Modeling plasma accelerators I: “Full” Electromagnetic Particle-In-Cell codes

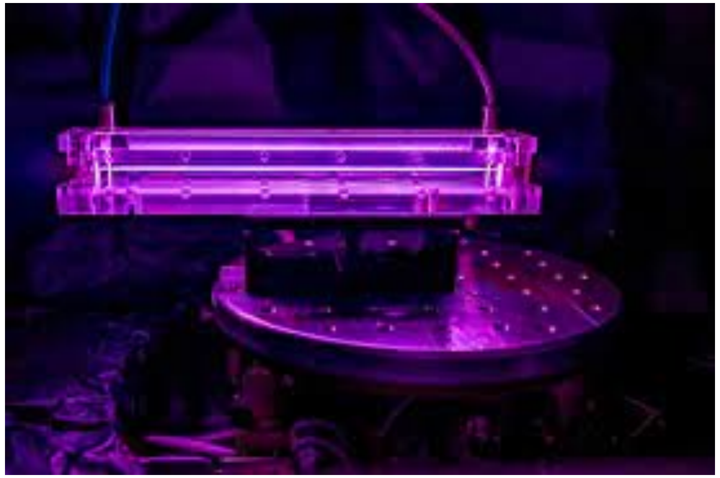
Remi Lehe

Lawrence Berkeley National Laboratory, Berkeley, California

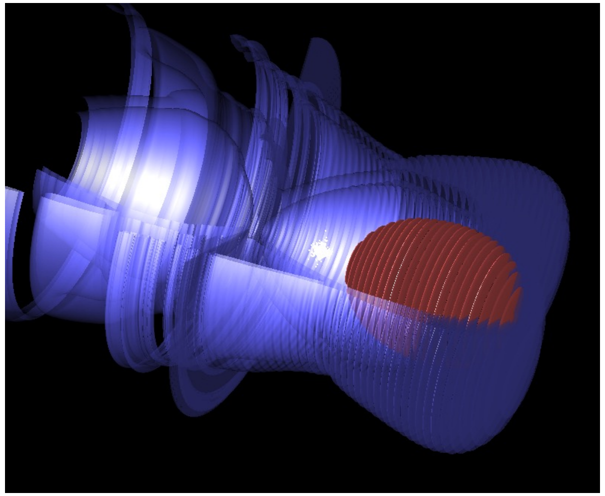
February 6, 2023

Numerical simulations of plasmas: motivations

- Typical **needs for modelling**:
  - Designing future experiments:
    e.g. quantitative estimate of beam properties
  - Interpreting existing experiments
    esp. with limited diagnostics



- **Analytical models** are **key** to our understanding,
  but cannot always capture all relevant phenomena, e.g.
  - Exact fields in the bubble regime
  - Non-linear focusing of complex lasers in plasmas
  - …



- Thus, in many cases, **numerical simulations** are needed.

Overview

Modeling plasma accelerators I:
"Full" Electromagnetic Particle-In-Cell codes

- Fundamental equations for plasma accelerators
- The electromagnetic Particle-In-Cell (PIC) algorithm
- Discretizing the field and particle equations
- Parallelization of PIC codes

Modeling plasma accelerators II:
Making the simulations faster

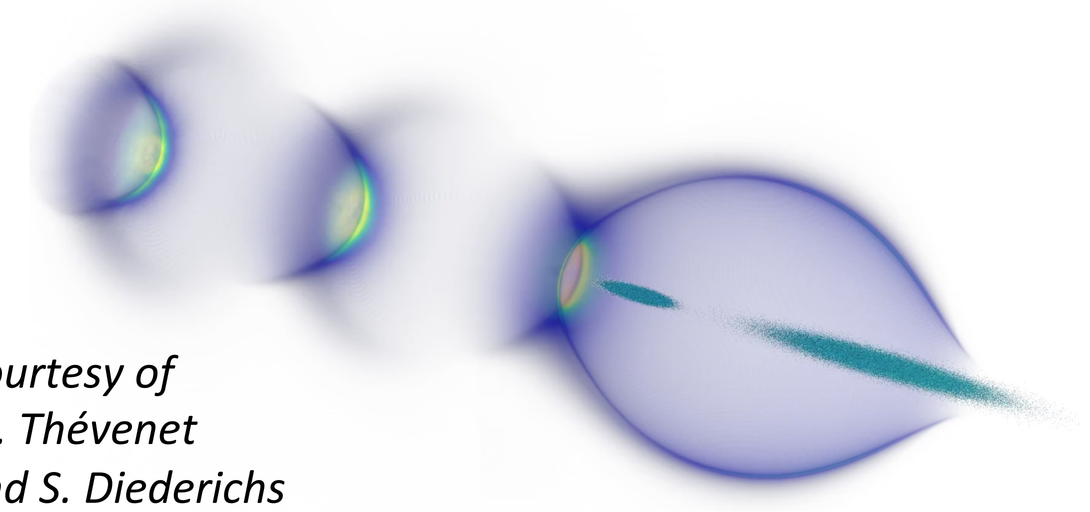
- The boosted-frame technique
- Cylindrical geometry
- Laser envelope model
- Quasi-static PIC codes

Outline

- **Fundamental equations for plasma accelerators**
- The electromagnetic Particle-In-Cell (PIC) algorithm
- Discretizing the field and particle equations
- Parallelization of PIC codes

Which fundamental equations capture the physics of beam-driven plasma acceleration?

- The driver beam creates strong E and B fields ("space-charge fields")
- These fields displace the background plasma particles.
- The displaced plasma particles generate E and B fields behind the driver (the "wakefield")
- The wakefield accelerates the witness beam (and decelerates the driver beam).



Courtesy of M. Thévenet and S. Diederichs

Maxwell's equations

$$\boldsymbol{\nabla}\cdot\boldsymbol{E} = \frac{\rho}{\epsilon_0} \qquad \partial_t\boldsymbol{B} = -\boldsymbol{\nabla}\times\boldsymbol{E}$$

$$\boldsymbol{\nabla}\cdot\boldsymbol{B} = 0 \qquad \frac{1}{c^2}\partial_t\boldsymbol{E} = \boldsymbol{\nabla}\times\boldsymbol{B} - \mu_0\boldsymbol{j}$$

Relativistic equations of motion, with Lorentz force

$$\frac{d\,\boldsymbol{p}}{dt} = q(\boldsymbol{E} + \boldsymbol{v}\times\boldsymbol{B})$$

$$\frac{d\,\boldsymbol{x}}{dt} = \boldsymbol{v} \qquad \left(\boldsymbol{p} = \frac{m\boldsymbol{v}}{\sqrt{1-\boldsymbol{v}^2/c^2}}\right)$$

Which fundamental equations capture the physics of beam-driven plasma acceleration?

Maxwell's equations

$$\boldsymbol{\nabla} \cdot \boldsymbol{E} = \frac{\rho}{\epsilon_0} \qquad \partial_t \boldsymbol{B} = -\boldsymbol{\nabla} \times \boldsymbol{E}$$

$$\boldsymbol{\nabla} \cdot \boldsymbol{B} = 0 \qquad \frac{1}{c^2}\partial_t \boldsymbol{E} = \boldsymbol{\nabla} \times \boldsymbol{B} - \mu_0 \boldsymbol{j}$$

ρ, j: contains the contributions from the **driver and witness beam** and the **plasma particles**

E, B: contains superposition of the space-charge field from **the driver and witness beam**, and the wakefield from the **plasma particles**.

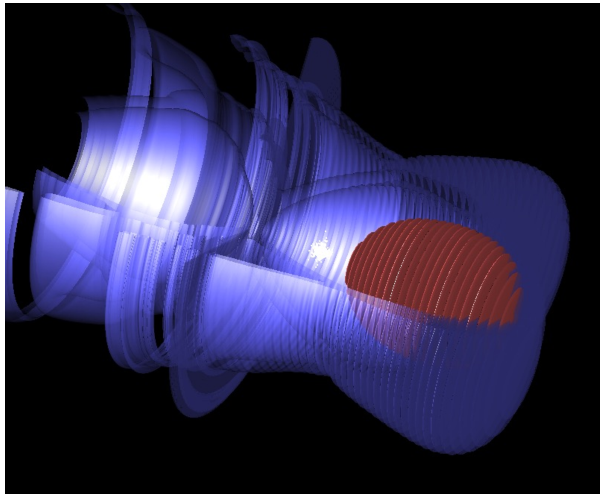
Relativistic equations of motion, with Lorentz force

$$\frac{d\boldsymbol{p}}{dt} = q(\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B})$$

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{v} \qquad \left(\boldsymbol{p} = \frac{m\boldsymbol{v}}{\sqrt{1 - \boldsymbol{v}^2/c^2}}\right)$$

Is applied to every particles of the **driver and witness beam** as well as the **plasma particles**

Which fundamental equations capture the physics of laser-driven plasma acceleration?



- The laser pulse is an electromagnetic wave (oscillating E and B field)
- The net effect of the oscillating E and B is to displace the plasma particles ("ponderomotive force")

Maxwell's equations

$$\boldsymbol{\nabla} \cdot \boldsymbol{E} = \frac{\rho}{\epsilon_0} \qquad \partial_t \boldsymbol{B} = -\boldsymbol{\nabla} \times \boldsymbol{E}$$

$$\boldsymbol{\nabla} \cdot \boldsymbol{B} = 0 \qquad \frac{1}{c^2}\partial_t \boldsymbol{E} = \boldsymbol{\nabla} \times \boldsymbol{B} - \mu_0 \boldsymbol{j}$$

- The displaced plasma particles generate E and B fields behind the driver (the "wakefield")
- The wakefield accelerates the witness beam (and decelerates the driver beam).
- The laser pulse evolves as it propagates through the plasma.

Relativistic equations of motion, with Lorentz force

$$\frac{d\boldsymbol{p}}{dt} = q(\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B})$$

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{v} \qquad \left(\boldsymbol{p} = \frac{m\boldsymbol{v}}{\sqrt{1 - \boldsymbol{v}^2/c^2}}\right)$$

Which fundamental equations capture the physics of laser-driven plasma acceleration?

Maxwell's equations

$$\boldsymbol{\nabla}\cdot\boldsymbol{E}=\frac{\rho}{\epsilon_0}\qquad \partial_t\boldsymbol{B}=-\boldsymbol{\nabla}\times\boldsymbol{E}$$

$$\boldsymbol{\nabla}\cdot\boldsymbol{B}=0\qquad \frac{1}{c^2}\partial_t\boldsymbol{E}=\boldsymbol{\nabla}\times\boldsymbol{B}-\mu_0\boldsymbol{j}$$

ρ, j: contains the contributions from the **witness beam** and the **plasma particles**

E, B: contains superposition of the **laser electromagnetic field**, the wakefield from the **plasma particles**, and the space charge field from the **witness beam**

Relativistic equations of motion, with Lorentz force

$$\frac{d\boldsymbol{p}}{dt}=q(\boldsymbol{E}+\boldsymbol{v}\times\boldsymbol{B})$$

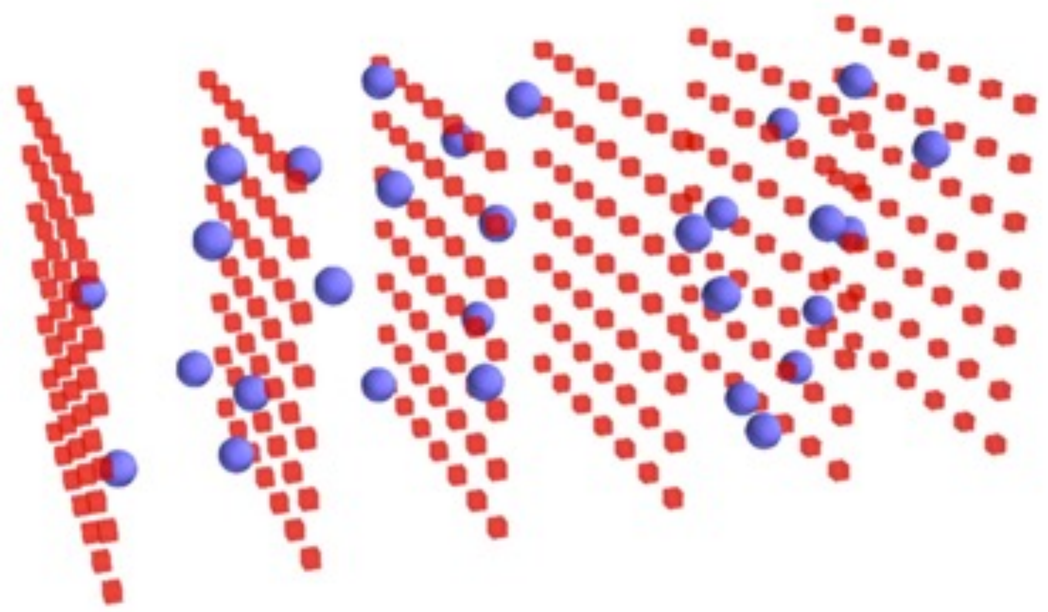
$$\frac{d\boldsymbol{x}}{dt}=\boldsymbol{v}\qquad\left(\boldsymbol{p}=\frac{m\boldsymbol{v}}{\sqrt{1-\boldsymbol{v}^2/c^2}}\right)$$

Is applied to every particles of the **witness beam** as well as the **plasma particles**

Outline

- Fundamental equations for plasma accelerators
- **The electromagnetic Particle-In-Cell (PIC) algorithm**
- Discretizing the field and particle equations
- Parallelization of PIC codes

The fields and particles are represented in different ways.



