My Parallel Electromagnetic Solver

A Star-Maxwell-P FDTD Propagator

18.337 Parallel Computing

Alejandro W. Rodriguez

Outline

Overview of nanophotonics

Statement of the problem

Parallelization: a data-parallel approach

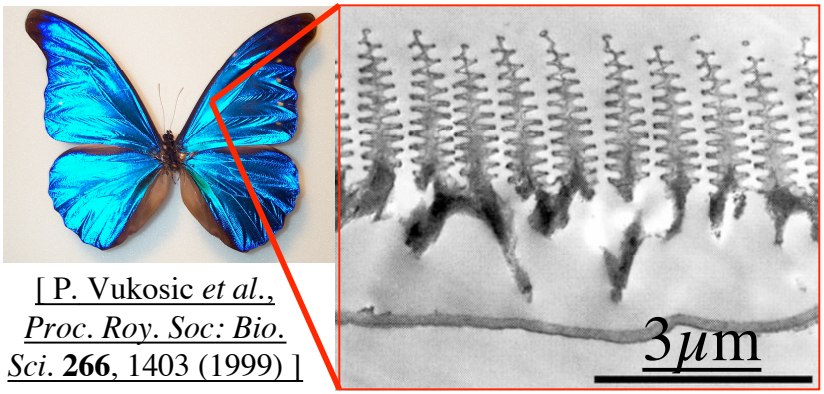
- Minimizing temporal and memory scalings

Future work

"New-School" Electromagnetism

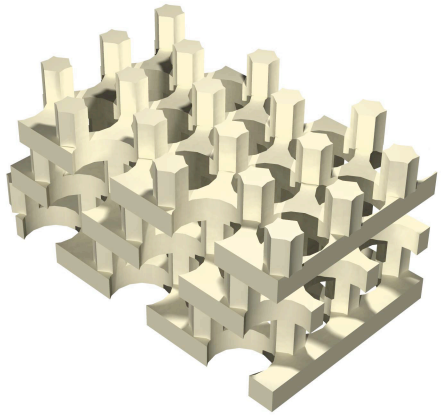
Beyond the geometric-optics limit

Design of frequency-selective structures



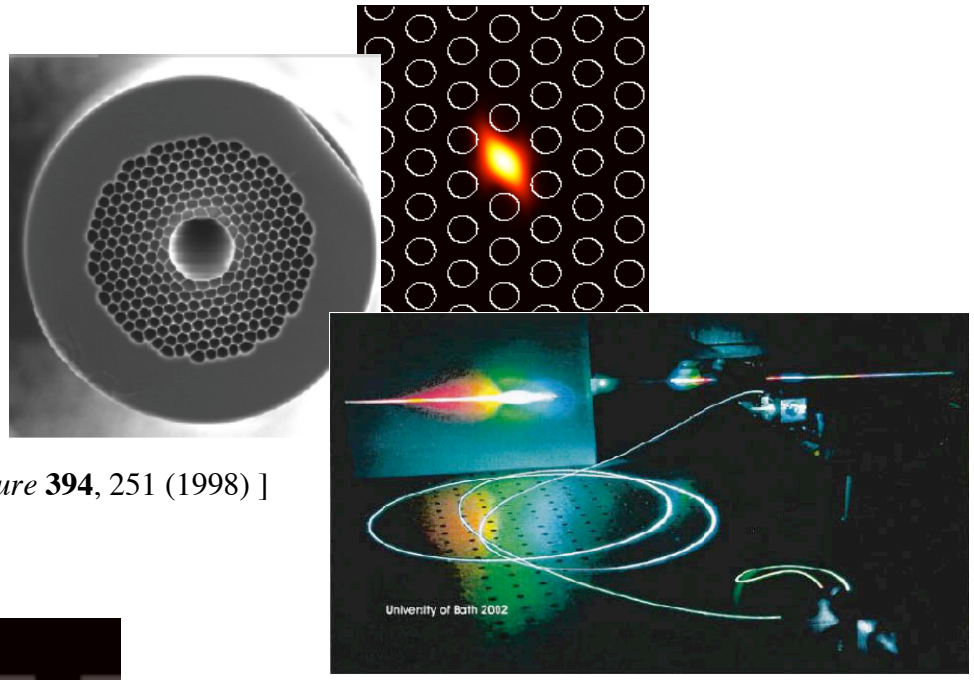
[P. Vukosic *et al.*, *Proc. Roy. Soc: Bio. Sci.* **266**, 1403 (1999)]

optical "insulators"



[A. Rodriguez *et al.*, *Opt. Lett.* **30**, (2005)]

Omniguides and Fiber optical guiding

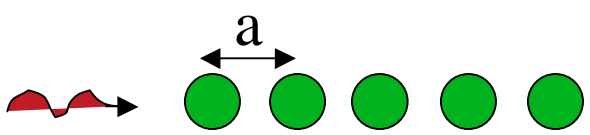


[S. Y. Lin *et al.*, *Nature* **394**, 251 (1998)]

[P. Vukosic *et al.*, *Proc. Roy. Soc: Bio. Sci.* **266**, 1403 (1999)]

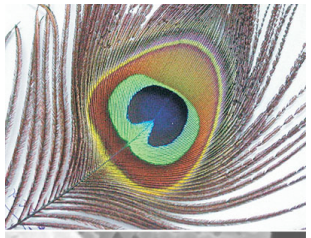
Nanophotonics — λ-scale geometries
(wavelength)

periodic geometries



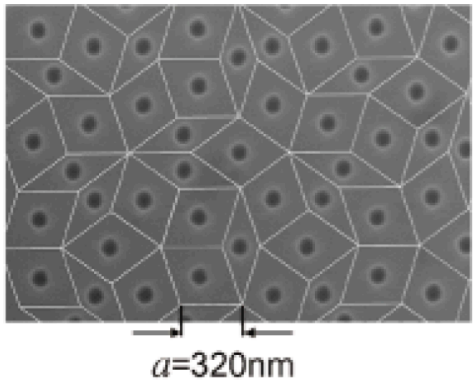
Coherent scattering!

