Supporting Information

Forward-mode Differentiation of Maxwell's Equations

Tyler W. Hughes, Ian A. D. Williamson, Momchil Minkov, Shanhui Fan

I. FREQUENCY-DOMAIN FORWARD-MODE DIFFERENTIATION

In a non-magnetic material with $\mu = \mu_0$, Maxwell's equations at steady state when driven by frequency ω are described as

$$\nabla \times \nabla \times \boldsymbol{e} - \left(\frac{\omega}{c_0}\right)^2 \epsilon \boldsymbol{e} = i\omega \boldsymbol{j}, \tag{S1}$$

where $\boldsymbol{e}$ is now a phasor and the physical electric fields have time dependence $\mathcal{R}\left\{\boldsymbol{e}\ e^{i\omega t}\right\}$. The magnetic fields $\boldsymbol{h}$ may be found by applying Maxwell's equations to the electric field solution $\boldsymbol{e}$.

Eq. (S1) is typically written in the more general form

$$A(\epsilon)\boldsymbol{e} = \boldsymbol{b}, \tag{S2}$$

which is solved for $\boldsymbol{e}$ by finding

$$\boldsymbol{e} = A(\epsilon)^{-1}\boldsymbol{b}, \tag{S3}$$

As in the time domain case, let us now consider a function $\boldsymbol{F}(\boldsymbol{\phi})$ where the i-th element is computed from the electric field distribution through $F_i = f_i(\boldsymbol{e}(t))$. We now compare the computation of $\frac{d\boldsymbol{F}}{d\boldsymbol{\phi}}$ through the adjoint and FMD methods. To do this, following a treatment given in (8), we apply the chain rule to obtain

$$\frac{d\boldsymbol{F}}{d\boldsymbol{\phi}} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{e}} \cdot \frac{d\boldsymbol{e}}{d\boldsymbol{\phi}} + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{e}^*} \cdot \frac{d\boldsymbol{e}^*}{d\boldsymbol{\phi}} \tag{S4}$$

$$= 2\mathcal{R}\left\{\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{e}} \cdot \frac{d\boldsymbol{e}}{d\boldsymbol{\phi}}\right\} \tag{S5}$$

$$= -2\mathcal{R}\left\{\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{e}} \cdot A^{-1}\frac{\partial A}{\partial \boldsymbol{\phi}}\boldsymbol{e}\right\} \tag{S6}$$

In the adjoint method, Eq. (S6) is solved by applying a transpose and then evaluating the adjoint field

$$\boldsymbol{e}_{\text{adj}} = -A^{-T}\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{e}}^T \tag{S7}$$

before plugging in the result to obtain

$$\frac{d\boldsymbol{F}}{d\boldsymbol{\phi}}^{(\text{adj})} = 2\mathcal{R}\left\{\boldsymbol{e}_{\text{adj}}^T\frac{\partial A}{\partial \boldsymbol{\phi}}\boldsymbol{e}\right\} \tag{S8}$$

In FMD, however, we directly apply A^{-1} to the right of this expression, solving for the FMD derivative field

$$\boldsymbol{e}_{\text{FMD}} = A^{-1}\frac{\partial A}{\partial \boldsymbol{\phi}}\boldsymbol{e} \tag{S9}$$

and expressing the final result as

$$\frac{d\boldsymbol{F}}{d\boldsymbol{\phi}}^{(\text{FMD})} = -2\mathcal{R}\left\{\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{e}} \cdot \boldsymbol{e}_{\text{FMD}}\right\} \tag{S10}$$

The correspondence with the time domain formalism of FMD given in the main text is apparent by inspetion.

II. DERIVATION OF ADJOINT SENSITIVITY FOR FDTD

In this Section, we derive the form of the adjoint sensitivity that was used in Section of the main text. As before, we wish to compute the Jacobian of function $\boldsymbol{f}$ with respect to its inputs $\boldsymbol{\phi}$, defined as follows

$$\boldsymbol{F}(\boldsymbol{\phi}) = \int_0^T dt\ \boldsymbol{f}(\boldsymbol{u}(t), t) \tag{S11}$$

where the vector $\boldsymbol{u}(t)$ is given by the solution to an FDTD simulation, defined as the constraint.

$$\boldsymbol{g}(\dot{\boldsymbol{u}}, u, \boldsymbol{\phi}, t) = A(\boldsymbol{\phi}) \cdot \dot{\boldsymbol{u}}(t) + B \cdot \boldsymbol{u}(t) + \boldsymbol{c}(t) = 0. \tag{S12}$$

For convenience we set the initial condition $\dot{\boldsymbol{u}}(0) = \boldsymbol{u}(0) = 0$. As shown in the main text, the adjoint method must be applied once to compute the derivative of each output of $\boldsymbol{F}$. Therefore, for convenience of notation, we consider a scalar function F, with the understanding that the same method may be applied to each output of a vector function $\boldsymbol{F}$.

