})$ and width $W = 8$ pixels (110 nm).

$$G_{1,2} = \sum_{n_y=1}^{30}\sum_{n_z=1}^{30}|E_z(n_y\Delta y, n_z\Delta y)|^2 \exp\left[-\frac{(n_y\Delta y - y_{1,2})^2 + (n_z\Delta z - z_{1,2})^2}{2W^2}\right] \tag{S90}$$

The gaussian center positions and widths are plotted over the optimized intensity profile in Supplementary Figure S2. The total objective function to be maximized for the system G is the minimum of the two individual wavelength objective functions $G = \min(G_1, G_2)$.

The gradient of G with respect to each of the 1800 permittivity values is computed using DD and the optimization of G is performed using simple gradient descent with a constant learning rate. The objective function for each of $G_{1,2}$ a function of iteration is plotted in Supplementary Figure S2b. The starting permittivity profile (in the normalized latent space units) is chosen to be uniformly 0 on the left and a random uniform distribution from 0 to

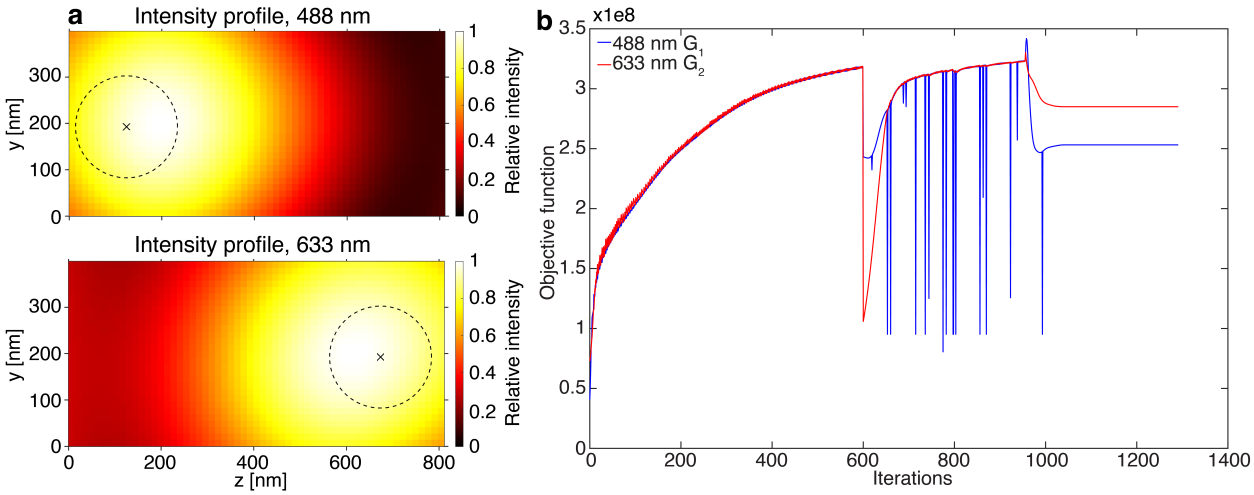

Figure S2: Optimization details for color sorter. **a** Location of the target gaussian intensity profiles (centered on the black crosses) and widths (dotted line), overlaid on the converged intensity profiles at each wavelength after optimization. **b** Variation of objective function values with iteration number for the individual wavelength objective functions $G_{1,2}$. A Gaussian smoothing filter is applied to the right hand side of the permittivity profile after iteration 599, resulting in a transient decrease in the objective function. The binarization step was started at iteration 956.

5 inclusive on the right. This starting configuration was chosen as the optimized structure significantly outperformed that of other starting configurations (uniform distribution, two halves with uniform distributions). In order to reduce the grainy permittivity pattern, a Gaussian kernel of five pixel width was applied to the right hand side of the latent space permittivity pattern after step 599 of optimization.

To obtain a final binary structure, we applied a binarization push from iteration step 956. This takes the form of multiplying the latent space vector by a constant 1.1 after every update step. Since negative latent space numbers are closer to the air refractive index and positive latent space numbers are closer to the TiO_2 refractive index, this multiplication has the effect of pushing the structure towards the refractive index bounds. This produces a slight reduction in deflection performance.