The E and B fields are represented on a grid (standard way to solve PDEs).

The beam and plasma particles are represented by discrete particles, that move through the grid.

The plasma and beams are actually represented by "macroparticles".

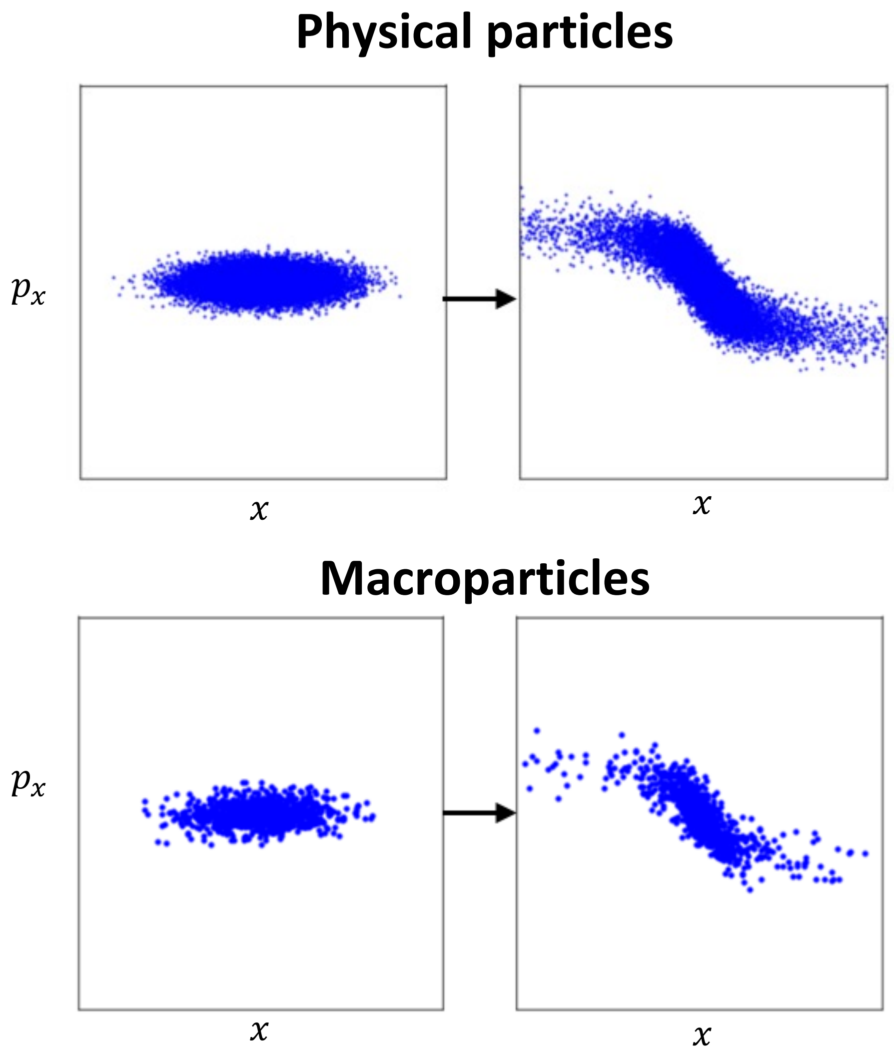
- Typical beams : $\sim 10^8 - 10^{10}$ particles.
  $(100\ \mu m)^3$ of plasma at $10^{18}\ cm^{-3}$: 10^{12} particles.
  $\Longrightarrow$ **It is too computationally expensive to represent and track every single physical particle!**

- In practice, particles that are **initially close** in 6D phase space follow **similar trajectories.**

- Simulations use **macroparticles**, which represent several physical particles that are close in phase space.

  Each macroparticle is characterized by its:

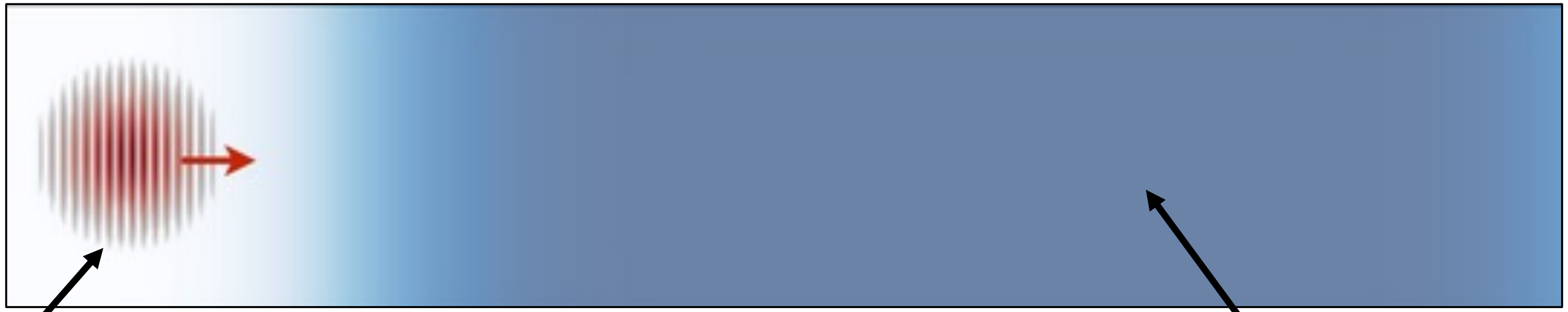
  - Position **x**
  - Momentum **p**
  - "Weight" **w** (= how many physical particles it represents)



The workflow of Particle-In-Cell simulation: initialization

1. Initialize the plasma and laser at t=0

Simulation box



Laser pulse

E and B fields can be initialized on the grid using known analytical formula e.g. Gaussian pulse:

$$E \propto a_0 \exp\left(-\frac{z-z0}{c\tau} - \frac{x^2+y^2}{w_0}\right) \cos\left(\frac{2\pi}{\lambda} z\right)$$

Plasma macroparticles

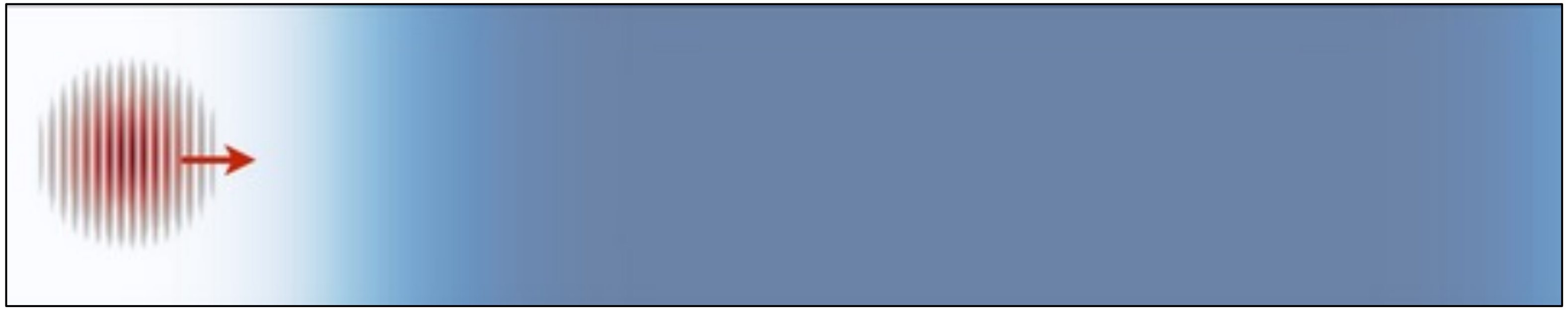
(neutral plasma, usually initialized at rest)

E = 0, B = 0

Note: another popular method is to emit the laser **continuously throughout the simulation**, from one of the boundaries or from an “antenna”

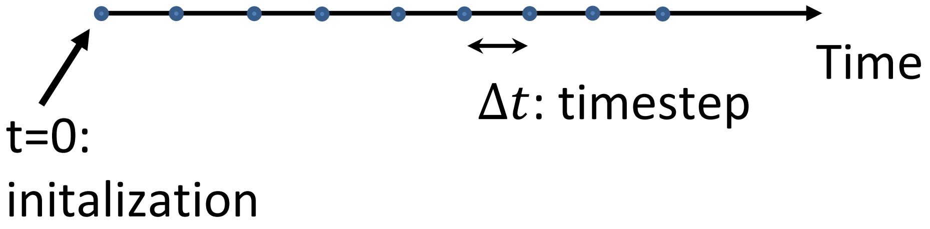
The workflow of Particle-In-Cell simulation: time evolution

1. Initialize the plasma and laser at t=0



2. Repeatedly update the fields and particles, in discrete timesteps

using a discretized version of the Maxwell equations and equations of motion



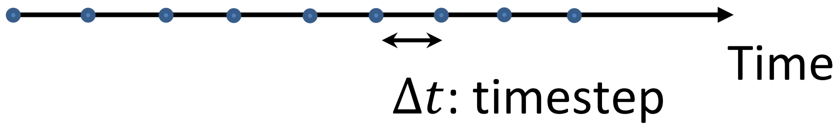
$$\partial_t \boldsymbol{B} = -\boldsymbol{\nabla} \times \boldsymbol{E}$$

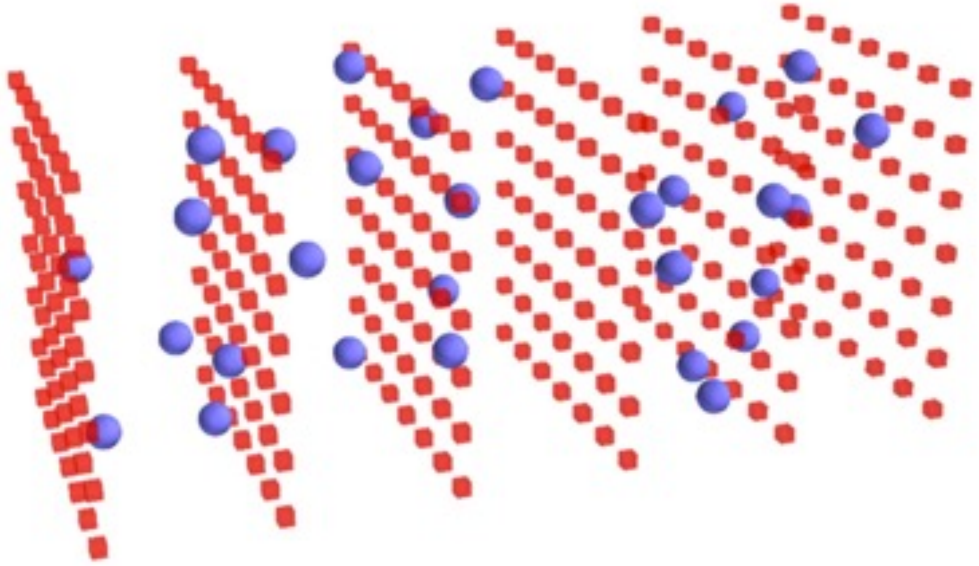
$$\partial_t \boldsymbol{E} = c^2 \boldsymbol{\nabla} \times \boldsymbol{B} - \mu_0 c^2 \boldsymbol{j}$$

$$\frac{d\boldsymbol{p}}{dt} = q(\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B})$$

