MIT Open Access Articles

High-performance hybrid time/frequency-domain topology optimization for large-scale photonics inverse design

The MIT Faculty has made this article openly available. Please share how this access benefits you. Your story matters.

Citation: Alec M. Hammond, Ardavan Oskooi, Mo Chen, Zin Lin, Steven G. Johnson, and Stephen E. Ralph, "High-performance hybrid time/frequency-domain topology optimization for large-scale photonics inverse design," Opt. Express 30, 4467-4491 (2022)

As Published: 10.1364/oe.442074

Publisher: The Optical Society

Persistent URL: https://hdl.handle.net/1721.1/139804

Version: Final published version: final published article, as it appeared in a journal, conference proceedings, or other formally published context

Terms of Use: Article is made available in accordance with the publisher’s policy and may be subject to US copyright law. Please refer to the publisher’s site for terms of use.
High-performance hybrid time/frequency-domain topology optimization for large-scale photonics inverse design

ALEC M. HAMMOND,1 ARDAVAN OSKOOI,2† MO CHEN,3 ZIN LIN,3 STEVEN G. JOHNSON,3 AND STEPHEN E. RALPH1*

1School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30308, USA
2Simpetus, LLC, San Francisco, CA 94109, USA
3Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA
†now at X, the Moonshot Factory, Mountain View, CA 94043, USA
*stephen.ralph@ece.gatech.edu

Abstract: We present a photonics topology optimization (TO) package capable of addressing a wide range of practical photonics design problems, incorporating robustness and manufacturing constraints, which can scale to large devices and massive parallelism. We employ a hybrid algorithm that builds on a mature time-domain (FDTD) package Meep to simultaneously solve multiple frequency-domain TO problems over a broad bandwidth. This time/frequency-domain approach is enhanced by new filter-design sources for the gradient calculation and new material-interpolation methods for optimizing dispersive media, as well as by multiple forms of computational parallelism. The package is available as free/open-source software with extensive tutorials and multi-platform support.

© 2022 Optical Society of America under the terms of the OSA Open Access Publishing Agreement

1. Introduction

Inverse design is a powerful class of methodologies for photonics design in which large-scale optimization is applied to maximize performance over thousands or even millions of degrees of freedom (DOF), often yielding non-intuitive geometries with unprecedented performance [1]. A particularly flexible form of inverse design is density-based topology optimization (TO), which allows each “pixel” to continuously evolve between two (or more) materials [2–10]. The full potential of TO is only unlocked, however, when it is applied to larger and more complex devices integrating multiple simultaneous functionalities, such as broad-bandwidth and/or large-area semiconductor-foundry devices [11–14], that are beyond the reach of intuitive design. Hence, there is a need for algorithms and software tools that expose the full flexibility of TO to scalable parallel-computing platforms while retaining a high-level user interface.

We present a free/open-source photonics TO package that addresses this need, consolidating many TO features individually explored throughout the literature, with enhancements that allow our methodology to address a wide range of practical photonics design problems and scale to arbitrarily large devices. Our approach employs a hybrid time/frequency-domain adjoint-variable method [15, Sec. 8.7] to efficiently and simultaneously compute gradients at multiple frequencies for any number of figures of merit (FOM) and/or design fields. We solve the forward and adjoint problems in the time domain using Meep, a popular, mature, and multi-featured free/open-source FDTD package that supports distributed-memory parallelism [16], combined with automatic differentiation (AD) [17–19] of arbitrary user-defined photonics-optimization objectives in a flexible Python interface. Since storage requirements for pure time-domain adjoint methods typically do not scale well with problem size [20], we leverage a hybrid adjoint formulation [21–24] that computes the forward and adjoint fields in the frequency domain, over an arbitrary bandwidth, using a discrete-time Fourier transform (DTFT). This methodology
merges the broad-bandwidth parallelism, reliability, and speed of time-domain solvers with the flexibility and simplicity of frequency-domain adjoint methods.

Our work brings together several different features from other photonics inverse-design packages including techniques for reducing the computational cost of large design regions (Sec. 4.2), arbitrary geometric parameterizations (Sec. 3), and a broad range of material definitions including dispersion and anisotropy (Sec. 2.1). For example, we formulate the adjoint problem for near-to-far field transformations, mirror and rotational symmetries, as well as cylindrical coordinates, combining them within a single unified software package. Optimization figures of merit (objective/loss functions) are easy to define as their gradient is computed using AD (Sec. 4). This same pipeline allows for arbitrary sequences of filtering and projection steps, which then map the design parameters to multiple, distinct “design regions” (e.g. for different foundry-fabrication layers). Similarly, common design constraints like minimum area, enclosed area, linewidth, linespacing, and curvature are all supported (Sec. 3.1) [25], while support for connectivity constraints [26] is forthcoming.

In addition, we present novel signal-processing and optimization techniques to consolidate many of the above features within a single environment. For example, broadband adjoint sources computed in the frequency domain must be efficiently transformed to an equivalent time-limited/band-limited time-domain source, a task for which a straightforward inverse-Fourier-transform approach is restricted by expensive storage requirements (proportional to the number of timesteps multiplied by the number of spatial points) [24]. To overcome this, we introduce a fitting routine using standard digital signal processing (DSP) window functions [27], yielding low storage requirements without sacrificing bandwidth or figure-of-merit (FOM) complexity (Sec. 5.2)—a solution closely related to filter-design methods that fit a desired frequency response to a finite impulse-response (FIR) filter [24, 28, 29]. Similarly, continuous material interpolation (a fundamental part of density-based TO methods [30, 31]) quickly becomes problematic for dispersive and lossy materials (esp. metals) as arbitrary shifting of their responses (e.g. refractive index “zero crossings” [32]) may induce field instabilities or problems with optimization. We present a material-interpolation scheme that adds broadband gradient support for anisotropic and dispersive materials while ensuring simulation stability (Sec. 2.1), inspired by a related technique for frequency-domain TO [32].

Finally, our software package combines three individually common (but rarely combined for TO) forms of parallelism that allow for predictive scaling as the problem size increases: (i) spatial parallelism, where the Maxwell solver divides the computational domain among hundreds or even thousands of compute nodes using a static load-balancing algorithm (Sec. 5.1); (ii) frequency parallelism, such that gradients for even complicated objective functions that involve multiple field quantities (e.g. eigenmode overlaps, scattering parameters, Poynting flux, etc.), corresponding to multiple adjoint sources, are simultaneously obtained across multiple frequencies of interest with minimal additional computational cost (Sec. 5.2); and (iii) simulation parallelism, where multiple design fields or objective functions can be simulated in tandem, but their results are globally gathered by the optimization algorithm at each iteration, enabling large-scale robust [33–36] and multi-objective [37] optimization (Sec. 5.3).

We illustrate our approach using two practical example problems: (i) a 3D silicon-photonics polarization splitter (Sec. 4.1) and (ii) a high numerical-aperture (NA) cylindrical metalens (Sec. 4.2). Some of these devices are large in comparison to the current state of the art, and they involve multiple TO features to demonstrate the methodology’s flexibility and diverse range of applications.

2. Photonics density-based topology optimization

We first review the fundamentals of photonics density-based TO, highlighting important features of our approach (e.g. automatic differentiation and hybrid time/frequency-domain adjoint methods) that enable large-scale inverse design. We then present a novel material-interpolation scheme
necessary for broadband TO over dispersive and/or anisotropic materials.