Any 1d-periodic (layered)
Structure has a band gap

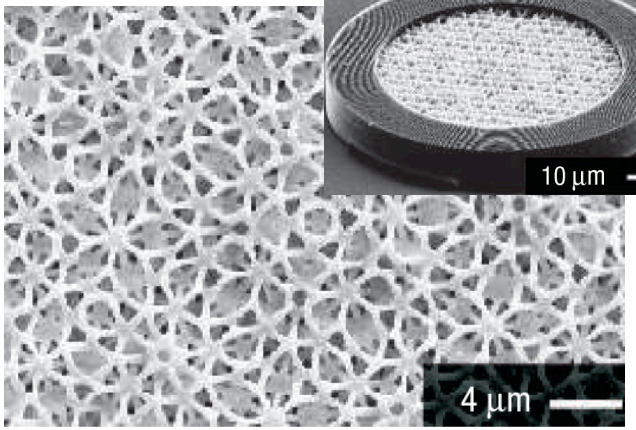


[J. Zi *et al*, *Proc. Nat. Acad. Sci. USA*, **100**, 12576 (2003)]
[figs: Blau, *Physics Today* **57**, 18 (2004)]

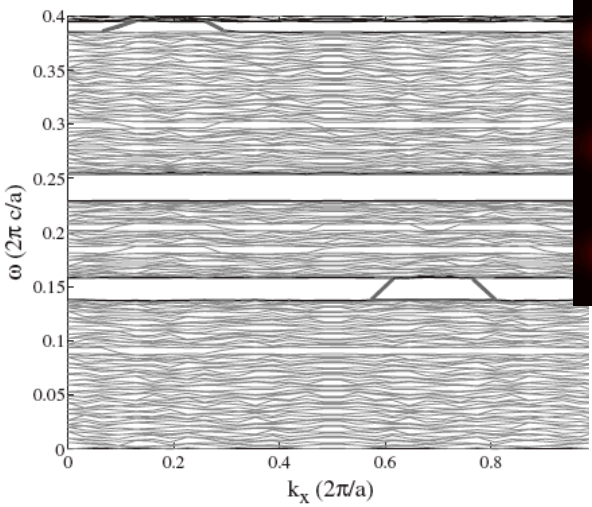
Quasi-periodic geometries

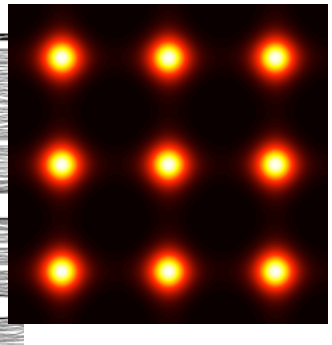


[Notomi, 2004]



[Lederman, 2006]





[A. Rodriguez *et. al.* Phys. Rev. B, 77, 104201, (2008)]

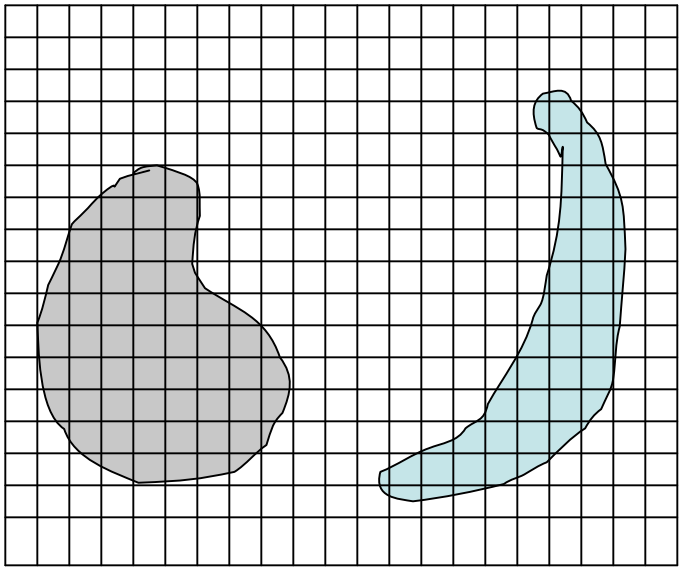
A Light-Speed Introduction to FDTD

Maxwell's Equations

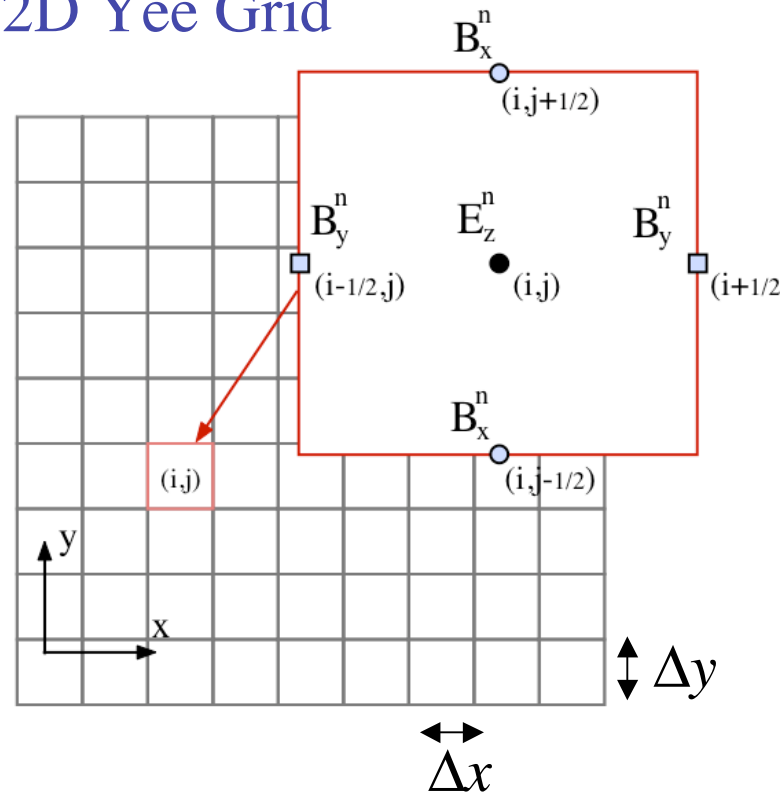
$$\nabla \times \vec{D} = \frac{\partial \vec{H}}{\partial t} \quad \nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} + 4\pi \vec{J}$$

