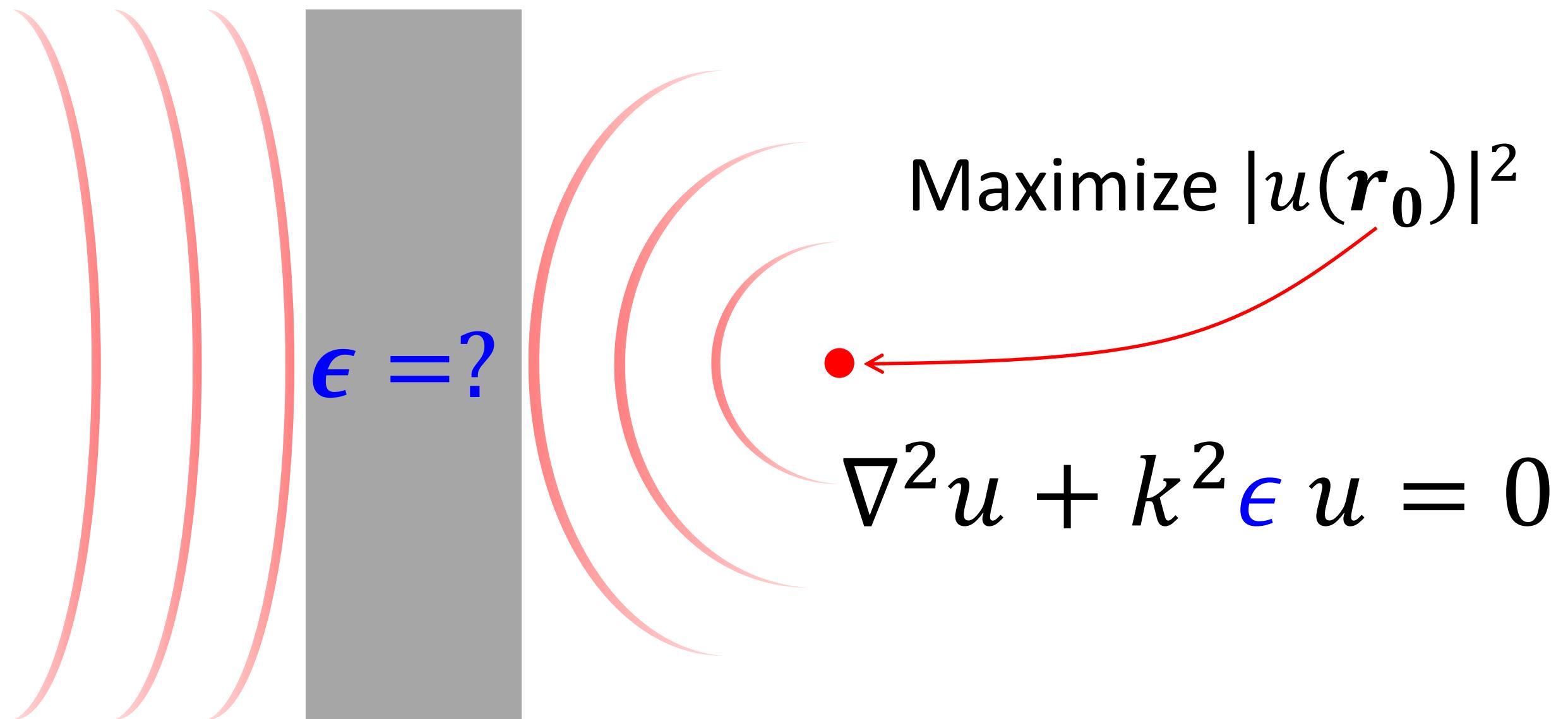


The adjoint method for differentiating complex computations ... with example applications in engineering design.

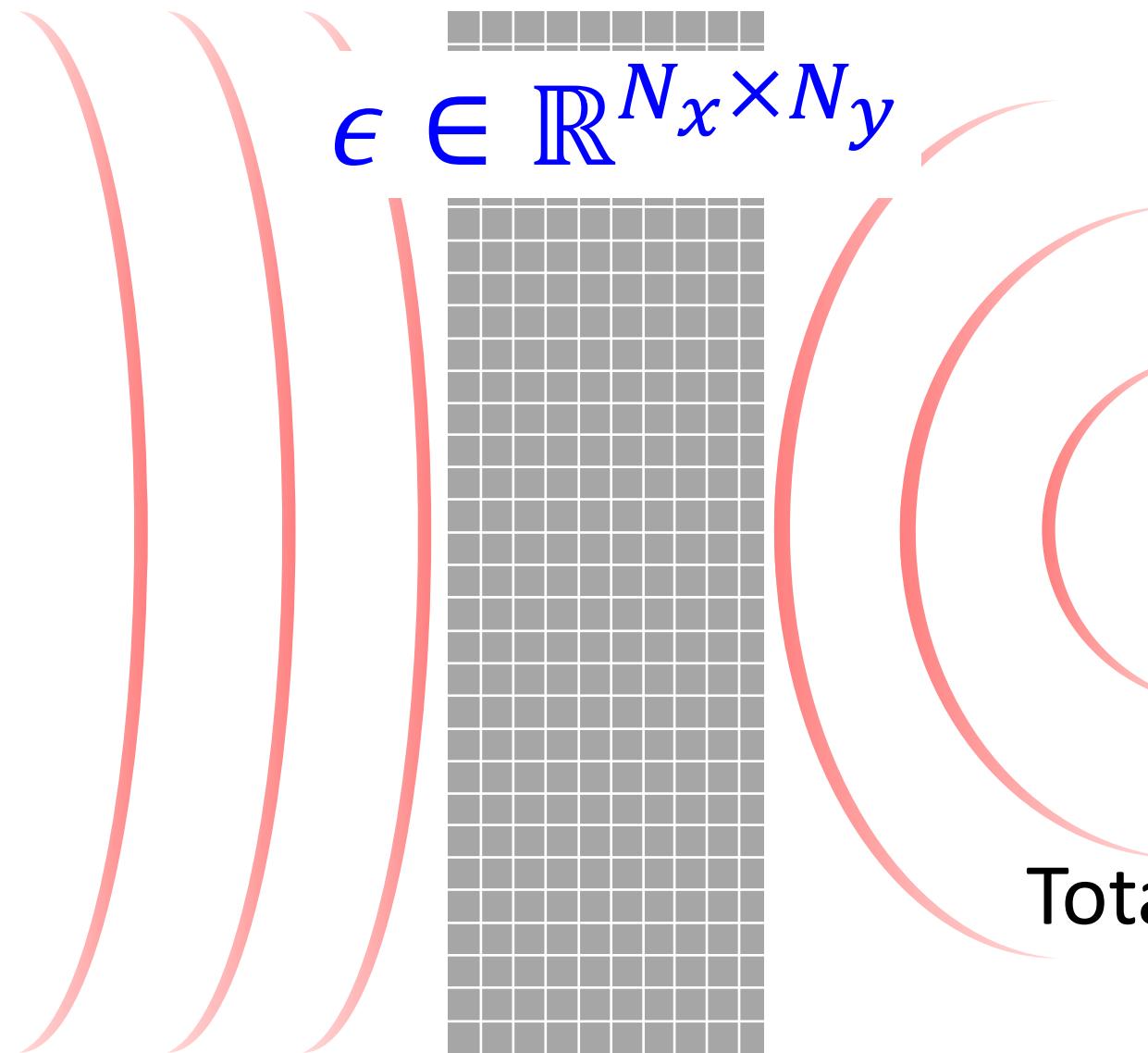
Dr. Zin Lin, MIT Applied Mathematics
guest lecture for [18.335](#) at MIT, Spring 2021 with Prof. Steven Johnson

