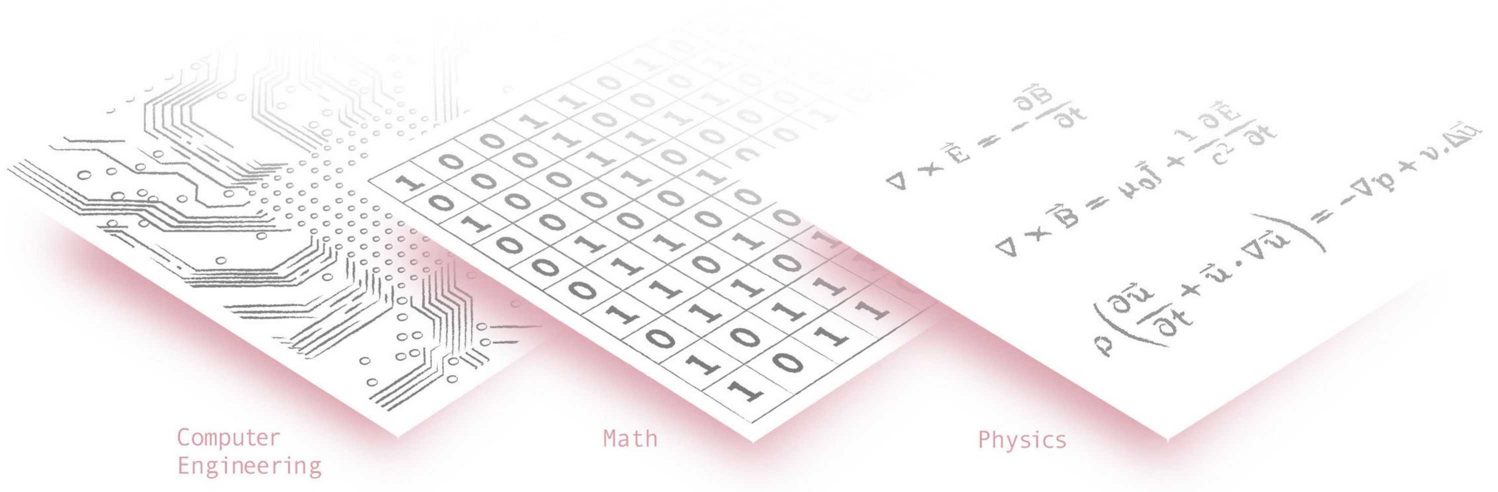


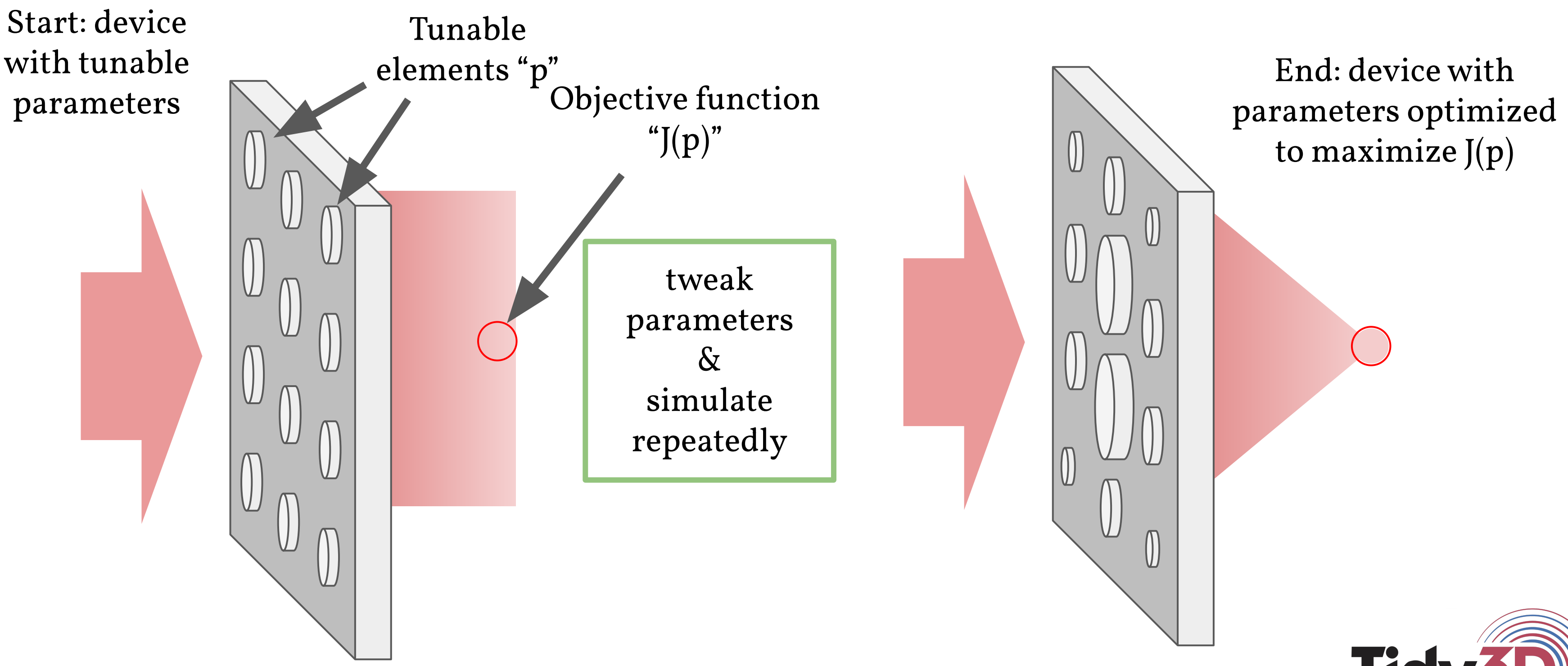
Inverse Design in Photonics

Tutorial 1: Introduction

FLEXCOMPUTE

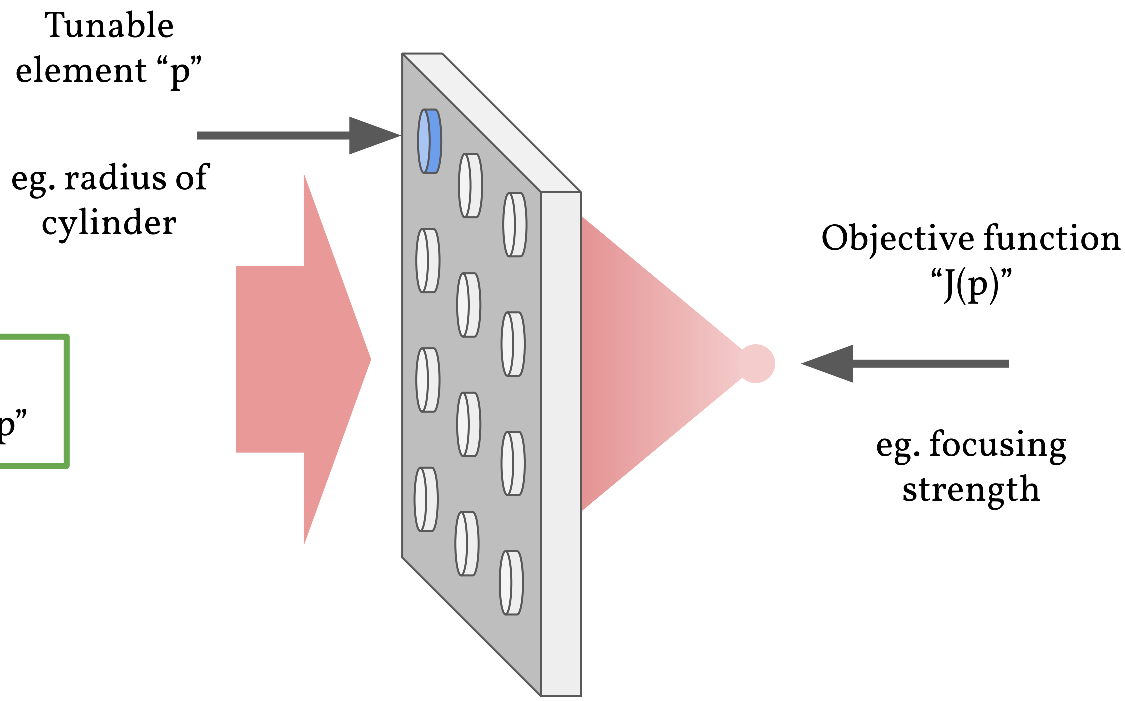
Tidy3D

Introduction to Computational Design