$$\vec{D} = \varepsilon \vec{E} \quad \vec{B} = \mu_0 \vec{H}$$

EM between produce



2D Yee Grid



$$\vec{E}(x_i, y_j, t_j) = \vec{E}^n_{i,j}$$

$$\vec{B}(x_i, y_j, t_n) = B^n_{i+1/2,\, j+1/2}$$

A Light-Speed Introduction to FDTD

2D Maxwell's Equations

continuum

$$\frac{\partial B_x}{\partial t} = \frac{\partial E_z}{\partial y} \qquad \frac{\partial B_y}{\partial t} = -\frac{\partial E_z}{\partial x}$$

discrete

$$B^n_{x,(i+,j+)} = B^{n-1}_{x,(i+,j+)} + \frac{\Delta t}{\Delta y}\left(E^{n-1}_{z,(i,j+1)} - E^{n-1}_{z,(i,j-1)}\right)$$

$$B^n_{y,(i+,j+)} = B^{n-1}_{y,(i+,j+)} - \frac{\Delta t}{\Delta x}\left(E^{n-1}_{z,(i+1,j)} - E^{n-1}_{z,(i-1,j)}\right)$$

$$\vec{H} = \mu_0^{-1}\vec{B}$$

$$H^n_{(i+,j+)} = \mu_0^{-1} B^n_{(i+,j+)}$$

$$\frac{\partial D_z}{\partial t} = \left(\frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y}\right)$$

$$D^n_{z,(i,j)} = D^{n-1}_{z,(i,j)} + \frac{\Delta t}{\Delta x}\left(H^{n-1}_{y,(i+,j)} - H^{n-1}_{y,(i-,j)}\right) - \frac{\Delta t}{\Delta y}\left(H^{n-1}_{x,(i,j+)} - H^{n-1}_{x,(i,j-)}\right) + 4\pi J^{n-1}_{z,(i,j)}$$

$$E_z = \varepsilon^{-1} D_z$$

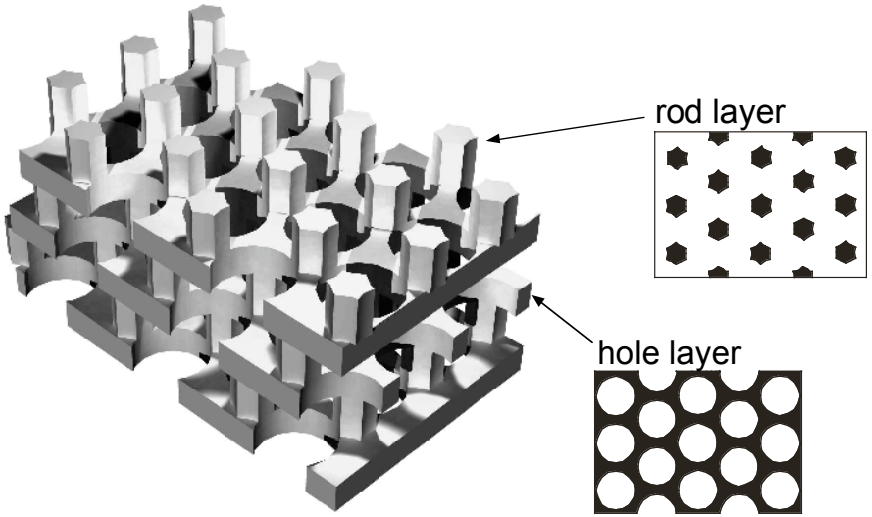
$$E^n_{(i,j)} = \varepsilon^{-1}_{(i,j)} D^n_{(i,j)}$$

So…why is this a hard problem?

Maxwell's Equations

$$\nabla \times \vec{D} = \frac{\partial \vec{H}}{\partial t} \qquad \nabla \times \vec{E} = \frac{\partial \vec{B}}{\partial t} + 4\pi \vec{J}$$

$$\vec{D} = \varepsilon \vec{E} \qquad \vec{B} = \mu_0 \vec{H}$$



FCC crystal (solved 1995)

Temporal complexity

- 2 complex (discrete) 3D fields

~ 100 flops / pixel / step

- 3D size ~ 20 x 18 x 18 a^3
- resolution (pixels / a) ~ 25

~ 10^8 pixels

- 100-400 time steps

~ 10^{12} flops!!

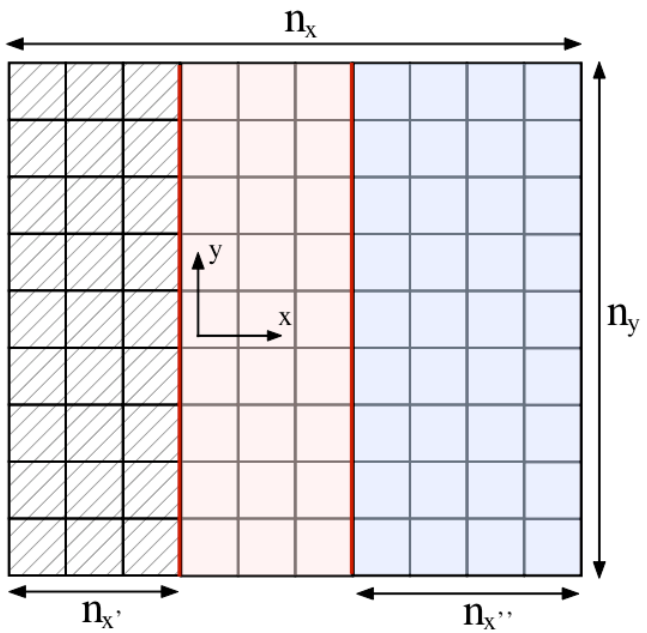
Memory ~ 20 GB

Now, let's create our own parallel code!

