ELISS2023
ELI Summer School | 29 Aug – 1 Sep 2023
Dolní Břežany, Czech Republic

Electromagnetic Particle-In-Cell simulations

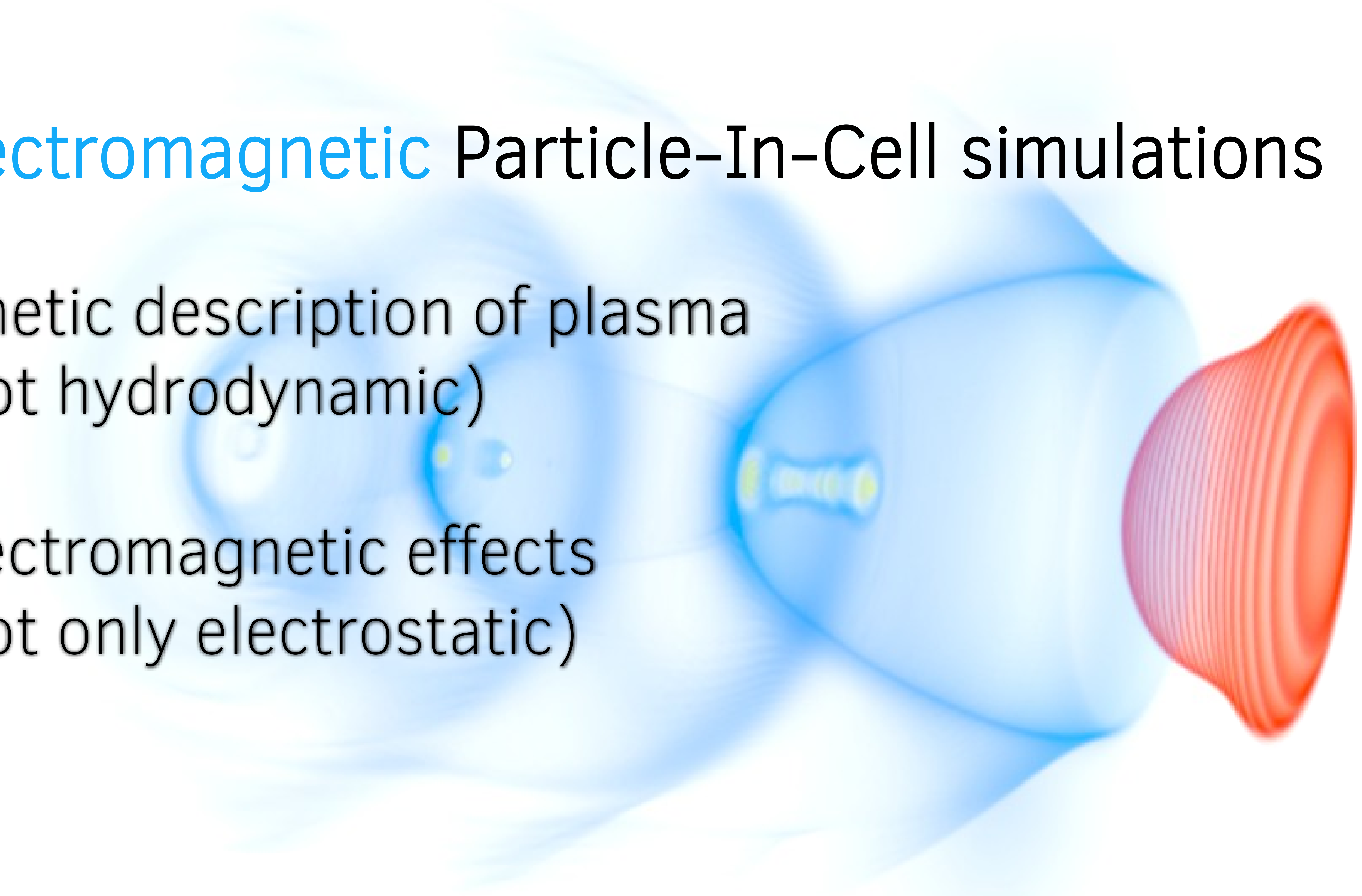
Frédéric Perez
LULI, CNRS (France)
frederic.perez@polytechnique.edu

August 31st, 2023
Dolní Břežany, Czech Republic

eli
IMPULSE

IMPULSE is funded by the European Union's Horizon 2020 programme under grant agreement No. 871161

Electromagnetic Particle-In-Cell simulations



Kinetic description of plasma
(not hydrodynamic)

Electromagnetic effects
(not only electrostatic)

An *Electromagnetic* PIC code:

- For which purpose?
- How does it work?
- Is it stable?
- What physics can it handle?
- On which computer can it work?

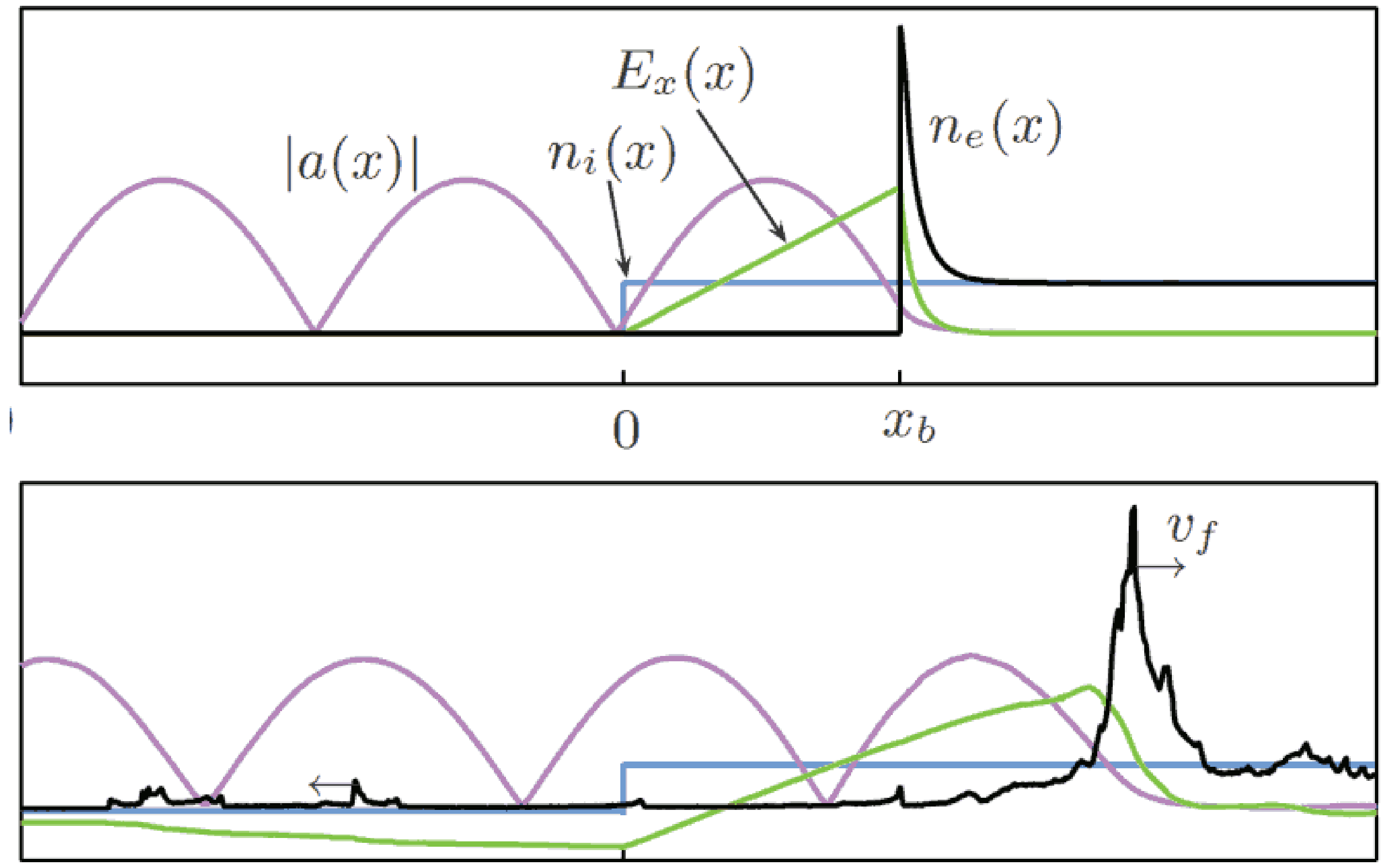
An *Electromagnetic* PIC code:

- **For which purpose?**
- How does it work?
- Is it stable?
- What physics can it handle?
- On which computer can it work?

PIC codes are an excellent support to theoretical modeling (even in 1D)

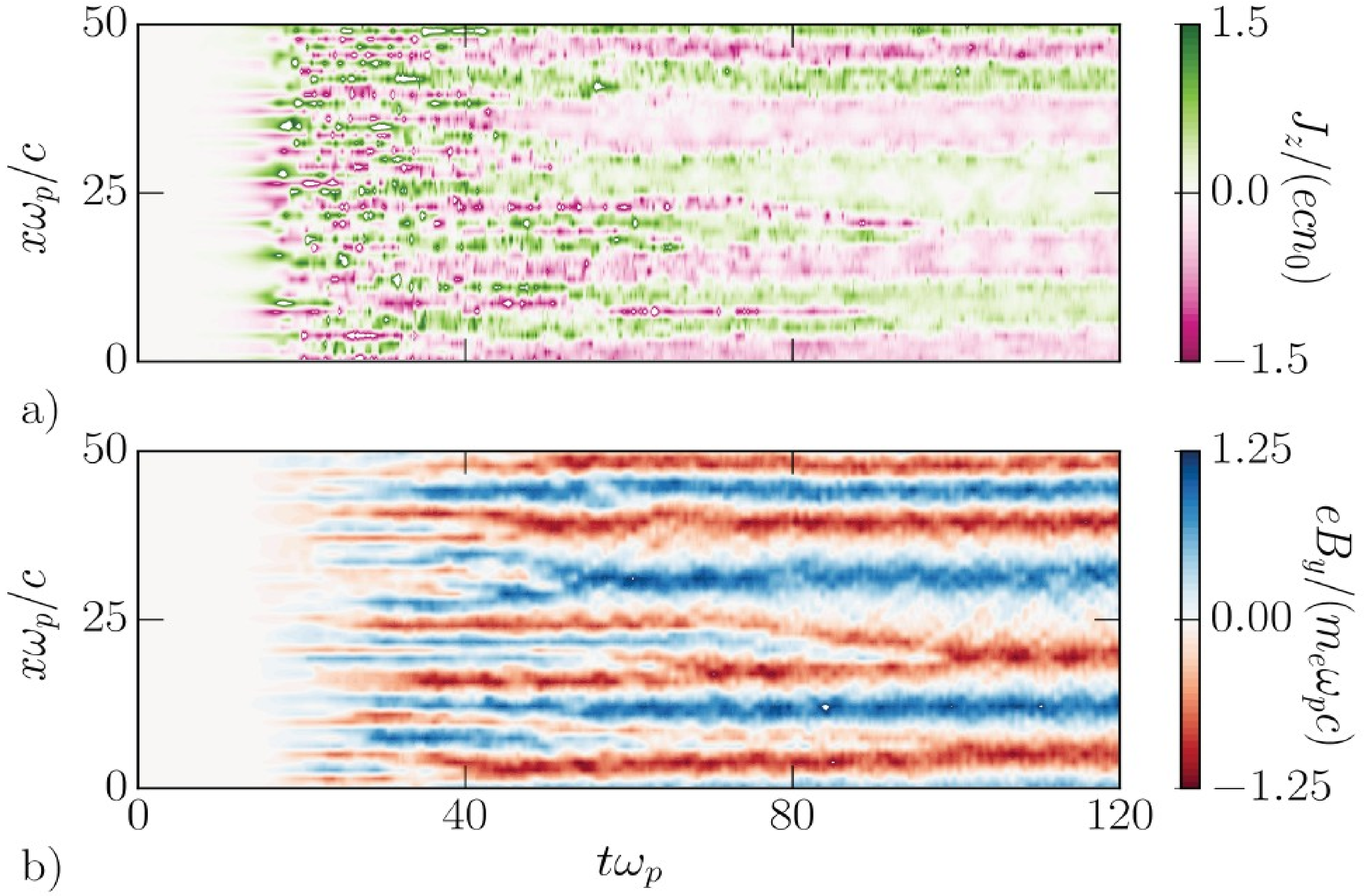
Relativistically-induced transparency

Siminos et al., PRE 86, 056404 (2012)