Here we derive the gradient of F using the adjoint method by an application of Lagrange multipliers. One first defines the Lagrangian

$$\mathcal{L} = \int_0^T dt \, \left[f(\boldsymbol{u}(t, \boldsymbol{\phi}), t) + \boldsymbol{\lambda}(t)^T \boldsymbol{g}(\dot{\boldsymbol{u}}, \boldsymbol{u}, \boldsymbol{\phi}, t) \right], \tag{S13}$$

where $\lambda(t)$ is a vector of Lagrange multipliers. We note that, when the constraints are satisfied, $\boldsymbol{g} = 0$, which means $\mathcal{L} = F$ and $\frac{d\mathcal{L}}{d\boldsymbol{\phi}} = \frac{dF}{d\boldsymbol{\phi}}$. Differentiating Eq. (S13) with respect to $\boldsymbol{\phi}$ gives the following

$$\frac{d\mathcal{L}}{d\boldsymbol{\phi}} = \int_0^T dt \, \left[\frac{\partial f}{\partial \boldsymbol{u}} \frac{d\boldsymbol{u}}{d\boldsymbol{\phi}} + \boldsymbol{\lambda}^T \frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}} \frac{d\dot{\boldsymbol{u}}}{d\boldsymbol{\phi}} + \boldsymbol{\lambda}^T \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}} \frac{d\boldsymbol{u}}{d\boldsymbol{\phi}} + \boldsymbol{\lambda}^T \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\phi}} \right] \tag{S14}$$

The second term may be integrated by parts as follows

$$\int_0^T dt \, \boldsymbol{\lambda}^T \frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}} \frac{d\dot{\boldsymbol{u}}}{d\boldsymbol{\phi}} = \boldsymbol{\lambda}^T \frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}} \frac{d\boldsymbol{u}}{d\boldsymbol{\phi}} \Big|_0^T - \int_0^T dt \, \left[\dot{\boldsymbol{\lambda}}^T \frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}} + \boldsymbol{\lambda}^T \frac{d}{dt} \frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}} \right] \frac{d\boldsymbol{u}}{d\boldsymbol{\phi}}, \tag{S15}$$

which, when reinserted into Eq. (S14), gives

$$\frac{d\mathcal{L}}{d\boldsymbol{\phi}} = \int_0^T dt \, \left[\left(\frac{\partial f}{\partial \boldsymbol{u}} - \dot{\boldsymbol{\lambda}}^T \frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}} + \boldsymbol{\lambda}^T \left[\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}} - \frac{d}{dt} \frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}} \right] \right) \frac{d\boldsymbol{u}}{d\boldsymbol{\phi}} + \boldsymbol{\lambda}^T \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\phi}} \right] + \boldsymbol{\lambda}^T \frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}} \frac{d\boldsymbol{u}}{d\boldsymbol{\phi}} \Big|_0^T \tag{S16}$$

We now wish to choose $\boldsymbol{\lambda}$ to eliminate terms of $\frac{du}{d\phi}$ from our expression, as these terms are not directly computable. First, we choose the condition $\boldsymbol{\lambda}(T) = \mathbf{o}$, which, along with our original condition of $\boldsymbol{u}(0) = \mathbf{o}$, eliminates the term outside of the integral. To handle the terms within the integral, we may express $\boldsymbol{\lambda}(t)$ as the solution to

$$\dot{\boldsymbol{\lambda}}^T \frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}} - \boldsymbol{\lambda}^T \left(\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}} - \frac{d}{dt} \frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}} \right) - \frac{\partial f}{\partial \boldsymbol{u}} = \mathbf{o}^T. \tag{S17}$$

Plugging this form of $\boldsymbol{\lambda}$ into Eq. (S16) gives the following expression for the gradient

$$\frac{d\mathcal{L}}{d\boldsymbol{\phi}} = \frac{dF}{d\boldsymbol{\phi}} = \int_0^T dt \, \left[\boldsymbol{\lambda}^T(t) \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\phi}}(t) \right]. \tag{S18}$$