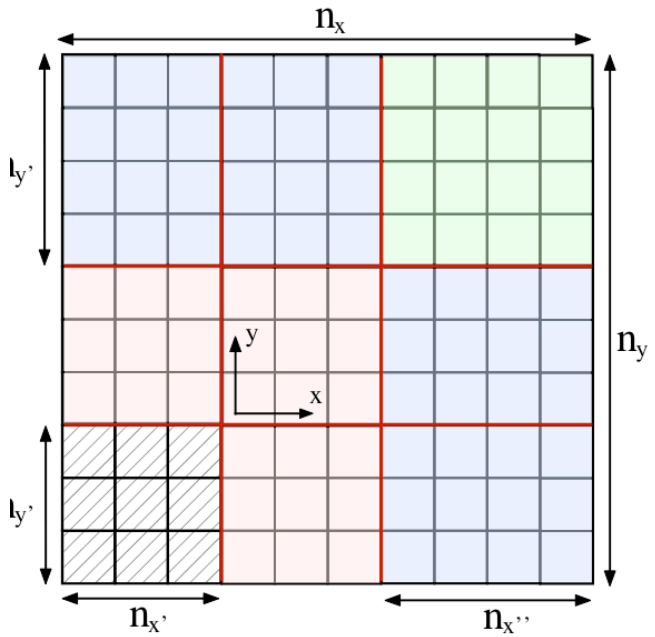
Parallelization Schemes

Optimizing complexity

1D



2D



$$\#\ \text{op. counts} \sim \alpha \frac{res^d}{n_p} + \beta\, res^{d-1} n_p$$

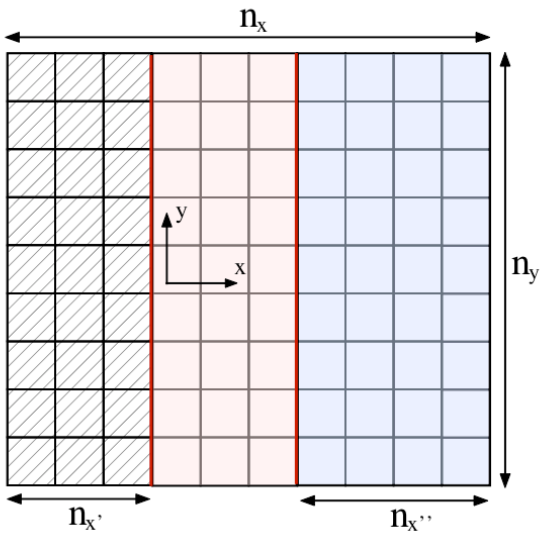
task ~ volume

comm. ~ area

Parallelization Schemes

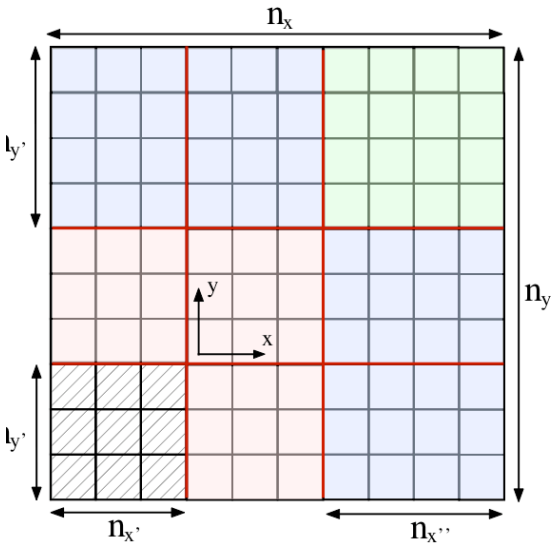
optimizing complexity

1D



$$\sim \mathrm{O}\left(\frac{n^2}{n_p} + nn_p\right)$$

2D



$$\sim \mathrm{O}\left(\frac{n^2}{n_p} + 4n\sqrt{n_p}\right)$$

Most common scenario

$$n >> n_p$$

$$\Rightarrow$$ 2D Wins!

Star-P Implementation

Power of vectorization

Simple 1D example

$$B, E \in \Re(N \times N)$$

consider

$$E_{N+1,\,j} = E_{0,\,j} = 0$$

$$B_{i,\,j} = B_{i,\,j} + \alpha\left(E_{i,\,j+1} - E_{i,\,j}\right)$$

Looped

```
for k=2:(N-1)
B(:,k)=B(:,k)+ α ( E(:,k+1)-E(:,k) )
end
```

Vectorized

```
B(:,2:end-1)=B(:,2:end-1)
+ α ( E(:,3:end-1)-E(:,1:end-2) )
```

Star-P Implementation

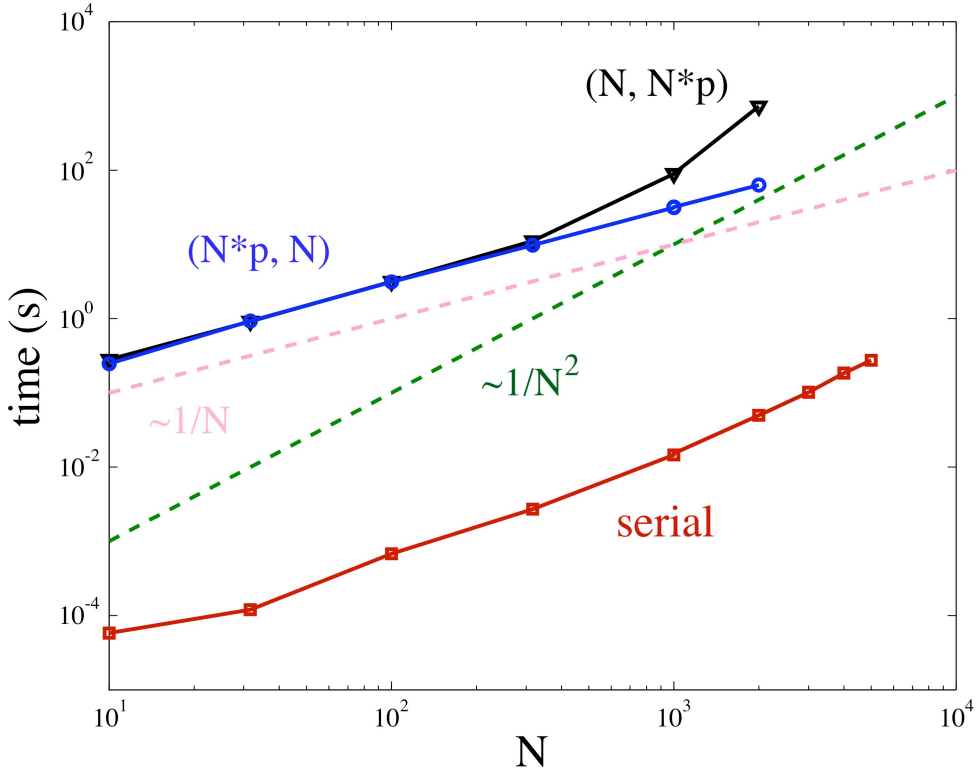
Power of vectorization

(1D parallelization)