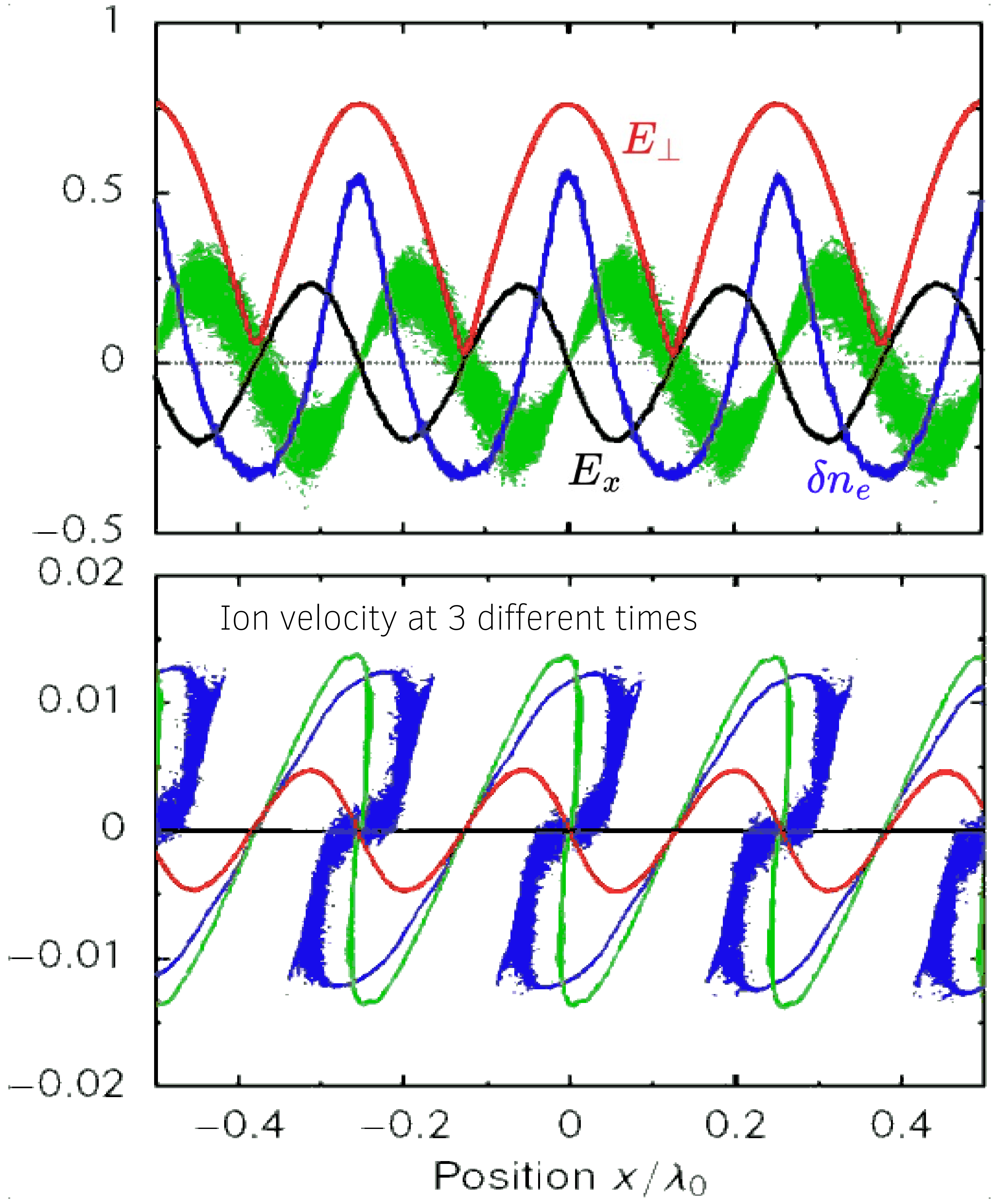
Weibel instability in the presence of an external magnetic field

Grassi et al., PRE 95, 023203 (2017)



Standing Whistler Waves

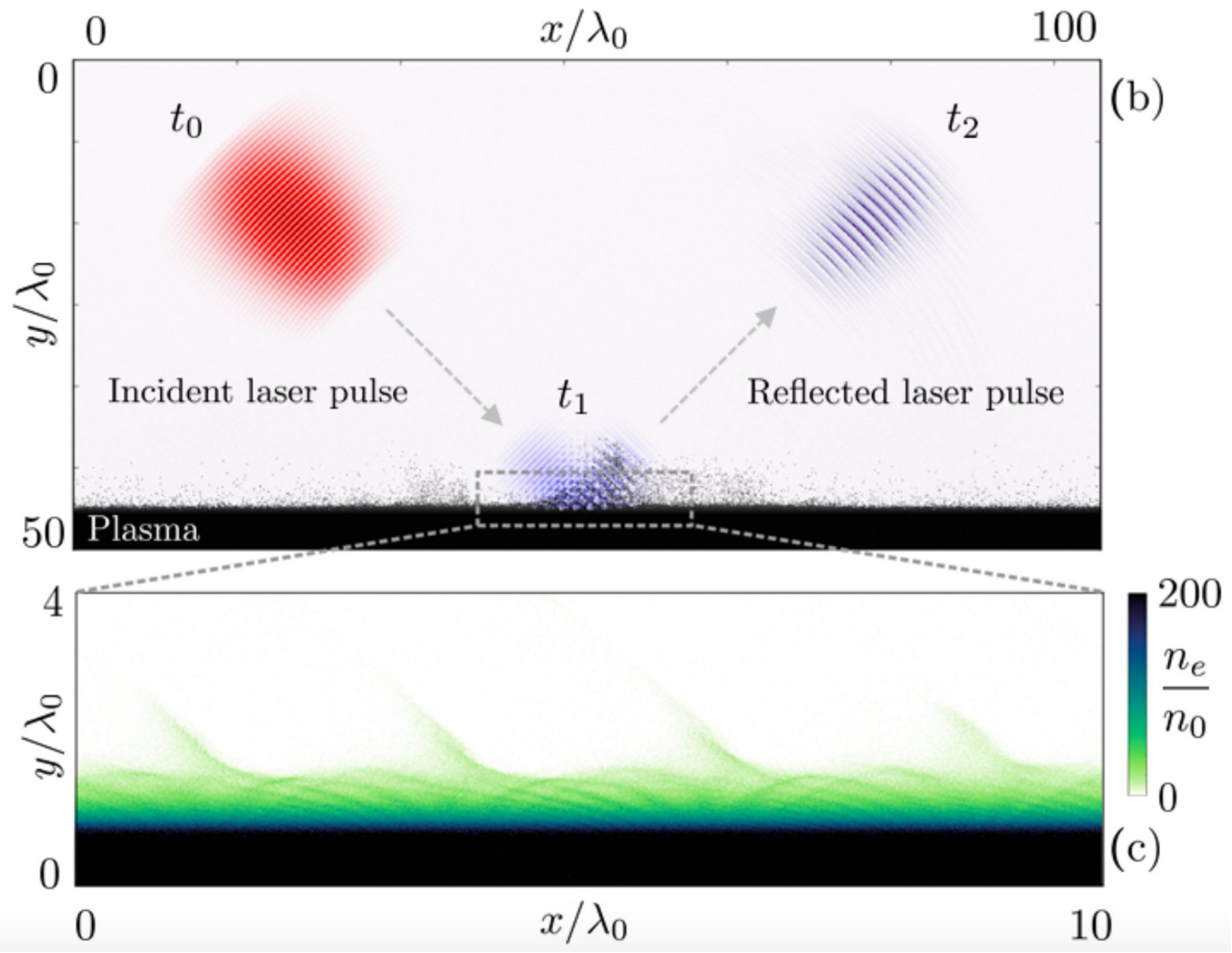
Sano et al., PRE 100, 053205 (2019)



2D or 3D simulations help design or understand experiments

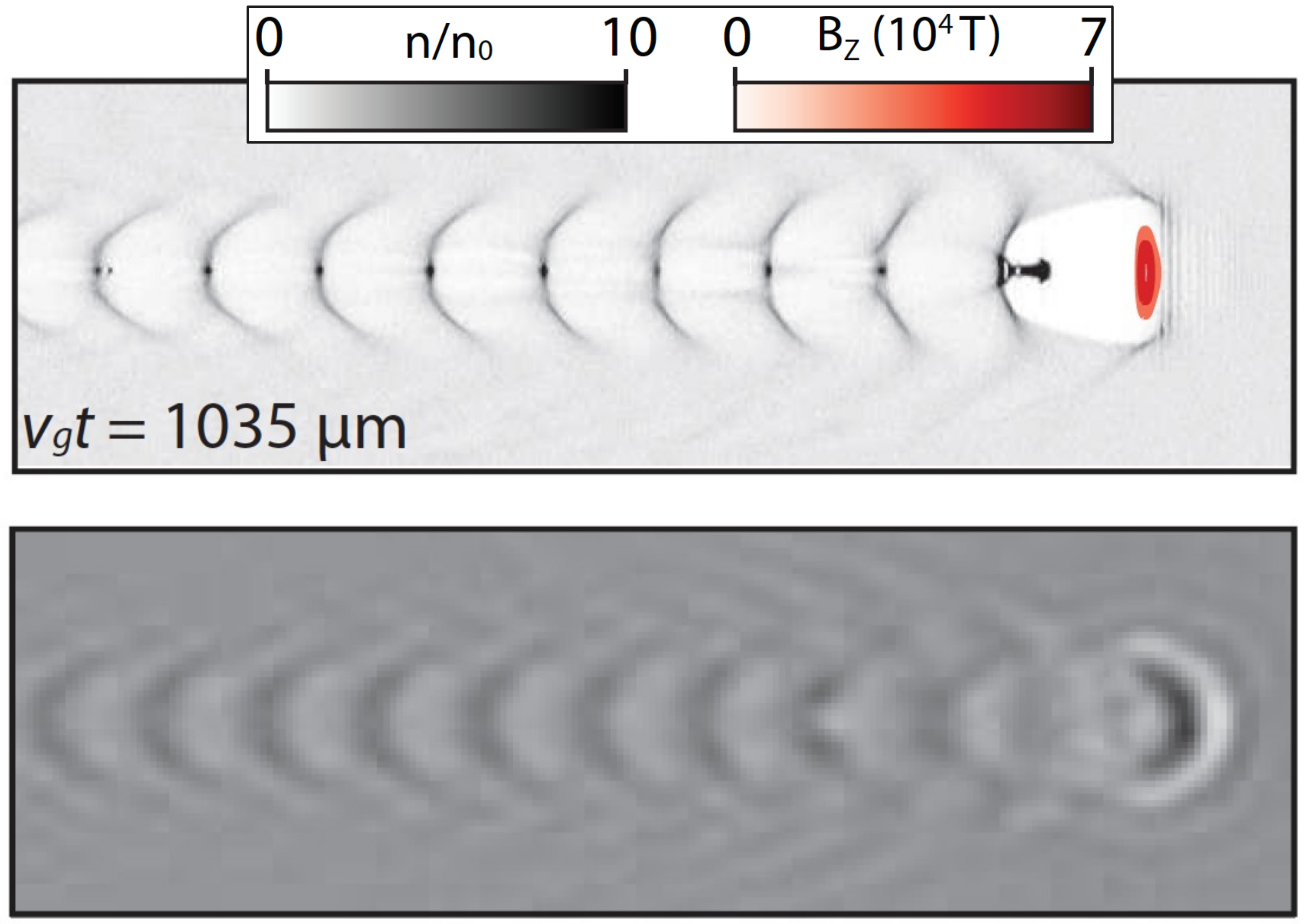
High-harmonic generation

G. Bouchard, F. Quéré, CEA/IRAMIS



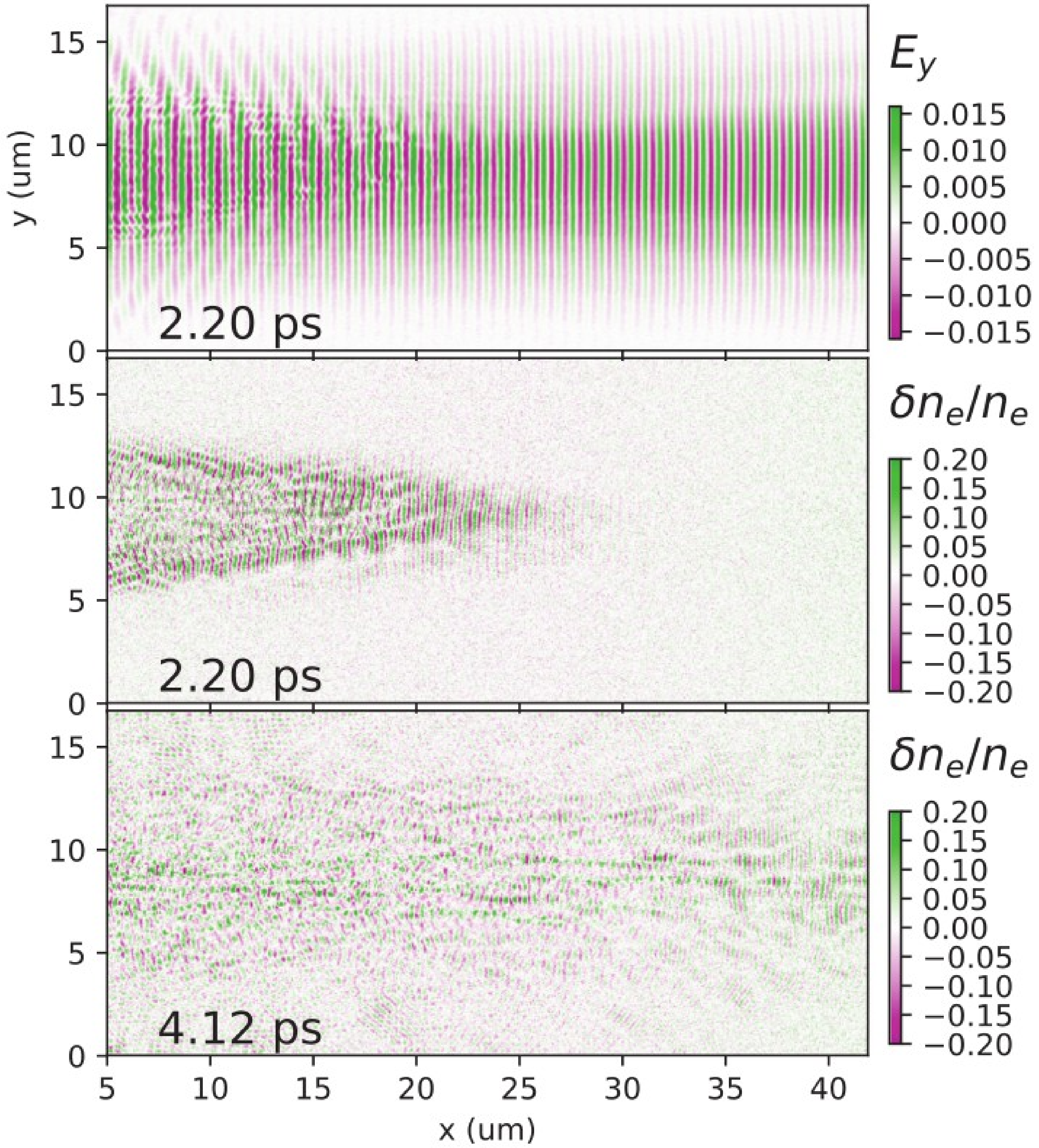
Laser wakefield acceleration

Sävert *et al.*,
PRL **115**, 055002 (2015)