Density-based TO parameterizes the design at each pixel such that each pixel is either “solid” (a high-index material), “void” (a low-index material), or some non-physical interpolation between the two. A sequence of transformations involving filters and projections are used to gradually binarize the design so that it is solid or void almost everywhere [31, 38], typically while also imposing various optimization constraints that enforce design rules (e.g. minimum lengthscale, area, and curvature) [25, 39, 40]. Since this approach allows pixels to adopt a fictitiously interpolated material “density” during the optimization process, the resulting cost function (and its gradient) are smooth and continuously differentiable, an important requirement for many gradient-based optimization algorithms [30]. (In contrast, level-set TO ensures that the material is binarized almost everywhere, but can make differentiation and optimization more difficult, especially for changes in topology [30].) The FOMs used to characterize the design’s performance are typically some function of the time- or frequency-dependent fields (Sec. 4), as dictated by the Maxwell equations. To accurately evaluate each FOM, one typically performs a “forward run” using a full-wave Maxwell solver (e.g. FDTD, FDFD, etc.). The gradient of each FOM is then efficiently calculated by performing an “adjoint run” using the same (or slightly modified) Maxwell solver with source terms determined from the forward run—in particular, the “adjoint currents” at a particular frequency are essentially the derivative of the FOM with respect to the electromagnetic fields from the forward run at that frequency [1, 23]. This process is repeated as the user gradually tunes a projection hyperparameter $\beta$ to increase binarization (Eq. (4)), yielding performant and oftentimes non-intuitive designs. Fig. 1 illustrates the basic design cycle for photonic TO driven by adjoint-variable methods, along with an example design evolution for a silicon-photonic three-way power splitter.

For example, one could formulate a broadband, multi-objective minimax optimization problem [41], based on performance over $M$ discrete frequency points ($\omega_m$) and $N$ figures of merit ($f_n$), with

$$\min_{\rho} \left[ \max_{n=1\ldots N, m=1\ldots M} f_n(x_m) \right]$$

s.t.  
$$A(\rho, \omega_m)x_m = b_m \quad m \in \{1, \ldots, M\}$$
$$g_k(\rho) \leq 0 \quad k \in \{1, \ldots, K\}$$
$$0 \leq \rho \leq 1$$

where $x_m$ describes the (discretized) electric and magnetic fields at $\omega_m$, $\rho$ are the design variables (densities) used to parameterize the design at each pixel (constrained pointwise to $0 \leq \rho \leq 1$), $g_k$ are inequality constraints implementing geometry constraints (Sec. 3.1), $A$ is the Maxwell operator in the frequency domain, and $b_m$ is a vector describing the forward problem’s current sources.

We emphasize that Eq. (1) and $A$ are only used conceptually in this presentation—we never form the frequency-domain operator $A$ or solve $Ax = b$ explicitly. Instead, we Fourier transform the time-domain fields of the Meep FDTD simulation in order to obtain the frequency-domain solution indirectly, as reviewed in Appendix A and Ref. 23. By using broadband sources for the forward and adjoint solves, this hybrid adjoint method calculates the gradients at multiple frequency points simultaneously (Sec. 5.2), meaning that each gradient calculation requires only two FDTD simulations for each figure of merit regardless of the number of design parameters or frequencies of interest.

The minimax problem of Eq. (1) is nondifferentiable, but it can be recast into a differentiable
Fig. 1. (a) Overview of density-based photonic topology optimization [30] for photonic inverse design via adjoint-variable methods demonstrated using a three-way silicon-photonic power splitter. First, the design variables are filtered and projected. Second, a forward run is executed to evaluate the objective function. Third, an adjoint run (one for each figure of merit) is executed using adjoint sources determined by the results of the forward run. The gradient of the objective function with respect to the design variables is computed from the Fourier-transformed fields in the design region from both the forward and adjoint runs. The optimization algorithm updates the design variables using the gradient information. A hyperparameter $\beta$ is gradually increased to “binarize” the design as optimization progresses.
form using an epigraph formulation [41, 42] by introducing a dummy parameter \( t \):

\[
\begin{align*}
\min_\rho, t & \\
\text{s.t.} & A(\rho, \omega_m) x_m = b_m, \quad m \in \{1, \ldots, M\} \\
& f_n(x_m) - t \leq 0, \quad n \in \{1, \ldots, N\} \quad (2) \\
& g_k(\rho) \leq 0, \quad k \in \{1, \ldots, K\} \\
& 0 \leq \rho \leq 1 
\end{align*}
\]

The process for mapping the raw design parameters \( \rho \) to a near-binary permittivity distribution \( \varepsilon_r \) typically begins with a linear-convolution filter step

\[
\tilde{\rho} = w(x) * \rho, \quad (3)
\]

where \( \tilde{\rho} \) is the filtered design field, \( w \) is the filter kernel, and \( * \) is a problem-dependent multidimensional convolution [31, 43]. Once the design variables are filtered, the resulting field is then projected onto a nearly binary value using a differentiable and nonlinear function. One common choice, for example, is the modified hyperbolic tangent (tanh) function [31]

\[
\tilde{\rho} = \frac{\tanh (\beta \eta) + \tanh (\beta (\tilde{\rho} - \eta))}{\tanh (\beta \eta) + \tanh (\beta (1 - \eta))}, \quad (4)
\]

where \( \beta \) and \( \eta \) are thresholding parameters and \( \tilde{\rho} \) is the projected design field.

A unique feature of our software package is the ability to quickly and easily prototype different combinations of filter kernels, projection functions, and their corresponding hyperparameters. This is useful because different filter kernels and projection functions are used to realize certain topological features, such as angled sidewalls [44] or minimum lengthscales [40]. Generally, deciding which kernel or projection function to use is a difficult and time-consuming process mainly because each new functional form requires a new gradient derivation. Our approach leverages JAX [17], a high-performance AD package that quickly and efficiently computes gradients of arbitrary filters. Since AD implements the chain rule in reverse via vector-Jacobian products, the resulting filter-project backpropagation step remains significantly cheaper than the forward and adjoint Maxwell solves, making it easy to employ multiple filter steps [45]. Enabling multiple cascaded steps is especially important for certain types of robust optimization [46]. While our package already provides many built-in projections (Heaviside, hyperbolic tangent, etc.), kernels (dilation, erosion, etc.), and even nonlinear filters (conic, cylindrical, Gaussian, linear, etc.), it is easy to add new functions to support new geometries and constraints.

### 2.1. Material interpolation

Once the design parameters are projected onto a binary design field, the next challenge is to map these scalar values to physical materials over a broad bandwidth such that the FDTD algorithm remains stable. In the simple case involving dispersionless dielectrics, linear-interpolation schemes tend to work well [47]:

\[
\varepsilon_r(\tilde{\rho}) = \varepsilon_{\min} + \tilde{\rho}(\varepsilon_{\max} - \varepsilon_{\min}), \quad (5)
\]

where the relative permittivity \( \varepsilon_r \) linearly varies between a low-index material \( \varepsilon_{\min} \) and a high-index material \( \varepsilon_{\max} \), and \( \tilde{\rho} \in [0, 1] \) are the projected design parameters [47]. More complicated design problems (e.g., those involving metals, whose relative permittivity is complex and may have a negative real part) require a more sophisticated interpolation scheme. For example, intermediate values of \( \rho \) can produce zero crossings in the permittivity profile that
induce non-physical resonances and inhibit optimization convergence [32]. One possible solution when dealing with single-frequency applications involves nonlinearly interpolating the refractive index $\eta(\tilde{\rho})$ and the extinction coefficient $\kappa(\tilde{\rho})$ [32]:

$$\varepsilon_r = (\eta(\tilde{\rho})^2 - \kappa(\tilde{\rho})^2) - i(2\eta(\tilde{\rho})\kappa(\tilde{\rho})) .$$

(6)

In this case, $\eta$ and $\kappa$ are both linearly interpolated as in Eq. (5). The final permittivity itself, however, is nonlinearly interpolated, preventing any possible zero crossings for intermediate values of $\rho$. While this approach works well for frequency-domain Maxwell solvers, the hybrid time/frequency-domain implementation employed in our work requires a more holistic interpolation scheme that works across the entire band of interest with the same set of design parameters and is numerically compatible with the FDTD algorithm.