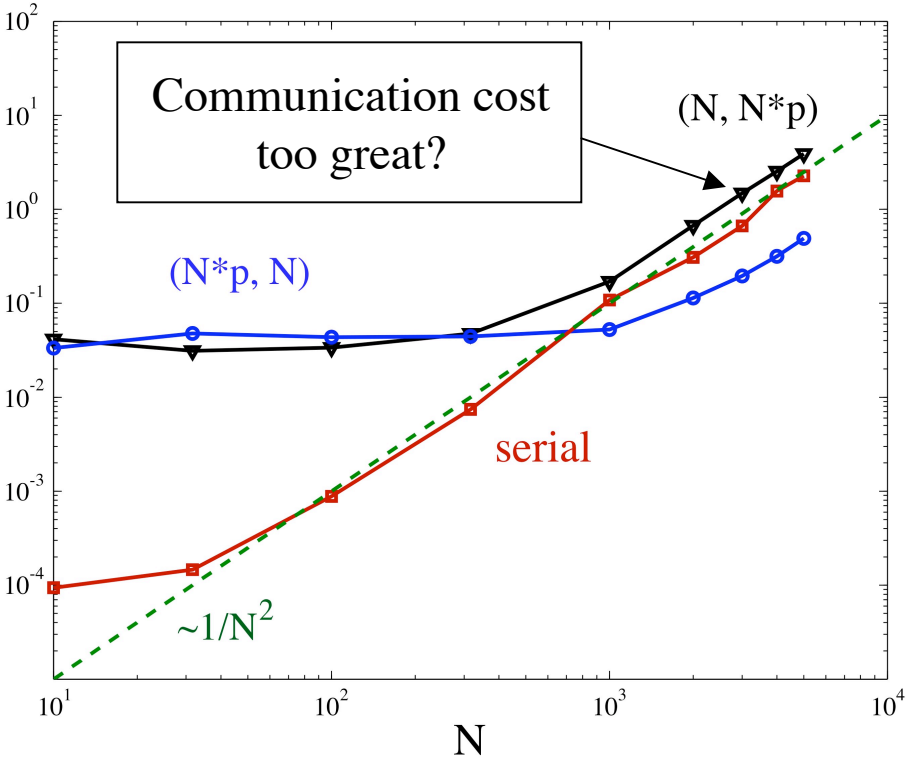
$$B_{i,j} = B_{i,j} + \alpha\left(E_{i,j+1} - E_{i,j}\right)$$

Communication over second index

looped



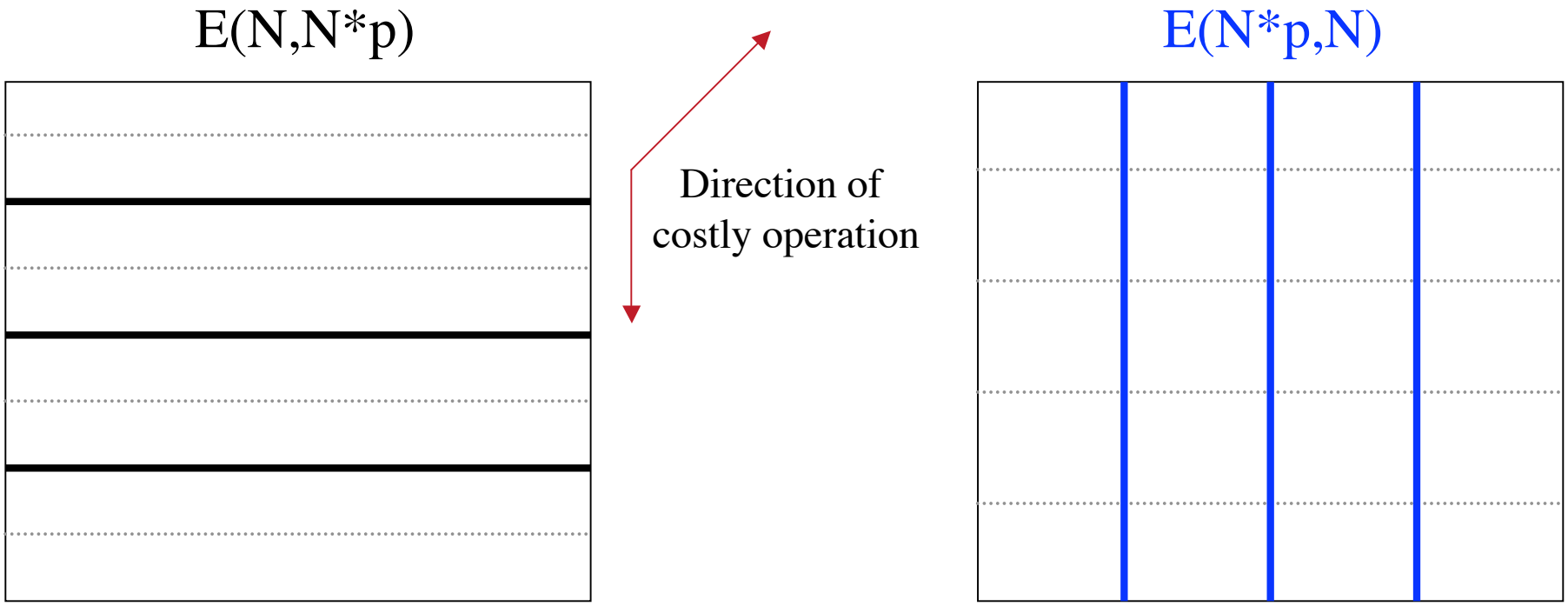
vectorized



Star-P Implementation

1D parallelization

$$B_{i,j} = B_{i,j} + \alpha\left(E_{i,j+1} - E_{i,j}\right)$$



$$\sim \mathrm{O}\left(\frac{n^2}{n_p} + nn_p\right)$$

Parallelizing over direction perpendicular to operation means constant comm. cost!

$$\sim \mathrm{O}\left(\frac{n^2}{n_p} + 1\right)$$

2D Maxwell Equations

Back to our problem

(1) $$B^{n}_{x,(i+,j+)} = B^{n-1}_{x,(i+,j+)} + \frac{\Delta t}{\Delta y}\left(E^{n-1}_{z,(i,j+1)} - E^{n-1}_{z,(i,j-1)}\right)$$