Plasma instabilities induced by laser

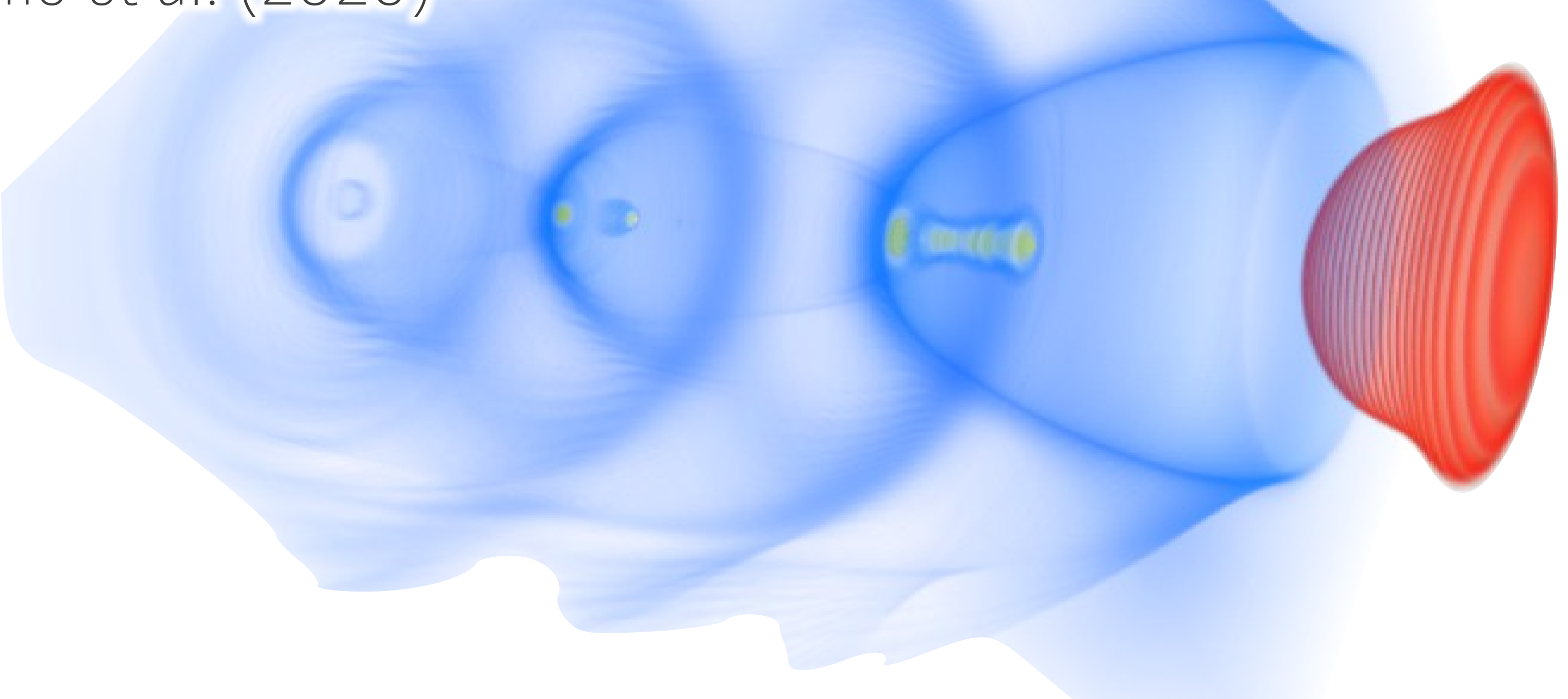
Perez *et al.* PoP 28, 043102 (2021)



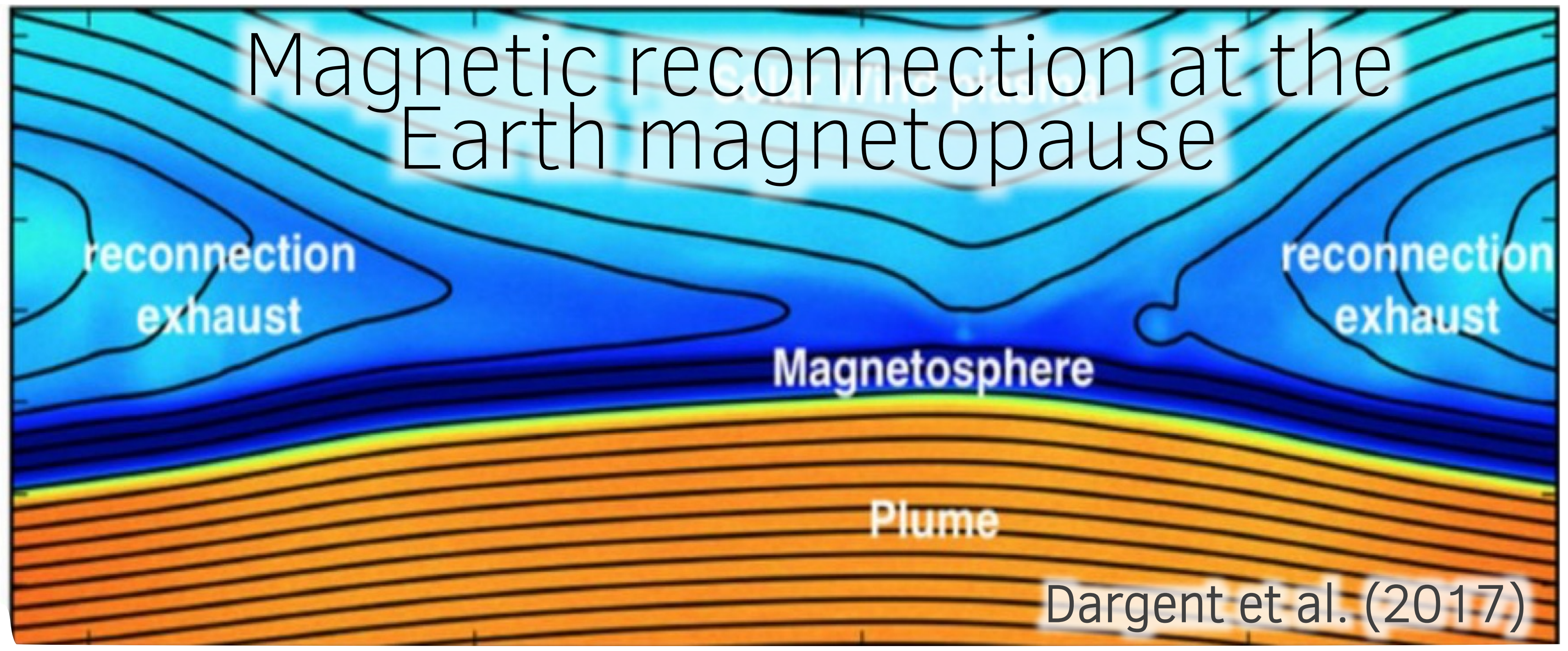
Overall, electromagnetic PIC codes are central to a wide range of plasma-physics-related studies

Laser wakefield acceleration

Massimo et al. (2020)



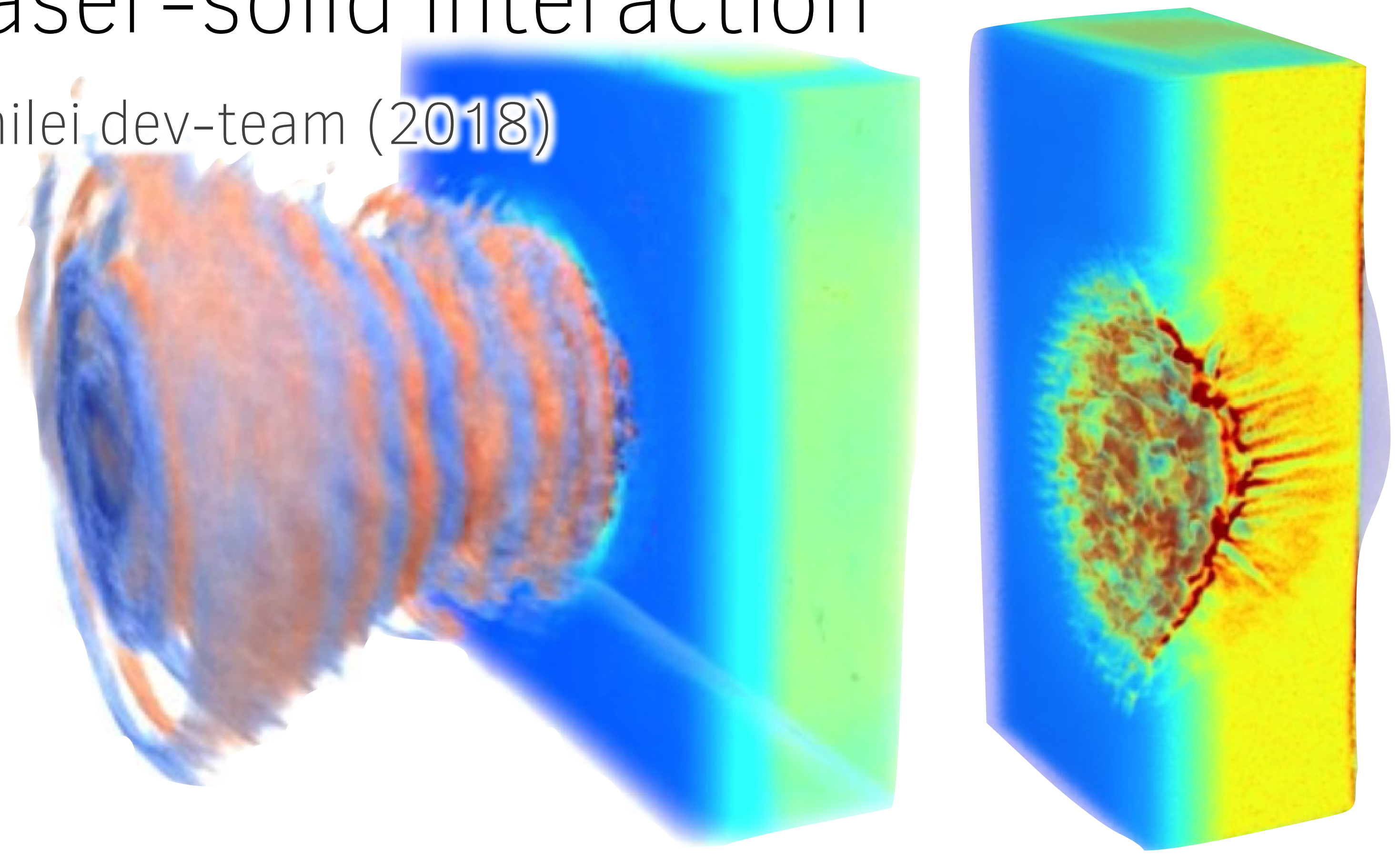
Magnetic reconnection at the Earth magnetopause