For example, linear-dispersive materials (including metals) are typically modeled within FDTD codes using Lorentz–Drude and conductivity models such that the position- ($\mathbf{r}$) and frequency- ($\omega$) dependent relative permittivity ($\varepsilon_r$) is described by [48]

$$\varepsilon_r(\omega, \mathbf{r}) = \left(1 + i\frac{\sigma_D(\mathbf{r})}{\omega}\right)\left(\varepsilon_{\infty} + \sum_n \frac{\sigma_n(\mathbf{r})\omega_n^2}{\omega_n^2 - \omega^2 - i\omega \gamma_n}\right) ,$$

(7)

where $\varepsilon_{\infty}$ is the instantaneous permittivity, $\sigma_D$ is the position-dependent electric conductivity, $\sigma_n$ is a position-dependent coupling strength, $\omega_n$ is a harmonic oscillator frequency, and $\gamma_n$ is a damping coefficient. Interpolating between two arbitrary material models described by Eq. (7) is more difficult, as one must minimize the number of parameters within the model that are interpolated (to also minimize storage requirements) while simultaneously ensuring simulation stability for each combination of material types.

We propose a new interpolation scheme that only requires interpolation of the conductivity terms ($\sigma_D$ and $\sigma_n$) and the instantaneous permeability ($\varepsilon_{\infty}$). First, we linearly interpolate the instantaneous permittivity, the conductivity term, and each Lorentz–Drude component,

$$\varepsilon_{\infty} = \varepsilon_{\infty,0} + \tilde{\rho}(\varepsilon_{\infty,1} - \varepsilon_{\infty,0}) ,$$

(8)

$$\tilde{\sigma}_D = \tilde{\sigma}_0 + \tilde{\rho}(\tilde{\sigma}_1 - \tilde{\sigma}_0) ,$$

(9)

$$\tilde{\varepsilon}_{\text{sus}} = (1 - \tilde{\rho})\sum_n^{(0)} \frac{\sigma_{n,0}\omega_{n,0}^2}{\omega_{n,0}^2 - \omega^2 - i\omega \gamma_{n,0}} + \tilde{\rho}\sum_m^{(1)} \frac{\sigma_{m,1}\omega_{m,1}^2}{\omega_{m,1}^2 - \omega^2 - i\omega \gamma_{m,1}} ,$$

(10)

where 0 and 1 subscripts correspond to the first and second materials respectively, $\varepsilon_{\infty}$ is the new material’s instantaneous permittivity, and $\tilde{\varepsilon}_{\text{sus}}$ describes the new material’s dispersive susceptibilities. Next, we introduce an artificial damping term needed to eliminate spurious bulk resonances from accidental “zero crossings” where $\varepsilon \approx 0$:

$$\tilde{\sigma}_D(\mathbf{r}) = \tilde{\rho}(1 - \tilde{\rho})\tilde{\omega} ,$$

(11)

where $\tilde{\omega}$ is the mean resonance frequency of all the susceptibilities. The final material response is then described by

$$\tilde{\varepsilon}_r(\omega, \mathbf{r}) = \left(1 + i\frac{\tilde{\sigma}_D(\mathbf{r}) + \tilde{\sigma}_D(\mathbf{r})}{\omega}\right)\left(\tilde{\varepsilon}_{\infty} + \tilde{\varepsilon}_{\text{sus}}\right) .$$

(12)

This new material interpolation scheme requires no modification to existing FDTD codes, provided the conductivity and instantaneous permittivity can be parameterized over the simulation domain (or in particular, the design regions of interest). Since the FDTD method uses a Yee grid in which different field components are stored on different grids [49], one must account for the interpolation from the design grid onto the different staggered grids, especially in the recombination step of the adjoint method used to compute the gradient of the objective function (Appendix A).
3. Geometric parameterization and the FDTD Yee grid

In this section, we discuss the geometric-parameterization Yee scheme used to enable density-based photonics TO over the FDTD Yee grid. In contrast to many density-based TO implementations (e.g. involving finite-element meshes) which use the same grid/mesh for both the design variables $\rho$ and the electromagnetic field unknowns [50], our approach allows the user to flexibly define arbitrary “material grids” [51] or design meshes in 1D, 2D, or 3D that have simulation-independent resolutions/orientations and are linearly interpolated onto the Yee grid (e.g. the simulation mesh) [16,48]. This generality, adapted from our Ref. 51, provides three key benefits: (i) a separation between the design and simulation mesh choices, where proper placement (and gradient calculation) with respect to the Yee grid is handled automatically; (ii) implicit dimensionality constraints (e.g. lithographic 2D designs that are projected onto a 3D polygonal prism via a 2D material grid) are supported without any additional optimization constraints; and (iii) complicated symmetries and periodic patterns are easily enforced by transforming and layering multiple material grids [51]. Fig. 2 illustrates the basic functionality of the material grid, along with its corresponding features.

Since the Yee grid involves staggered placement of a material’s permittivity tensor components and field components, even simple density-based parameterization methods can be difficult to...
formulate. The initial simulation setup requires the user to properly interpolate the design DOFs onto each field-component location. The adjoint recombination step after the forward and adjoint runs have been executed (Appendix A) must also interpolate the field components from the Yee grid back onto the corresponding design grid using the transpose of the forward pass’s coefficients (a process referred to as restriction [16]). Careful interpolation and bookkeeping is therefore required. As such, we generalized the user-specified design grid to accept any arbitrary resolution or dimensionality, such that the design grid correctly interpolates/backpropagates to/from the Yee grid. This is consistent with *Meep*’s overall philosophy of “pervasive interpolation,” [16] in which the user is presented with an illusion of continuous spatial coordinates (transparently interpolated to/from the Yee grid internally).

This new data structure, which we call a “material grid,” fills user-specified geometric shapes, such as blocks, so that the user can apply the same (or different) material grid(s) to multiple discrete geometric objects (covering any portions of the simulation domain). The objects containing each grid can be arbitrarily rotated/flipped/translated. Different (or identical) material grids can occupy multiple geometric objects to represent distinct fabrication layers and design planes or periodic structures within a supercell. Similarly, the user can adapt a two-dimensional material grid to a three-dimensional geometric primitive (e.g. a rectangular prism): our approach extrudes the design into the third dimension, simplifying the optimization pipeline for integrated photonics and other lithographic-based designs.

Geometric design symmetries are easy to impose by layering multiple material grids on top of one another, because the net structure is obtained by averaging any overlapping grids [51]. For example, by layering a grid with a mirror image of itself, the resulting design must have mirror symmetry. While symmetries could alternatively be enforced as explicit constraints on the design variables (e.g. before the filtering and projection stages) using an AD package, that approach is only straightforward for symmetry operations that map the material grid exactly onto itself. In contrast, the overlapping-grid approach allows arbitrary rotational and translational symmetries, such as threefold symmetries, glide-plane symmetries, and crystallographic symmetry groups [51].

In short, the material-grid approach maximizes flexibility and expressivity, without requiring complicated optimization constraints or non-intuitive design parameterizations. With this increased level of flexibility, however, one must explicitly constrain the resulting topological evolution to comply with fabrication design rules in a manner that is compatible with existing TO algorithms.

3.1. Fabrication constraints

In order to constrain the vast design freedom typical of most density-based TO methods, our package readily supports differentiable constraint functions commonly used to prescribe a design’s minimum lengthscale [39], minimum area/enclosed area [25], curvature [40], connectivity [3], or any other topological feature. Performing optimization sweeps over different design “rulebooks” is often a simple matter of parameterizing each designated constraint function, as we described in detail in Ref. 25. For example, Fig. 3 compares four broadband, three-way splitters designed to comply with different minimum lengthscale constraints. Each design performs with no more than 6% insertion loss across the band and ≤ 2% variation in splitting ratio across the band.

Often, constraints are enforced implicitly using density filters [39, 40] or even cascades of filters [46]. Many techniques, however, require explicit constraint functions that may even work in tandem with the density filters to enforce a particular topological feature requirement. Using the same AD approach for the density filters and projection steps (Sec. 1), our package supports AD for arbitrary constraint functions. This feature is essential for prototyping and “inventing” new constraint functions and methodologies. Indeed, choosing the proper constraints and filtering steps often seems to "regularize" the optimization problem to avoid nonphysical local minima [52]. When combined with the proper objective functions, the fabrication constraints
Fig. 3. Design of a three-way power splitter in 2D across 10 wavelengths from 1.5 μm to 1.6 μm with a design region of size 3 μm × 3 μm. (a) From left to right, the devices were designed with minimum lengthscale fabrication constraints of 60 nm, 90 nm, 120 nm, and 150 nm. Each constraint is represented by the diameter of a black circle in the upper-left corner of each design. The steady-state field pattern at λ = 1.55 μm is superimposed on top of each design. (b) The transmission across all wavelengths is depicted for the top/bottom ports (the same due to induced symmetry) and the right (output) port. Each device shows good performance with ≤ 6% insertion loss and ≤ 2% variation in splitting ratio averaged across the entire bandwidth.

help ensure the optimizer quickly yields a suitable design after a limited number of iterations.