see also notes: <https://math.mit.edu/~stevenj/18.336/adjoint.pdf>

Example: optimizing *every* pixel of design...



Optimizing *every* pixel of your design ...

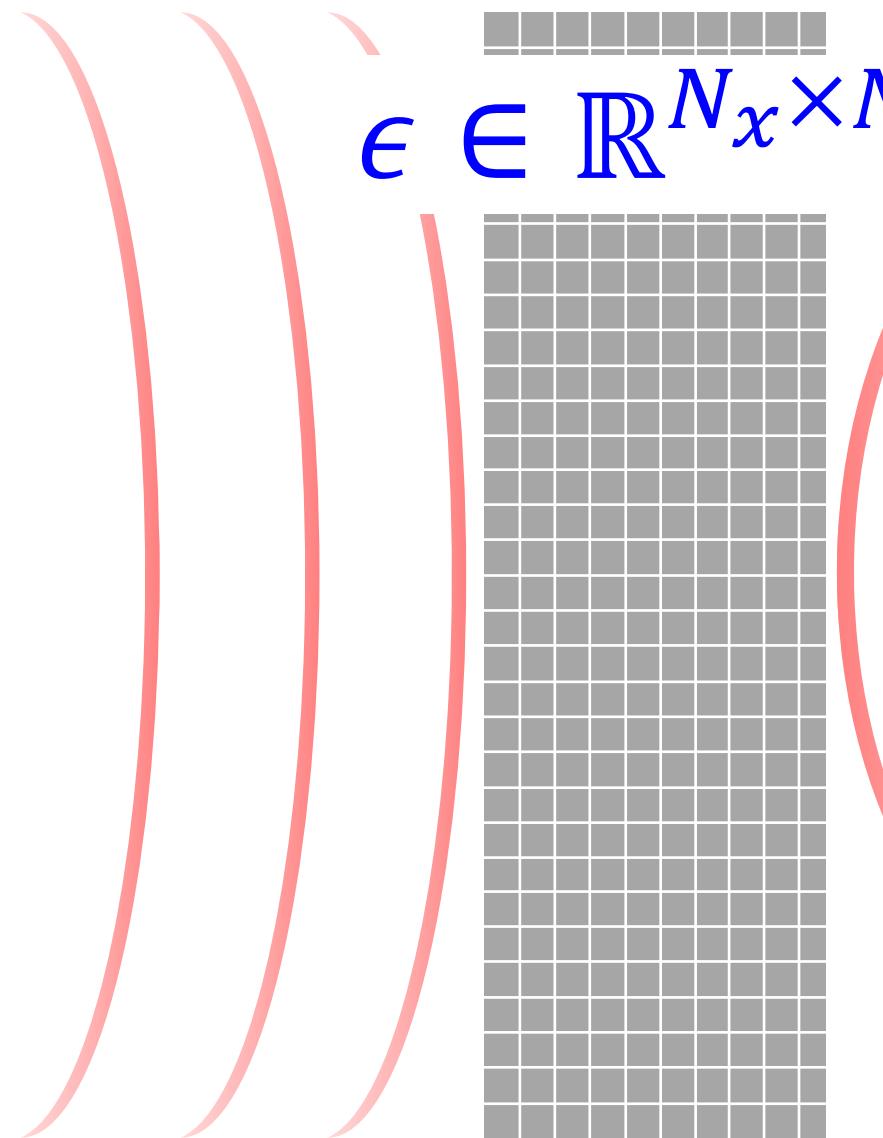


Maximize $|u(r_0)|^2$

$\nabla^2 u + k^2 \epsilon u = 0$

Total # of params. = $N_x \times N_y (\times N_z)$
 $\sim 10^3 - 10^9$

Optimizing *every* pixel of your design ...



Maximize $|u(r_0)|^2$

A red curve starts at a point on the right and curves downwards towards a red dot representing a minimum. A red arrow points from the text below to this minimum point.

$$\nabla^2 u + k^2 \epsilon u = 0$$

We need to compute $\frac{\partial |u|^2}{\partial \epsilon} \in \mathbb{R}^{N_x \times N_y}$
fast and accurate.

Adjoint method (\sim Back prop, Backward differentiation, etc)

Optimize

$$g(\mathbf{u}(\mathbf{p}), \mathbf{p})$$

g is a scalar-valued function $\in \mathbb{R}$.

Subject to

$$\mathbf{A}(\mathbf{p})\mathbf{u} = \mathbf{b}(\mathbf{p})$$

\mathbf{u} is a “physics” vector of size N .
(electric, displacement, pressure,
temperature, etc.)

Requires

$$\frac{dg}{d\mathbf{p}} = ?$$

gradient vector
of size P

\mathbf{p} is a param. vector of size P .
(dielectric, mass density,
conductivity, etc.)

\mathbf{A} is an $N \times N$ matrix.
(PDE: Maxwell, elastodynamics, heat
equation, etc.)