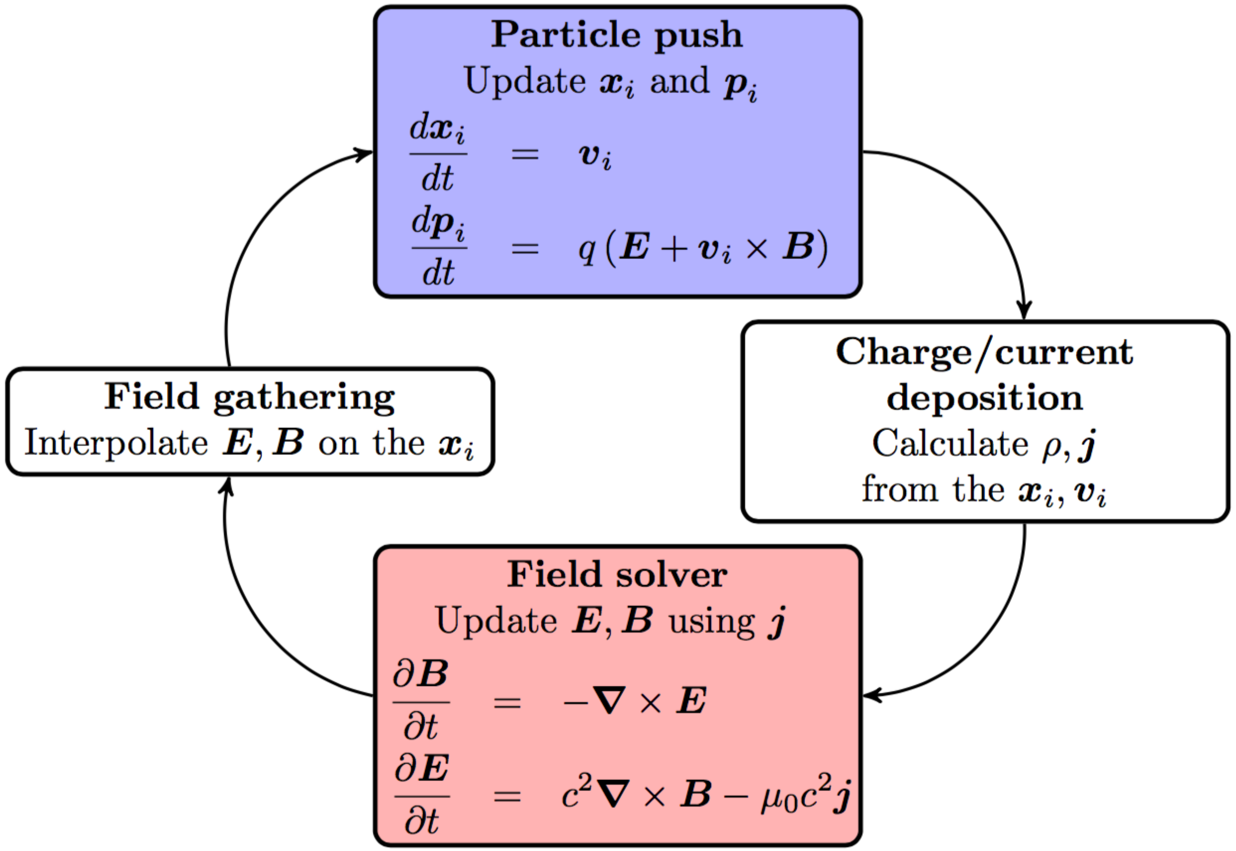
$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{v} \qquad \left(\boldsymbol{p} = \frac{m\boldsymbol{v}}{\sqrt{1 - \boldsymbol{v}^2/c^2}} \right)$$

The time evolution of particles and fields is computed over discrete timesteps, using the PIC loop

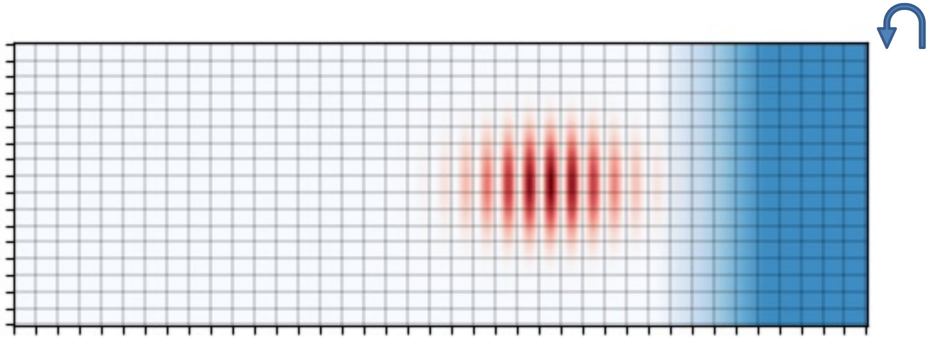




At each Δt : one iteration of the following loop

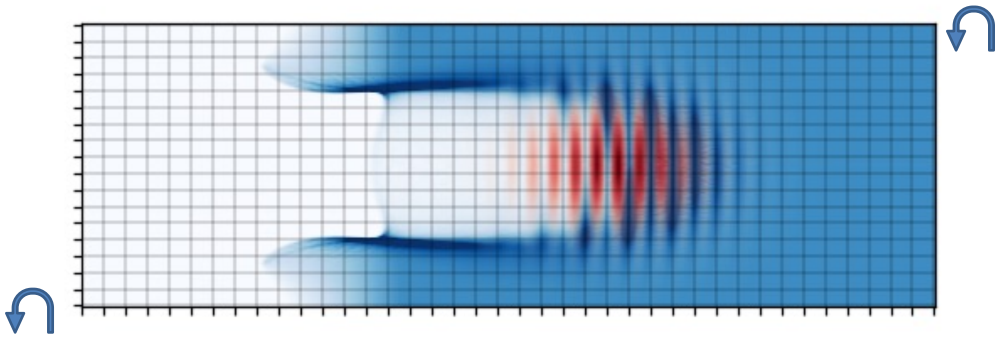


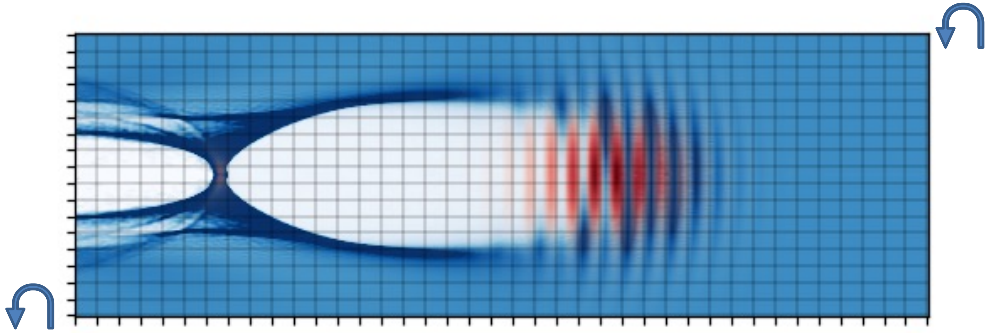
Instead of having the simulation box cover the whole plasma, PIC simulations often use a moving window



At each Δt: add a new row of grid cells, filled with fresh plasma macroparticles

At each Δt: remove last row of grid cell



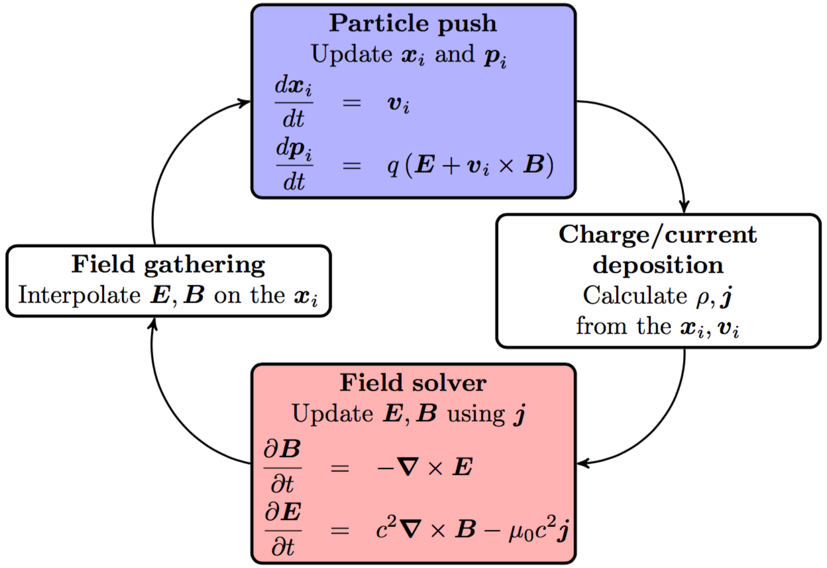


Note: the grid cells are not drawn to scale. (They should be much finer, to resolve the laser wavelength.)

Ensuring that $\nabla \cdot \boldsymbol{B} = 0$

- The PIC loop only uses 2 out of the 4 Maxwell equations.
  How do we ensure that $\boldsymbol{\nabla} \cdot \boldsymbol{B} = 0$?

- For the **continuous** Maxwell equations:



> - $\boldsymbol{\nabla} \cdot \boldsymbol{B} = 0$ at $\boldsymbol{t} = \boldsymbol{0}$
> - $\boldsymbol{B}$ evolves according to: $\partial_t \boldsymbol{B} = -\boldsymbol{\nabla} \times \boldsymbol{E}$

$\Longrightarrow$

> $\boldsymbol{\nabla} \cdot \boldsymbol{B} = 0$ at **any** $\boldsymbol{t}$

- This remains true for the **discretized** Maxwell equations.
  (with the most common discretization schemes)

- It is therefore important to ensure $\boldsymbol{\nabla} \cdot \boldsymbol{B} = 0$ in the simulation box, when **initializing** the simulation

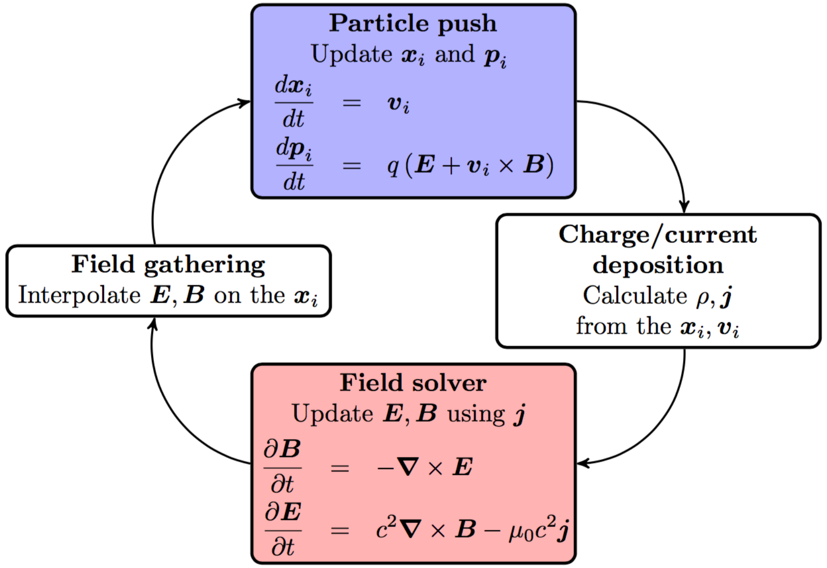
Ensuring that $\nabla \cdot \boldsymbol{E} = \rho/\epsilon_0$

- The PIC loop only uses 2 out of the 4 Maxwell equations.
  How do we ensure that $\nabla \cdot \boldsymbol{E} = \rho/\epsilon_0$?

- For the **continuous** Maxwell equations:

  - $\nabla \cdot \boldsymbol{E} = \rho/\epsilon_0$ at $\boldsymbol{t = 0}$
  - $\boldsymbol{E}$ evolves according to:
    $\partial_t \boldsymbol{E} = c^2 \nabla \times \boldsymbol{B} - \mu_0 c^2 \boldsymbol{j}$
  - $\boldsymbol{j}, \rho$ satisfy:
    $\partial_t \rho + \nabla \cdot \boldsymbol{j} = 0$

  $\Longrightarrow$ $\boxed{\nabla \cdot \boldsymbol{E} = \rho/\epsilon_0 \text{ at } \textbf{any } \boldsymbol{t}}$



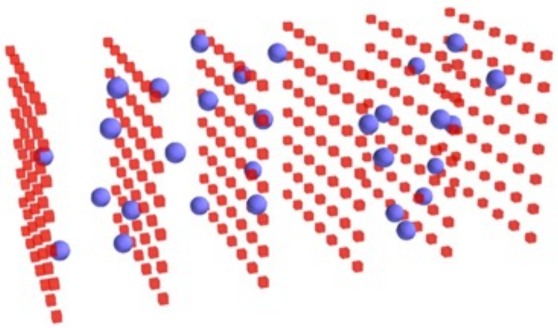
- This remains true for the **discretized** Maxwell equations
  (with the most common discretization schemes).

- However, making sure that the deposited $\boldsymbol{j}, \rho$ satisfy $\partial_t \rho + \nabla \cdot \boldsymbol{j} = 0$ is not trivial!