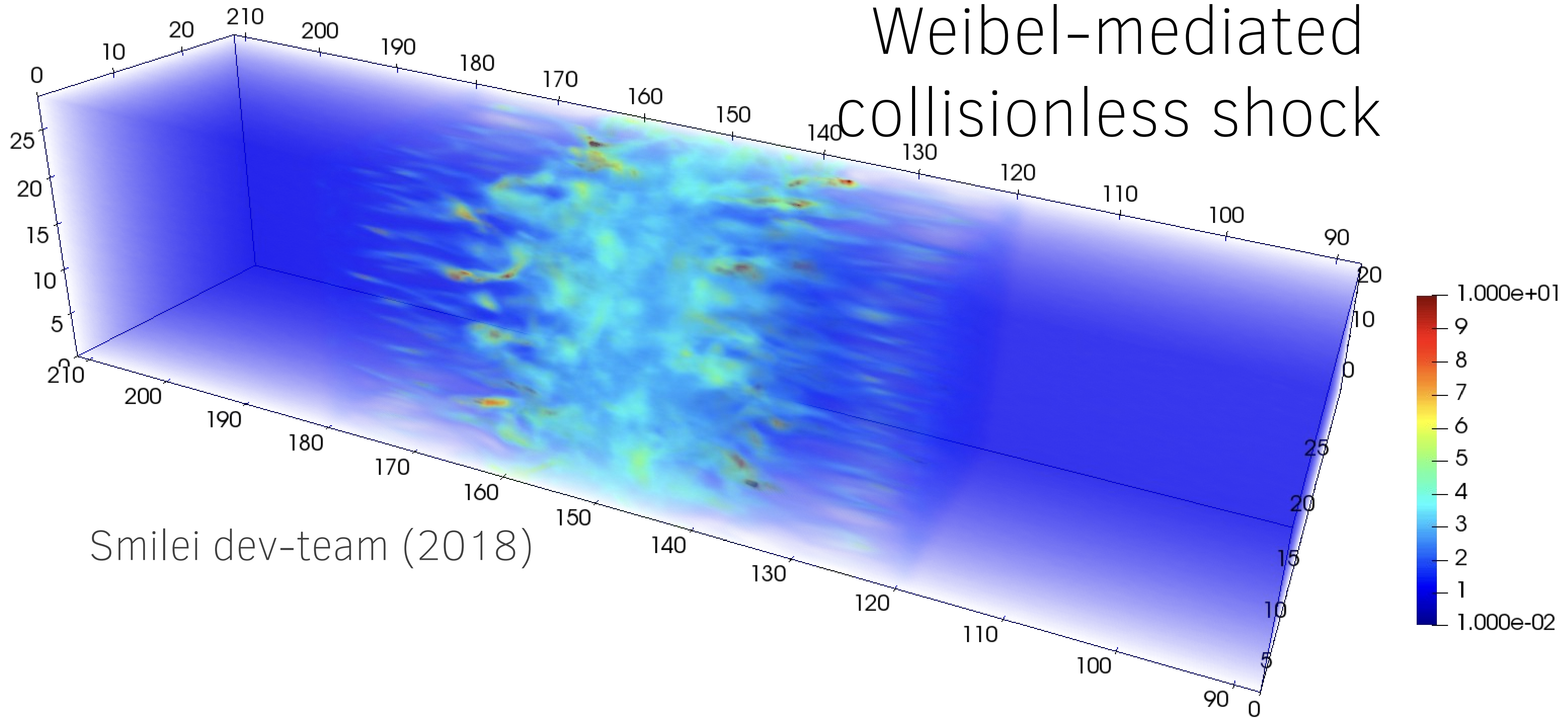
Dargent et al. (2017)

Laser-solid interaction

Smilei dev-team (2018)



Weibel-mediated collisionless shock



Smilei dev-team (2018)

+ plasma propulsion, cosmology, ...

An *Electromagnetic* PIC code:

- For which purpose?
- **How does it work?**
- Is it stable?
- What physics can it handle?
- On which computer can it work?

Quick reminder: special relativity

Momentum $$\mathbf{p} = \gamma\, m\, \mathbf{v}$$

Newton's 2nd law $$\frac{d\mathbf{p}}{dt} = \mathbf{F}$$

Lorentz factor $$\gamma = \frac{1}{\sqrt{1-(\mathbf{v}/c)^2}} = \sqrt{1+(\mathbf{p}/(mc))^2}$$

A kinetic description of plasma

$f_s(t, \mathbf{x}, \mathbf{p})$ Distribution function

species

density $\rho(t, \mathbf{x}) = \int d^3\mathbf{p}\ f_s(t, \mathbf{x}, \mathbf{p})$

current density $\mathbf{J}(t, \mathbf{x}) = q_s \int d^3\mathbf{p}\ \mathbf{v}\ f_s(t, \mathbf{x}, \mathbf{p})$

$$\partial_t f_s + \mathbf{v} \cdot \nabla f_s + \mathbf{F}_L \cdot \nabla_{\mathbf{p}} f_s = 0$$

Vlasov's equation

Average distribution

Average Lorentz force

Add Fields and Maxwell's equations

$\mathbf{E}(t, \mathbf{x})$ Electric field

$\mathbf{B}(t, \mathbf{x})$ Magnetic field

→ Lorentz force $\mathbf{F}_L = q_s (\mathbf{E} + \mathbf{v} \times \mathbf{B})$

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\epsilon_0} \qquad \partial_t \mathbf{E} = -\frac{1}{\epsilon_0}\mathbf{J} + c^2 \nabla \times \mathbf{B}$$

$$\nabla \cdot \mathbf{B} = 0 \qquad \partial_t \mathbf{B} = -\nabla \times \mathbf{E}$$

Maxwell's equations

Combine Vlasov + Maxwell

Particles (Vlasov)

$$\partial_t f_s + \mathbf{v}\cdot\nabla f_s + \mathbf{F}_L\cdot\nabla_{\mathbf{p}} f_s = 0$$

$$\mathbf{F}_L$$

The goal of an EM PIC code is to solve this Maxwell-Vlasov system of equations

$$\rho\ ,\ J$$

Fields (Maxwell)

$$\nabla\cdot\mathbf{E} = \frac{\rho}{\epsilon_0} \qquad \partial_t\mathbf{E} = -\frac{1}{\epsilon_0}\mathbf{J} + c^2\nabla\times\mathbf{B}$$

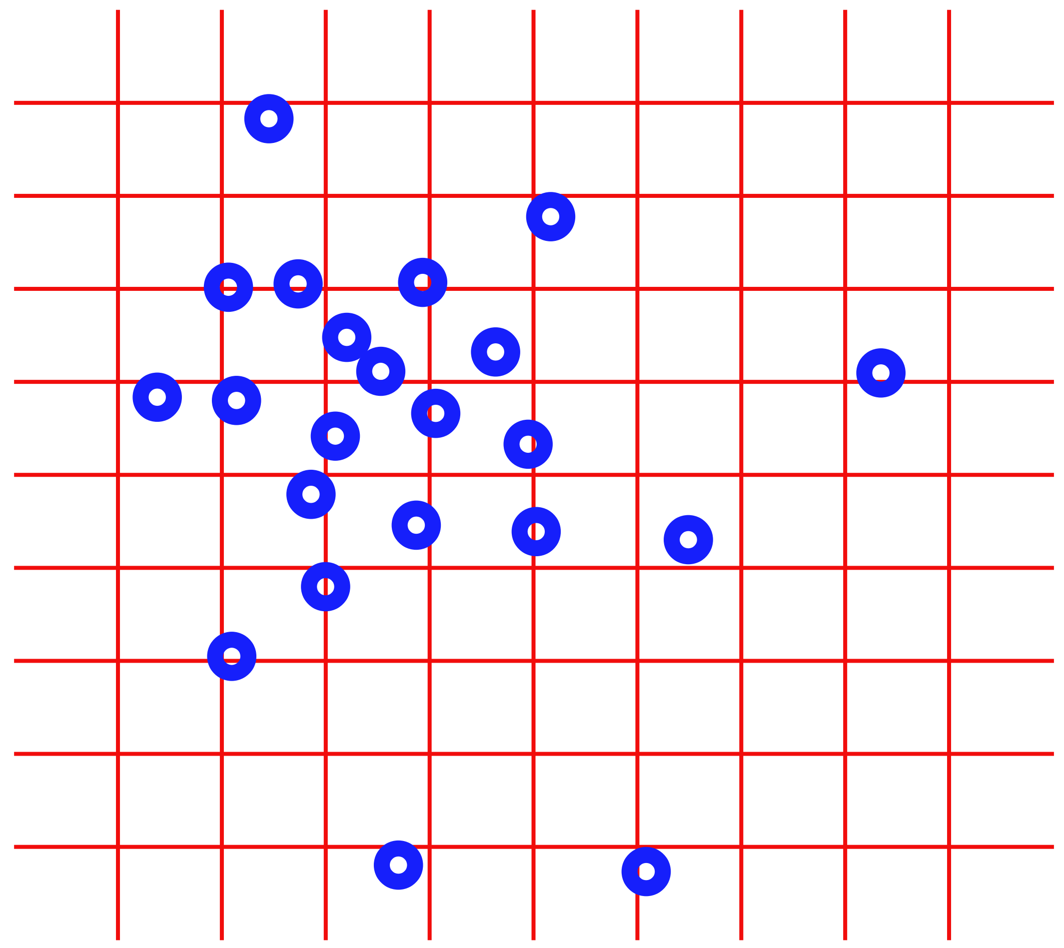
$$\nabla\cdot\mathbf{B} = 0 \qquad \partial_t\mathbf{B} = -\nabla\times\mathbf{E}$$

How to solve Vlasov + Maxwell numerically?

Particle-in-Cell

Vlasov — quasi-particles

Maxwell — discrete grid for fields



Distribution function = sum of quasi-particles

$$\partial_t f_s + \mathbf{v}\cdot\nabla f_s + \mathbf{F}_L\cdot\nabla_{\mathbf{p}} f_s = 0$$

Vlasov = partial differential equations in a 6D space.
Direct integration (*see Vlasov codes*) has tremendous computational cost!

In a PIC code, the distribution function is approximated as a sum over quasi-particles

$$f_s(t,\mathbf{x},\mathbf{p}) = \sum_{p=1}^{N} w_p\, S(\mathbf{x}-\mathbf{x}_p(t))\,\delta(\mathbf{p}-\mathbf{p}_p(t))$$

Statistical weight (w_p)

Shape function (S)

Now, inject this distribution in Vlasov's equation ...

PIC distribution + Vlasov equation = quasi-particle motion

Integrate over momentum → $\partial_t \mathbf{x}_p = \mathbf{v}_p$

Multiply by $\mathbf{p}$ then integrate over momentum and space → $\partial_t \mathbf{p}_p = q_s \mathbf{E}_p + q_s \mathbf{v}_p \times \mathbf{B}_p$

where
$$\begin{cases} \mathbf{E}_p = \int \mathbf{E}(\mathbf{x})\, S(\mathbf{x} - \mathbf{x}_p)\, d^3\mathbf{x} \\ \mathbf{B}_p = \int \mathbf{B}(\mathbf{x})\, S(\mathbf{x} - \mathbf{x}_p)\, d^3\mathbf{x} \end{cases}$$

Fields interpolated at the location of quasi-particles.

Note that the fields $\boldsymbol{E}$ and $\boldsymbol{B}$ exist only at grid points.

The movement of quasi-particles is essentially that of real particles

Replacing Vlasov by the quasi-particle motion & introducing field interpolation

The PIC loop

Pusher

$$\partial_t \mathbf{x}_p = \mathbf{v}_p$$

$$\partial_t \mathbf{p}_p = q_s\, \mathbf{E}_p + q_s\, \mathbf{v}_p \times \mathbf{B}_p$$

ρ , J

Fields (Maxwell)

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\epsilon_0} \qquad \partial_t \mathbf{E} = -\frac{1}{\epsilon_0}\mathbf{J} + c^2 \nabla \times \mathbf{B}$$

$$\nabla \cdot \mathbf{B} = 0 \qquad \partial_t \mathbf{B} = -\nabla \times \mathbf{E}$$

Field interpolation

$$\mathbf{E}_p = \int \mathbf{E}(\mathbf{x})\, S(\mathbf{x} - \mathbf{x}_p)\, d^3\mathbf{x}$$

$$\mathbf{B}_p = \int \mathbf{B}(\mathbf{x})\, S(\mathbf{x} - \mathbf{x}_p)\, d^3\mathbf{x}$$

Maxwell's equations can be simplified

Assume Maxwell-Ampere, Maxwell-Faraday and conservation of charge.

$$\partial_t \mathbf{E} = -\frac{1}{\epsilon_0}\mathbf{J} + c^2 \nabla \times \mathbf{B} \qquad \partial_t \mathbf{B} = -\nabla \times \mathbf{E} \qquad \partial_t \rho + \nabla \cdot \mathbf{J} = 0$$

Take the divergence

$$\partial_t \nabla \cdot \mathbf{E} + \nabla \cdot \mathbf{J} = 0$$
$$\partial_t (\nabla \cdot \mathbf{E} - \rho) = 0$$