Adjoint method (Linear equations)

$$\frac{dg}{dp} = \frac{\partial g}{\partial p} + \frac{\partial g}{\partial u} \frac{\partial u}{\partial p} \quad (\text{Chain rule})$$



$1 \times N$



$1 \times P$



$N \times P$

Adjoint method (Linear equations)

$$\frac{dg}{dp} = \frac{\partial g}{\partial p} + \frac{\partial g}{\partial u} \frac{\partial u}{\partial p} \quad (\text{Chain rule})$$

$$A\mathbf{u} = \mathbf{b} \Rightarrow A \frac{\partial \mathbf{u}}{\partial p} = \frac{\partial \mathbf{b}}{\partial p} - \frac{\partial A}{\partial p} \mathbf{u}$$

Adjoint method (Linear equations)

$$\frac{dg}{dp} = \frac{\partial g}{\partial p} + \frac{\partial g}{\partial u} \frac{\partial u}{\partial p} \quad (\text{Chain rule})$$

$$A\mathbf{u} = \mathbf{b} \Rightarrow A \frac{\partial \mathbf{u}}{\partial p} = \frac{\partial \mathbf{b}}{\partial p} - \frac{\partial A}{\partial p} \mathbf{u}$$

$$\frac{dg}{dp} = \frac{\partial g}{\partial p} + \frac{\partial g}{\partial u} \left[A^{-1} \left(\frac{\partial \mathbf{b}}{\partial p} - \frac{\partial A}{\partial p} \mathbf{u} \right) \right]$$

Adjoint method (Linear equations)

$$\frac{dg}{dp} = \frac{\partial g}{\partial p} + \underbrace{\frac{\partial g}{\partial u}}_{1 \times N} \left[A^{-1} \underbrace{\left(\frac{\partial b}{\partial p} - \frac{\partial A}{\partial p} u \right)}_{N \times P} \right]$$

Adjoint method (Linear equations)

$$\frac{dg}{dp} = \frac{\partial g}{\partial p} + \underbrace{\frac{\partial g}{\partial u}}_{1 \times N} \left[A^{-1} \underbrace{\left(\frac{\partial b}{\partial p} - \frac{\partial A}{\partial p} u \right)}_{N \times P} \right]$$

☞ You have to solve $A[\dots]_i = \frac{\partial b}{\partial p_i} - \frac{\partial A}{\partial p_i} u$ for *every* p_i , $i = 1, \dots, P$!!!

Adjoint method (Linear equations)

$$\frac{dg}{dp} = \frac{\partial g}{\partial p} + \left[\begin{array}{cc} \frac{\partial g}{\partial u} & A^{-1} \end{array} \right] \left(\frac{\partial b}{\partial p} - \frac{\partial A}{\partial p} u \right)$$

$1 \times N \quad N \times N \quad N \times P$

$1 \times N \qquad N \times N \qquad N \times P$

$\underbrace{\qquad\qquad}_{1 \times N} \quad \underbrace{\qquad\qquad}_{N \times N} \quad \underbrace{\qquad\qquad}_{N \times P}$

KEY Trick:
switch the bracket

Adjoint method (Linear equations)

$$\frac{dg}{dp} = \frac{\partial g}{\partial p} + \left[\begin{array}{cc} \frac{\partial g}{\partial u} & A^{-1} \end{array} \right] \left(\frac{\partial b}{\partial p} - \frac{\partial A}{\partial p} u \right)$$

$1 \times N \quad N \times N \quad N \times P$

$1 \times N$

$$\lambda^T = \frac{\partial g}{\partial u} A^{-1} \Rightarrow A^T \lambda = \frac{\partial g^T}{\partial u}$$

Adjoint method (Linear equations)

$$\frac{dg}{dp} = \frac{\partial g}{\partial p} + \lambda^T \left(\frac{\partial b}{\partial p} - \frac{\partial A}{\partial p} u \right)$$

$\underbrace{\hspace{10em}}$ $1 \times N$ $\overbrace{\hspace{10em}}$ $N \times P$

$$\lambda^T = \frac{\partial g}{\partial u} A^{-1} \Rightarrow A^T \lambda = \frac{\partial g^T}{\partial u}$$

Adjoint method (Linear equations)

Optimize $g(u(p), p)$

Subject to $A(p)u = b(p)$

Requires $\frac{dg}{dp} = \frac{\partial g}{\partial p} + \lambda^T \left(\frac{\partial b}{\partial p} - \frac{\partial A}{\partial p} u \right)$

Where $A^T \lambda = \frac{\partial g^T}{\partial u}$

Takeaway: you can obtain the entire gradient by solving the linear equations just one more time.

Adjoint method (Nonlinear equations)

Optimize $g(u(p), p)$

Subject to $f(u, p) = 0$

Requires $\frac{dg}{dp} = \frac{\partial g}{\partial p} + \frac{\partial g}{\partial u} \frac{\partial u}{\partial p}$

f is a vector-valued function of size N .
(nonlinear Maxwell equations, Navier-Stokes equations, etc.)

Adjoint method (Nonlinear equations)

$$f(u, p) = 0 \quad \Rightarrow \quad \frac{\partial u}{\partial p} = - \left(\frac{\partial f}{\partial u} \right)^{-1} \frac{\partial f}{\partial p}$$

$N \times N \quad N \times P$

$$\frac{dg}{dp} = \frac{\partial g}{\partial p} + \frac{\partial g}{\partial u} \left[- \left(\frac{\partial f}{\partial u} \right)^{-1} \frac{\partial f}{\partial p} \right]$$

Adjoint method (Nonlinear equations)

Again, switch the bracket

$$\frac{dg}{dp} = \frac{\partial g}{\partial p} - \left[\frac{\partial g}{\partial u} \left(\frac{\partial f}{\partial u} \right)^{-1} \right] \frac{\partial f}{\partial p}$$

$$= \frac{\partial g}{\partial p} - \lambda^T \frac{\partial f}{\partial p}, \quad \frac{\partial f^T}{\partial u} \lambda = \frac{\partial g^T}{\partial u}$$

N×N

Note: the adjoint problem is a **linear** problem (of lesser complexity than the forward **nonlinear** problem).