Outline

- Fundamental equations for plasma accelerators
- The electromagnetic Particle-In-Cell (PIC) algorithm
- **Discretizing the field and particle equations**
- Parallelization of PIC codes

Discretizing the field and particle equations



Particle push

Update $\boldsymbol{x}_i$ and $\boldsymbol{p}_i$

$$\frac{d\boldsymbol{x}_i}{dt} = \boldsymbol{v}_i$$
$$\frac{d\boldsymbol{p}_i}{dt} = q\left(\boldsymbol{E} + \boldsymbol{v}_i \times \boldsymbol{B}\right)$$

Charge/current deposition

Calculate $\rho, \boldsymbol{j}$ from the $\boldsymbol{x}_i, \boldsymbol{v}_i$

Field solver

Update $\boldsymbol{E}, \boldsymbol{B}$ using $\boldsymbol{j}$

$$\frac{\partial \boldsymbol{B}}{\partial t} = -\boldsymbol{\nabla} \times \boldsymbol{E}$$
$$\frac{\partial \boldsymbol{E}}{\partial t} = c^2 \boldsymbol{\nabla} \times \boldsymbol{B} - \mu_0 c^2 \boldsymbol{j}$$

Field gathering

Interpolate $\boldsymbol{E}, \boldsymbol{B}$ on the $\boldsymbol{x}_i$

Discretizing the equations of motion: derivatives are replaced by finite-difference approximations

Continuous equation:

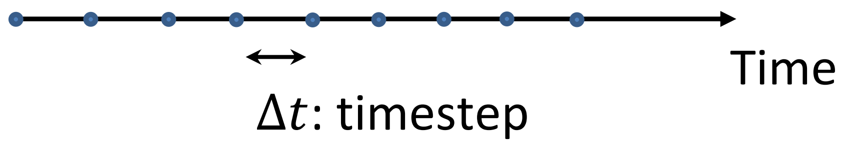
$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{v}$$

$\longrightarrow$

Discretized equation:

$$\frac{\boldsymbol{x}^{new} - \boldsymbol{x}^{old}}{\Delta t} = \boldsymbol{v}$$

i.e. $\boldsymbol{x}^{new} = \boldsymbol{x}^{old} + \boldsymbol{v}\Delta t$



Discretizing the equations of motion: derivatives are replaced by finite-difference approximations

Continuous equation:

$$\frac{d\,\boldsymbol{p}}{dt} = q(\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B})$$

$\longrightarrow$

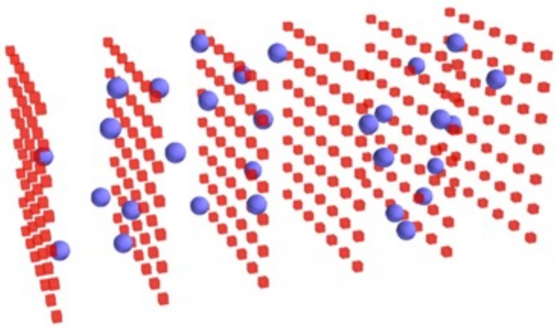
Discretized equation:

$$\frac{\boldsymbol{p}^{new} - \boldsymbol{p}^{old}}{\Delta t} = q(\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B})$$

$$\left(\boldsymbol{p} = \frac{m\boldsymbol{v}}{\sqrt{1 - \boldsymbol{v}^2/c^2}}\right)$$

Different algorithms exists (e.g. Boris, Vay, Higuera-Cary, …) depending on the details of how the term $\boldsymbol{v}\times\boldsymbol{B}$ is treated.

Discretizing the field and particle equations



Particle push

Update $\boldsymbol{x}_i$ and $\boldsymbol{p}_i$

$$\frac{d\boldsymbol{x}_i}{dt} = \boldsymbol{v}_i$$

$$\frac{d\boldsymbol{p}_i}{dt} = q\left(\boldsymbol{E} + \boldsymbol{v}_i \times \boldsymbol{B}\right)$$

Charge/current deposition

Calculate $\rho, \boldsymbol{j}$ from the $\boldsymbol{x}_i, \boldsymbol{v}_i$

Field solver

Update $\boldsymbol{E}, \boldsymbol{B}$ using $\boldsymbol{j}$

$$\frac{\partial \boldsymbol{B}}{\partial t} = -\boldsymbol{\nabla} \times \boldsymbol{E}$$

$$\frac{\partial \boldsymbol{E}}{\partial t} = c^2 \boldsymbol{\nabla} \times \boldsymbol{B} - \mu_0 c^2 \boldsymbol{j}$$

Field gathering

Interpolate $\boldsymbol{E}, \boldsymbol{B}$ on the $\boldsymbol{x}_i$

Discretizing the Maxwell equations: derivatives are replaced by finite-difference approximations

Continuous equation:

$$\frac{\partial \boldsymbol{B}}{\partial t} = -\boldsymbol{\nabla} \times \boldsymbol{E}$$

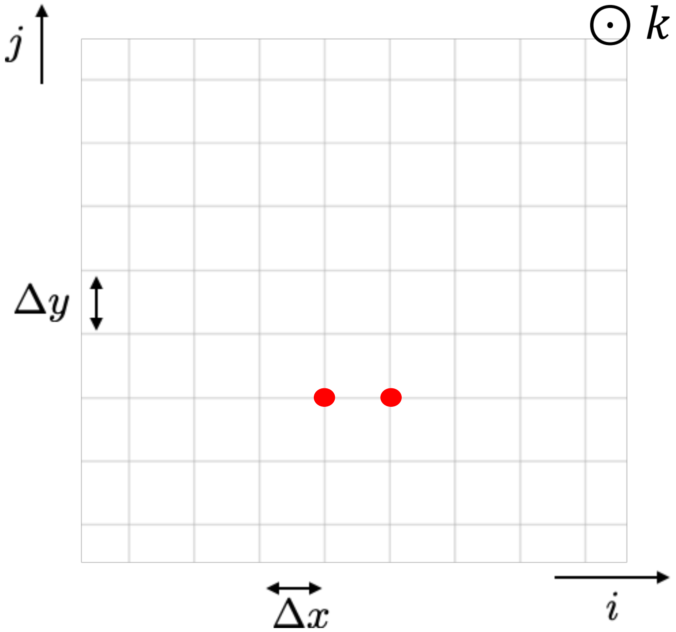
Discretized equation:

$$\frac{\boldsymbol{B}^{new} - \boldsymbol{B}^{old}}{\Delta t} = -\hat{\boldsymbol{\nabla}} \times \boldsymbol{E}$$

i.e. $\boldsymbol{B}^{new} = \boldsymbol{B}^{old} - \Delta t\, \hat{\boldsymbol{\nabla}} \times \boldsymbol{E}$

$\hat{\boldsymbol{\nabla}}$: discretized spatial derivative

e.g. $\hat{\partial}_x \boldsymbol{E} = \dfrac{\boldsymbol{E}_{i+1,j,k} - \boldsymbol{E}_{i,j,k}}{\Delta x}$



$\Delta x, \Delta y, \Delta z$: cell size

(In practice, the components of E and B are staggered in time and space in order to increase the accuracy of this discretization.)

Discretizing the Maxwell equations: derivatives are replaced by finite-difference approximations

Continuous equation:

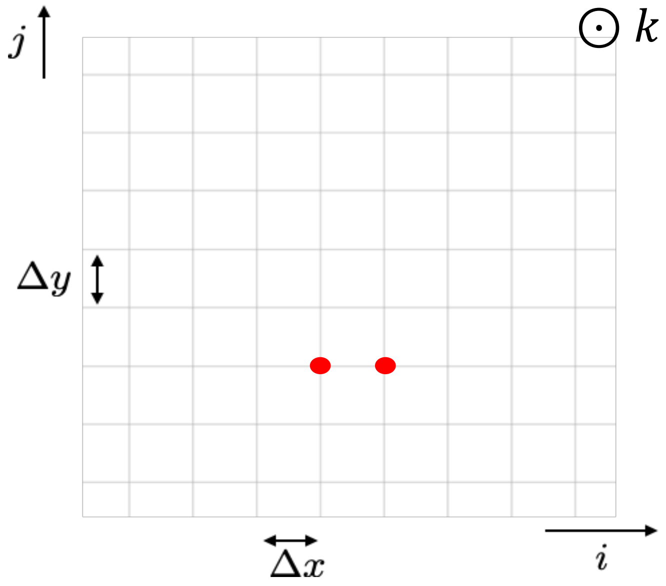
$$\frac{\partial \boldsymbol{E}}{\partial t} = c^2 \boldsymbol{\nabla} \times \boldsymbol{B} - \mu_0 c^2 \boldsymbol{j}$$

→

Discretized equation:

$$\frac{\boldsymbol{E}^{new} - \boldsymbol{E}^{old}}{\Delta t} = c^2 \hat{\boldsymbol{\nabla}} \times \boldsymbol{B} - \mu_0 c^2 \boldsymbol{j}$$

i.e. $\boldsymbol{E}^{new} = \boldsymbol{E}^{old} + \Delta t\, c^2 \hat{\boldsymbol{\nabla}} \times \boldsymbol{B} - \Delta t\, \mu_0 c^2 \boldsymbol{j}$



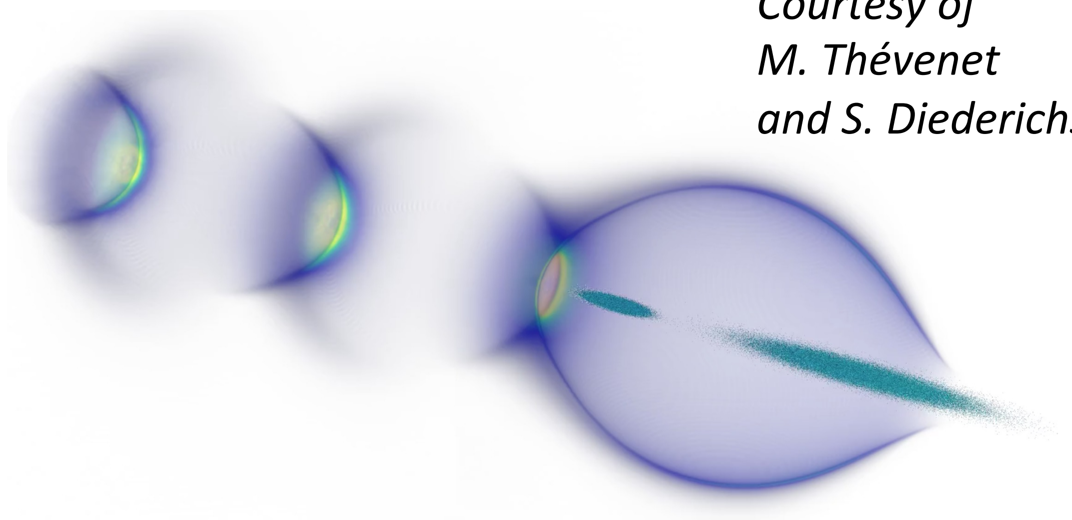
- This discretization scheme is called the **Yee scheme**.
- Other discretization schemes are also commonly used (e.g. using more points on the grid to evaluate the finite-difference spatial derivative)

Resolution requirement: beam-driven acceleration

- **Linear regime:**
  $\Delta x, \Delta y, \Delta z$ should be much **smaller** than λ_p and the beam(s) size

- **Nonlinear regime:**
  $\Delta x, \Delta y, \Delta z$ may need to be **smaller** than the thin sheath along the bubble

- If in doubt, studying **convergence** with respect to grid size is always a good idea.

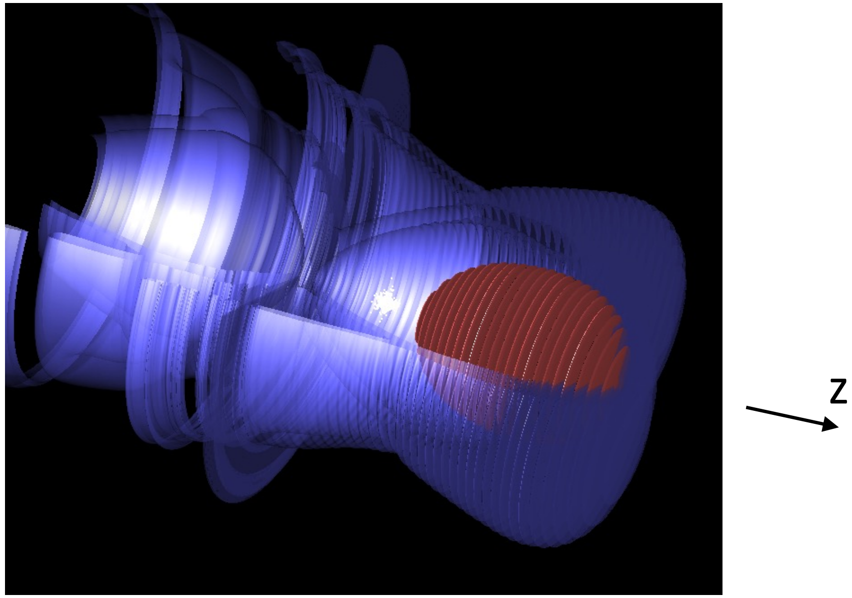


Courtesy of M. Thévenet and S. Diederichs

Resolution requirement: laser-driven acceleration

Same requirements as for beam-driven,
but **in addition**, Δz needs to be **much smaller** than the **laser wavelength**.

e.g. $$\Delta z \sim \frac{\lambda}{40}$$



Imposes that the grid be very fine in z!

The CFL limit constrains the timestep Δt.

Courant-Friedrichs-Lewy (CFL) limit

In order for the Yee discretization scheme to be **numerically stable**:

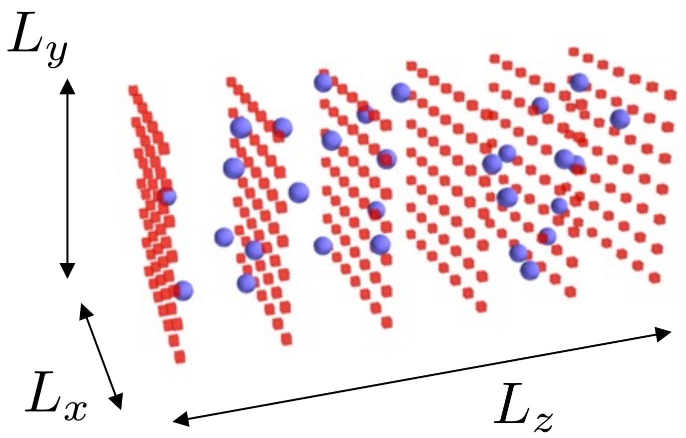
$$\Delta t < \frac{1}{c} \frac{1}{\sqrt{1/\Delta x^2 + 1/\Delta y^2 + 1/\Delta z^2}}$$

(Most other discretization schemes also have a CFL limit.)