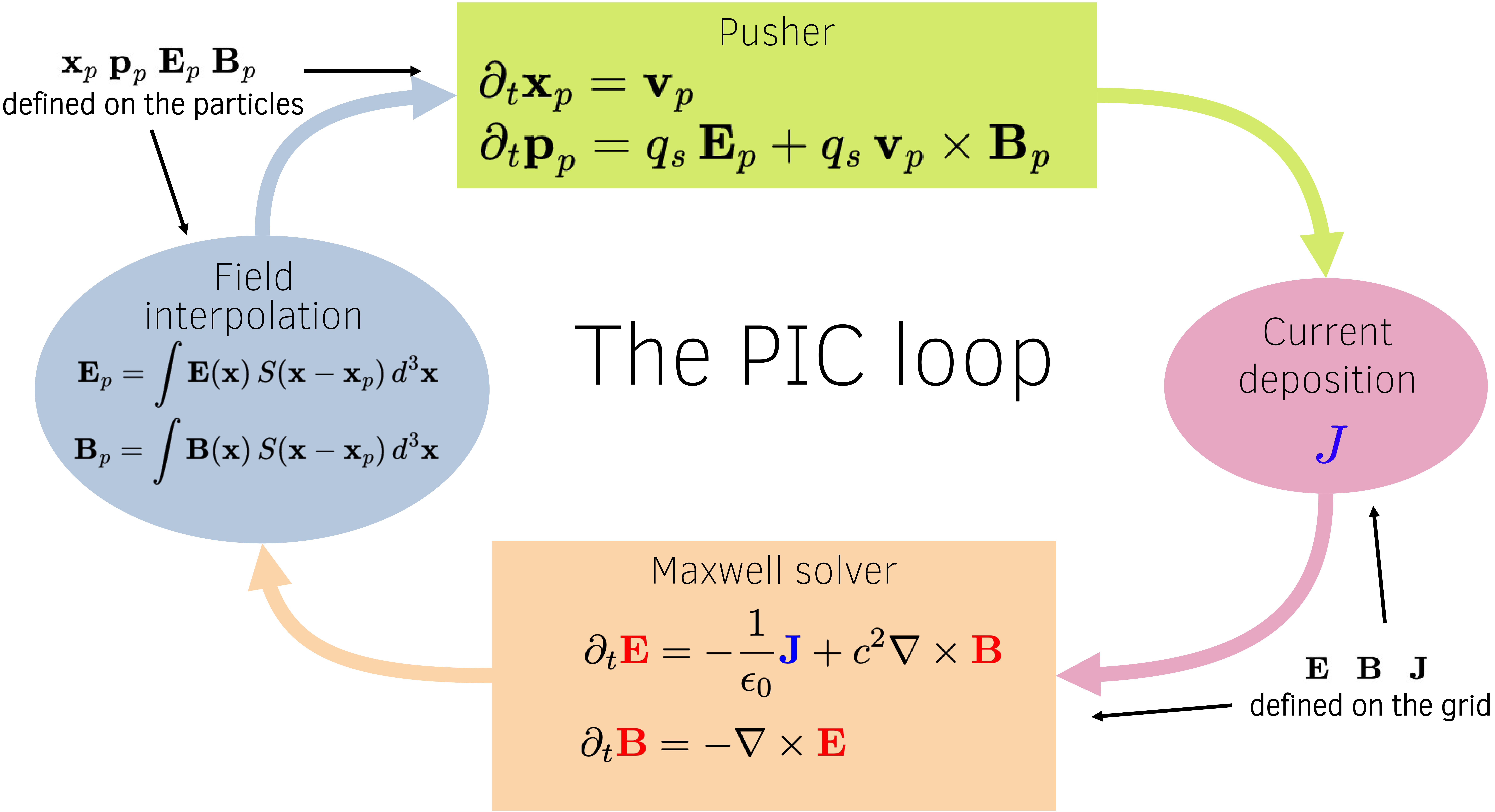
→ Maxwell-Poisson conserved

$$\partial_t (\nabla \cdot \mathbf{B}) = 0$$

→ Maxwell-Gauss conserved

If Maxwell-Gauss and Maxwell-Poisson are satisfied initially, it is not necessary to solve them.

(Then it is also unnecessary to calculate the density)



The PIC loop
Pusher
$\partial_t \mathbf{x}_p = \mathbf{v}_p$
$\partial_t \mathbf{p}_p = q_s\, \mathbf{E}_p + q_s\, \mathbf{v}_p \times \mathbf{B}_p$
Current deposition
J
Maxwell solver
$\partial_t \mathbf{E} = -\frac{1}{\epsilon_0}\mathbf{J} + c^2 \nabla \times \mathbf{B}$
$\partial_t \mathbf{B} = -\nabla \times \mathbf{E}$
Field interpolation
$\mathbf{E}_p = \int \mathbf{E}(\mathbf{x})\, S(\mathbf{x} - \mathbf{x}_p)\, d^3\mathbf{x}$
$\mathbf{B}_p = \int \mathbf{B}(\mathbf{x})\, S(\mathbf{x} - \mathbf{x}_p)\, d^3\mathbf{x}$
$\mathbf{x}_p$ $\mathbf{p}_p$ $\mathbf{E}_p$ $\mathbf{B}_p$
defined on the particles
$\mathbf{E}$ $\mathbf{B}$ $\mathbf{J}$
defined on the grid

In practice, many other aspects must be considered

- System of units
- Shape function $\boldsymbol{S}$
- Geometry of the grid
- Ensure conservation of charge
- Boundary conditions
- Initial conditions
- …

The natural units system

Velocity	c
Charge	e
Mass	m_e
Momentum	$m_e\, c$
Energy, Temperature	$m_e c^2$
Time	ω_r^{-1}
Length	c/ω_r
Number density	$n_r = \epsilon_0\, m_e\, \omega_r^2/e^2$
Current density	$e\, c\, n_r$
Pressure	$m_e\, c^2\, n_r$
Electric field	$m_e\, c\, \omega_r/e$
Magnetic field	$m_e\, \omega_r/e$
Poynting flux	$m_e\, c^3\, n_r/2$

ω_r is the reference angular frequency. It does not need to be defined in the code. Results of the simulation can be scaled *a posteriori* by changing the value of ω_r

Fields (Maxwell)

$$\nabla \cdot \mathbf{E} = \rho \qquad \partial_t \mathbf{E} = -\mathbf{J} + \nabla \times \mathbf{B}$$

$$\nabla \cdot \mathbf{B} = 0 \qquad \partial_t \mathbf{B} = -\nabla \times \mathbf{E}$$

Maxwell's equations are simplified when all quantities are normalized with the natural units.

Choose the shape function $\boldsymbol{S}$

$$\hat{s}^{(0)}(x) = \Delta x\,\delta(x),$$

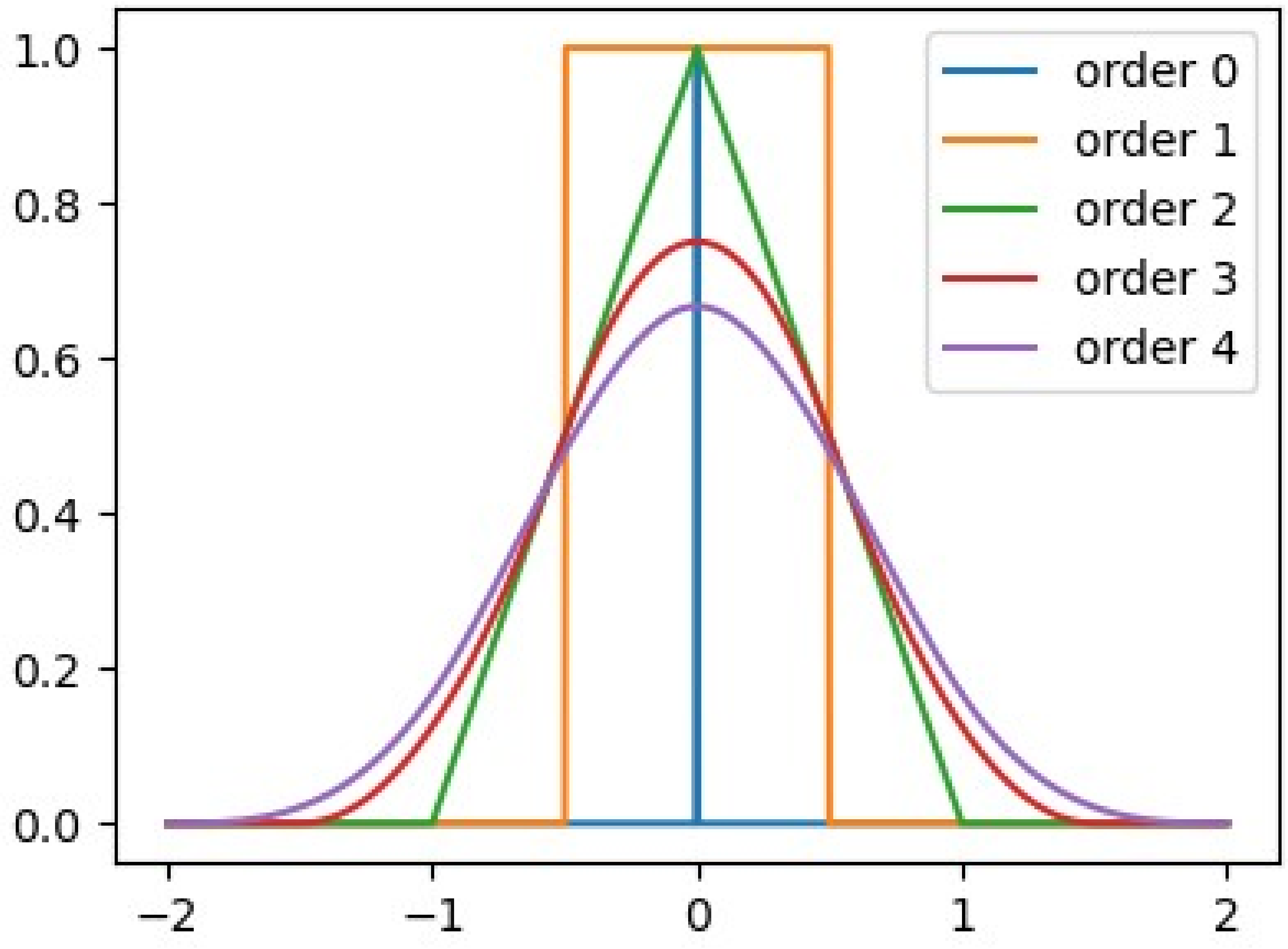
$$\hat{s}^{(1)}(x) = \begin{cases} 1 \text{ if } |x| \leq \frac{1}{2}\Delta x, \\ 0 \text{ otherwise}, \end{cases}$$

$$\hat{s}^{(2)}(x) = \begin{cases} \left(1 - \left|\frac{x}{\Delta x}\right|\right) & \text{if } |x| \leq \Delta x, \\ 0 & \text{otherwise}, \end{cases}$$

$$\hat{s}^{(3)}(x) = \begin{cases} \frac{3}{4}\left[1 - \frac{4}{3}\left(\frac{x}{\Delta x}\right)^2\right] & \text{if } |x| \leq \frac{1}{2}\Delta x, \\ \frac{9}{8}\left(1 - \frac{2}{3}\left|\frac{x}{\Delta x}\right|\right)^2 & \text{if } \frac{1}{2}\Delta x < |x| \leq \frac{3}{2}\Delta x, \\ 0 & \text{otherwise}, \end{cases}$$

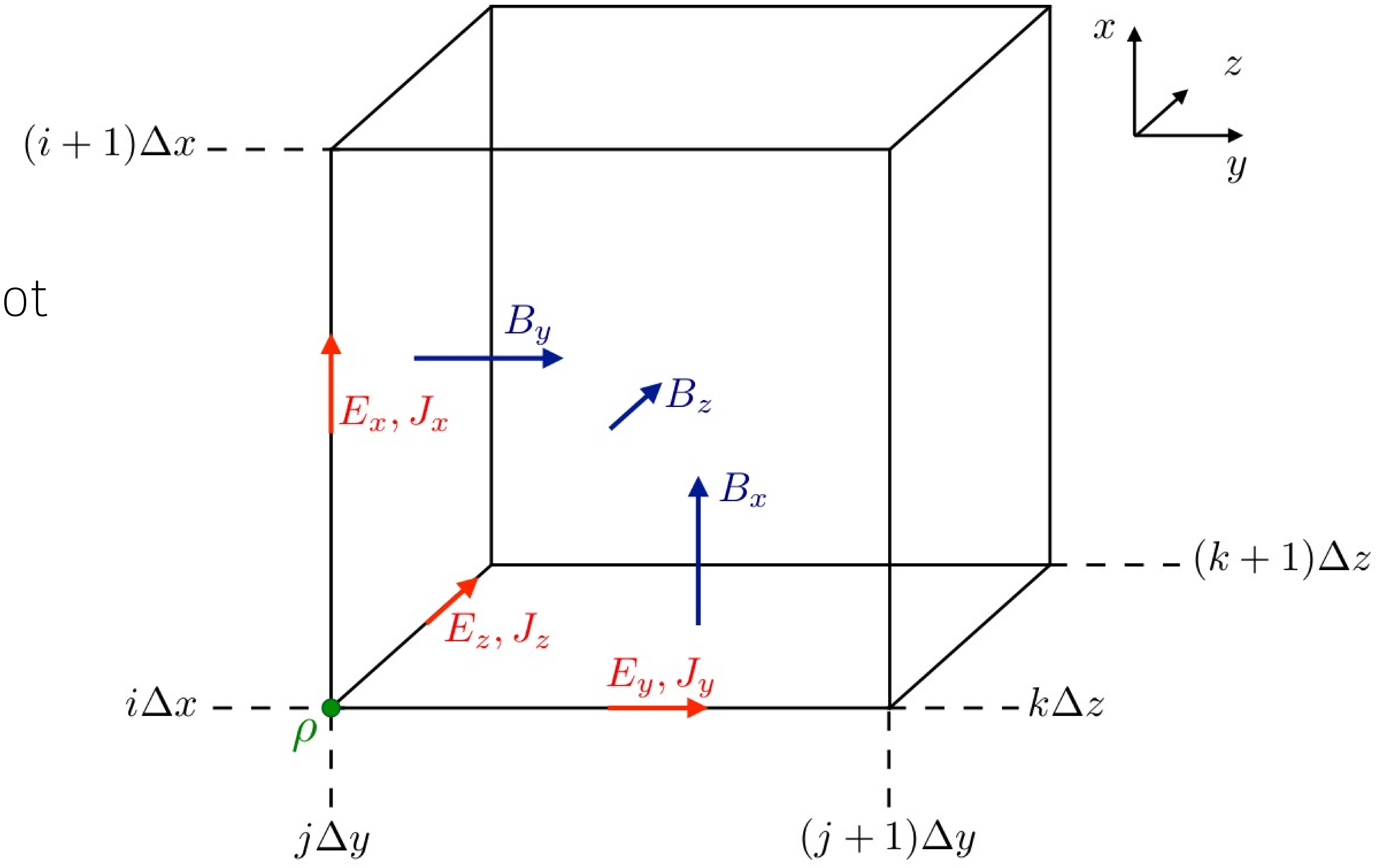
$$\hat{s}^{(4)}(x) = \begin{cases} \frac{2}{3}\left[1 - \frac{3}{2}\left(\frac{x}{\Delta x}\right)^2 + \frac{3}{4}\left|\frac{x}{\Delta x}\right|^3\right] & \text{if } |x| \leq \Delta x, \\ \frac{4}{3}\left(1 - \frac{1}{2}\left|\frac{x}{\Delta x}\right|\right)^3 & \text{if } \Delta x < |x| \leq 2\,\Delta x, \\ 0 & \text{otherwise}. \end{cases}$$

The shape function may be of different orders that increase the smoothing of field interpolation.
A higher order reduces noise, but longer to compute.