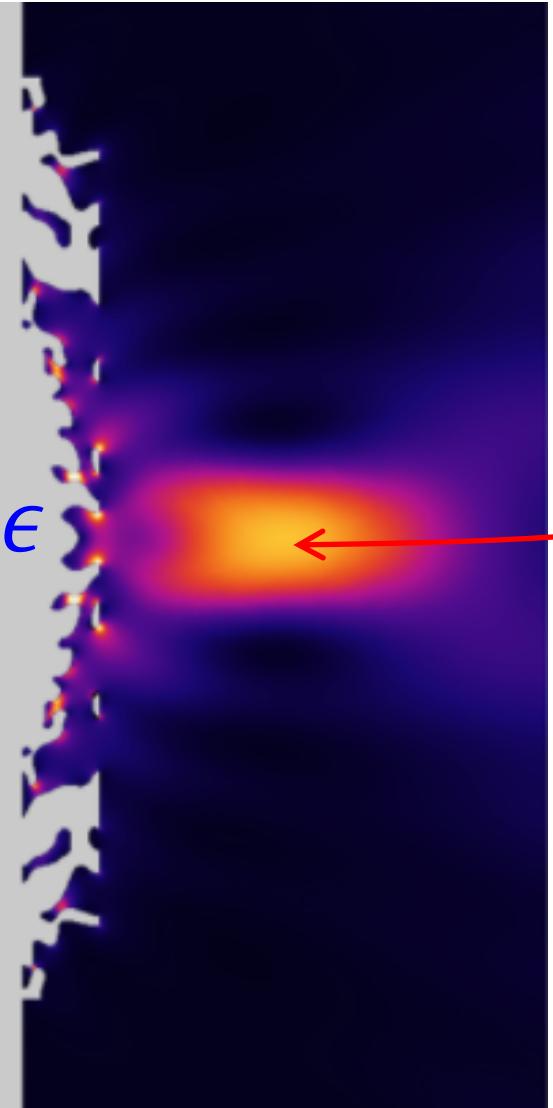
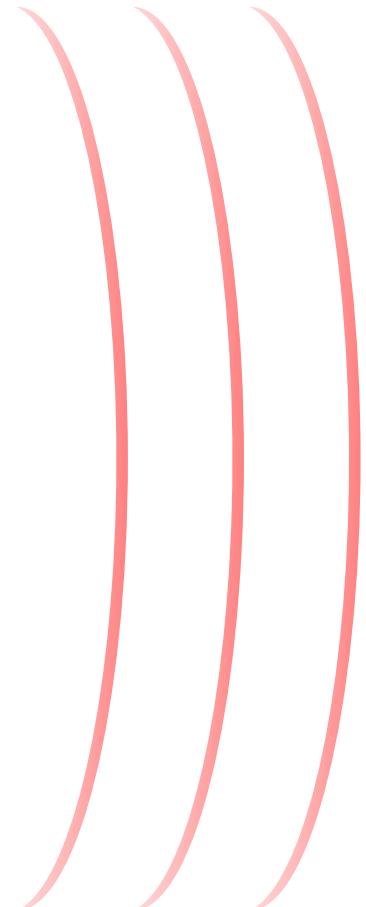
Example applications

Engineering optimization (structural “topology” optimization):

Find the physical structure that optimizes some objective (e.g. focusing light, minimizing drag, supporting weight, ...).

Adjoint methods are key! “Forward” model is a **huge simulation**, but gradient requires only **one additional simulation**.