In practice, rather than solving Eq. (S17) backwards in time from $\boldsymbol{\lambda}(T) = \mathbf{o}$ to $\boldsymbol{\lambda}(0)$, we may instead define the time-reversed Lagrange multipliers $\tilde{\boldsymbol{\lambda}}(t) \equiv \boldsymbol{\lambda}(T-t)$ which have the initial condition $\tilde{\boldsymbol{\lambda}}(0) = \mathbf{o}$ and may be solved forward in time. After substituting $\tilde{\boldsymbol{\lambda}}$ into Eq. (S17), and applying a transpose, the time-reversed Lagrange multipliers are the solution to

$$\frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}}^T \dot{\tilde{\boldsymbol{\lambda}}} + \left(\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}}^T - \frac{d}{dt} \frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}}^T \right) \tilde{\boldsymbol{\lambda}} + \frac{\partial f}{\partial \boldsymbol{u}}^T = 0. \tag{S19}$$

Importantly, we note that while $\tilde{\boldsymbol{\lambda}}$ is evaluated at time t in Eq. (S19), all other terms are evaluated at time $T-t$ as they were not time reversed. Finally, in terms of $\tilde{\boldsymbol{\lambda}}$, the gradient is

$$\frac{dF}{d\boldsymbol{\phi}} = \int_0^T dt \, \left[\tilde{\boldsymbol{\lambda}}^T(T-t) \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\phi}}(t) \right], \tag{S20}$$

or, written in terms of the original adjoint solution

$$\frac{dF}{d\boldsymbol{\phi}} = \int_0^T dt \, \left[\boldsymbol{\lambda}^T(t) \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\phi}}(t) \right], \tag{S21}$$

A. Application to Maxwell's Equations

We now evaluate Eq. (S20) for a system obeying Maxwell's equations. We take our unknown to be the set of electric and magnetic fields, $\boldsymbol{u}(t) = [\boldsymbol{g}(t),\ \boldsymbol{e}(t)]^T$, and may derive the following expression for the constraint $\boldsymbol{g}(\dot{\boldsymbol{u}}, \boldsymbol{u}, \boldsymbol{\phi}, t)$

$$\boldsymbol{g}(\dot{\boldsymbol{u}}, \boldsymbol{u}, \boldsymbol{\phi}, t) = A(\boldsymbol{\phi}) \cdot \dot{\boldsymbol{u}}(t) + B \cdot \boldsymbol{u}(t) + \boldsymbol{c}(t) = \mathbf{o} \tag{S22}$$

$$= \begin{bmatrix} -\mu & 0 \\ 0 & \epsilon(\boldsymbol{\phi}) \end{bmatrix} \cdot \begin{bmatrix} \dot{\boldsymbol{g}} \\ \dot{\boldsymbol{e}} \end{bmatrix} + \begin{bmatrix} -\sigma_H & \nabla\times \\ \nabla\times & -\sigma_E \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{g} \\ \boldsymbol{e} \end{bmatrix} + \begin{bmatrix} \mathbf{o} \\ \boldsymbol{j} \end{bmatrix} = \mathbf{o} \tag{S23}$$

This is assuming a time-independent permittivity, permeability and conductivity, although the same analysis can be applied in the time-dependent case. Note also that we have assumed that only the only $\boldsymbol{\phi}$ dependence appears in the permittivity distribution $\epsilon(\phi)$, although this can be generalized to other degrees of freedom, such as the current source, J, or conductivity distributions, σ, without complication.

The derivatives of $\boldsymbol{g}(\dot{\boldsymbol{u}}, \boldsymbol{u}, \boldsymbol{\phi}, t)$ are needed in the adjoint and gradient equations of Eq. (S19) and Eq. (S20), which are given by

$$\frac{\partial \boldsymbol{g}}{\partial \dot{\boldsymbol{u}}} = A(\boldsymbol{\phi}) = \begin{bmatrix} -\mu & 0 \\ 0 & \epsilon(\boldsymbol{\phi}) \end{bmatrix} \tag{S24}$$

$$\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}} = B = \begin{bmatrix} -\sigma_H & \nabla\times \\ \nabla\times & -\sigma_E \end{bmatrix} \tag{S25}$$