4. Objective functions

In this section, we describe the process for formulating objective functions and highlight some unique features that enable rapid prototyping and sophisticated optimization. Our package supports several “differentiable measurements” (DMs) for which the adjoint source can be automatically generated. These DMs, which are derived from the Fourier-transformed fields, include: (1) eigenmode-decomposition coefficients (or Scattering parameters), (2) Poynting flux, (3) near-to-far transformation, and (4) the Fourier-transformed fields themselves. By using the same AD package (JAX) described in Sec. 2, the user can programatically define any arbitrary function of these DMs, such that the adjoint source for each objective function is scaled and placed appropriately.

For example, a typical objective function, \( f \), may seek to maximize the squared magnitude of the \( S_{21} \) parameter of a device’s scattering matrix (\( S \) matrix) equivalent to the power transmitted in Port 2 given an input source in Port 1,

\[
  f = |S_{21}|^2, \tag{13}
\]

where the \( S_{21} \) parameter is a ratio of the outgoing and incoming eigenmode coefficients (each
being a DM) such that

$$ S_{21} = \frac{\alpha_{1,2}^+}{\alpha_{1,1}^+}, $$

(14)

where \( \alpha_{m,n}^\pm \) is the mode coefficient (complex amplitude) of the \( m \)th mode for either the forward (+) or backward (−) directions in the \( n \)th port. The mode coefficient itself is determined by an overlap integral over a cross-section \( A \) [53]:

$$ \alpha_{m,n}^\pm = C \int_A \left[ \hat{E}^\pm(r) \times \hat{H}_{m,n}^\pm(r) + \hat{E}_{m,n}^\pm(r) \times \hat{H}^\mp(r) \right] \cdot \hat{n} dA, $$

(15)

where \( \hat{E}(r) \) and \( \hat{H}(r) \) are the Fourier-transformed monitor fields at a particular frequency, \( \hat{E}_{m,n}^\pm(r) \) and \( \hat{H}_{m,n}^\pm(r) \) are the mode profiles corresponding to the \( m \)th mode at the \( n \)th port at the same frequency for the forward (+) and backward (−) propagating modes (obtained by an eigenmode solver [54]), and \( C \) is a normalization constant chosen such that

$$ |\alpha_{m,n}^\pm|^2 = P_{m,n}^\pm, $$

(16)

where \( P_{m,n}^\pm \) is the total power propagating in that particular mode. Since the adjoint source for a particular mode-overlap coefficient corresponds to another eigenmode source pointing in the opposite direction [22], we can choose a time-domain profile that scales the eigenmode source according to the results of the forward run (Sec. 5.2).

The user can further customize the objective function to additionally depend on other Fourier-transformed field quantities, like a far-field profile (corresponding to multiple broadband adjoint sources). Furthermore, different normalization functions, DOF filters, and FOM arrangements are easy to prototype, due to the automated AD process used to place each adjoint source. This flexibility is essential for selecting hyperparameters and identifying which objective function is best for a particular optimization problem. We describe two examples that highlight the flexibility and simplicity of our approach.

### 4.1. Example I: Silicon-photonics polarization splitter

As an initial example, we design a 3D silicon-photonics polarization splitter compatible with a commercial CMOS foundry using a stack of two independent design layers (an SOI layer and a polysilicon layer [55]). To quantify the performance of each output port, we construct two objective functions (to be evaluated in parallel as separate simulations) each consisting of four overlap coefficients over a 100 nm bandwidth. Fig. 4 illustrates the simulation configuration and final device design. The FOMs are designed to maximize transmission of the quasi-TE (Q-TE) and quasi-TM (Q-TM) modes in Ports 2 and 3, respectively, while minimizing return loss in Port 1 as well as channel crosstalk between Ports 2 and 3. The FOM for the Q-TE mode \( f_{Q-TE} \) is defined by

$$ f_{Q-TE} = g_1 \left( \frac{\alpha_{1,2}^+}{\alpha_{1,1}^+} \right)^2 + g_2 \left( \frac{\alpha_{1,3}^+}{\alpha_{1,1}^+} \right)^2 + g_3 \left( \frac{\alpha_{1,1}^-}{\alpha_{1,1}^+} \right)^2 $$

(17)

where \( \alpha_{1,2}^+ \) corresponds to the forward (+) propagating coefficient of the first mode (Q-TE) at the second port, such that \( |\alpha_{1,2}^+|^2 \) corresponds to the \( S_{21} \) entry, \( |\alpha_{1,1}^+|^2 \) corresponds to the \( S_{31} \) entry, and \( |\alpha_{1,1}^-|^2 \) corresponds to the \( S_{11} \) entry. The functions, \( g_1, g_2, \) and \( g_3 \) are mapping functions used to ensure units are consistent and of the same scale (optimization direction) as one another, such that the transmission is maximized, and the return loss and channel crosstalk are minimized. In
this example, the mapping we used is a differentiable spline interpolater based on user-specified thresholds to map each individual “sub-objective” to a compatible global FOM [56]. Similarly, the FOM for the Q-TM mode, $f_{Q-TM}$, is defined by

$$f_{Q-TM} = g_1 \left( \frac{\alpha_{2,2}^+}{\alpha_{2,1}^+} \right)^2 + g_2 \left( \frac{\alpha_{2,2}^+}{\alpha_{2,1}^+} \right)^2 + g_3 \left( \frac{\alpha_{2,2}^-}{\alpha_{2,1}^-} \right)^2$$

(18)

where $\left( \frac{\alpha_{2,2}^+}{\alpha_{2,1}^+} \right)^2$ corresponds to the $S_{21}$ entry, $\left( \frac{\alpha_{2,3}^-}{\alpha_{2,1}^-} \right)^2$ corresponds to the $S_{31}$ entry, and $\left( \frac{\alpha_{2,2}^-}{\alpha_{2,1}^-} \right)^2$ corresponds to the $S_{11}$ entry.

Both FOMs are defined over ten frequency points spanning a 100 nm bandwidth covering the C-band, from 1.5 μm to 1.6 μm, resulting in twenty unique FOMs that each produce four broadband adjoint sources. The full 3D structure was optimized using the minimax formulation outlined in Eq. (2) to maximize the minimum (worst-case) FOM out of this set. While the gradient for each frequency point was calculated using the same forward and adjoint run, each objective function ($f_{Q-TE}$ and $f_{Q-TM}$) required its own forward and adjoint runs, such that two simulations were always run in parallel (Sec. 5.3). Fig. 5 illustrates the final results of the polarization splitter for all six scattering parameters of interest across the entire band, along with a steady-state field profile for both polarization states at 1.55 μm.