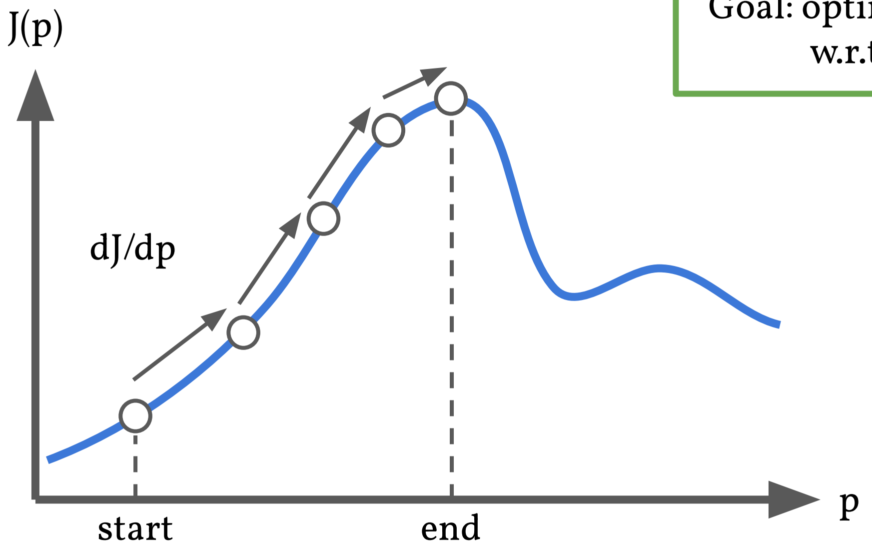
[1] Hughes, Tyler W., et al. *Applied Physics Letters* 119.15 (2021): 150502.
[2] Mansouree, Mahdad, et al. *ACS Photonics* 8.2 (2021): 455-463.

Sensitivity Analysis



Goal: compute sensitivity "dJ/dp"