$$\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\phi}} = \frac{\partial A}{\partial \boldsymbol{\phi}} \dot{\boldsymbol{u}} = \begin{bmatrix} 0 & 0 \\ 0 & \epsilon' \end{bmatrix} \cdot \begin{bmatrix} \dot{H} \\ \dot{E} \end{bmatrix} = \begin{bmatrix} 0 \\ \epsilon' \cdot \dot{E} \end{bmatrix} \tag{S26}$$

where $\epsilon' \equiv \frac{\partial \epsilon}{\partial \boldsymbol{\phi}}$ defines how the permittivity distribution changes with a change of parameters $\boldsymbol{\phi}$. With these, the expression for the time-reversed Lagrange multipliers $\tilde{\boldsymbol{\lambda}}$ from Eq. (S19) becomes

$$A^T \cdot \dot{\tilde{\boldsymbol{\lambda}}} + B^T \cdot \tilde{\boldsymbol{\lambda}} + \frac{\partial f}{\partial \boldsymbol{u}}^T = 0, \tag{S27}$$

To express these in terms of Maxwell's Equations, we define the electromagnetic adjoint fields $\boldsymbol{\lambda}(t) \equiv [\boldsymbol{h}_{\text{adj}},\ \boldsymbol{e}_{\text{adj}}]^T$ and their evaluations at $T - t$ given by $\tilde{\boldsymbol{\lambda}}(t) \equiv [\tilde{\boldsymbol{g}}_{\text{adj}}(t),\ \tilde{\boldsymbol{e}}_{\text{adj}}(t)]^T$. The latter are given by the solution to Eq. (S27), written with A^T and B^T from by Maxwell's equations as

$$\begin{bmatrix} -\mu^T & 0 \\ 0 & \epsilon(\boldsymbol{\phi})^T \end{bmatrix} \cdot \begin{bmatrix} \dot{\tilde{\boldsymbol{g}}}(\tau) \\ \dot{\tilde{\boldsymbol{e}}}(\tau) \end{bmatrix} + \begin{bmatrix} -\sigma_H^T & \nabla\times \\ \nabla\times & -\sigma_E^T \end{bmatrix} \cdot \begin{bmatrix} \tilde{\boldsymbol{g}}(\tau) \\ \tilde{\boldsymbol{e}}(\tau) \end{bmatrix} + \begin{bmatrix} (\partial f / \partial \boldsymbol{h})^T (T - \tau) \\ (\partial f / \partial \boldsymbol{e})^T (T - \tau) \end{bmatrix} = \mathbf{o} \tag{S28}$$

Finally, in terms of $\tilde{\boldsymbol{g}}$ and $\tilde{\boldsymbol{e}}$, the gradient can be expressed as

$$\frac{dF}{d\boldsymbol{\phi}} = \int_0^T dt \left[\boldsymbol{\lambda}(t)^T \frac{\partial \boldsymbol{g}(t)}{\partial \boldsymbol{\phi}} \right] \tag{S29}$$

$$= \int_0^T dt \left[\tilde{\boldsymbol{\lambda}}(T - t)^T \frac{\partial A}{\partial \boldsymbol{\phi}} \dot{\boldsymbol{u}}(t) \right] \tag{S30}$$

$$= \int_0^T dt \left[\Big[\boldsymbol{h}_{\text{adj}}(t),\ \boldsymbol{e}_{\text{adj}}(t) \Big]^T \cdot \begin{bmatrix} 0 \\ \epsilon' \cdot \dot{\boldsymbol{e}}(t) \end{bmatrix} \right] \tag{S31}$$

$$= \int_0^T dt \left[\boldsymbol{e}_{\text{adj}}(t) \cdot \epsilon' \cdot \dot{\boldsymbol{e}}(t) \right] \tag{S32}$$

$$. \tag{S33}$$

Practically, when solving these equations numerically, it can be more convenient to put this in a slightly different form. By integrating by parts, one can move the time derivative to the adjoint field

$$\frac{dF}{d\boldsymbol{\phi}} = \int_0^T dt \left[\boldsymbol{e}_{\text{adj}}(t) \cdot \epsilon' \cdot \dot{\boldsymbol{e}}(t) \right] \tag{S34}$$

$$= -\int_0^T dt \left[\dot{\boldsymbol{e}}_{\text{adj}}(t) \cdot \epsilon' \cdot \boldsymbol{e}(t) \right] - \boldsymbol{e}_{\text{adj}}(t) \cdot \epsilon' \cdot \boldsymbol{e}(t) \Big|_0^T \tag{S35}$$