We ran the optimization on two Intel Xeon Gold 6226 2.7 GHz CPU nodes using resources provided by the Partnership for an Advanced Computing Environment (PACE) at the Georgia Institute of Technology. Each node contained 24 cores (48 total) and 192 GB of RAM (384 GB total). The optimization procedure consisted of 250 iterations (500 total Maxwell solves) and took approximately three days to complete at an FDTD resolution of 30 pixels/μm. As expected,

Fig. 4. Design of a 3D silicon-photonic polarization splitter using multiple objective functions and design regions. (a) A 2D cross section in the XY plane showing the bottom design region (SOI layer) with the eigenmode source (red) and three eigenmode monitors (blue) used to compute the scattering parameters over a broad bandwidth. (b) The full 3D design with an SOI base layer and polysilicon top layer. The size of the two design regions are each 5 μm × 5 μm. (c) The XZ cross section showing the SOI design region (green) and the polysilicon design region (blue) as well as the sources and monitors.
Fig. 5. Structure and performance of an optimized 3D silicon-photonic polarization splitter. (a) Steady-state field profile at a wavelength of 1.55 μm for the $H_z$ (quasi-TE) and $E_z$ (quasi-TM) polarization states superimposed on the SOI cross section of the full 3D design (lower right inset). (b) Broadband power spectrum of the quasi-TE (left) and quasi-TM (right) polarization states showing the three scattering parameters $S_{11}$, $S_{21}$, and $S_{31}$. The spectra of the two polarization states shows minimal return and insertion loss as well as strong polarization selectivity.
the polarization splitter device performs well across the entire band. We enforced standard semiconductor foundry design rule constraints during the final “epoch” of optimization (Sec. 3.1). The effect of the constraints on the optimization evolution is demonstrated in Fig. 6, which tracks the evolution of the minimax objective, along with the actual scattering parameters as a function of optimization evolution. The performance worsens significantly once the constraints are applied, as the optimizer adjusts the topology for features and areas that are too small. Once these features satisfy the constraint, the optimization algorithm continues to adjust the broadband performance for all objective functions.

4.2. Example II: High numerical-aperture cylindrical metalens

The second example involves a planar metalens [57–60] with high numerical aperture (NA) that uses a near-to-far field transformation and cylindrical symmetry to significantly reduce the computational cost at each optimization iteration, similar to techniques we have previously demonstrated for frequency-domain TO [3]. We design a metalens that has a diameter of 30 μm, is 2 μm thick, and has a desired focal length of 30 μm (corresponding to a numerical aperture of 0.5) and a multi-wavelength operation regime spanning 1.54–1.56 μm. Fig. 7 describes the steady-state performance of the optimized device at 1.55 μm and compares the reduced cell used in the optimization to the full cell without any domain-decomposition techniques. The final device focuses an incoming plane wave at the desired focal length across the entire band.

In order to inverse-design a metalens, we simply maximize the intensity at the desired focal point [3, 61], so that the objective function is:

$$f_n = |\hat{\mathbf{E}}(\omega_n, r_0, z_0)|^2,$$

where $\hat{\mathbf{E}}$ is the Fourier-transformed electric field at frequency $\omega$ and position $(r_0, z_0)$. Naively, this requires a simulation domain that includes the metalens, the desired focus, and the uniform propagation region in between (which could be rather large relative to the size of the lens). The simulation cost would then scale poorly with the focal length—not just because of the number of voxels that require updating at each timestep, but also because the electromagnetic fields must propagate through the entire cell (affecting the total number of timesteps needed for convergence), a typical challenge with time-domain methods.

Since the hybrid time/frequency-domain method used throughout this work already operates on frequency-domain quantities (Sec. 2), we employed a near-to-far transformation to significantly reduce the size of the computational cell. The near-to-far transformation is a well-known technique, based on Huygens’ principle, where the Fourier-transformed tangential fields in the “near region” can be treated as currents that are convolved with the corresponding Green’s function (fields produced by time-harmonic point sources) to yield the Fourier-transformed fields at arbitrary user-defined far-field points [28, 48]. Using near-to-far transformations, the new computational cell need only include the actual metalens structure and the fields just beyond its surface [61]. In this example, we use a cylindrical-coordinates Green’s function, but our package also supports 2- and 3-dimensional near-to-far transformations in Cartesian coordinates (with both distributed-memory and thread-level parallelism).

The backpropagation step of the near-to-far transformation is just another (transposed) Green’s function convolution. This extra step is efficiently implemented using the same machinery as the rest of the FDTD solver, such that the entire adjoint pipeline can be used with far-field transformations (or even combinations of far-field transformations and other fundamental differentiable quantities). This method could be extended to other Green’s functions, such as those modeling propagation through inhomogeneous regions [62], which is useful for integrating multilayer films [14].
Fig. 6. Design field evolution for the SOI (a) and polysilicon (b) layers at iterations 1, 31, 101, 121, and 250 for the (3D) polarization splitter. (c) The evolution of the worst-case frequency differentiable quantity for both FOMs, (worst-ω), used in Eq. (2). The fabrication constraints were applied at iteration 120, which resulted in a significant drop in performance. Each FOM is a function of the transmission, crosstalk, and return loss, as described by the corresponding scattering parameters. The evolution of each independent quantity as a function of iteration is found in (d), (e), and (f) respectively.
Fig. 7. Broadband metalens with a high numerical aperture (NA) efficiently designed using near-to-far field transformation and cylindrical \((RZ)\) coordinates. (a) The full 3D design is reduced to a compact 2D simulation. The computational cell \((15 \, \mu\text{m} \times 12 \, \mu\text{m})\) consists of the design region \((15 \, \mu\text{m} \times 2 \, \mu\text{m})\), an incident planewave source, a near-field monitor used to calculate a far-field projection, and perfectly-matched layer (PML) absorbing boundaries on three sides of the cell. The design objective is a focal point of \(30 \, \mu\text{m}\) from the lens center, a distance that is more than four times the length of the truncated cell. (b) The metalens structure after 280 optimization iterations and the corresponding field profile from an incoming planewave source. As expected, the lens strongly focuses the beam at \(30 \, \mu\text{m}\). (c) The evolution of the figure of merit (FOM) with iteration number. Large dips in performance tend to be associated with increases in the thresholding parameter \((\beta)\) which can significantly change the design. The best- and worst-case performance over a 20 nm bandwidth are depicted in the light-blue shaded region and the mean performance is depicted by a solid line. (d) Broadband performance of the final device depicted by the Strehl ratio and transmission efficiency.
Fig. 8. The three forms of computational parallelism embodied by our approach. (a) Domain parallelism. The forward and adjoint simulations are load balanced for execution using an MPI cluster by dividing the computational cell into chunks with nearly equivalent cost. The heterogeneous physics of the simulation involving different phenomena or calculations at different points in the computational domain often requires chunks of unequal sizes. (b) Frequency parallelism. The FDTD algorithm is used to calculate the gradient at multiple frequency points in parallel, even with complicated objective functions that have several adjoint sources or with simulation domains involving lossy and dispersive materials. (c) Objective parallelism. Multiple objective functions and design fields are run using “in-the-loop” parallelism, allowing the results of each run to be communicated to the optimizer after each iteration. This is essential for multi-objective optimization (i.e. optimizing along a Pareto front) and robust optimization (i.e. optimizing for fabrication variability).

5. Computational parallelism

We now describe three forms of computational parallelism implemented within our package that enable large-scale photonics topology optimization: (1) spatial parallelism, which involves dividing the cell into load-balanced workloads across multiple processor cores of an MPI cluster; (2) frequency parallelism, which involves computing gradients for multiple frequencies simultaneously; and (3) simulation parallelism, which involves distributing gradient computations for different design fields or objective functions among different processor groups. Fig. 8 illustrates each of these forms of parallelism.

5.1. Data-driven load balancing for massively parallel simulations

Simulating large devices (i.e. tens to hundreds of wavelengths in physical dimensions) in which the total memory requirements exceed those of a single multicore machine is made possible in Meep by dividing the simulation region into multiple subregions or “chunks” [16]. Each chunk is assigned to a different processor core which, at each timestep, updates the fields at every voxel and communicates the fields on its exterior surfaces to neighboring chunks. Processors are assigned to the chunks in a way that ensures that adjacent chunks are assigned to a single cluster node as much as possible to exploit fast intra-node communication via shared memory for the chunk boundaries. A key challenge for performance optimization of massively parallel simulations involving hundreds or thousands of chunks/processes is the heterogeneous physics: different phenomena or calculations are modeled at different points in the computational domain. Because different materials or data processing often require vastly disparate computational resources, load balancing the simulation involves partitioning the cell into unevenly sized chunks which have (nearly) equivalent computational work. To enable this feature, we developed a
data-driven cell-partitioning strategy based on estimating the computational work required by a given subvolume [63, Sec. 3].