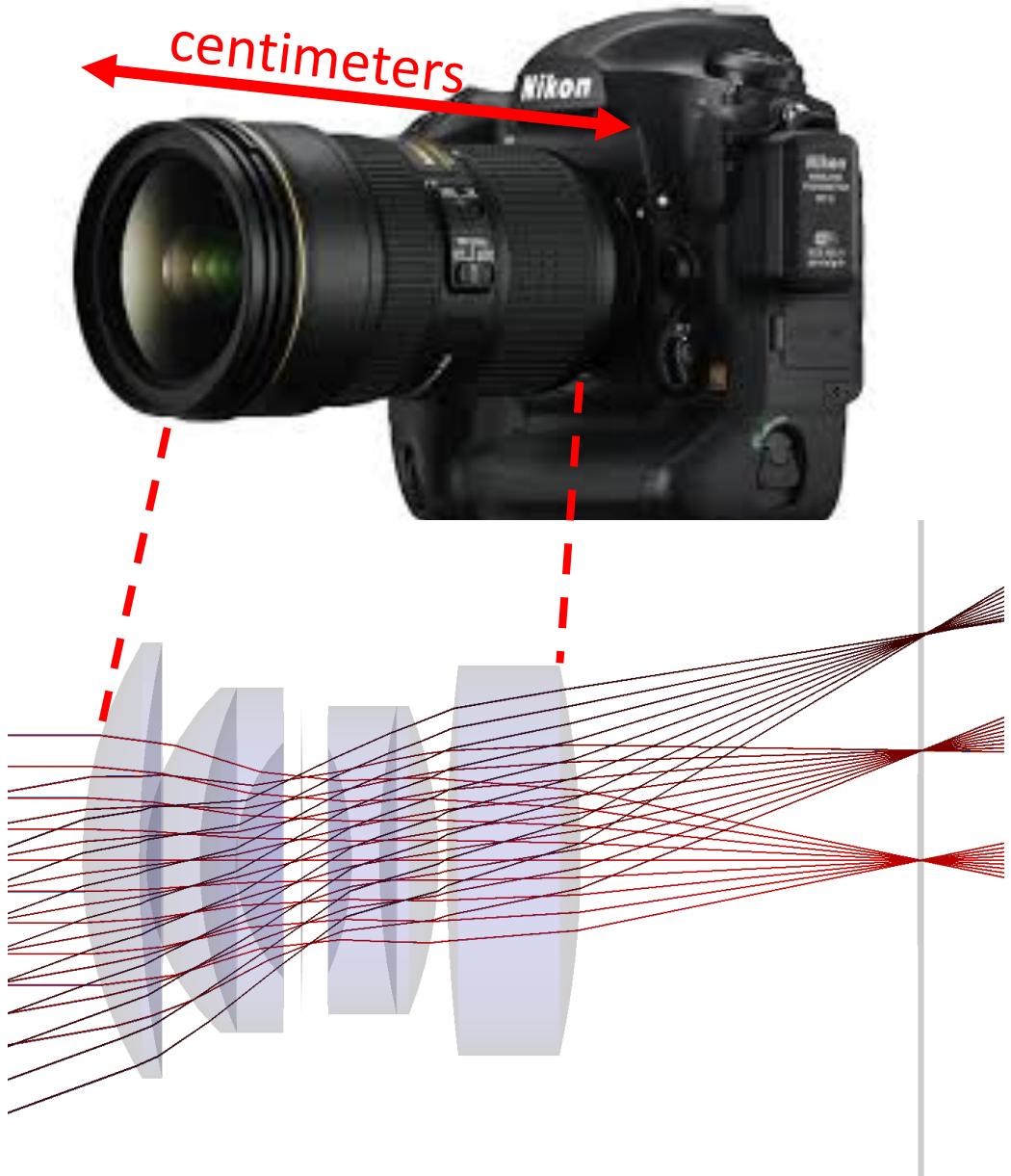
Topology-optimized nanophotonic lens



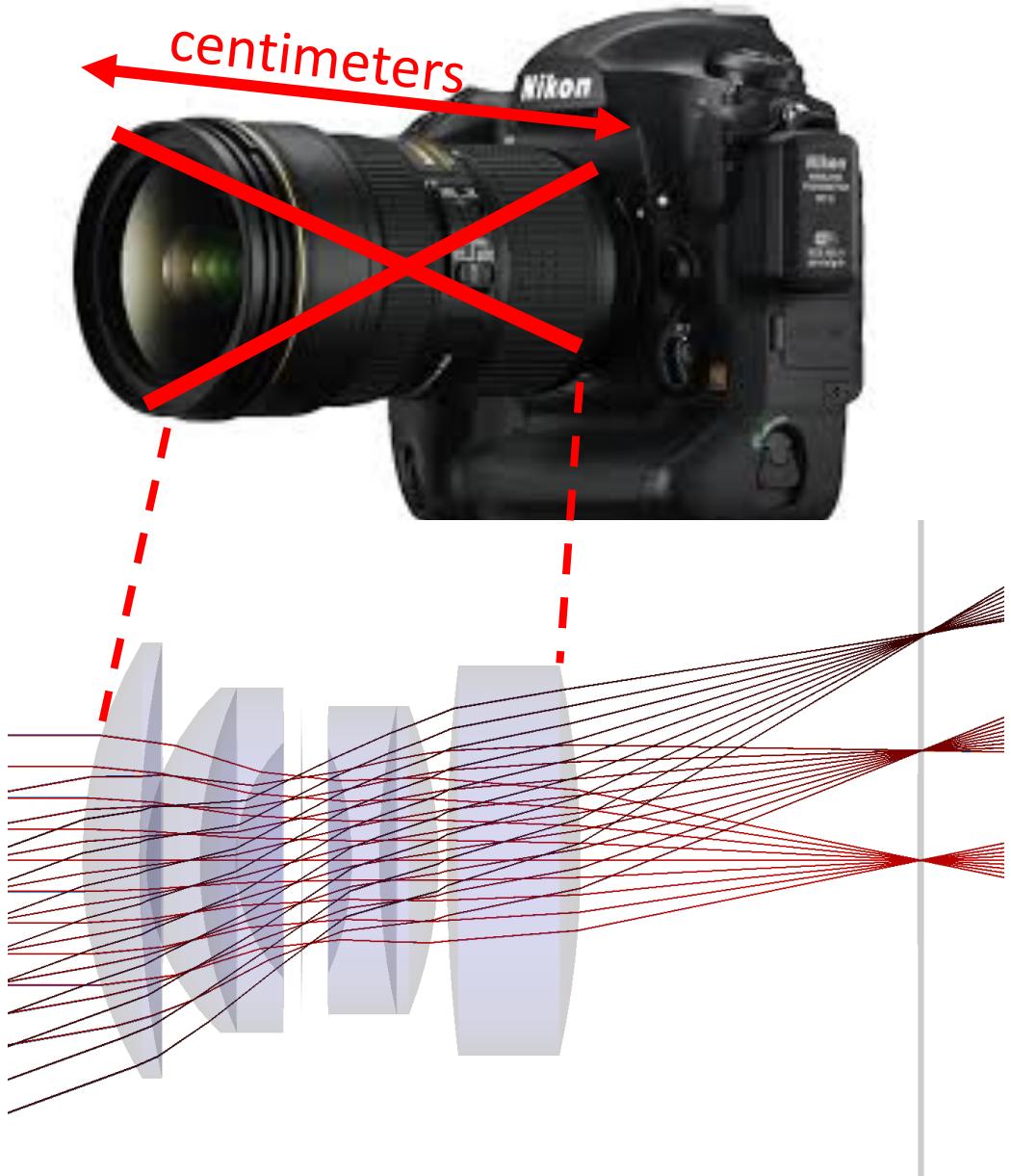
$$\text{Max. } |u(r_0)|^2$$
$$\nabla^2 u + k^2 \epsilon u = 0$$

Christiansen, Rasmus E., and Ole Sigmund. "Inverse design in photonics by topology optimization: tutorial." *JOSA B* 38.2 (2021): 496-509.

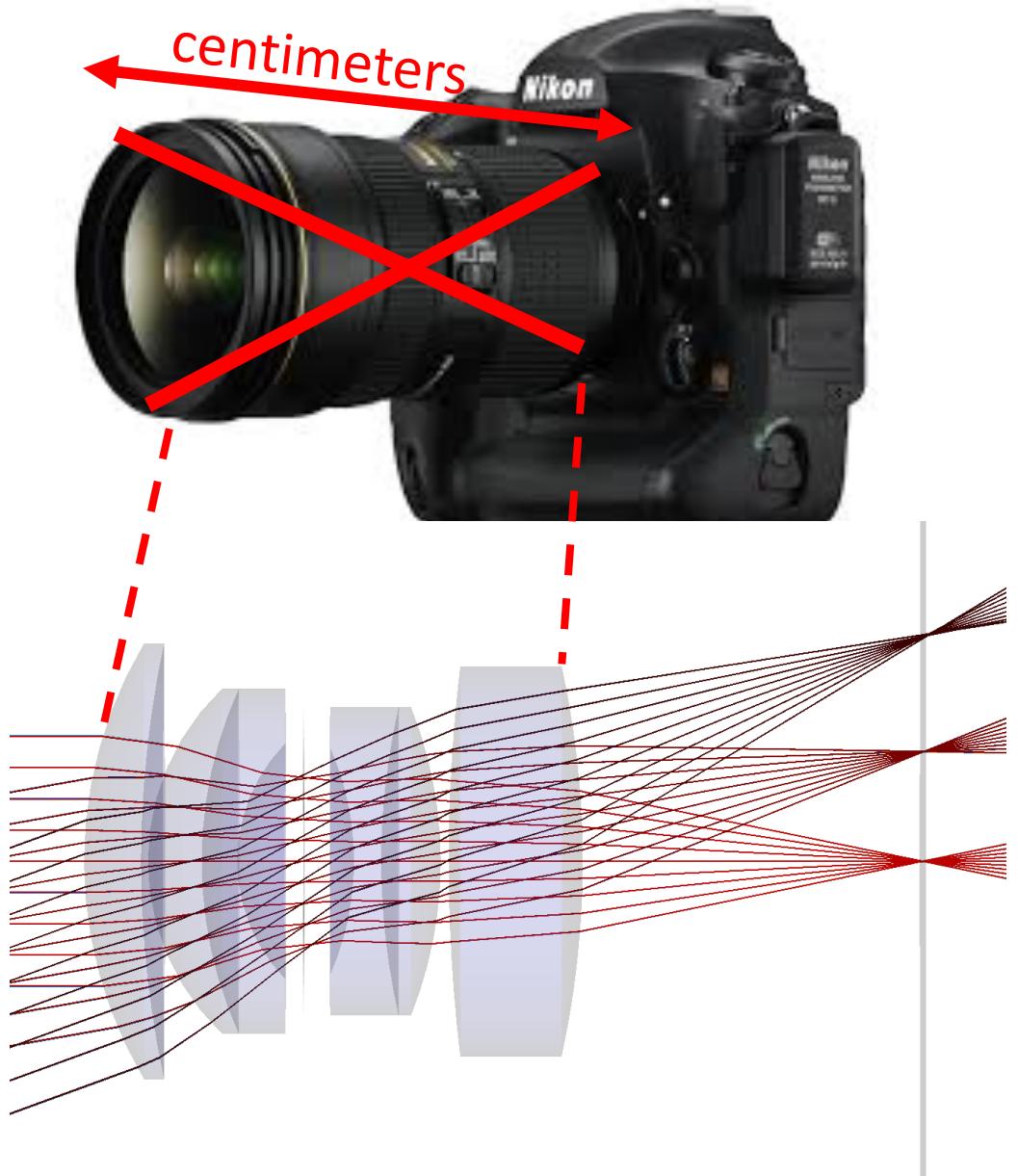
A squished camera lens?