(2) $$B^{n}_{y,(i+,j+)} = B^{n-1}_{y,(i+,j+)} - \frac{\Delta t}{\Delta x}\left(E^{n-1}_{z,(i+1,j)} - E^{n-1}_{z,(i-1,j)}\right)$$

$$H^{n}_{(i+,j+)} = \mu_0^{-1} B^{n}_{(i+,j+)}$$

(3) $$D^{n}_{z,(i,j)} = D^{n-1}_{z,(i,j)} + \frac{\Delta t}{\Delta x}\left(H^{n-1}_{y,(i+,j)} - H^{n-1}_{y,(i-,j)}\right) - \frac{\Delta t}{\Delta y}\left(H^{n-1}_{x,(i,j+)} - H^{n-1}_{x,(i,j-)}\right) + 4\pi J^{n-1}_{z,(i,j)}$$

$$E^{n}_{(i,j)} = \varepsilon^{-1}_{(i,j)} D^{n}_{(i,j)}$$

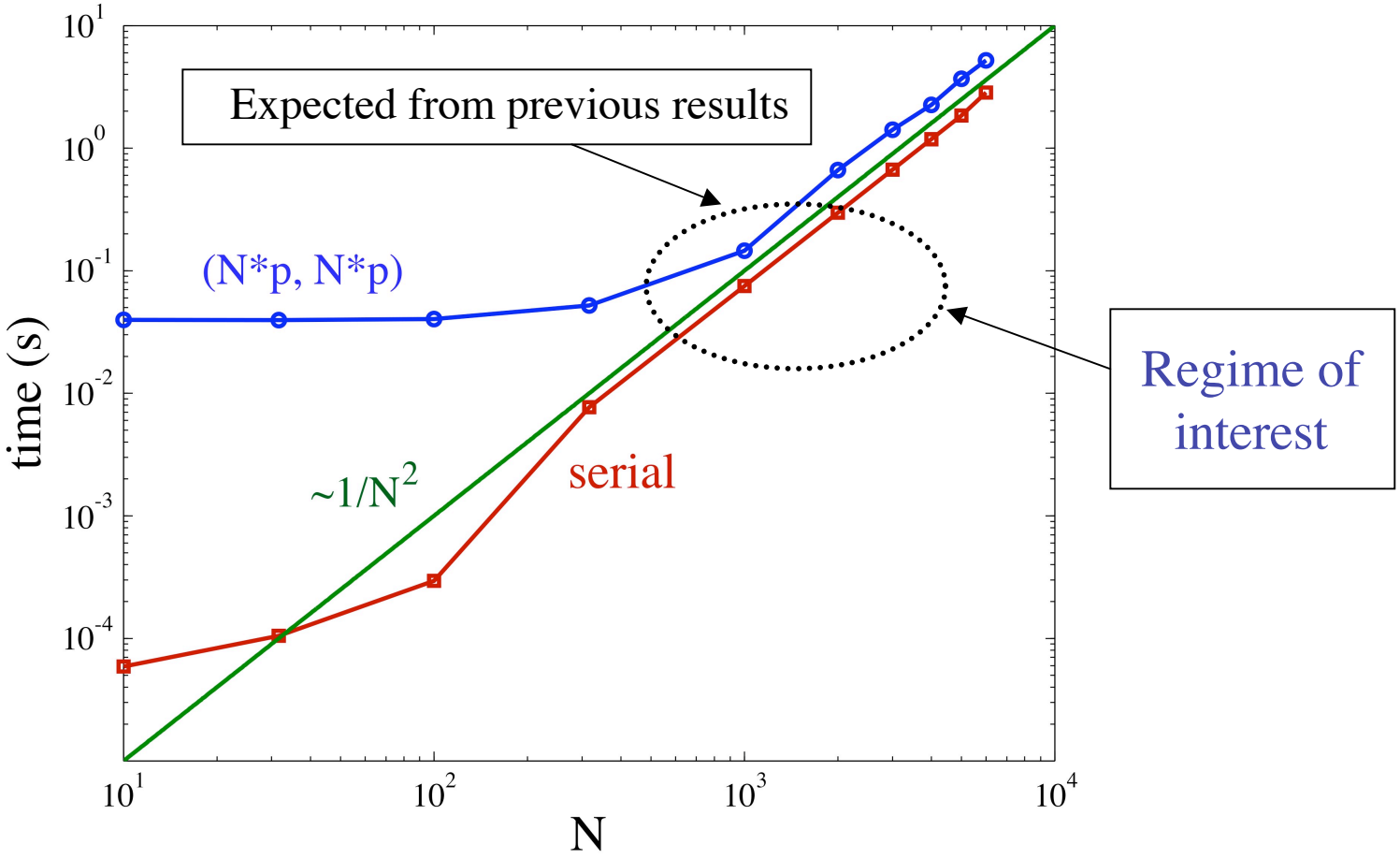
Our problems mixes direction of cost-operations

1D parallelization will be susceptible to communication costs due to either (1) or (2) and certainly due to (3)

Let's try 2D parallelization!

2D Maxwell Equations

2D parallelization?



2D Maxwell Equations

A solution: hybridization!

$$B^{n}_{x,(i+,j+)} = B^{n-1}_{x,(i+,j+)} + \frac{\Delta t}{\Delta y}\left(E^{n-1}_{z,(i,j+1)} - E^{n-1}_{z,(i,j-1)}\right)$$

$$B^{n}_{y,(i+,j+)} = B^{n-1}_{y,(i+,j+)} - \frac{\Delta t}{\Delta x}\left(E^{n-1}_{z,(i+1,j)} - E^{n-1}_{z,(i-1,j)}\right)$$

$$H^{n}_{(i+,j+)} = \mu_0^{-1} B^{n}_{(i+,j+)}$$

$$D^{n}_{z,(i,j)} = D^{n-1}_{z,(i,j)} + \frac{\Delta t}{\Delta x}\left(H^{n-1}_{y,(i+,j)} - H^{n-1}_{y,(i-,j)}\right) - \frac{\Delta t}{\Delta y}\left(H^{n-1}_{x,(i,j+)} - H^{n-1}_{x,(i,j-)}\right) + 4\pi J^{n-1}_{z,(i,j)}$$