Because the forward and adjoint simulations involve computing the Fourier-transformed fields over the entire design region which can be a large fraction of the total cell, the cost of the DTFT field updates can often dominate the overall stepping. We therefore implemented two performance-related features: (i) reducing the memory bandwidth by storing the DTFT fields (as well as the time-domain fields) using single- rather than double-precision floating point since the error in the simulation is almost always dominated by discretization error of the discontinuous material boundaries rather than floating-point rounding errors and (ii) decimating the DTFT time series updates (i.e. performing the DTFT updates at every \( n \)th timestep rather than every timestep, without any loss of accuracy due to aliasing thanks to the band-limited nature of the sources we employ) [27]. The degree of decimation is chosen automatically at runtime based on the Nyquist rate set by the bandwidth of the sources and monitors. We also implement a recursive “cache-oblivious” loop tiling [64] approach for the field updates to improve memory locality for updating different field components. Taken together, we have demonstrated through benchmarking studies using Intel Xeon processors that these features can provide more than a 5× speedup for the stepping.

5.2. Frequency parallelism, broadband optimization, and convergence criteria

To generate the correct frequency response over a broad bandwidth, a key challenge is the construction of the time-domain adjoint sources. For a given frequency-dependent FOM, the frequency-domain adjoint method prescribes a corresponding frequency-dependent current density \( \hat{J}(x, \omega) \) at each of the design frequencies \( \omega_m \) [1], as reviewed in Sec. 2. In a time-domain simulation, one must then construct a time-dependent current \( J(x, t) \) whose (discrete-time) Fourier transform (DTFT) matches \( \hat{J}(x, \omega_m) \) for \( m = 1, \ldots, M \). By Fourier-transforming the “adjoint fields” resulting from this current, the gradient of the FOM can be computed at all frequencies using one additional time-domain simulation. Because there are infinitely many functions \( J(x, t) \) whose Fourier transform would be suitable, for efficiency we desire a current source that is highly localized in time (time-limited), enabling a short-duration time-domain adjoint simulation, as well as being well-localized in frequency (approximately bandlimited) to enable decimation without aliasing as described in Sec. 5.1. In this section, we describe a novel fitting approach to efficiently construct a time-domain adjoint current with these properties.

Our method fits a series of band-limited filter windows (acting as basis functions) in the frequency domain to the desired adjoint-source frequency response. The weight factors are then used to modulate the corresponding time-domain profiles, which are analytically derived using the inverse DTFT. The resulting algorithm only requires \( O(NM) \) storage for the basis weights, where \( M \) is the number of frequency points and \( N \) is the number of spatial points spanning the source. This approach is independent of the number of timesteps and works with any number of adjoint sources and any number of frequency points with arbitrary spacing. Fig. 9 illustrates the fitting process for an eigenmode adjoint source, along with its corresponding time-domain profile.

The fitting routine solves the least squares problem

\[
\min_{b_m(x)} \left\| \hat{J}(x, \omega) - \sum_{m=0}^{M} b_m(x) \hat{W}_m(\omega) \right\|_2^2 ,
\]

where \( \hat{J}(x, \omega) \) is a frequency- and position-dependent adjoint source corresponding to the \( n \)th adjoint quantity of interest (eigenmode overlap, far-field profile, etc.) and \( b_m(x) \) is a basis weight at position \( x \) corresponding to the \( n \)th frequency basis function, \( \hat{W}_m(\omega) \). Since the adjoint quantities are only evaluated at the discrete frequencies of interest, the fitted function can take
Fig. 9. Overview of the fitting procedure using Nuttall basis functions for the multi-frequency adjoint source used for the gradient calculation. The Nuttall basis functions $W$ in the (a) frequency and (b) time domain. (c) The desired profile of the frequency-domain source is fit using $m$ basis functions, one for each frequency of interest. Only the values at each specified frequency point are used in the fit. (d) An analytic time profile for each source is quickly recovered using the inverse DTFT of the Nuttall window function. (e) This procedure is repeated at each spatial point for all three adjoint sources using the polarization-splitter example from Fig. 5.
any value for all other frequency points. Given these weights $b_m$, the adjoint simulation then uses a time-domain current source

$$\mathbf{J}(\mathbf{x}, t) = \sum_{m=0}^{M} b_m(\mathbf{x}) W_m(t),$$

(21)

where $W_m(t)$ is the inverse DTFT of $\hat{W}_m(\omega)$.

It is desirable to have a basis function $W$ that is localized in both the time and frequency domains, with an analytically tractable form in both domains. (For example, a Gaussian window is somewhat inconvenient because it does not have a simple analytical DTFT formula, although it could still be used as a basis function by, e.g., numerically integrating the corresponding DTFT at each timestep [24].) We chose to employ a Nuttall window [65] as our basis function, since its functional forms are simple and because it demonstrates a balance between resolution and dynamic range (an important trade-off when fitting several basis functions at closely spaced frequencies). The Nuttall window is defined by

$$\hat{W}_m(\omega) = \frac{1}{2} \sum_{k=0}^{3} a_k (\omega - \omega_m) \sum_{k=0}^{3} d_k \left( 1 - e^{i(N+1)(\omega - \omega_m + k\Delta\omega)\Delta t} + \frac{1}{1 - e^{i(N+1)(\omega - \omega_m + k\Delta\omega)\Delta t}} \right),$$

(22)

where $N$ is the total length of the sequence, $\Delta t$ is the timestep, $\omega_m$ is the basis-function center frequency, the spacing between adjacent sinc functions is $\Delta \omega = \frac{2\pi}{N\Delta t}$, and the Nuttall coefficients $a_k$ are

$$a_0 = 0.355768, \quad a_1 = 0.4873960, \quad a_2 = 0.144232, \quad a_3 = 0.012604.$$  

(23)

This approach centers a basis function at each frequency of interest ($\omega_0$). The corresponding (complex) time-domain sequence is defined by

$$W_m[n] = \begin{cases} e^{-i\omega_m n \Delta t} \sum_{k=0}^{3} a_k (\omega - \omega_m) \sum_{k=0}^{3} d_k \left( 1 - e^{i(N+1)(\omega - \omega_m + k\Delta\omega)\Delta t} + \frac{1}{1 - e^{i(N+1)(\omega - \omega_m + k\Delta\omega)\Delta t}} \right) \cos \left( \frac{2\pi kn\Delta t}{N} \right), & 0 \leq t \leq N \\ 0 & \text{otherwise} \end{cases}$$

(24)

where $n$ is the current (integer) timestep such that the current time is $t = n\Delta t$. When necessary, we cache the current evaluation of each respective basis function and weight it with the corresponding spatial profile. We note that if the simulation allows for purely real fields, we take the real part of the above complex current source; the resulting additional frequency content at negative frequencies does not affect our analysis at purely positive frequencies.

The frequency bandwidth of a basis function is inversely proportional to its temporal width, so that narrower-bandwidth basis functions require longer adjoint runs. The tradeoff is that wide bandwidths cause the basis functions in the fit (20) to overlap and result in an ill-conditioned fit and potentially inaccurate results. In consequence, we choose the bandwidth of the basis functions to be proportional to the spacing of the user-defined frequency points $\omega_m$, such that the time duration of each (broadband) adjoint source is $N = \lfloor \frac{2\pi}{\Delta \omega \Delta t} \rfloor$, where $\Delta \omega$ corresponds to the minimum frequency spacing in the objective functions (which support arbitrary nonuniform frequency sampling). Requesting closely spaced $\omega_m$ values therefore leads to long-running adjoint simulations. However, the main reason to request such finely spaced frequencies is if the spectrum exhibits fine features (e.g., sharp peaks), which correspond to long-lived resonant waves that would require a long-running simulation anyway.

5.3. Simulation parallelism

Finally, our approach supports simulation parallelism, where multiple design fields or objective functions can be simulated in tandem, but their results are globally gathered by the optimization
algorithm at each iteration, enabling large-scale robust and multi-objective TO. Similar to Sec. 5.1, our package automatically distributes different simulations to different groups of processors. Each group load-balances its respective simulation, and each simulation runs in parallel. After a gradient is calculated for each group, each group sends its corresponding gradient calculation to every other group (and every processor within each group) using an "all-to-all" operation [66]. This way, the same deterministic optimization algorithm can run on each individual processor and converge to the same result, even though each processor only propagates part of the fields for a single group.