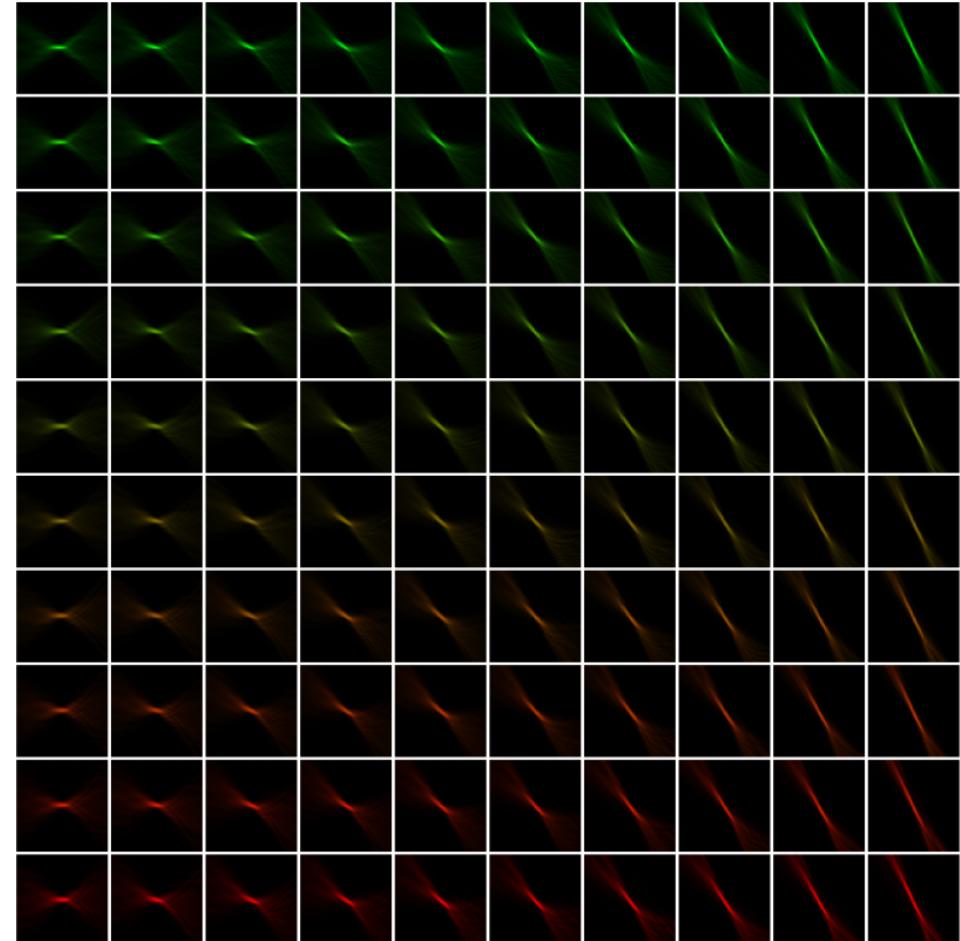
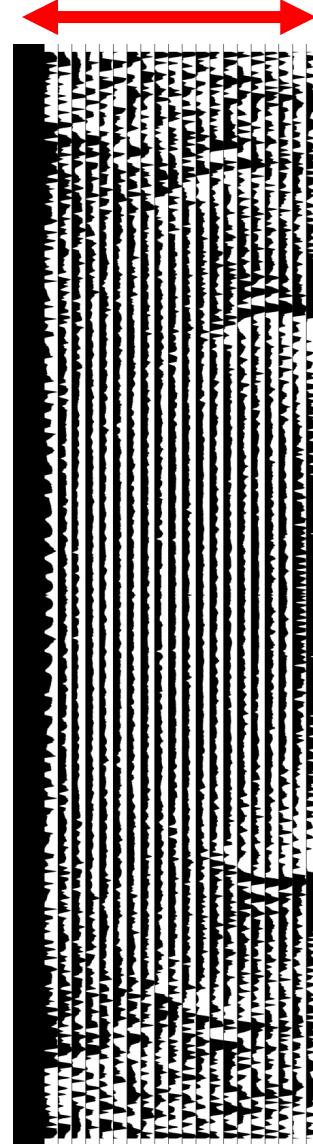
A squished camera lens?



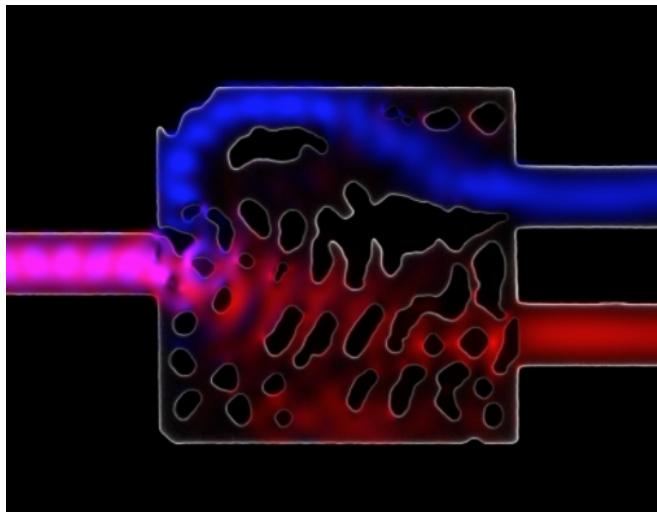
A squished camera lens?



micrometers



Many more applications in optics and photonics ...