$$E^{n}_{(i,j)} = \varepsilon^{-1}_{(i,j)} D^{n}_{(i,j)}$$

Hybrid parallelization scheme

$$B^{n}_{x} \in (N * p \times N) \sim E^{n}_{z,x} \in (N * p \times N)$$

$$B^{n}_{y} \in (N \times N * p) \sim E^{n}_{z,y} \in (N \times N * p)$$

Auxiliary fields

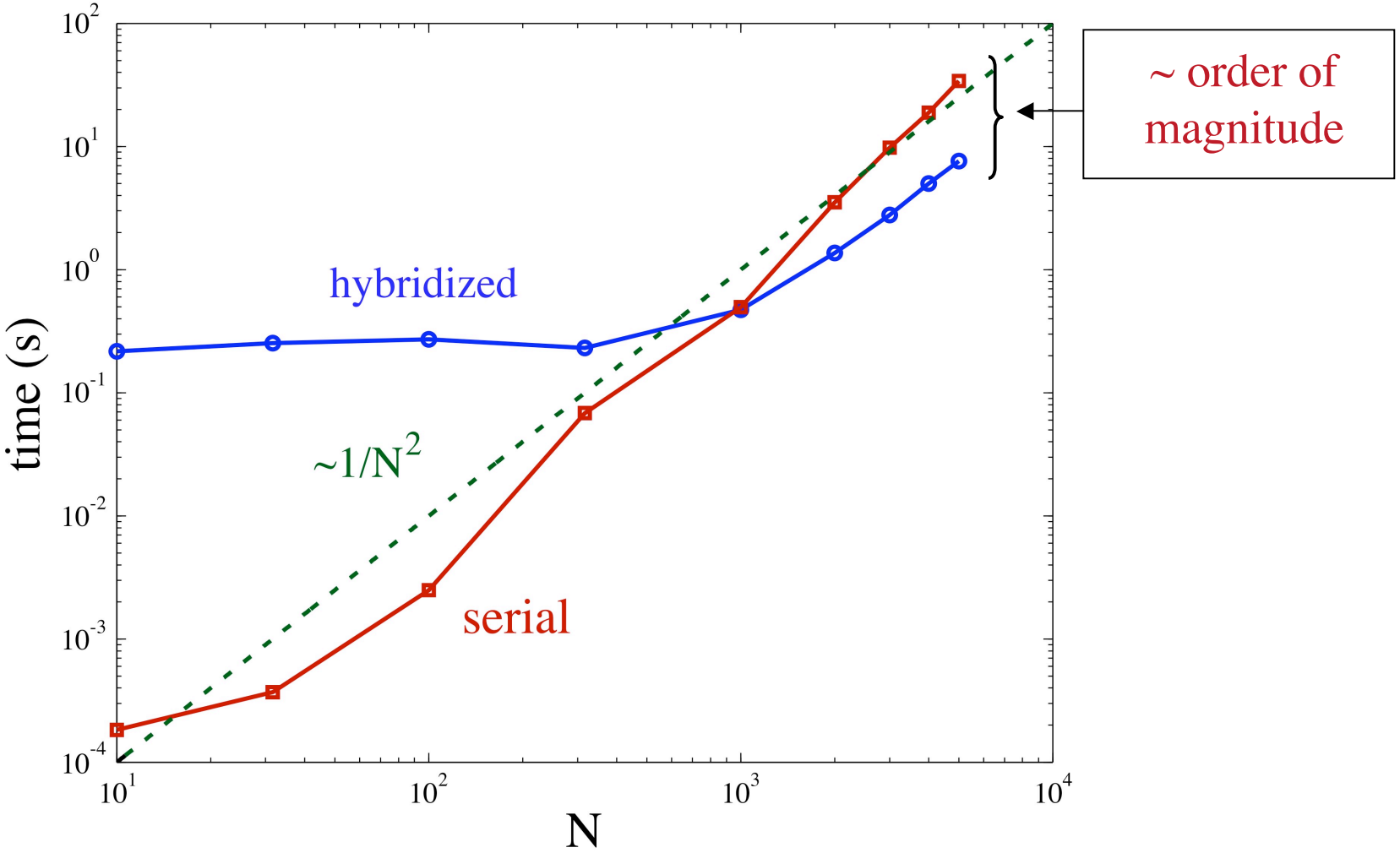
$$D^{n}_{z} \in (N * p \times N * p) \sim B^{n}_{y} \in (N \times N * p) + B^{n}_{x} \in (N * p \times N)$$

$$E^{n}_{z,x} \in (N \times N * p) \sim D^{n}_{z} \in (N * p \times N * p)$$

$$E^{n}_{z,y} \in (N * p \times N) \sim D^{n}_{z} \in (N * p \times N * p)$$

2D Maxwell Equations

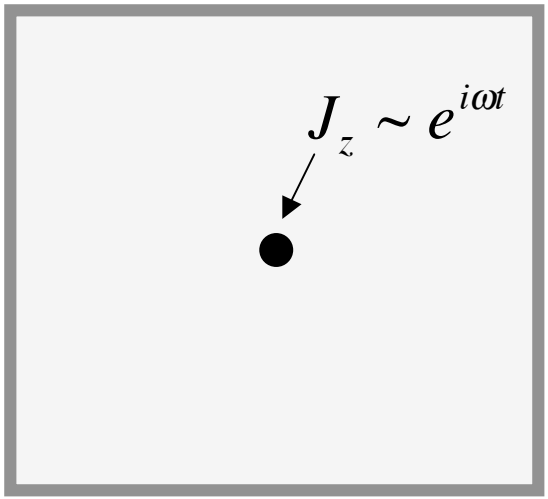
hybridization wins!