For example, the polarization rotator described in Sec. 4.1 assigned an independent objective function to two independent groups of processors, where each group was allocated an independent Intel Xeon Gold 6248 node with 40 cores (for a total of 80 cores). One group computed the gradient corresponding to the quasi-TE objective function ($f_{Q-TE}$), while the other group computed the gradient corresponding to the quasi-TM objective function ($f_{Q-TM}$). Again, we note that each group is able to compute gradients at multiple frequency points simultaneously, such that 20 unique gradients are broadcasted during each optimization iteration (2 objective functions x 10 frequency points). After broadcasting, the optimization algorithm running on each core chooses the same set of parameters for the next iteration.

A more efficient approach might also distribute the work (and storage) of the actual optimization algorithm, such that each core only stores a subset of the design parameters and other arrays used while computing the next optimization step. Indeed, such an approach is especially important for large-scale designs where the DOF span three independent dimensions within the design region (unlike many lithographic applications, where a 2D space is projected onto a 3D thin film). To meet this challenge, we’ve implemented a hybrid thread-level parallelism within the fundamental timestepping algorithm, such that a single process is allocated on each compute node, but multiple shared-memory threads (corresponding to each core) propagate the fields in parallel. Such a hybrid approach [67] combines the benefits of the distributed-memory and multithreaded approaches to parallelism. Since only one process is allocated on each node, only one instance of the optimization algorithm (and all its accompanying data structures) is stored on the node. Rather than assign one process to each core, each core spawns a thread to distribute common tasks, like timestepping and near-to-far transformations. If more memory and/or cores are needed (especially when needing to run multiple simulations in parallel) then more nodes (and processes) can be allocated as needed.

Another common multi-objective task is robust optimization, which performs a worst-case (e.g. minimax) optimization over various design field perturbations [68]. For example, one might optimize over an ideal design and systematic over/under-etched variants (corresponding to eroded and dilated versions of its geometry, respectively). Each gradient calculation is performed in tandem by assigning one group to each design field. This approach seamlessly scales to any number of design variations (an appealing feature if one wishes to impose robustness over an ensemble of random variations [33]).

6. Conclusion

We present a free/open-source photonics topology-optimization software package that builds on the popular FDTD solver Meep which is capable of scaling to arbitrarily large design problems by exploiting massive computational parallelism. By combining powerful techniques scattered throughout several previous works, such as automatic differentiation and a hybrid time/frequency-domain adjoint-variable method, along with novel features such as new filter-design sources for the gradient calculation and material-interpolation methods for dispersive media, our package can flexibly inverse design complicated, broad bandwidth devices and subsystems for a wide range of photonic applications.
Leveraging a hybrid time-/frequency-domain adjoint-variable formulation brings several important benefits to the inverse design community not seen with formulations strictly implemented in one domain or the other. For example, our approach allows for broadband optimization without the prohibitive scaling costs of time-domain adjoint formulations. Similarly, our approach enjoys the flexibility and generalizability of frequency-domain adjoint solvers, without the subtle convergence issues of iterative solvers. Despite these advantages, no single approach is a panacea for all photonics problems, and certain design criteria are better addressed using alternate formulations. Plasmonic design problems, for example, tend to favor non-uniform meshes and finite-element or boundary-element methods (FEM or BEM) that are most easily implemented in the frequency domain [4,69–73]. Another interesting case is the optimization of nonlinear optical devices. The steady-state time-harmonic response of a nonlinear system cannot be computed by Fourier-transforming a pulse input (as in our hybrid approach), whereas nonlinear frequency-domain methods are applicable [74–76]. Non-steady-state transient nonlinear effects, or perhaps self-pulsing or chaotic phenomena, might conversely require a purely time-domain approach. On the other hand, there are a number of nonlinear phenomena that can be modeled using cascaded linear calculations, which could utilize the hybrid time-/frequency-domain approach—for example, optimizing second-harmonic generation (SHG) [77], Raman scattering [4], or other devices where the nonlinearity can be treated perturbatively.

Appendix A. Time-harmonic adjoint formulation

This appendix derives the hybrid time/frequency-domain adjoint-variable method used throughout this work. Our implementation solves the Maxwell equations in the time domain using the FDTD software package Meep. Since the objective functions themselves, however, are defined in the frequency domain, the frequency-domain fields (and the corresponding adjoint sources) must be transformed from time-domain quantities and vice-versa. While this hybrid approach significantly reduces the storage requirements required of time-domain adjoint methods without compromising the flexibility of frequency-domain adjoint methods, there are several nuances that must be considered when constructing the proper adjoint formulation.

First, we examine the Maxwell equations in the time domain as described by

\[
\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E} - \mathbf{K},
\]

\[
\frac{\partial \mathbf{D}}{\partial t} = +\nabla \times \mathbf{H} - \mathbf{J},
\]

where \( \mathbf{E} \) and \( \mathbf{H} \) are the macroscopic electric and magnetic fields, \( \mathbf{D} \) and \( \mathbf{B} \) are the electric displacement and magnetic induction fields, \( \mathbf{J} \) is the electric current density, and \( \mathbf{K} \) is a fictitious magnetic current density. Throughout our work, we assume linear time-invariant materials that may be dispersive and/or anisotropic (Eq. (7)).

The FDTD algorithm discretizes Eq. (25) and Eq. (26) using the Yee grid [49], formulating an implicit “timestep operator” [16], \( \tilde{T}_0 \), that iteratively updates the fields after every timestep, \( n \), spaced by \( \Delta t \), such that

\[
\mathbf{x}^{n+1} = \tilde{T}_0 \mathbf{x}^n + \mathbf{s}^n,
\]

where \( \mathbf{x}^n \) represents all the fields (E, D, H, and B) at timestep, \( n \), and \( \mathbf{s}^n \) are the source terms from that timestep.

While the FDTD algorithm computes the fields in the time domain, the equivalent frequency-domain response, \( \tilde{\mathbf{x}}(\omega) \), can be obtained by accumulating a discrete-time Fourier transform (DTFT) at a frequency \( \omega \) after each timestep:

\[
\tilde{\mathbf{x}}(\omega) = \sum_n e^{i\omega n \Delta t} \mathbf{x}(n \Delta t) \Delta t.
\]
Accumulating the DTFT only at the desired frequencies during time-stepping allows one to avoid the expensive storage that would be required to save the fields at every monitor point and at every timestep and then Fourier transform these time-domain fields in post-processing [16].

Two criteria must be met in order for the results of the DTFT to accurately approximate the results of an explicit frequency-domain solver. First, the Fourier-transformed fields must converge after time-stepping. As the DTFT is updated after every timestep, any particular Fourier component will continue to update so long as its time-domain equivalent has not decayed to a negligible value due to absorption (by materials or by absorbing boundaries [48]). Sufficient runtime is especially important when designing highly resonant devices or explicitly specifying adjacent sources with long time profiles (due to dense frequency spacing, Sec. 5.2).

Second, since the FDTD algorithm approximates the continuous time-derivative operator, \( \frac{\partial}{\partial t} \), using a finite-difference operator, \( \hat{D} \), the corresponding frequency term, \( \omega \), requires correction to retain second-order accuracy in \( \Delta t \). Specifically, the time discretization is described by

\[
\left. \frac{\partial x}{\partial t} \right|_{\Delta t/2} \approx \hat{D} x = \frac{x(\Delta t) - x(0)}{\Delta t}.
\]  

(28)

While a continuous time-harmonic field \( x(t) = e^{-i\omega t} x(0) \) would have the following DTFT relationship:

\[
\frac{\partial x}{\partial t} \xrightarrow{\text{DTFT}} -i\omega \hat{x}(\omega),
\]

(29)

the DTFT relationship for the finite-difference operator is described by

\[
\hat{D} x \xrightarrow{\text{DTFT}} e^{-i\omega \Delta t} - 1 \Delta t \frac{\hat{x}(\omega)}{-i\hat{\omega}},
\]

(30)

where \( \hat{\omega} \) is an adjusted frequency term, which can be equivalently expressed as

\[-i\hat{\omega} = -i\omega + O(\Delta t) .
\]

As expected, the two transforms agree as \( \Delta t \rightarrow 0 \).