Finite-difference time-domain (FDTD) (or Yee's grid)
A popular method to solve Maxwell

Fields and currents are not defined on the nodes.



The PIC algorithms may also take many different forms

- Cylindrical or spherical geometry
- Various models for pusher (ex: implicit)
- Various models for solver (ex: spectral, envelope)
- …

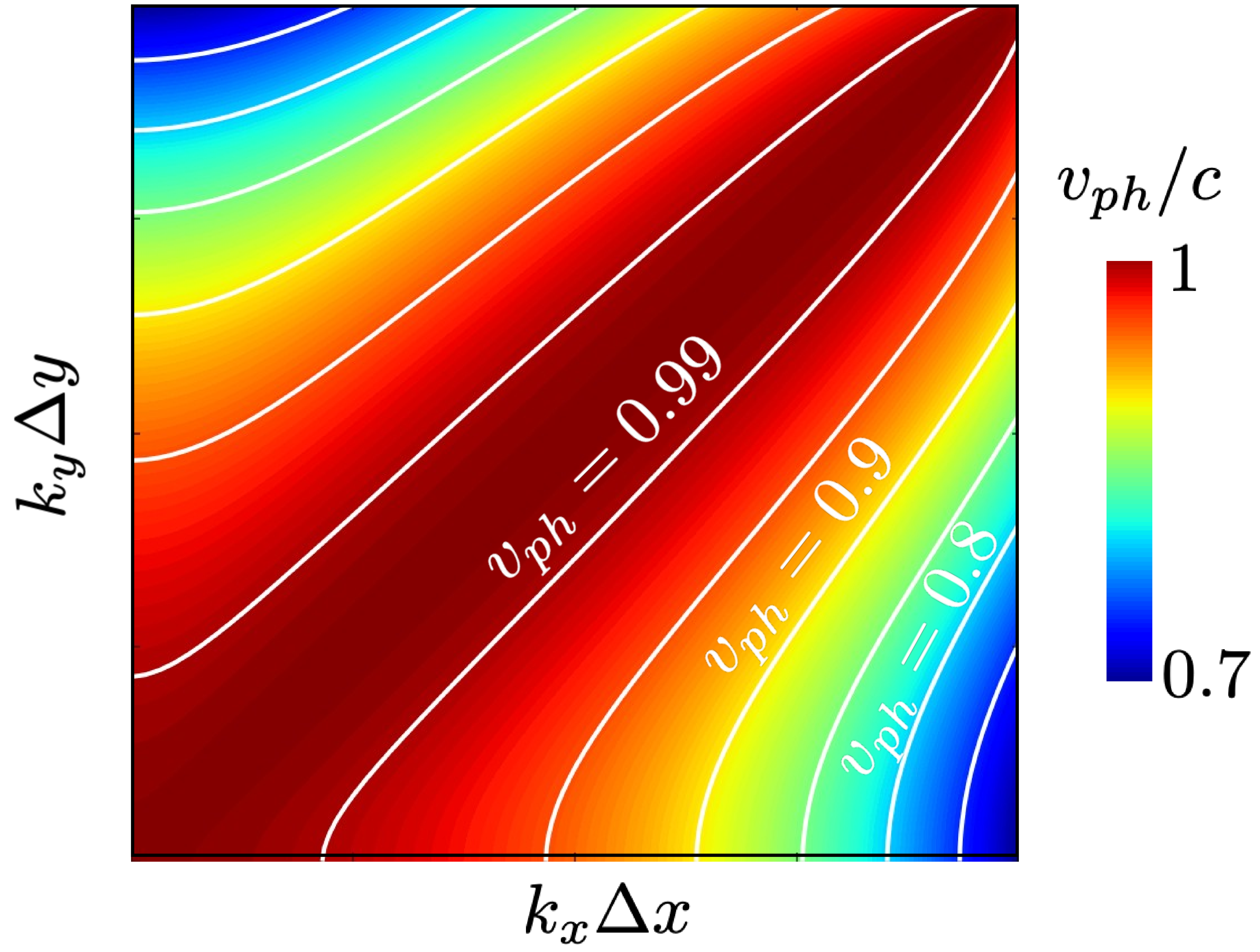
An *Electromagnetic* PIC code:

- For which purpose?
- How does it work?
- **Is it stable?**
- What physics can it handle?
- On which computer can it work?

Numerical analysis of the FDTD solvers shows some of their limitations

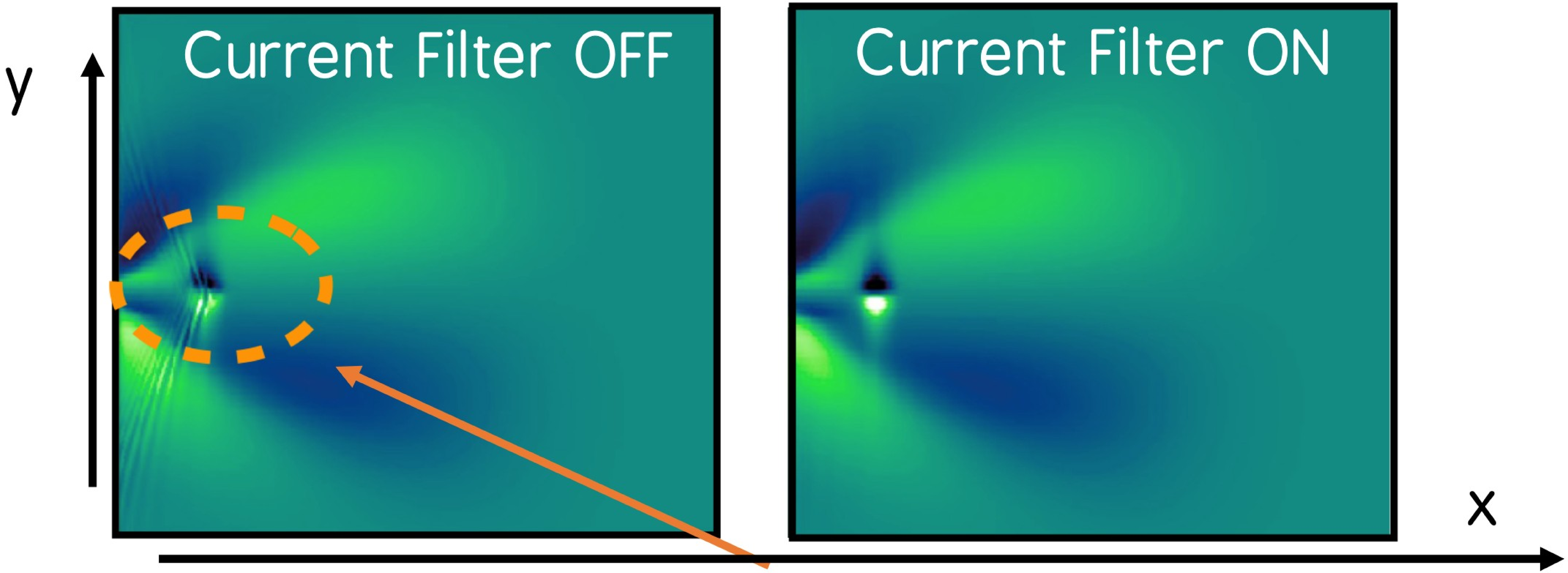
Write Maxwell's equations on a grid, and search for wave-like solutions

→ numerical dispersion relation $\Delta t^{-2}\ \sin^2(\omega\Delta t/2) = \sum_{a=x,y,z} \Delta a^{-2}\ \sin^2(k_a\Delta a/2)$



The numerical vacuum is dispersive and anisotropic!

Dispersive vacuum → numerical Cherenkov



Filtering can reduce Numerical Cherenkov Radiation

The time step cannot be too large

From the dispersion relation, we can calculate the phase velocity.

Stability requires $v_{ph}\Delta t < \textbf{cell size}$

(i.e. light waves cannot cross 2 cells in 1 time step)

$$\rightarrow \quad \Delta t^{-2} > \sum_{a=x,y,z} \Delta a^{-2}$$

Courant-Friedrich-Levy (CFL) condition

The cell size cannot be too large either

Depending on the situation you may need to resolve:

- ✓ The Debye length (or the simulation will have numerical heating)
- ✓ The laser wavelength (or it won't propagate)
- ✓ The skin depth

Often, a PIC simulation won't crash when the results are meaningless.
Users must understand the limitations and *test*.

An *Electromagnetic* PIC code:

- For which purpose?
- How does it work?
- Is it stable?
- **What physics can it handle?**
- On which computer can it work?

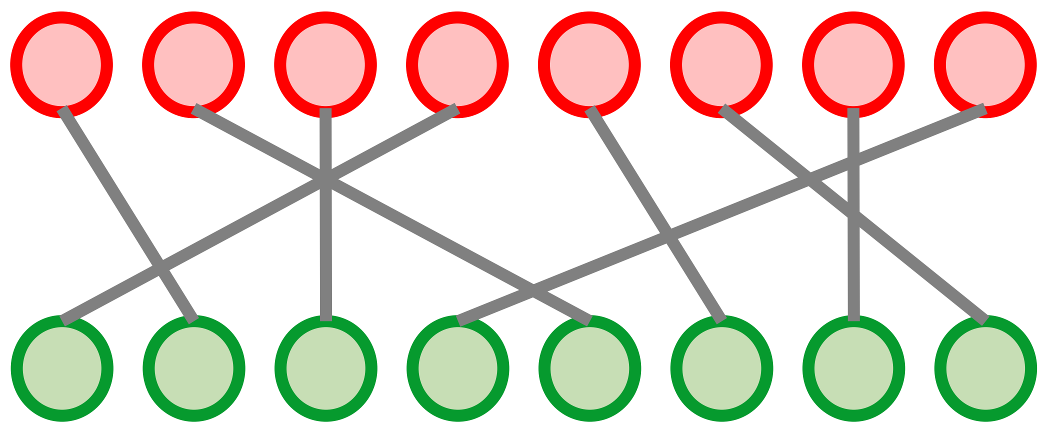
What physics are included? Or not?

PIC has ...