Example 1

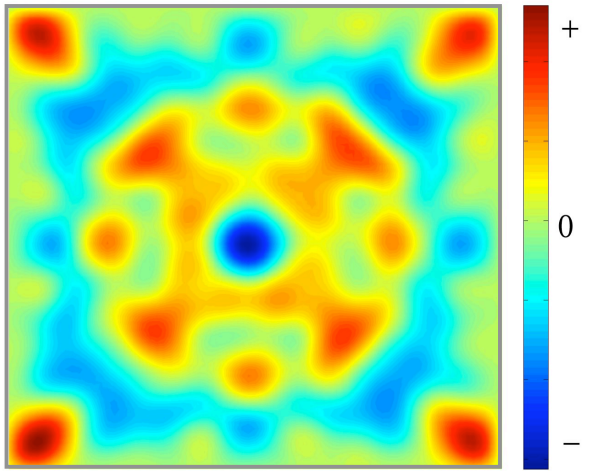
Field visualization

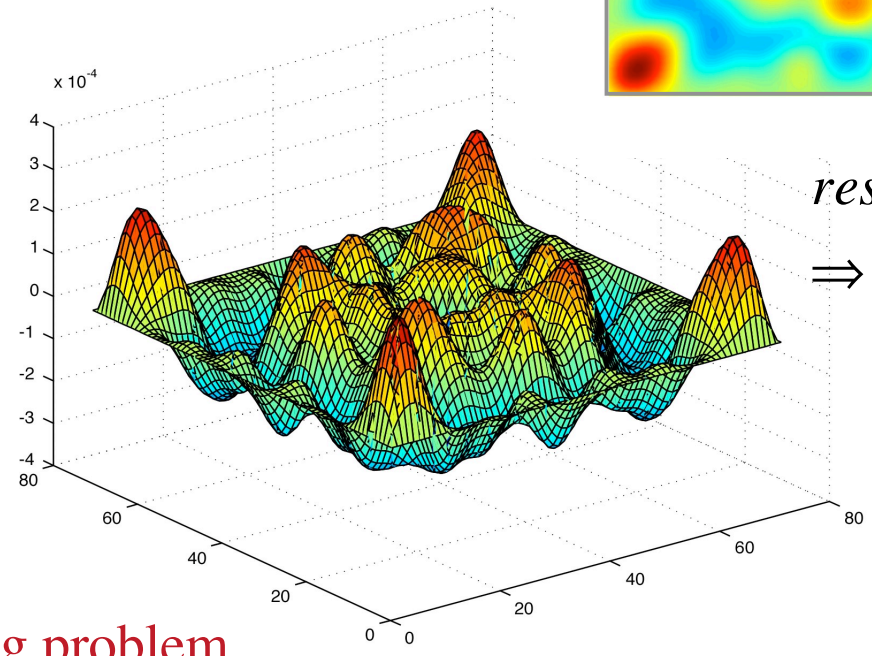
Metallic geometry



A quadrupole is born!

Steady-state field E_z





$res = 100$

$\Rightarrow N_{pixels} = 10{,}000$

Serial ~ 1 minute

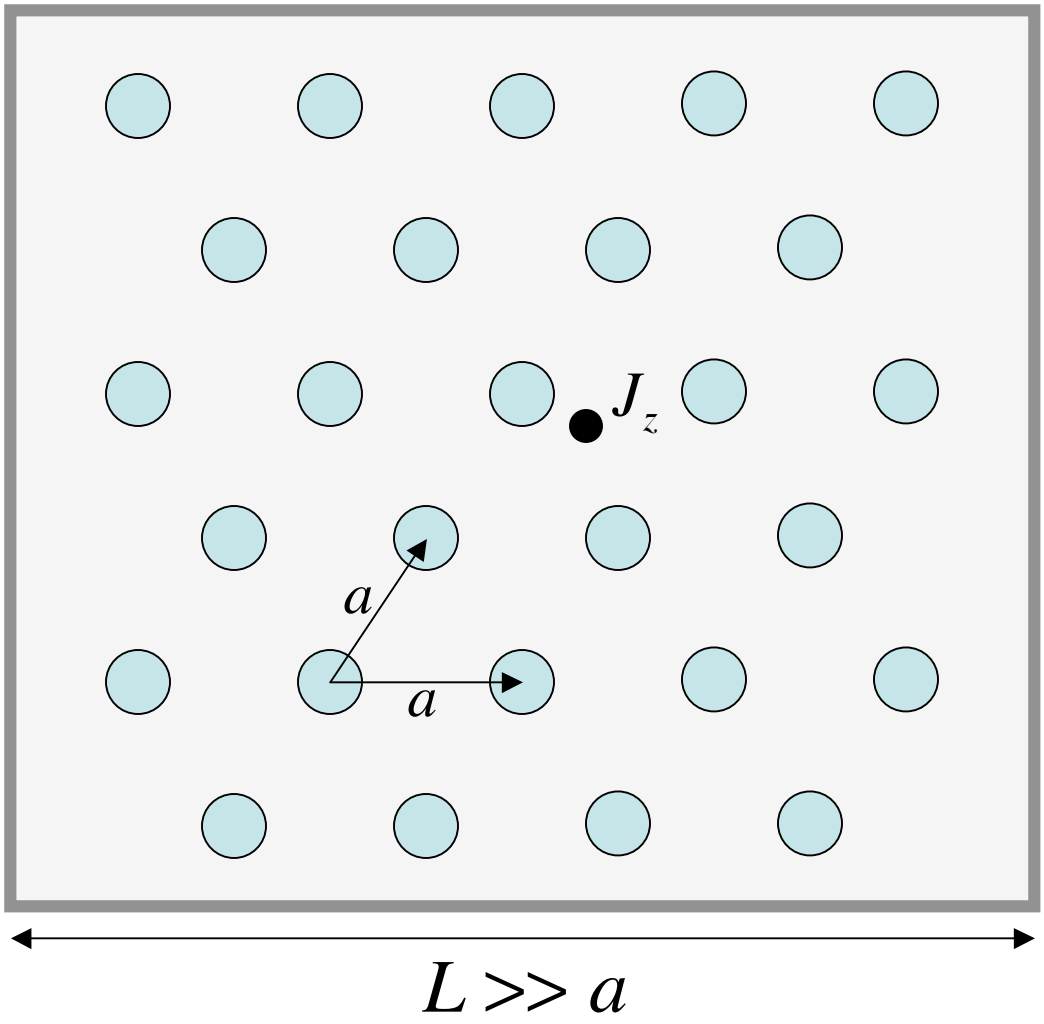
Parallel ~ 10 minutes

Let's go to a more interesting problem…

Example 2

Field visualization

PhC-Metal geometry



Using moderate resolutions

$$res \sim 30$$

$$\Rightarrow N_{pixels} = O(10^8)$$

Is this a job for star-Maxwell?

serial = impossible

parallel ~ hour

Future Work

(coming months)

C++ for loops much faster than Matlab's

(try MPI implementation --- task-parallel approach)

A promising optimization (3D geometries):