We now describe the operator, \( A(\rho, \hat{\omega}_m) \), that corresponds to the steady-state field evolution calculated using the FDTD algorithm and the DTFT:

\[
\begin{bmatrix}
-j\hat{\omega} & \nabla \times \\
1 & -\epsilon_0 \epsilon_r(\omega, \rho) \\
-\nabla \times & -j\hat{\omega} \\
1 & -\mu_0
\end{bmatrix}
\begin{bmatrix}
\hat{x} \\
\hat{D} \\
\hat{E} \\
\hat{B}
\end{bmatrix}
= \begin{bmatrix}
\hat{J} \\
0 \\
0 \\
0
\end{bmatrix}.
\]

(32)

To compute the gradient of an objective function, \( f_n(x) \), satisfying the constraint \( A(\rho, \hat{\omega}_m) \hat{x} = \hat{b} \), an adjoint-variable formulation is used such that

\[
\frac{\partial f_n}{\partial \rho} = \hat{\lambda}^T \frac{\partial A(\rho, \hat{\omega})}{\partial \rho} \hat{x},
\]

(33)

where \( \hat{x} \) are the fields stored after the forward simulation and \( \hat{\lambda} \) are the fields stored after the adjoint simulation, with sources determined by both the objective function and results of the forward run [15, Sec. 8.7].
Since only the permittivity (and not the permeability) varies with \( \rho \), only the electric fields need to be stored during the forward \( \hat{E}_f \) and adjoint \( \hat{E}_a \) solves. The above inner product becomes a matrix-tensor-matrix product yielding the gradient at each point in space:

\[
\frac{\partial f_n}{\partial \rho} = -\hat{E}^T_a(\rho, \omega) \frac{\partial e_r(\rho, \omega)}{\partial \rho} \hat{E}_f(\rho, \omega). \tag{34}
\]

This analysis produces sufficiently accurate gradients provided the interpolation and restriction steps (due to the Yee grid) are properly implemented (Sec. 3). That being said, we note that improvements could still be made by implementing a true “discretize-then-differentiate” scheme [78]. Specifically, the adjoint method should explicitly account for the timestepping operator, \( \hat{T}_0 \), and not just the effects of the discrete-time operator. The approach taken here mixes the two methodologies (the other being “differentiate-then-discretize”) in order to simplify the implementation. Future efforts could consolidate the approach while simultaneously integrating important features, like subpixel smoothing [21, 79], further reducing inaccuracies and artifacts that stem from the discretization.

**Funding.** AMH was supported by the Department of Defense (DoD) through the National Defense Science & Engineering Graduate (NDSEG) Fellowship Program. AO was supported by the National Science Foundation (NSF) via Small Business Innovation Research (SBIR) Phase I and II awards 1647206 and 1758596. MC, ZL, and SGJ were supported in part by the U.S. Army Research Office through the Institute for Soldier Nanotechnologies (Award No. W911NF-18-2-0048), and by the Simons Foundation collaboration on Extreme Wave Phenomena. SER was supported by the Georgia Electronic Design Center of the Georgia Institute of Technology.

**Acknowledgments.** The authors are grateful to M.T. Homer Reid for initial work on this project. The authors would also like to thank Rasmus E. Christiansen, Andrew S. Michaels, and Owen D. Miller for useful discussions. This research was supported in part through research cyberinfrastructure resources and services provided by the Partnership for an Advanced Computing Environment (PACE) at the Georgia Institute of Technology, Atlanta, Georgia, USA.

**Disclosures.** The authors declare no conflicts of interest.

**Data availability.** Data underlying the results presented in this paper are available in Ref. 80.

**References**

1. S. Molesky, Z. Lin, A. Y. Piggott, W. Jin, J. Vucković, and A. W. Rodriguez, “Inverse design in nanophotonics,” Nat. Photonics 12, 659–670 (2018).
2. E. W. Wang, D. Sell, T. Phan, and J. A. Fan, “Robust design of topology-optimized metasurfaces,” Opt. Mater. Express 9, 469–482 (2019).
3. R. E. Christiansen, Z. Lin, C. Roques-Carmes, Y. Salamin, S. E. Kooi, J. D. Ioannopoulos, M. Soljačić, and S. G. Johnson, “Fullwave maxwell inverse design of axisymmetric, tunable, and multi-scale multi-wavelength metasurfaces,” Opt. Express 28, 33854–33868 (2020).
4. R. E. Christiansen, J. Michon, M. Benzaouia, O. Sigmund, and S. G. Johnson, “Inverse design of nanoparticles for enhanced raman scattering,” Opt. Express 28, 4444–4462 (2020).
5. F. Wang, R. E. Christiansen, Y. Yu, J. Mørk, and O. Sigmund, “Maximizing the quality factor to mode volume ratio for ultra-small photonic crystal cavities,” Appl. Phys. Lett. 113, 241101 (2018).
6. A. Y. Piggott, J. Lu, K. G. Lagoudakis, J. Petykiewicz, T. M. Babinec, and J. Vucković, “Inverse design and demonstration of a compact and broadband on-chip wavelength demultiplexer,” Nat. Photonics 9, 374–377 (2015).
7. K. Y. Yang, J. Skarda, M. Cotrufo, A. Dutt, G. H. Ahn, M. Sawably, D. Vercruyssse, A. Arabbian, S. Fan, A. Alù, and J. Vucković, “Inverse-designed non-reciprocal pulse router for chip-based lidar,” Nat. Photonics 14, 369–374 (2020).
8. L. Su, A. Y. Piggott, N. V. Supra, J. Petykiewicz, and J. Vuckovic, “Inverse design and demonstration of a compact on-chip narrowband three-channel wavelength demultiplexer,” ACS Photonics 5, 301–305 (2018).
9. L. Su, R. Trivedi, N. V. Supra, A. Y. Piggott, D. Vercruysse, and J. Vucković, “Fully-automated optimization of grating couplers,” Opt. Express 26, 4023–4034 (2018).
10. C. Dory, D. Vercruysse, K. Y. Yang, N. V. Supra, A. E. Rugar, S. Sun, D. M. Lukin, A. Y. Piggott, J. L. Zhang, M. Radulaski, K. G. Lagoudakis, L. Su, and J. Vucković, “Inverse-designed diamond photonics,” Nat. Commun. 10, 1–7 (2019).
11. A. Y. Piggott, E. Y. Ma, L. Su, G. H. Ahn, N. V. Supra, D. Vercruysse, A. M. Netherton, A. S. Khope, J. E. Bowers, and J. Vuckovic, “Inverse-designed photonics for semiconductor foundries,” ACS Photonics 7, 569–575 (2020).
74. T. W. Hughes, M. Minkov, I. A. D. Williamson, and S. Fan, “Adjoint method and inverse design for nonlinear nanophotonic devices,” ACS Photonics 5, 4781–4787 (2018).
75. Y. Shi, W. Shin, and S. Fan, “Multi-frequency finite-difference frequency-domain algorithm for active nanophotonic device simulations,” Optica 3, 1256–1259 (2016).
76. S. Esterhazy, D. Liu, M. Liertzer, A. Cerjan, L. Ge, K. Makris, A. Stone, J. Melenk, S. Johnson, and S. Rotter, “Scalable numerical approach for the steady-state ab initio laser theory,” Phys. Rev. A 90, 023816 (2014).
77. Z. Lin, X. Liang, M. Lončar, S. G. Johnson, and A. W. Rodriguez, “Cavity-enhanced second-harmonic generation via nonlinear-overlap optimization,” Optica 3, 233–238 (2016).
78. M. D. Gunzburger, Perspectives in Flow Control and Optimization (SIAM, 2002).
79. A. Farjadpour, D. Roundy, A. Rodriguez, M. Ibanescu, P. Bermel, J. D. Joannopoulos, S. G. Johnson, and G. Burr, “Improving accuracy by subpixel smoothing in FDTD,” Opt. Lett. 31, 2972–2974 (2006).
80. “Meep software package,” https://github.com/NanoComp/meep (2021).