We validate the performance of the inverse-designed device by simulating it in a commercial FDTD suite (Ansys Lumerical 2023 R1.1) in addition to the DD FDTD. A staircasing mesh method is used where the permittivity mesh coincides with the spatial Cartesian mesh.

The commercial FDTD simulation is run for 100 fs without an automatic early shutoff. The transmitted intensity plots in Figure 3c and Supplementary Figure S2a are of the commercial FDTD electric field profiles.

10 Design of resonator array for imposing group delay

To simulate infinitely periodic arrays in the transverse dimension, we introduce periodic boundary conditions in the transverse direction instead of TFSF boundary conditions. The boundary conditions in the x propagation direction remain as 10 layer thick PMLs. The pixel size for this simulation is 20 nm, the Courant number is kept at 0.9, and the total simulation $x \times y \times z$ dimensions are $128 \times 25 \times 25$ pixels ($2560 \times 500 \times 500$ nm), including the PMLs in the x-direction. The tunable parameters are the 9900 dielectric permittivities in a $44 \times 15 \times 15$ pixel block ($880 \times 300 \times 300$ nm) centered in the yz plane. The dielectric permittivities are parametrized in the same latent space on the infinite real line as in the previous sections. There is no glass substrate in this simulation and the structure is illuminated by a z-polarized transversely uniform Gaussian pulse located 25 pixels (500 nm) behind the structure, propagating in the $+x$ direction, with the form:

$$E_z(t) = \sin(\omega_0 t) \exp\left[-\left(\frac{t - t_0}{\tau}\right)^2\right] \tag{S91}$$

where $\omega_0 = 2\pi(564 \text{ THz})$ is the center frequency corresponding to a vacuum wavelength of 532 nm, t_0 is a time displacement of 20 fs, and τ is a temporal width of 8 fs.

The objective function for this calculation is the transmitted z-polarized electric field summed over the full transverse cross-section A of the simulation, which can be related to the far-field on-axis electric field. Specifically, the objective function to be maximized is:

$$G = \sum_{t=62.4fs}^{t=64.8fs} \int_A E_z(\boldsymbol{x}, t) d^2\boldsymbol{x} \tag{S92}$$

where the cross-section A is located 25 pixels (500 nm) away from the top of the structure.

The objective function represents the total transmitted electric field between timesteps 1800 and 1870 of the simulation, which is about 38.7 fs delayed from the envelope peak of the illuminating pulse had there been no structure present. This delay is greater than the temporal delay of $(n_{max} - 1)L/c = 4.1$ fs that can be attained by simply replacing a slab of $n = 1$ air with thickness L with a uniform slab of $n_{max} = 2.404$, thereby requiring the structure to exploit resonances in order to perform the group delay.

The initial permittivities in the structure block are uniformly sampled over -10 and 10 in the permittivity latent space and the gradient descent is performed with a fixed step size. To remove isolated pixels and force filled pixels to cluster together in space, we apply a 3D Gaussian blur on the 3D latent space distribution at various points in the optimization, which has the effect of penalizing isolated pixel structures. This blur is imposed at step 7 (Gaussian width 2 pixels) and every 15 iterations (Gaussian width 1 pixel) between steps 439 and 544. From iteration 545 onwards, no Gaussian blurring is implemented and the device is allowed to converge. The optimization is terminated at iteration 602. Each optimization step takes 17 seconds using two CPU cores on an Intel E5-2690 v2 processor (3.00 GHz base frequency, 3.60 GHz turbo frequency). The objective value as a function of iteration is plotted in Supplementary Figure S3.