For instance:

- For square cells ($\Delta x = \Delta y = \Delta z$): $\Delta t < \frac{\Delta z}{c\sqrt{3}}$
- If $\Delta z \ll \Delta x, \Delta y$ (e.g. laser-driven): $\Delta t < \frac{\Delta z}{c}$

Computational cost rapidly increases with grid resolution.



The number of computational operations scales like:

$$N_{comp} \propto \underbrace{\left(\frac{L_x}{\Delta x}\right) \times \left(\frac{L_y}{\Delta y}\right) \times \left(\frac{L_z}{\Delta z}\right)}_{\text{Number of grid points}} \times \underbrace{\left(\frac{T_{interaction}}{\Delta t}\right)}_{\text{Number of timesteps}}$$

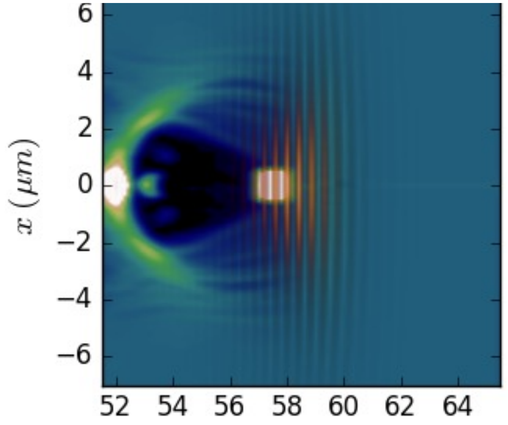
Number of timesteps (i.e. number of iterations of the PIC loop)

$T_{interaction}$ = physical duration of the simulated phenomenon (e.g. time it takes for the driver to propagate through the whole plasma)

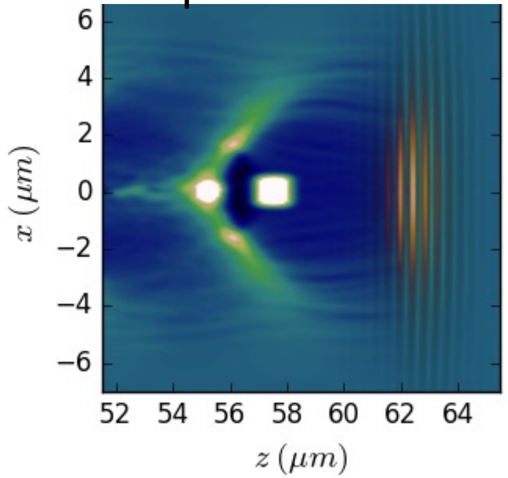
Reducing $\Delta x, \Delta y, \Delta z$ by a factor 2 increases the computational cost by a factor 16!

One consequence of using low resolution: spurious numerical dispersion

Yee scheme with low resolution (unphysical)



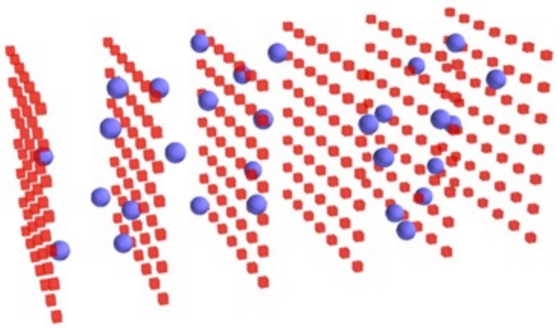
Expected result



If Δz is not **small enough** compared to the **laser wavelength**, the **group velocity** of the simulated laser pulse may be erroneous. (“spurious numerical dispersion”)

(Picking a different discretization scheme than the Yee scheme can mitigate spurious numerical dispersion.)

Discretizing the field and particle equations



Particle push

Update $\boldsymbol{x}_i$ and $\boldsymbol{p}_i$

$$\frac{d\boldsymbol{x}_i}{dt} = \boldsymbol{v}_i$$

$$\frac{d\boldsymbol{p}_i}{dt} = q\left(\boldsymbol{E} + \boldsymbol{v}_i \times \boldsymbol{B}\right)$$

Charge/current deposition

Calculate $\rho, \boldsymbol{j}$ from the $\boldsymbol{x}_i, \boldsymbol{v}_i$

Field solver

Update $\boldsymbol{E}, \boldsymbol{B}$ using $\boldsymbol{j}$

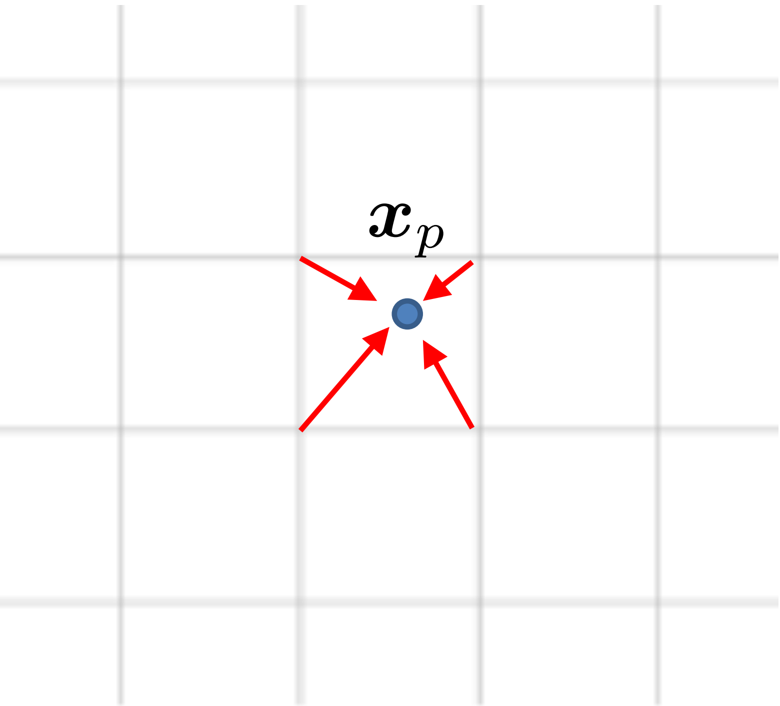
$$\frac{\partial \boldsymbol{B}}{\partial t} = -\boldsymbol{\nabla} \times \boldsymbol{E}$$

$$\frac{\partial \boldsymbol{E}}{\partial t} = c^2 \boldsymbol{\nabla} \times \boldsymbol{B} - \mu_0 c^2 \boldsymbol{j}$$

Field gathering

Interpolate $\boldsymbol{E}, \boldsymbol{B}$ on the $\boldsymbol{x}_i$

The gathered field is a weighted sum of the field on the grid.



The field felt by a macroparticle is a **weighted sum** of the field at **neighboring grid points**:

"Shape factor"

$$\boldsymbol{E}(\boldsymbol{x}_p) = \sum_{i,j,k \in \mathcal{V}(\boldsymbol{x}_p)} S(\boldsymbol{x}_p - \boldsymbol{x}_{i,j,k})\boldsymbol{E}_{i,j,k}$$

Macroparticle position

Sum over grid points in the vicinity of the macroparticle

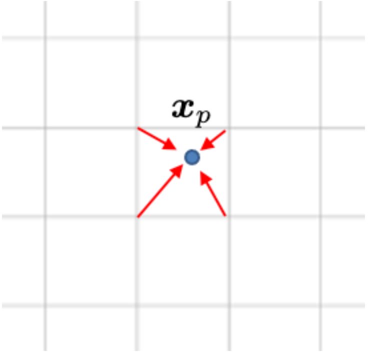
Grid point position

Field value at grid point

Different shape factors are commonly used.

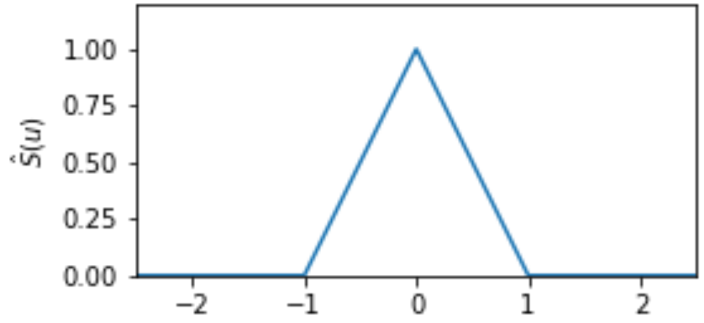
$$\boldsymbol{E}(\boldsymbol{x}_p) = \sum_{i,j,k \in \mathcal{V}(\boldsymbol{x}_p)} S(\boldsymbol{x}_p - \boldsymbol{x}_{i,j,k}) \boldsymbol{E}_{i,j,k}$$

$$S(\boldsymbol{x}_{\boldsymbol{p}} - \boldsymbol{x}_{i,j,k}) = \hat{S}\left(\frac{x_p - x_i}{\Delta x}\right) \hat{S}\left(\frac{y_p - y_i}{\Delta y}\right) \hat{S}\left(\frac{z_p - z_i}{\Delta z}\right)$$

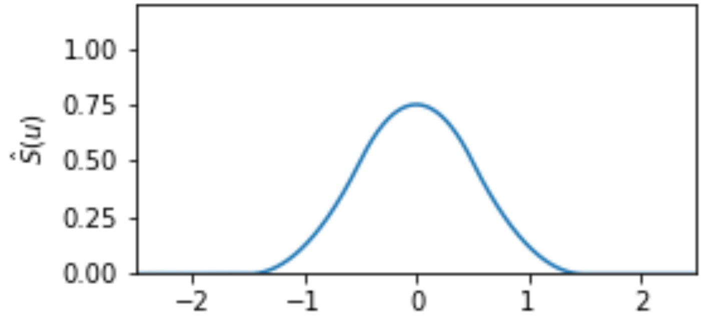


High-order particle shapes are computationally more **expensive** but often lead to **smoother** results.

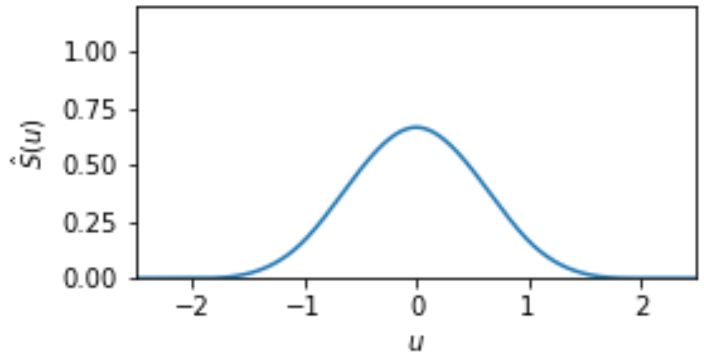
- **Linear shape factor** (a.k.a. "1st order", "CIC") involves **2 grid points** along each direction



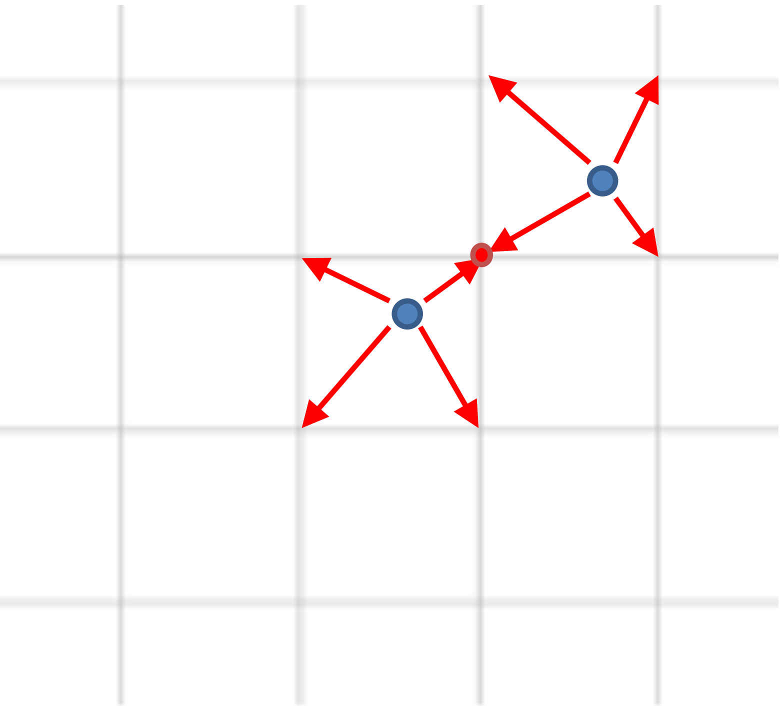
- **Quadratic shape factor** (a.k.a. "2nd order", "TSC") involves **3 grid points** along each direction



- **Cubic shape factor** (a.k.a. "3rd order", "PQS") involves **4 grid points** along each direction