Optimization (single element)



Goal: optimize J(p) w.r.t p

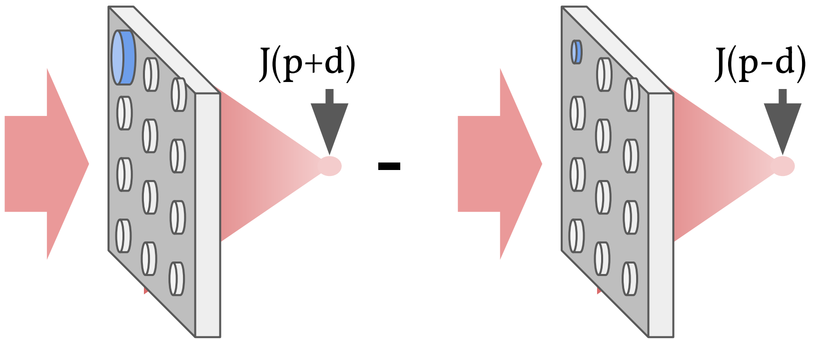
Approach:

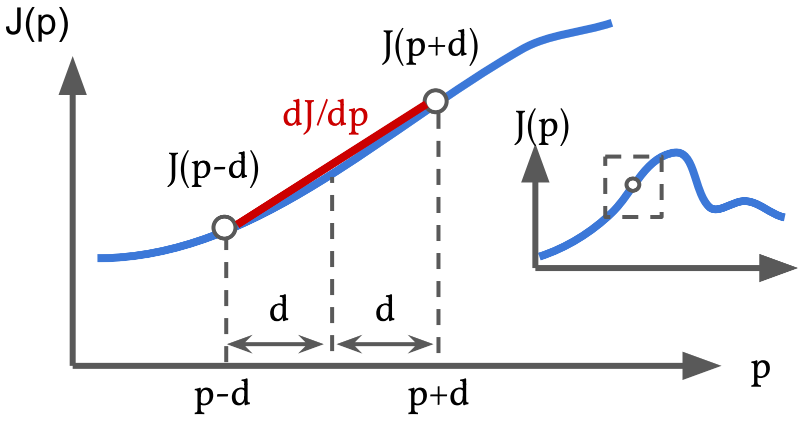
- Start with “p”
- Compute dJ/dp
- Adjust p in the direction of dJ/dp
- Repeat until dJ/dp = 0