Splitting wavelengths into different channels
Piggott et al, Nature Photonics (2016)

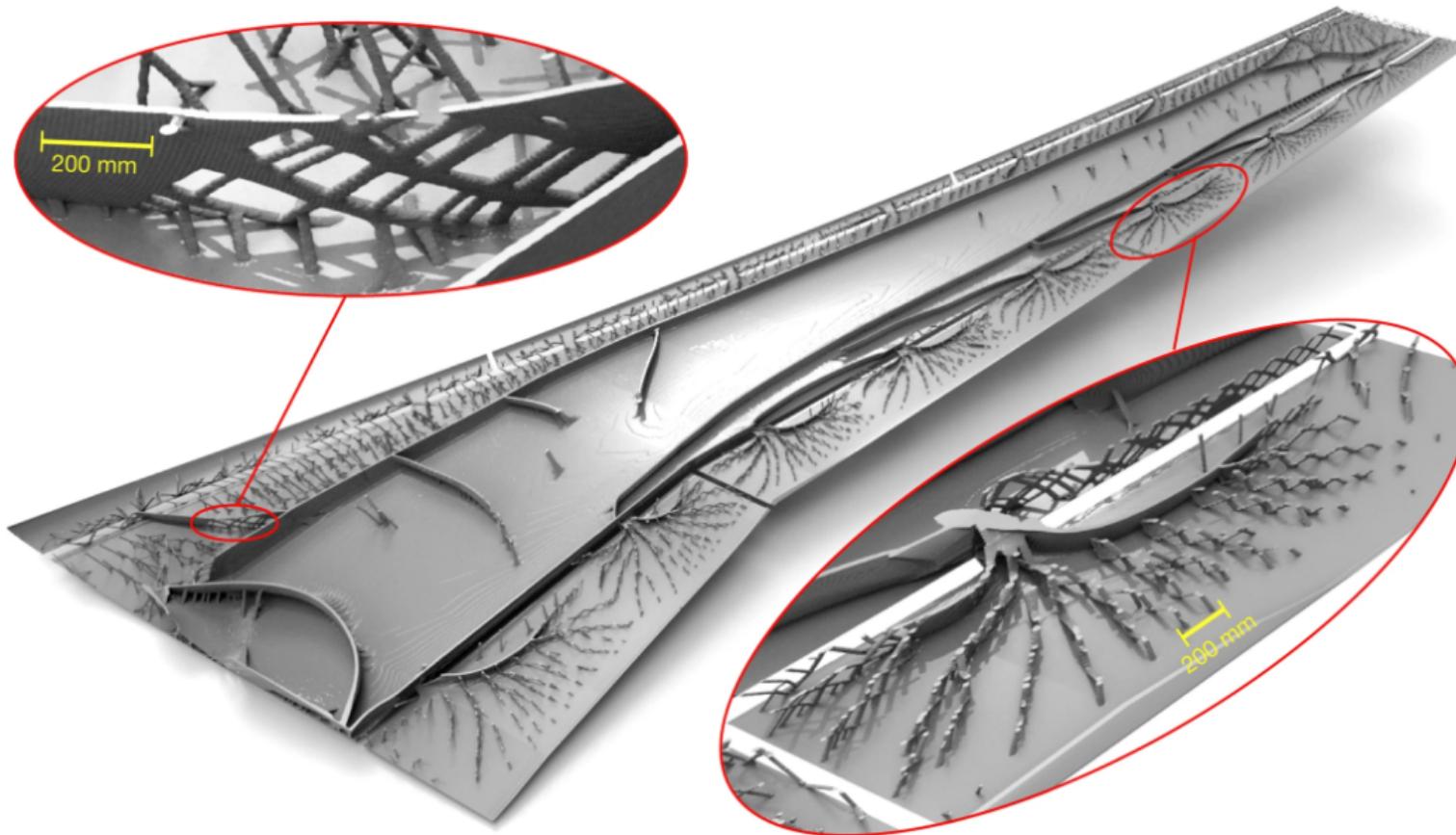


Trapping light in a very small volume
Wang et al, APL (2018)

Optimize $g(E(\epsilon), \epsilon)$ over ϵ
Subject to $\nabla \times \nabla \times E - k^2 \epsilon E = i \omega J$

Topology-optimized aircraft wing

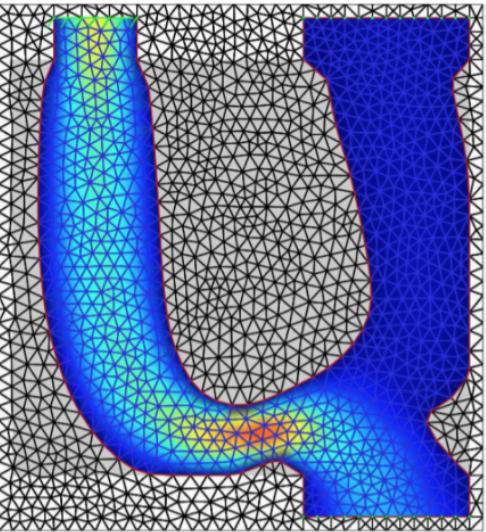
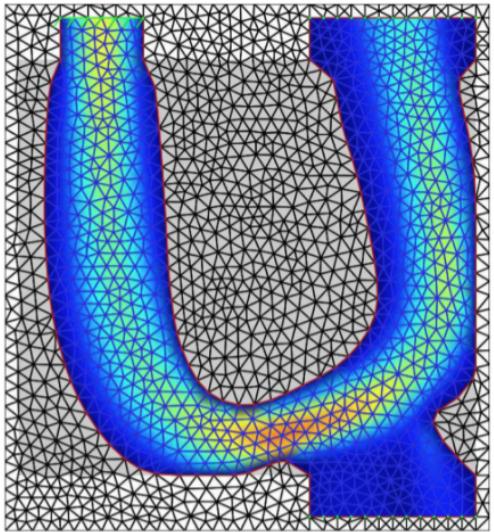
~ 10^9 parameters



Goal: maximize stiffness under external loads, utilizing limited amount of material
→ Light but strong
→ 100s tonnes of fuel saving

Aage, Niels, et al. "Giga-voxel computational morphogenesis for structural design." *Nature* 550.7674 (2017): 84-86.