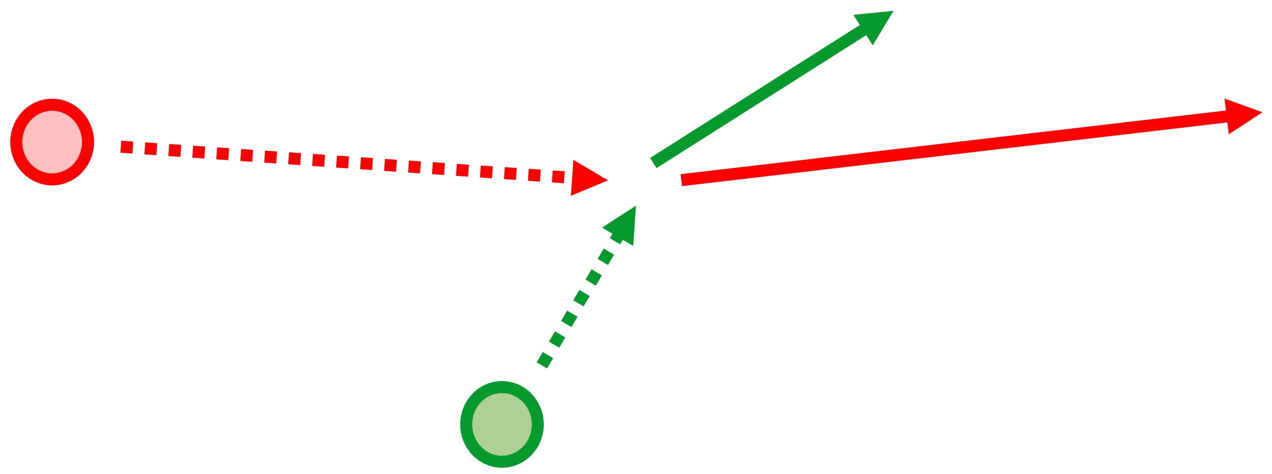
Vlasov	→	No interaction between individual particles
Maxwell	→	No quantum mechanics
Fields on a grid	→	No high-frequency photons
Particles with fixed charge	→	No atomic physics
Particles with fixed mass	→	No nuclear physics

Collisions can be introduced with a Monte-Carlo method (other methods exist)

- Instead of calculating $n!$ interactions, quasi-particles are associated 2-by-2 randomly



- The collision rate is computed for each pair
  Usually: small-angle Rutherford cross-section.
- A random deflection is computed accordingly

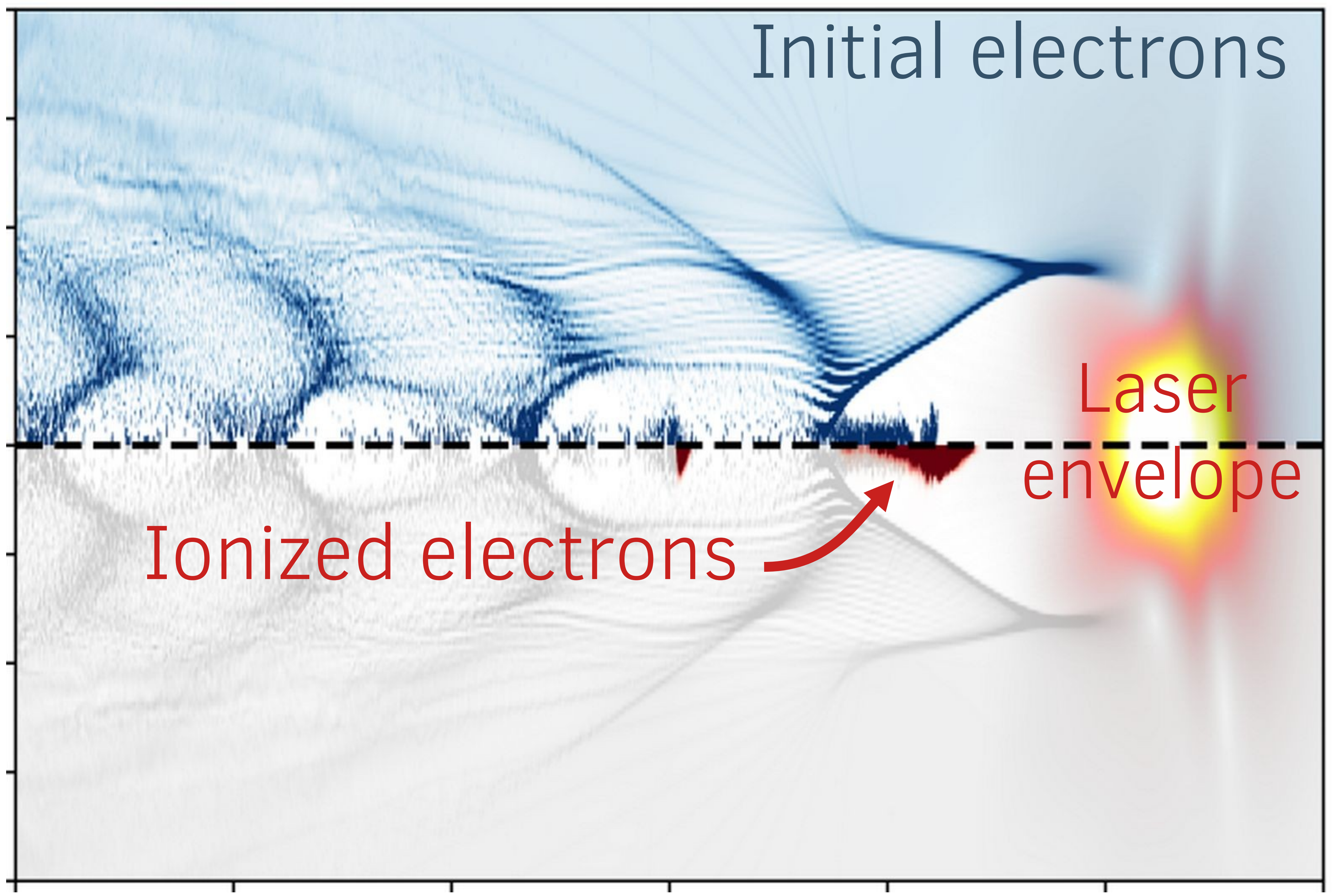


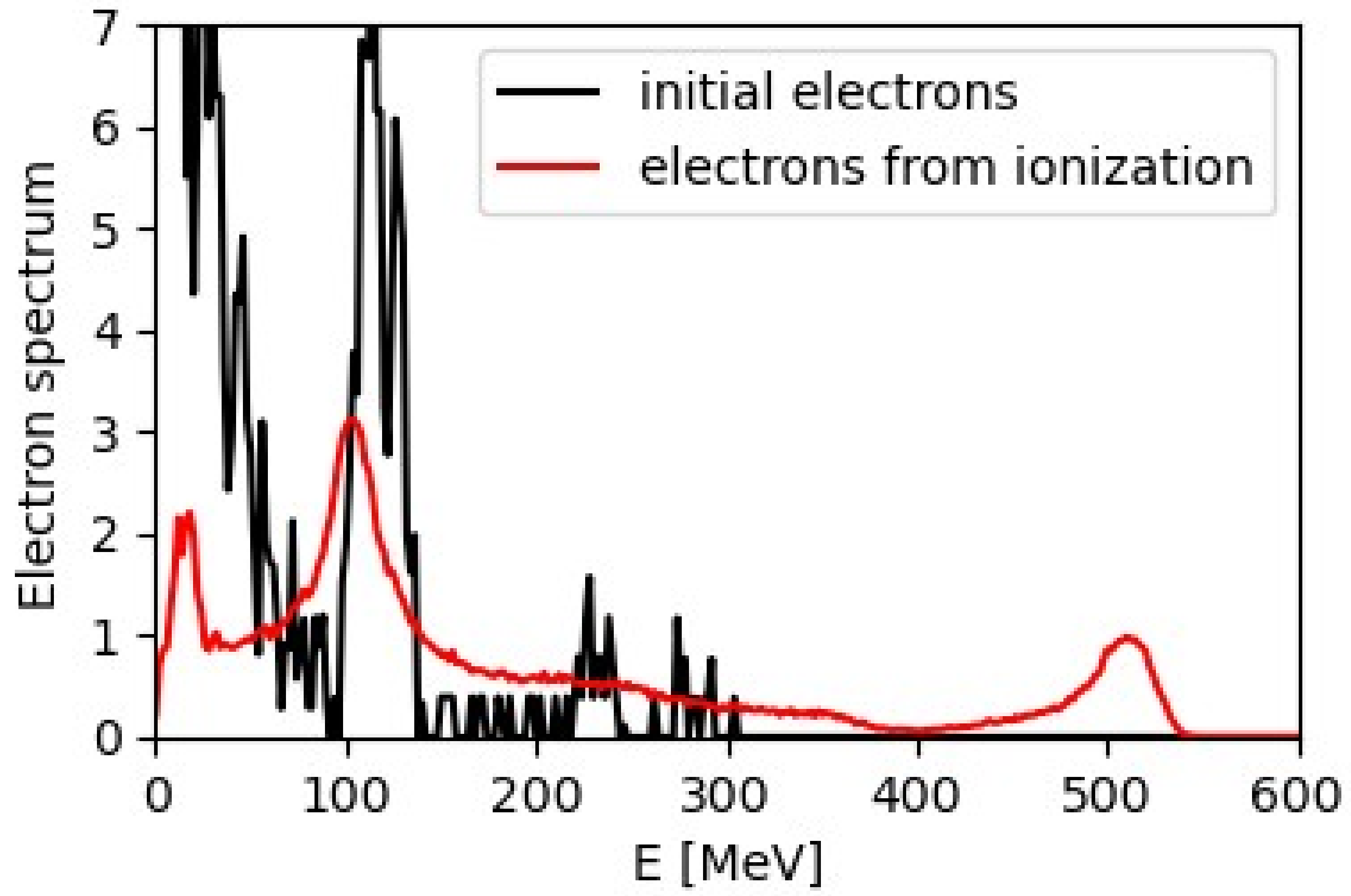
→ provides additional physics:

stopping power, resistivity, heat transport, …

Ionization by fields and by collisions may also be introduced with a Monte-Carlo method

Example with wakefield acceleration

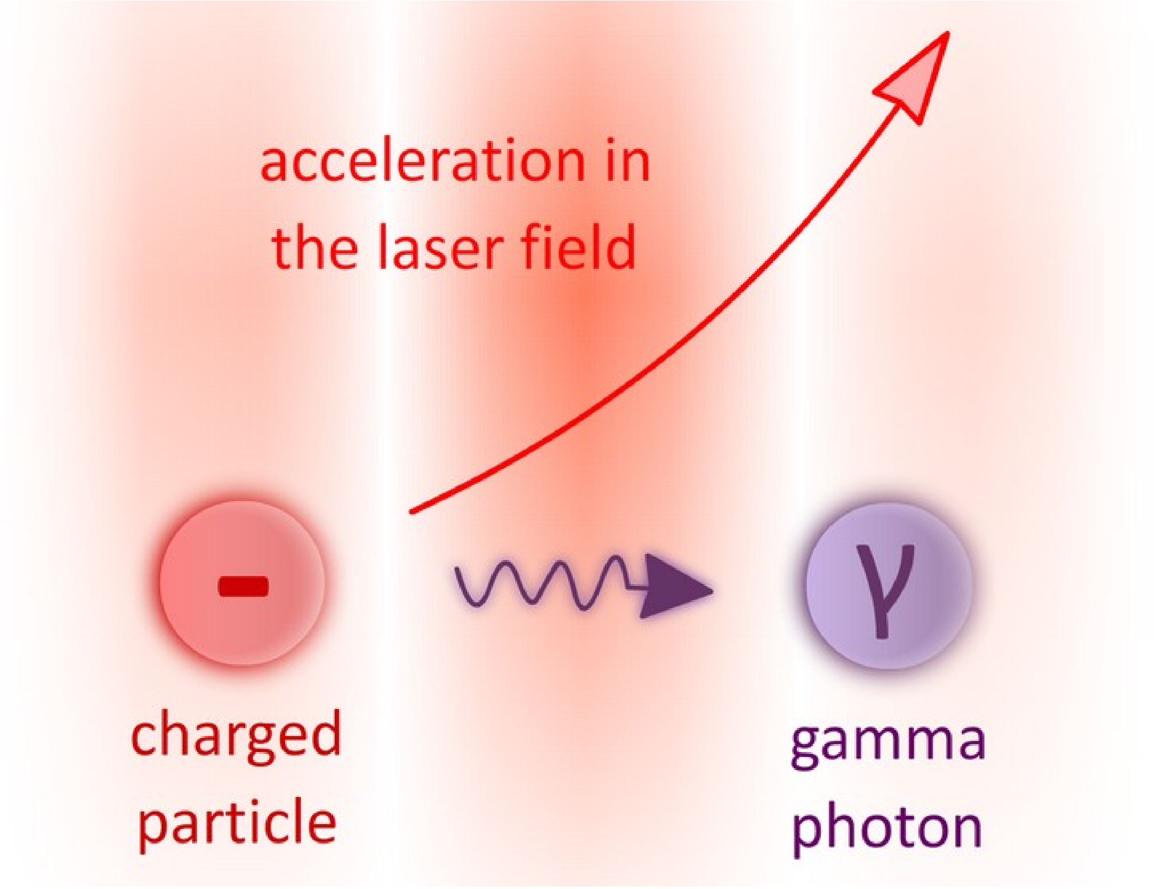




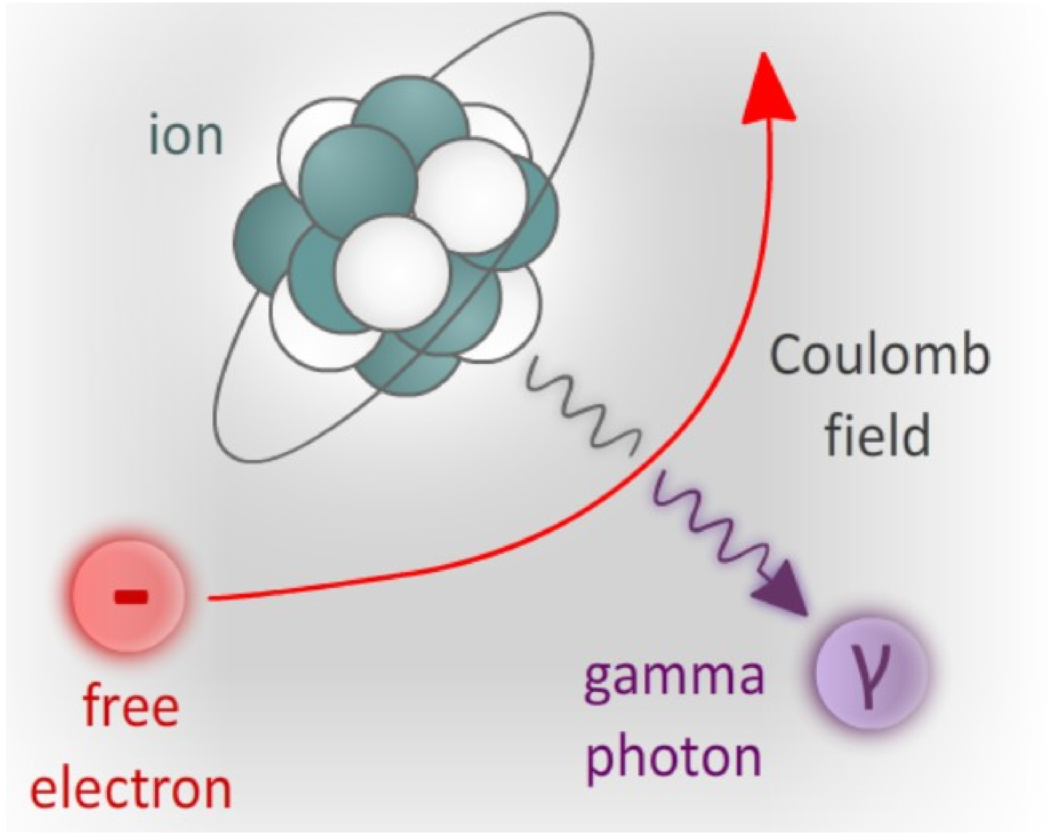
Strong-field quantum electrodynamics are relevant to multi-petawatt laser facilities