Macroparticles distribute charge/current to neighboring points



Macroparticles distribute their **charge** to neighboring grid points:

$$\rho_{i,j,k} = \frac{\sum_{p \in \mathcal{V}(\boldsymbol{x}_{i,j,k})} S(\boldsymbol{x}_{i,j,k} - \boldsymbol{x}_p) q_p}{\Delta x \Delta y \Delta z}$$

Field value at grid point

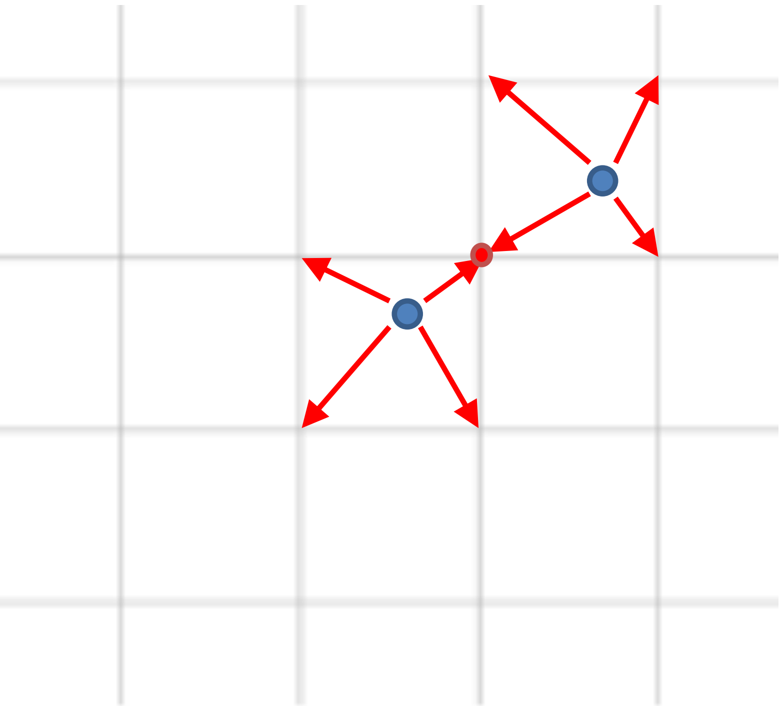
Sum over macroparticles in the vicinity of the grid point

Cell volume

Total charge of the macroparticle

(takes into account weight)

Macroparticles distribute charge/current to neighboring points



Macroparticles distribute their **current** to neighboring grid points:

$$\boldsymbol{J}_{i,j,k} = \frac{\sum_{p \in \mathcal{V}(\boldsymbol{x}_{i,j,k})} S(\boldsymbol{x}_{i,j,k} - \boldsymbol{x}_p) q_p \boldsymbol{v}_p}{\Delta x \Delta y \Delta z}$$

Note: that are also more advanced deposition schemes for **J** (e.g. Esirkepov scheme), that automatically ensure

$$\partial_t \rho + \boldsymbol{\nabla} \cdot \boldsymbol{J} = 0$$

Outline

- Fundamental equations for plasma accelerators
- The electromagnetic Particle-In-Cell (PIC) algorithm
- Discretizing the field and particle equations
- **Parallelization of PIC codes**

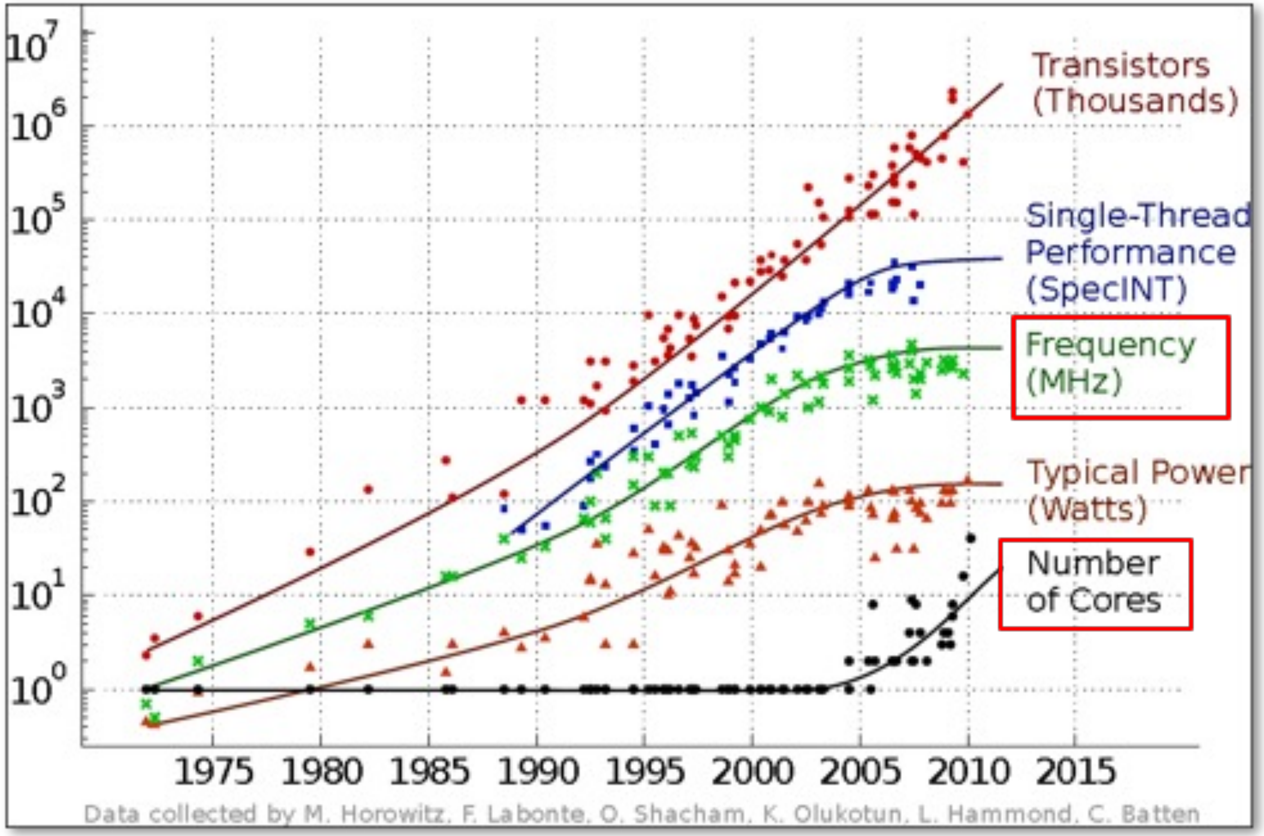
Full PIC simulations can rapidly be very computationally expensive.

$$N_{comp} \propto \underbrace{\left(\frac{L_x}{\Delta x}\right) \times \left(\frac{L_y}{\Delta y}\right) \times \left(\frac{L_z}{\Delta z}\right)}_{\text{Number of grid points}} \times \underbrace{\left(\frac{T_{interaction}}{\Delta t}\right)}_{\substack{\text{Number of timesteps} \\ \text{(i.e. number of iterations} \\ \text{of the PIC loop)}}}$$

Example: typical 3D laser-wakefield simulation
3D grid with 200 x 200 x 2500 grid points
140,000 timesteps (CFL limit!)
$\Longrightarrow$ **~60,000 hours on 1 core (7 years!)**

We need either **faster cores** or **more cores in parallel.**

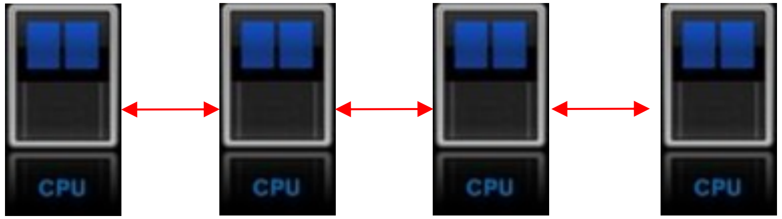
Single-core performance is saturating nowadays.

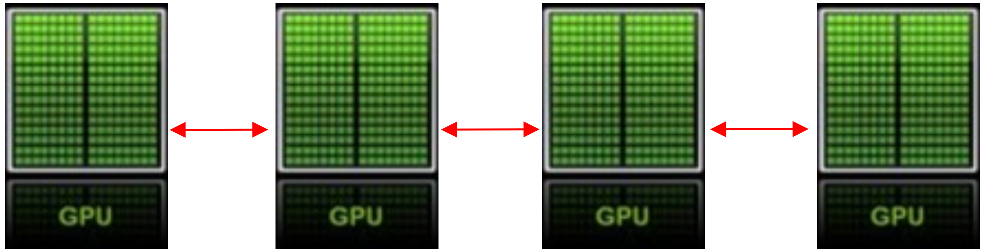


Nowadays, individual cores do not get faster.
We need to use **many cores in parallel.**

Modern supercomputers are organized in nodes, with multiples cores.

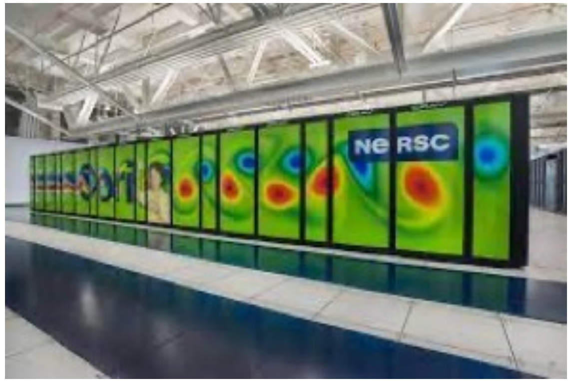
Computational work is parallelized across multiple **cores** and across multiples **nodes**.





”Fast” network (~10 Gb/s)

CPUs: tens of cores per node

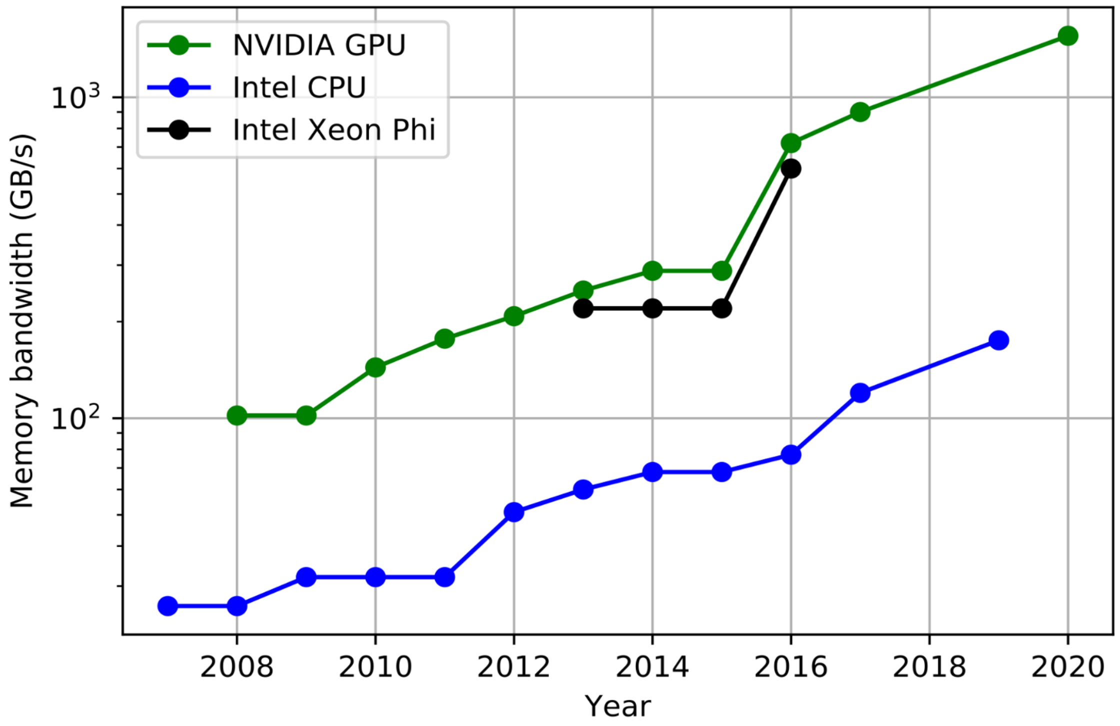


GPUs: thousands of (slow) cores per node

Advantages of GPUs: high memory bandwidth and energy efficiency

Memory bandwidth of different CPU/GPU models

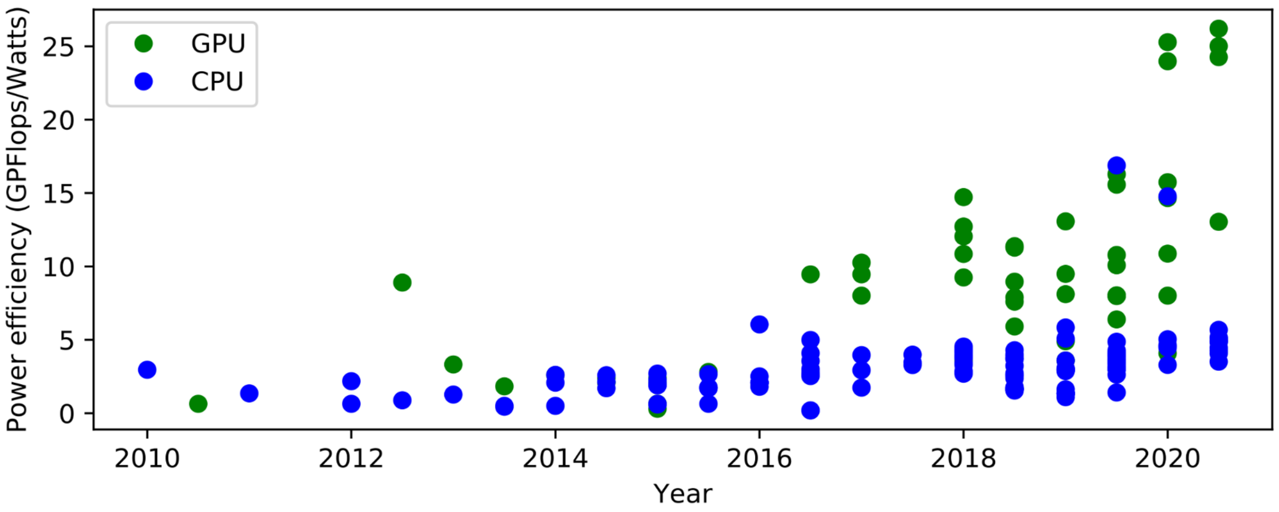
(= how fast can the compute cores fetch data in memory)