To evaluate the performance of the optimized device, the simulation is run for 30000 timesteps (1040 fs) and the fields at the cross-sectional plane A are recorded for the system with and without the optimized device. The time-domain fields are plotted in Figure 4c, which demonstrate that the device successfully maximized the electric field values in the desired shaded time range by delaying the signal envelope. The complex mean electric fields for each situation are obtained by fast Fourier transformation of the averaged E_z fields across A and the complex transmission coefficient of the optimized devices is obtained by taking the ratio of these complex electric fields. Figure 4d-e plots the transmission intensity (squared absolute value of the amplitude) and unwrapped transmission phase as a function

of frequency for the optimized device, respectively. As expected, the transmission phase exhibits a linear decrease across the center frequency of 564 THz, indicating that the device implements a group delay on the illuminating pulse. To quantify this group delay, we fit the transmission phase to a quadratic polynomial for a range of ±50 THz about the center frequency, for which the group delay (GD) and group delay dispersion (GDD) parameters can be estimated using the fitted polynomial coefficients:

$$\phi(\omega) = \phi_0 + GD(\omega - \omega_0) + \frac{1}{2}GDD(\omega - \omega_0)^2 \tag{S93}$$

such that:

$$GD = \left.\frac{\partial \phi}{\partial \omega}\right|_{\omega=\omega_0}, \quad GDD = \left.\frac{\partial^2 \phi}{\partial \omega^2}\right|_{\omega=\omega_0} \tag{S94}$$

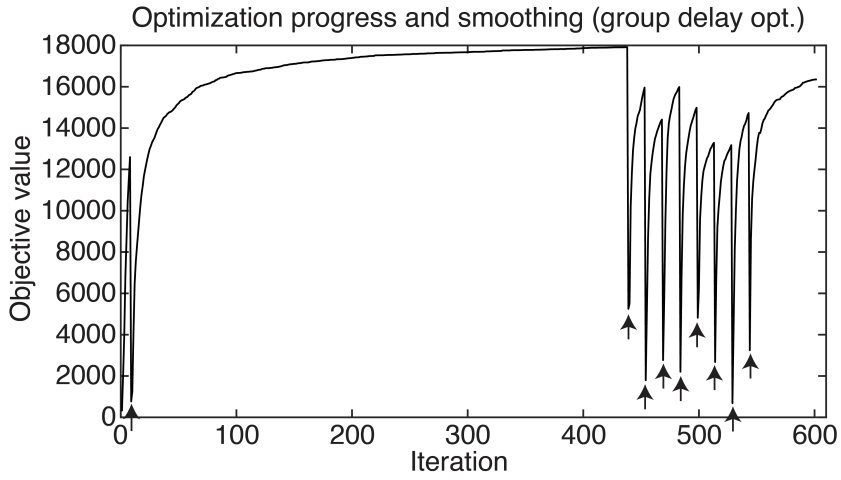

Figure S3: Optimization progress for time-domain resonator inverse design. To remove isolated pixels and improve fabrication robustness, 3D gaussian smoothing is performed over the device permittivity distribution in the permittivity latent space. The iterations at which the smoothing is performed is indicated with an arrow. The gaussian width used is 2 pixels for the first smoothing iteration and 1 pixel for subsequent smoothing implementations.

References

(1) Li, Z.; Pestourie, R.; Lin, Z.; Johnson, S. G.; Capasso, F. Empowering Metasurfaces with Inverse Design: Principles and Applications. *ACS Photonics* **2022**, *9*, 2178–2192.

(2) Lalau-Keraly, C. M.; Bhargava, S.; Miller, O. D.; Yablonovitch, E. Adjoint shape optimization applied to electromagnetic design. *Optics Express* **2013**, *21*, 21693.

(3) Miller, O. D. Photonic Design: From Fundamental Solar Cell Physics to Computational Inverse Design. Ph.D. thesis, University of California, Berkeley, 2013.

(4) Molesky, S.; Lin, Z.; Piggott, A. Y.; Jin, W.; Vucković, J.; Rodriguez, A. W. Inverse design in nanophotonics. *Nature Photonics* **2018**, *12*, 659–670.

(5) Yunpeng Song,; Nikolova, N. Memory-Efficient Method for Wideband Self-Adjoint Sensitivity Analysis. *IEEE Transactions on Microwave Theory and Techniques* **2008**, *56*, 1917–1927.

(6) Sullivan, D. M. *Electromagnetic simulation using the FDTD method*; IEEE Press, 2000.

(7) Griewank, A. On Automatic Differentiation. *Mathematical Programming: Recent Developments and Applications* **1989**, 83–108.