$$= -\int_0^T dt \left[\dot{\boldsymbol{e}}_{\text{adj}}(t) \cdot \epsilon' \cdot \boldsymbol{e}(t) \right] - \cancel{\boldsymbol{e}_{\text{adj}}(T)} \cdot \epsilon' \cdot \boldsymbol{e}(T) - \boldsymbol{e}_{\text{adj}}(0) \cdot \epsilon' \cdot \cancel{\boldsymbol{e}(0)} \tag{S36}$$

$$= -\int_0^T dt \left[\dot{\boldsymbol{e}}_{\text{adj}}(t) \cdot \epsilon' \cdot \boldsymbol{e}(t) \right]. \tag{S37}$$

where the last line comes from the boundary conditions of $\boldsymbol{e}(0) = \boldsymbol{e}_{\text{adj}}(T) = \mathbf{o}$.

It is straightforward to show that $\dot{\boldsymbol{e}}_{\text{adj}}(t)$ can be identified as the electric fields (no time derivative) found when running the adjoint problem with source $\frac{d}{dt}\frac{\partial f}{\partial \boldsymbol{u}}$ instead of $\frac{\partial f}{\partial \boldsymbol{u}}$, which is more practical when dealing with the algorithm where computing time derivatives requires careful treatment of finite-difference derivatives.

III. EFFICIENT COMPUTATION OF THE ADJOINT INTEGRAL

To compute the adjoint sensitivity, one must compute Eq. (14), copied below as

$$\frac{dF_i}{d\boldsymbol{\phi}} = \int_0^T dt \; \boldsymbol{\lambda}_i(t)^T \cdot \frac{\partial A}{\partial \boldsymbol{\phi}} \cdot \boldsymbol{u}(t), \tag{S38}$$

where $\boldsymbol{\lambda}$ is the adjoint field, $\boldsymbol{u}(t)$ is the forward field and $\frac{\partial A}{\partial \boldsymbol{\phi}}$ is a rank three tensor of dimension $(N \times m \times N)$ and N is the number of grid points in the domain.

Computed naively, this integral would amount to a time complexity of $\mathcal{O}(N^2 T m)$ and the computation of the full Jacobian would be $\mathcal{O}(N^2 T m n)$. However, in practice, we may eliminate the m-dependence of this integral, which results in the standard constant scaling of the adjoint method with respect to the number of inputs.

To do this, we first write Eq. (S39) in index notation, giving

$$\left(\frac{dF}{d\boldsymbol{\phi}} \right)_k = \int_0^T dt \; \sum_{i,j} \lambda_i \frac{\partial A_{ij}}{\partial \phi_k} u_j. \tag{S39}$$

We note from Eq. (2) that A is diagonal, and therefore we may make the substitution

$$\frac{\partial A_{ij}}{\partial \phi_k} = \delta_{ij} \frac{d a_i}{d \phi_k}, \tag{S40}$$

where a_i is the value of A_{ii}, corresponding to either a relative permittivity or permeability. Substituting into Eq. (S39) gives

$$\left(\frac{dF}{d\boldsymbol{\phi}} \right)_k = \int_0^T dt \; \sum_{i,j} \lambda_i \frac{\partial A_{ij}}{\partial \phi_k} u_j \tag{S41}$$

$$= \int_0^T dt \; \sum_{i,j} \lambda_i \delta_{ij} \frac{d a_i}{d \phi_k} u_j \tag{S42}$$

$$= \int_0^T dt \; \sum_{i} \frac{d a_i}{d \phi_k} \lambda_i u_i \tag{S43}$$

$$= \int_0^T dt \; \frac{d \boldsymbol{a}}{d \boldsymbol{\phi}}^T \cdot \left(\boldsymbol{\lambda}(t) \odot \boldsymbol{u}(t) \right) \tag{S44}$$

$$= \frac{d \boldsymbol{a}}{d \boldsymbol{\phi}}^T \cdot \left[\int_0^T dt \; \left(\boldsymbol{\lambda}(t) \odot \boldsymbol{u}(t) \right) \right], \tag{S45}$$

where $\odot$ is element-wise vector multiplication and $\boldsymbol{a}$ refers to the vector along the diagonal of A. Thus, we may first perform $\int_0^T dt\ \boldsymbol{\lambda}(t) \odot \boldsymbol{u}(t)$ in $\mathcal{O}(NT)$ time. In general, the matrix $\frac{d\boldsymbol{a}}{d\boldsymbol{\phi}}^T$ is sparse because each free parameter ϕ_k will only affect some subset of the spatial domain. This means that the complexity of the multiplication in Eq. (S45) will be between $\mathcal{O}(NT)$ and $\mathcal{O}(NTm)$, but closer to $\mathcal{O}(NT)$ in general. Thus, the complexity of the total adjoint calculation is $\mathcal{O}(NTn)$, as expected.