Data from github.com/karlrupp/cpu-gpu-mic-comparison

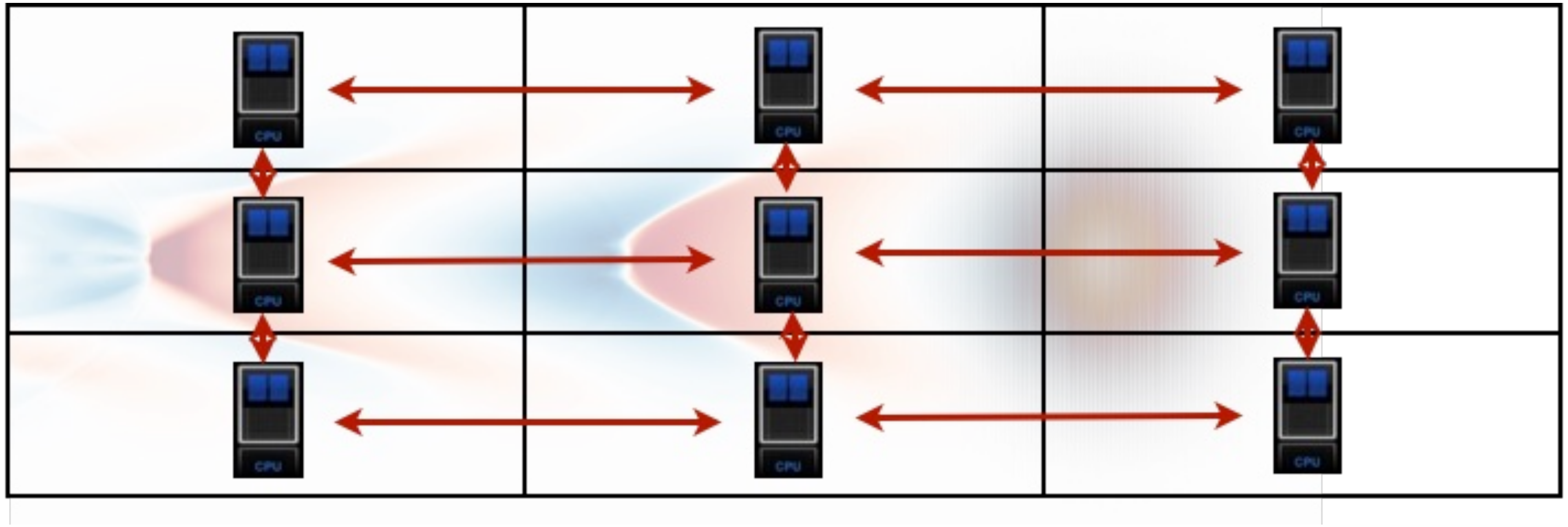
Power efficiency of different supercomputers

(= how much computation per unit of energy consumption)



Data from top500.org

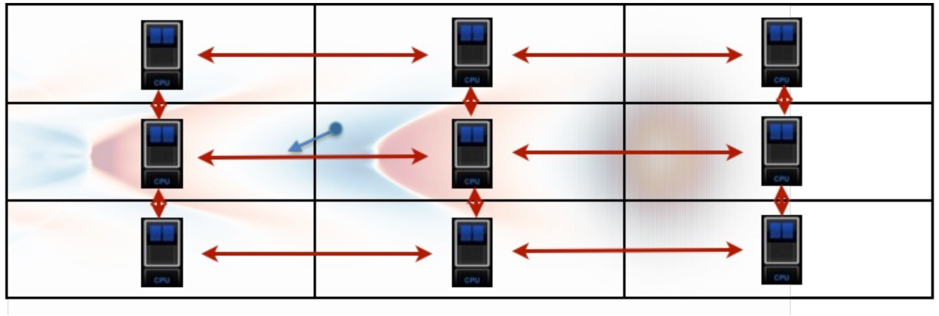
Domain-decomposition: groups of cores handle a fixed chunk of space

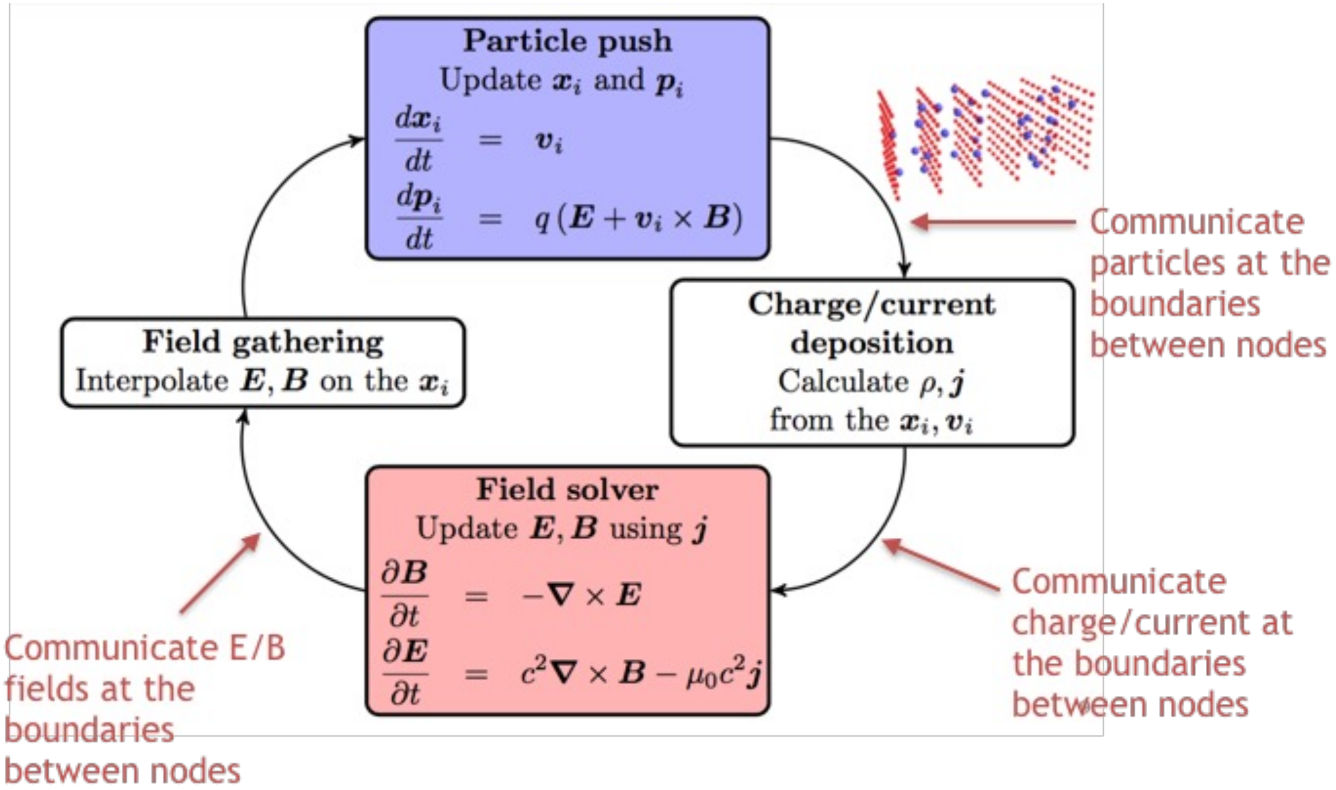


- Groups of cores (e.g. part of a CPU, one full GPU) work together on the same chunk of space (memory is shared among this group)

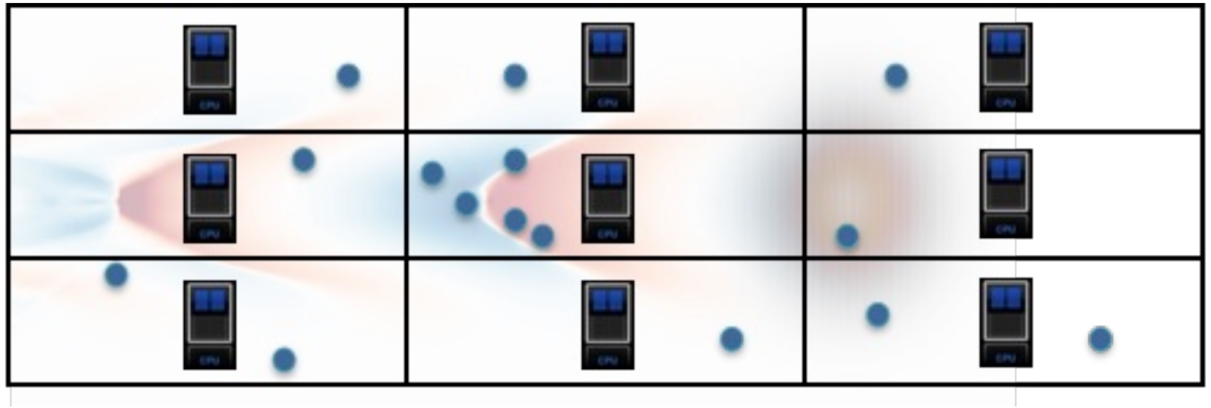
- Information is exchanged between groups of core over the network (using MPI)

Domain-decomposition: groups of cores handle a fixed chunk of space





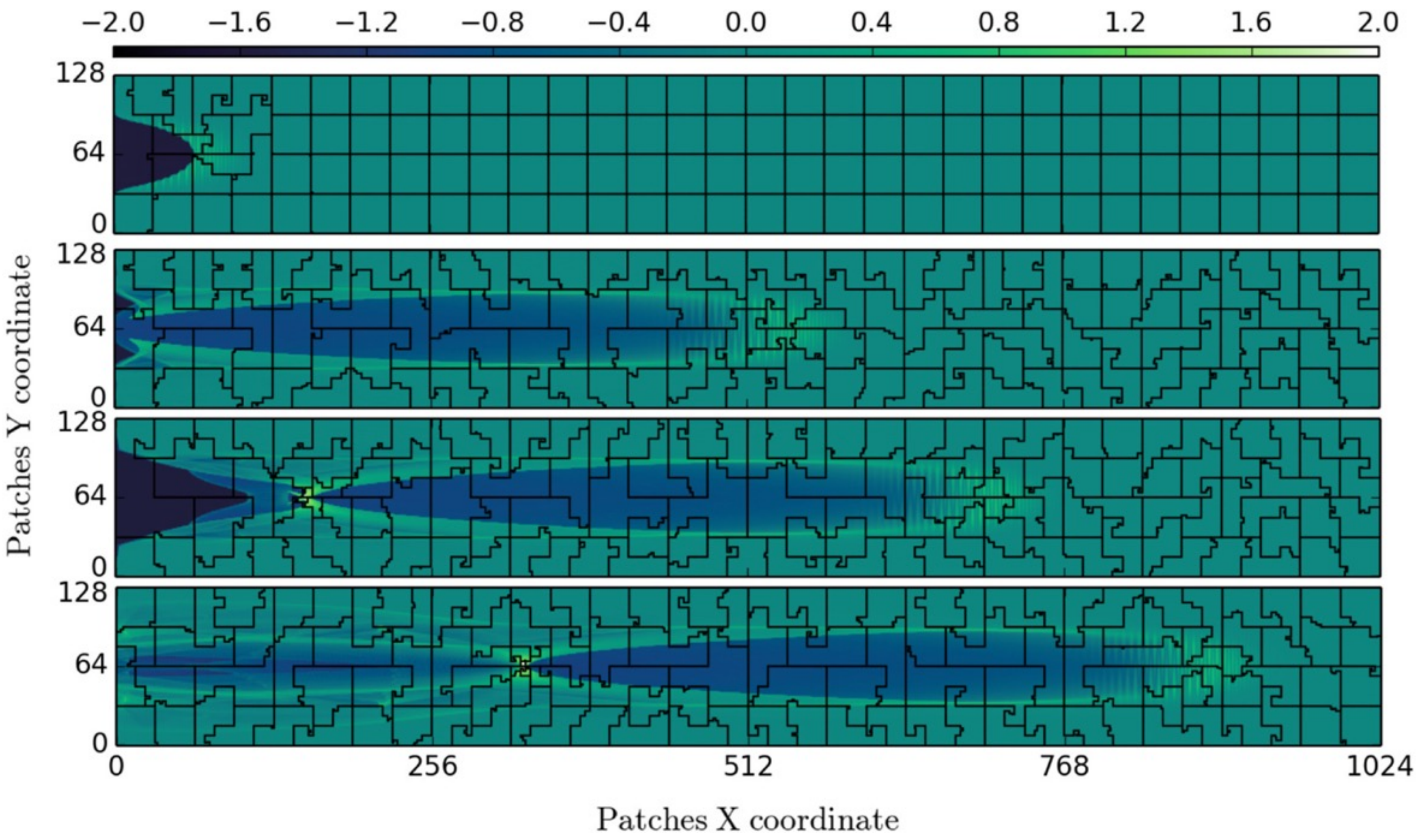
Load imbalance can significantly slow down the simulation.