Finite Difference Derivatives

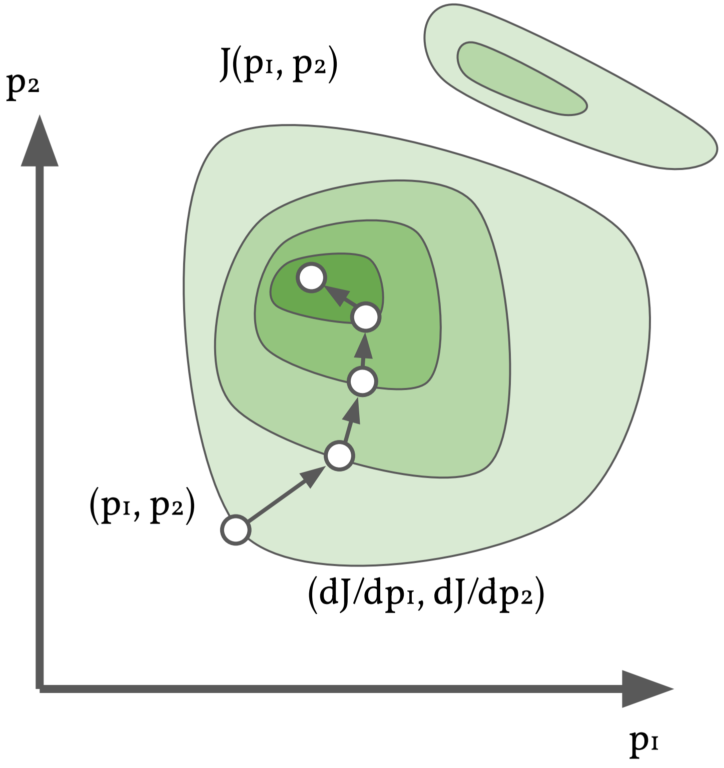
Goal: compute dJ/dp

$$\frac{dJ}{dp} \approx \frac{J(p+d) - J(p-d)}{2d}$$





Multi-Dimensional Sensitivity

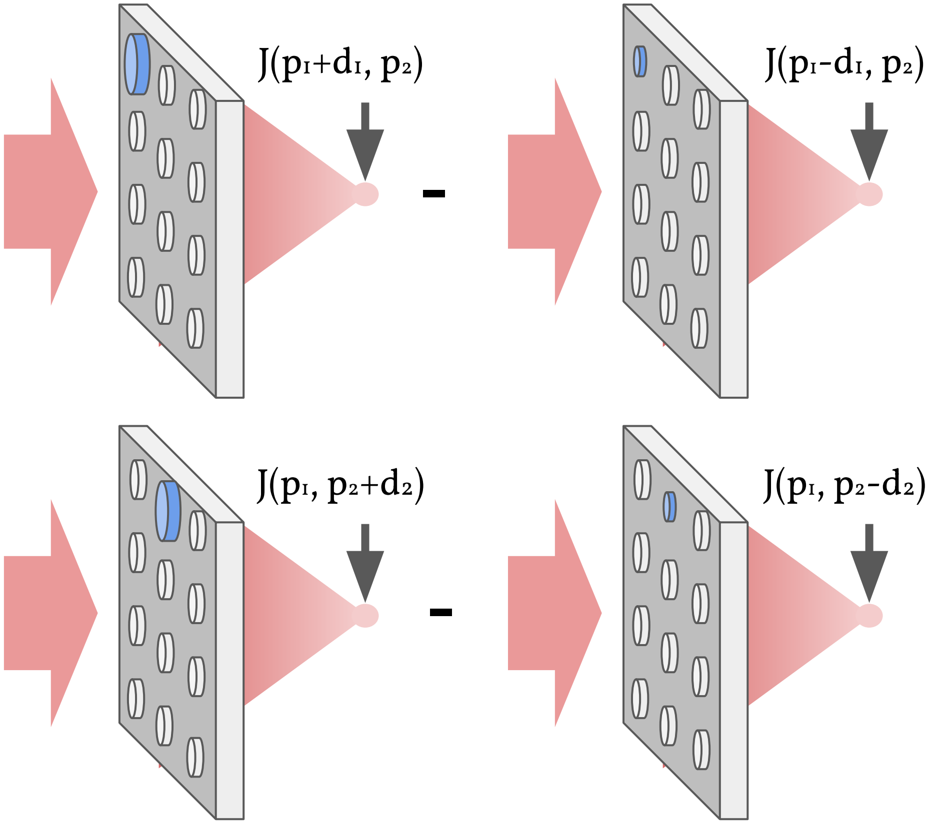


Goal: optimize $J(p_1, p_2)$

Approach (similar to 1D):

- Start with "(p_1, p_2)"
- Compute derivatives $(dJ/dp_1, dJ/dp_2)$
- Adjust p_1 in the direction of dJ/dp_1
- Adjust p_2 in the direction of dJ/dp_2
- Repeat until $dJ/dp_1 = 0$ and $dJ/dp_2 = 0$

2D Finite-Difference Derivatives



Goal: compute (dJ/dp1, dJ/dp2)

$$\frac{dJ}{dp_1} \approx \frac{J(p_1 + d_1, p_2) - J(p_1 - d_1, p_2)}{2d_1}$$

Note: need 4 simulations.

$$\frac{dJ}{dp_2} \approx \frac{J(p_1, p_2 + d_2) - J(p_1, p_2 - d_2)}{2d_2}$$

N-Dimensional Derivative Calculation

Vector Notation:

$\mathbf{p} = (p_1, p_2, p_3, \ldots, p_N)$

$J = J(\mathbf{p})$

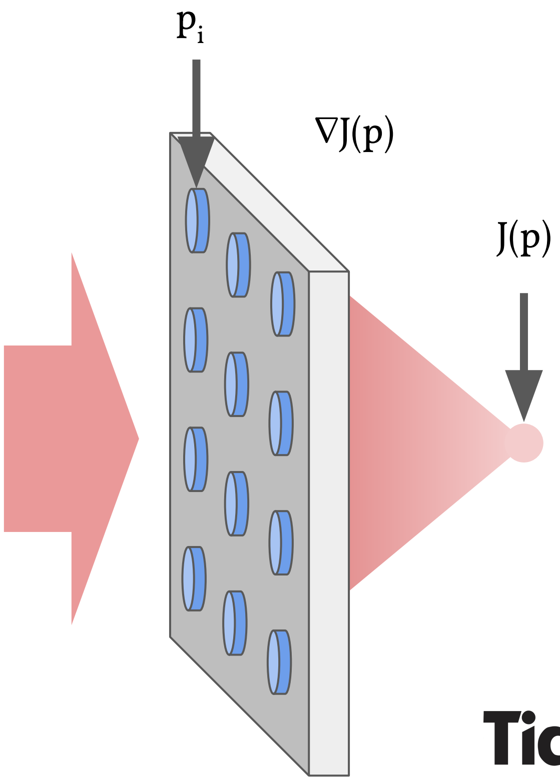
$\nabla_{\mathbf{p}} J = (dJ/p_1, dJ/p_2, dJ/p_3, \ldots, dJ/p_N)$ = the "gradient"

Computing $\nabla_{\mathbf{p}} J$ through finite difference:

$$\nabla_{\mathbf{p}} J \approx \frac{J(\mathbf{p} + \mathbf{d}_i) - J(\mathbf{p} - \mathbf{d}_i)}{2|\mathbf{d}_i|}$$

Repeat for all N parameters.

- Requires 2N simulations!



Issues with Finite Difference

BUILT BY FLEXCOMPUTE – A SOLVER TECHNOLOGY COMPANY

- In practice, there are significant issues with finite difference:
  - For inverse design, N ~ thousands / millions.
  - Requires choosing step size "d_i" for each parameter.
  - Derivatives are approximate.
- Alternative approach: *adjoint method* [3,4]
  - requires only 2 simulations, regardless of N.
  - No need to chose step size.
  - Gives exact gradient, generally without approximation.

[3] https://math.mit.edu/~stevenj/18.336/adjoint.pdf
[4] G. Veronis, R. W. Dutton, and S. Fan, Optics Letters 29, 2288 (2004)

Tidy3D