IV. SCALING COMPARISON OF METHODS

We now examine how the speed and memory scaling of each of these methods compare methods compare as a function of the simulation parameters. As before, we assume a function F with m input parameters and n output parameters. The evaluation of F involves running an FDTD simulation containing N points in the spatial grid and T time steps. In each case, the storage of Jacobian itself requires a memory storage of $\mathcal{O}(mn)$ and a single FDTD run may be completed in $\mathcal{O}(NT)$ time complexity and requires a memory storage of $\mathcal{O}(N)$.

In the finite-difference approach, one must perform one independent FDTD simulation per input parameter. This leads to a time complexity of $\mathcal{O}(NTm)$. The finite-difference calculation itself requires storage of a column of the Jacobian and the subtraction of two vectors of length m, which adds no additional overhead in the memory complexity, which scales as $\mathcal{O}(N + mn)$.

In FMD, one must first compute and store the full $\boldsymbol{u}(t)$ sequence. Then, for each input parameter, one FDTD simulation must be performed along with the integral in Eq. (5), which gives a time complexity of $\mathcal{O}(NT(n + m))$. The corresponding memory complexity is $\mathcal{O}(NT + mn)$

For the adjoint method, one must also compute and store the full $\boldsymbol{u}(t)$ sequence. Then, for each *output* parameter, an adjoint FDTD simulation must be performed and the integral in Eq. (14) must be computed.

For the adjoint case, without any special techniques applied, one must store the entire forward field solution over time. This results in a memory cost that scales as $\mathcal{O}(NT)$. Although the final gradient computation requires knowledge of the tensor, ϵ', which is of size $N \times P \times N$, there is no explicit need to store the full tensor in memory, so we ignore this contribution.

By contrast, the numerical derivative only requires storage of the electric fields at each time step, which is already required for the FDTD simulation, as well as the final figures of merit for each design parameter. This results in a memory storage of $\mathcal{O}(N) + \mathcal{O}(P)$.

Like the numerical derivative, the forward difference approach requires running P additional FDTD simulations. However, in this case, one must use the forward field solution to construct a source for each of these simulations. Therefore, in a naive implementation, one must run each of these simulations in parallel with the forward simulation, resulting in a memory storage of $\mathcal{O}(NP)$. However, this memory storage is independent of T, which may result in significant improvements in memory requirements in the case that the number of parameters is far fewer than the number of time steps.

A. Time scaling

We now examine the computational time complexity required for each method. The calculation of $u(t)$ requires a full FDTD simulation, which has a time complexity of $\mathcal{O}(NT)$. The first step of the adjoint gradient requires running two FDTD simulations to compute the forward and adjoint fields. Therefore, this step has a time complexity of $\mathcal{O}(NT)$. The integral needed to compute the full sensitivity of Eq. (14), while generally requiring $\mathcal{O}(NTP)$ operations, can be neglected in almost all practical considerations, as discussed in the Appendix Material. Therefore, the time complexity of the adjoint gradient is $\mathcal{O}(NT)$.

To compute the derivative using finite differences, one must run P independent FDTD simulations then compute the finite difference derivative of numerical derivative expression from Eq. (4). This gives a time complexity of $\mathcal{O}(NTP)$.

The forward difference approach also requires running P independent simulations. To compute the gradient, one may integrate Eq. (5), within the FDTD loop, which does not add additional scaling to the time complexity. Therefore, like numerical derivatives, the forward differentiation method has a time complexity of $\mathcal{O}(NTP)$.

B. Summary

A summary of the time and memory requirements of each method are given in Table S1.

Method	**Time Complexity**	**Memory Complexity**
Numerical	$\mathcal{O}(NTm)$	$\mathcal{O}(N)$
FMD	$\mathcal{O}(NTm)$	$\mathcal{O}(NT + Nn)$
Adjoint	$\mathcal{O}(NTn)$	$\mathcal{O}(NT + Nm)$

TABLE S1. Comparison of memory and speed complexity of the three different gradient computation methods examined in this work. N is the number of grid cells. T is the number of time steps. m and n are the number of input and output variables of the function F. We assume n and m are each less than or equal to N.