Topology optimization with fluid dynamics

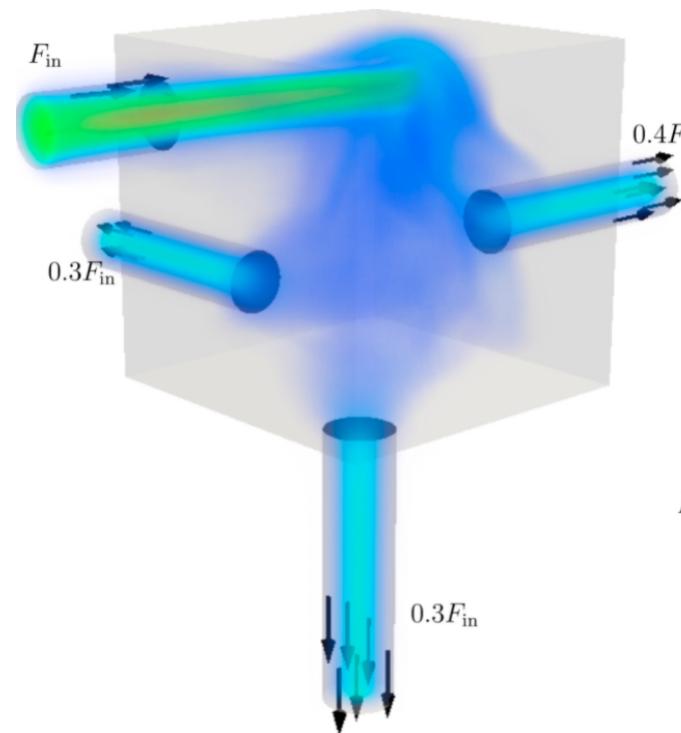


(A) High Re flow, velocity magnitude

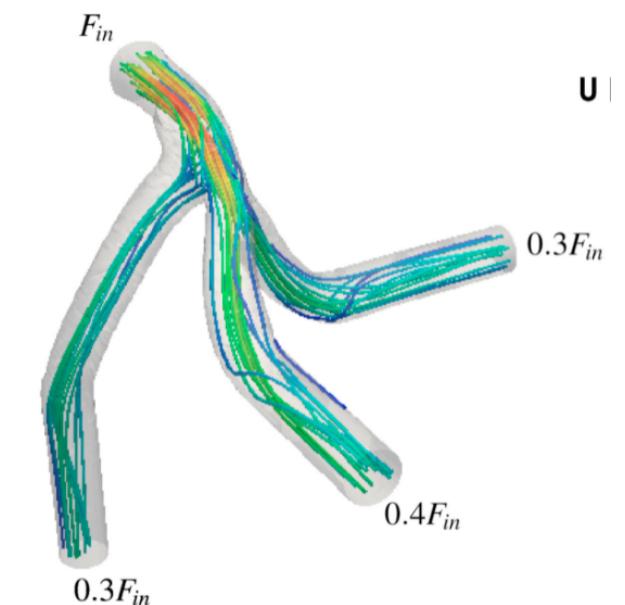
(B) Low Re flow, velocity magnitude

Switching flow channels for high vs. low viscosity

Zhou, Mingdong, et al. "Shape morphing and topology optimization of fluid channels by explicit boundary tracking." *International Journal for Numerical Methods in Fluids* 88.6 (2018): 296-313.



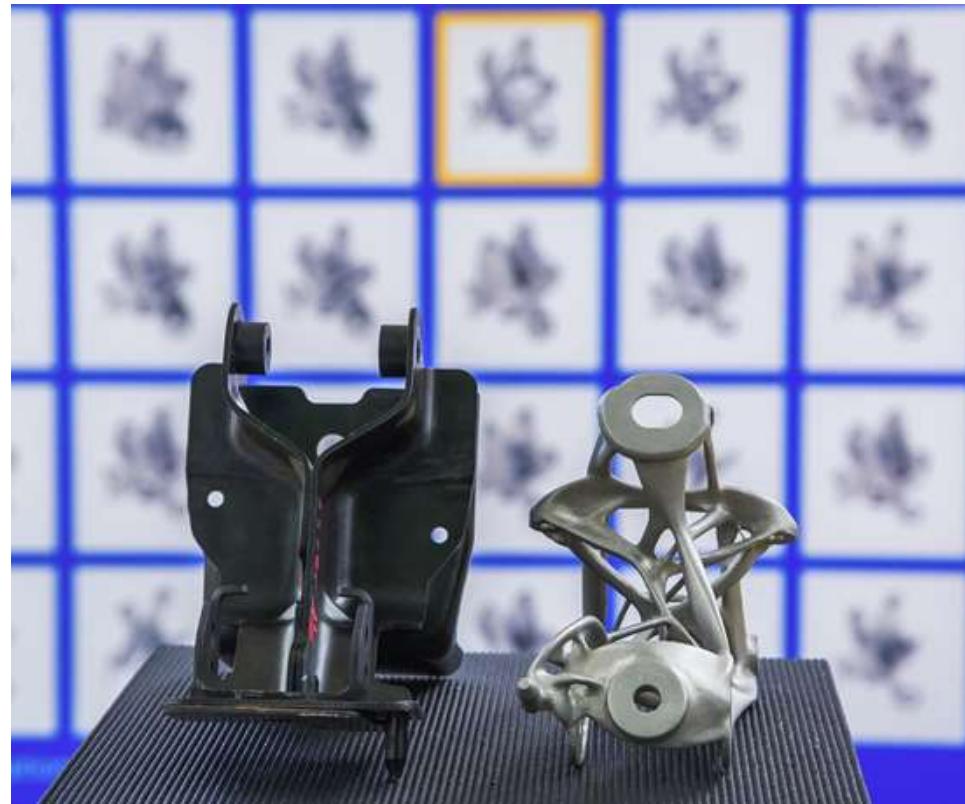
Dilgen, Cetin B., et al.
"Topology optimization of
turbulent flows." *Computer
Methods in Applied
Mechanics and
Engineering* 331 (2018):
363-393.



Some industrial applications



Topology-optimized
3D-printed hip
replacement (Altair)



Topology-optimized 3D-printed
seat bracket (General Motors)

And many others ...

Not just for physical problems!

Key point is that if you **have any complicated calculation** with lots of parameters, you can compute **gradient** (sensitivity) of a scalar output $g(u)$ with **respect to every parameter** with roughly **one additional calculation**.

Enabling factor for large-scale optimization in machine learning [g = loss function, u = network outputs, p = network weights & other parameters], statistics, finance, and **many other fields**.

Central trick is to **evaluate chain rule from “left-to-right”**, i.e. **“back-propagating”** in **“reverse”** order from **“outputs to inputs”**, i.e. using left-to-right **“vector–Jacobian products”**.