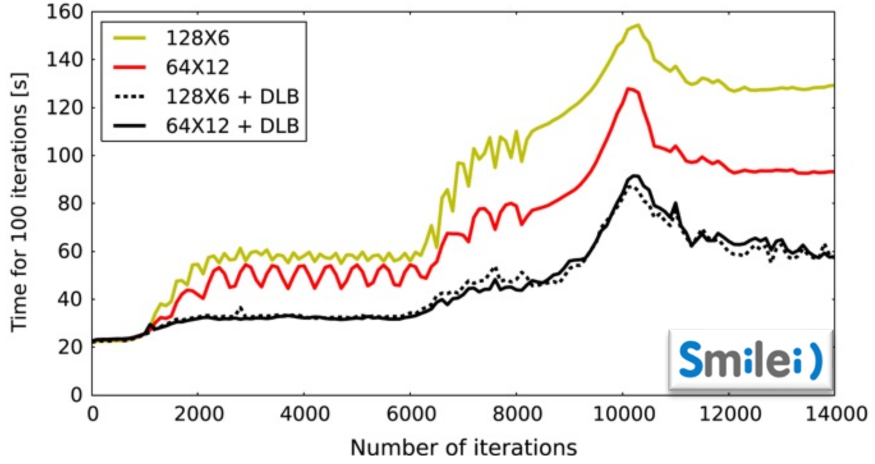
Load imbalance:
Some groups of cores have many more macroparticles (i.e. more work) than others.

The simulation will always progress at the pace of the slowest group.

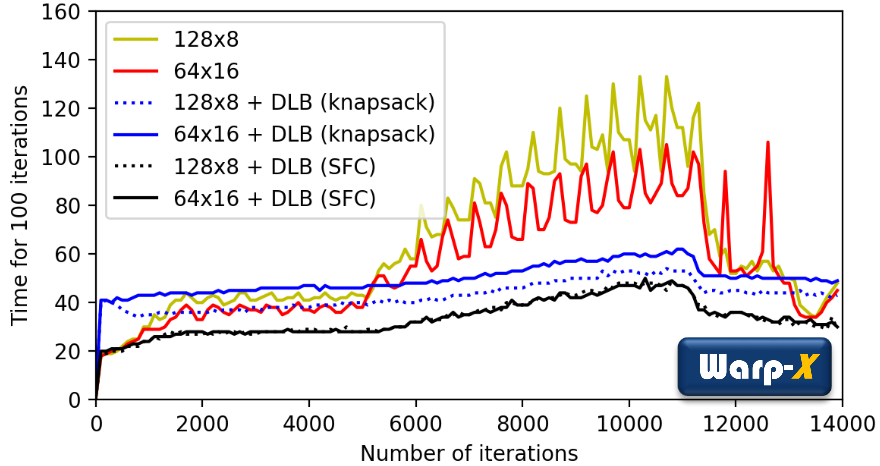
Dynamic load-balancing changes the shape of subdomains in order to even out the work load.



Simulation using Smilei[1]



Simulation using Warp-X[2]



[1]J. Derouillat *et al*, *Comput. Phys. Comm.* **222,** 351-373 (2018).

[2]J.-L. Vay *et al*, *Proc.* AAC 2018.

Conclusion

- **Full electromagnetic PIC codes** model the **Maxwell equations**, along with **the equations of motion** for plasma and beam particles

- **Very generic**, but the **computational cost** can be very high. (esp. for laser-driven acceleration)

- Therefore, simulations often **require massively parallel** computing architectures.

- Examples of common full electromagnetic PIC codes in our community:

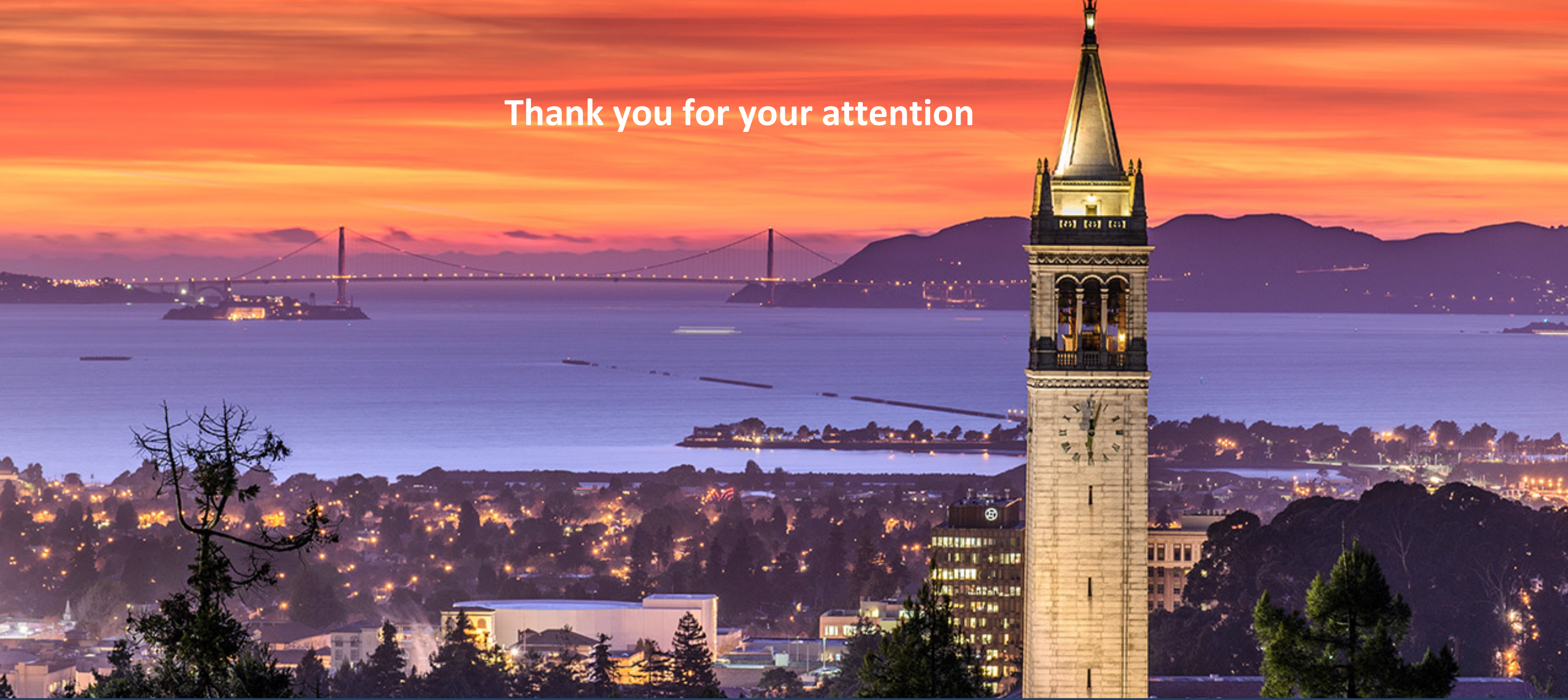
  EPOCH OSIRIS Smilei PIConGPU WarpX ….

Quick announcement: practical exercises

Monday, 6.2.	Tuesday, 7.2.	Wednesday, 8.2.	Thursday, 9.2.	Friday, 10.2.
Breakfast				
Laser-driven plasma acceleration I	Diagnostic techniques I	Laser-driven plasma acceleration II	Diagnostic techniques II	Plasma accelerator applications I
Discussion	Discussion	Discussion	Discussion	Discussion
Coffee & Tea				
Beam-driven plasma acceleration I	Modeling plasma accelerators I	Beam-driven plasma acceleration II	Modeling plasma accelerators II	Plasma accelerator applications II
Discussion	Discussion	Discussion	Discussion	Discussion
Lunch				
High-intensity lasers	Artificial intelligence and controlling acc.	Excursion	High-average power lasers	Departure
Discussion	Discussion		Discussion	
Coffee & Tea			Coffee & Tea	
Poster session I	Poster session II		Practical exercises	
Dinner				
	Special evening talk			

Please make sure you bring your **laptop**, and have an **active Google account**.

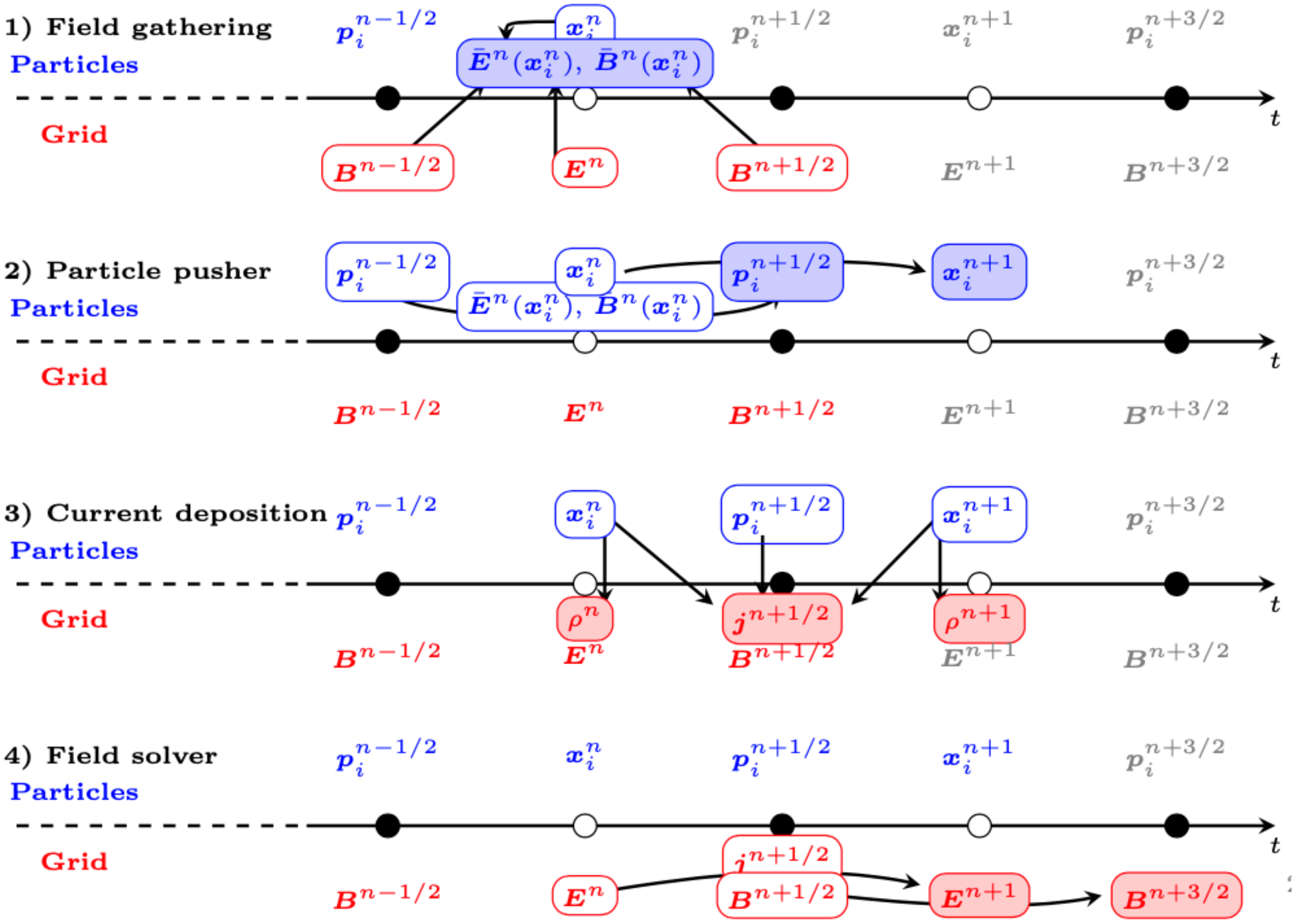
We will be using Google Colab.
(no need to install anything)



Thank you for your attention

There are multiple open post-doctoral positions at the BELLA Center (theory & experiments).
If interested, please visit jobs.lbl.gov and search for “BELLA”.

Staggering in time



Staggering in space