electron → photon

Non-linear Compton scattering

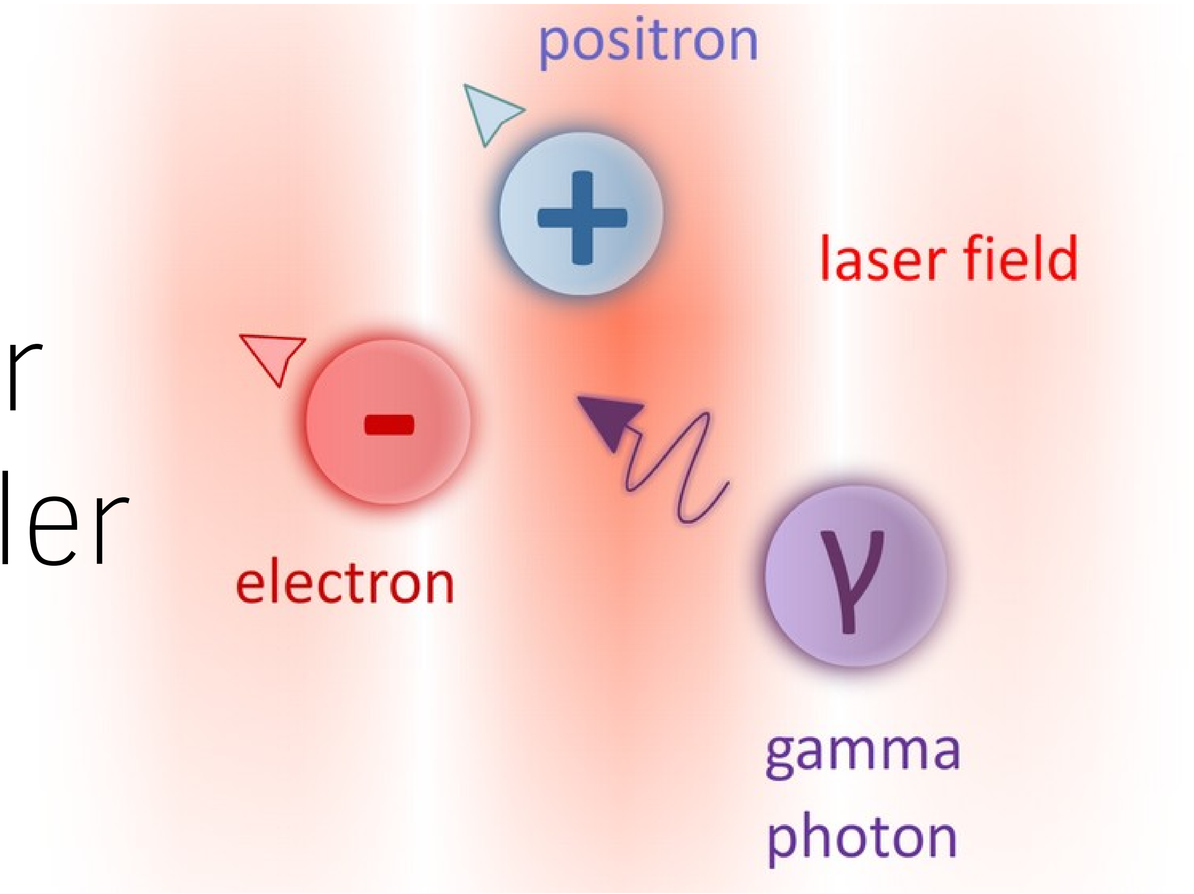


Bremsstrahlung

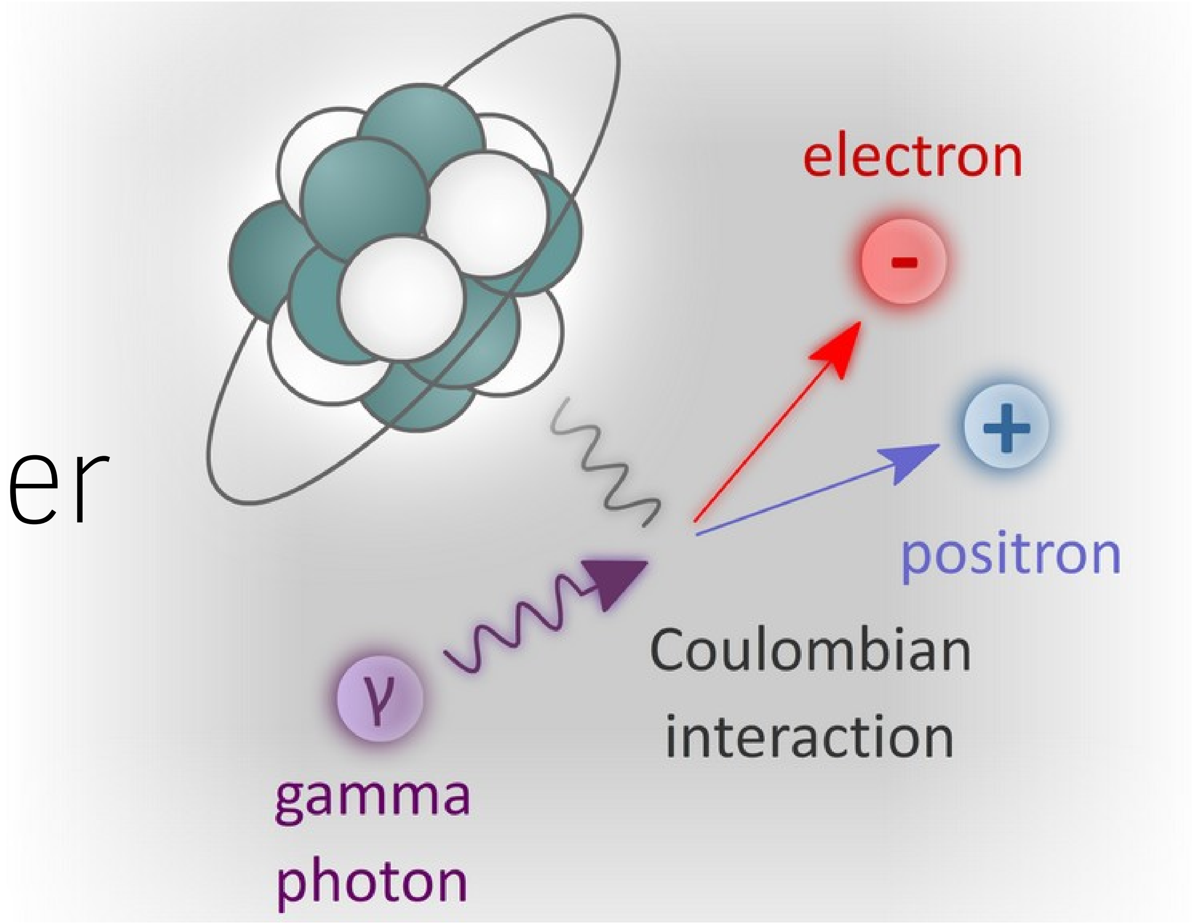


photon → e^-/e^+ pair

Non-linear Breit-Wheeler



Bethe-Heitler



Many other physics modules are possible

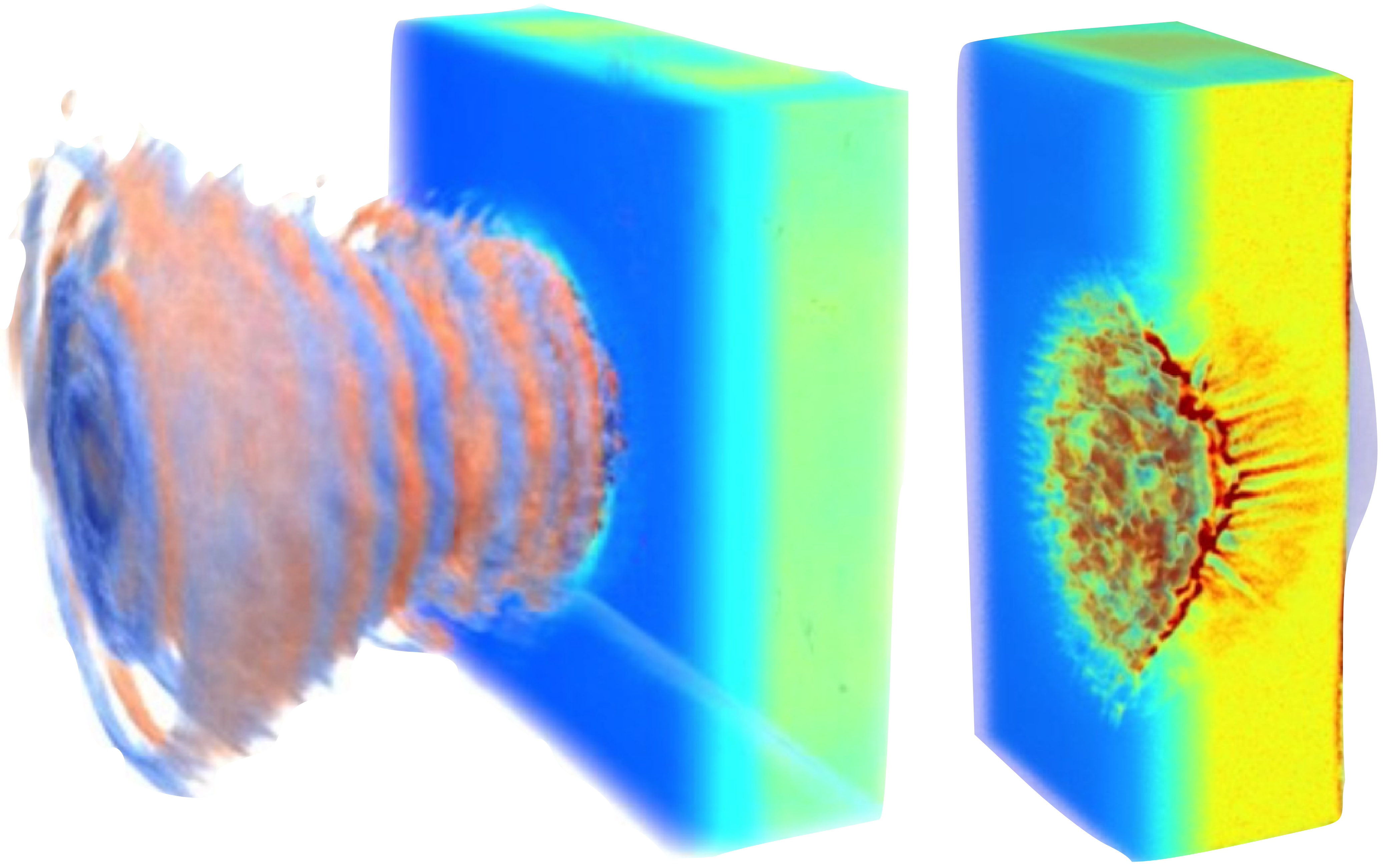
- Nuclear reactions
- Radiation transport
- Fluid species
- Particles injected from the boundary
- ...

An *Electromagnetic* PIC code:

- For which purpose?
- How does it work?
- Is it stable?
- What physics can it handle?
- **On which computer can it work?**

How long does a simulation take?

Example: Ultra-high intensity
laser-solid interaction



High density's Debye length ~ 2 nm
target size ~ 1 µm x 10 µm x 10 µm

→ 10^{10} cells
→ 10^{11} quasi-particles
→ 10^{13} bytes = **10 TeraBytes !**

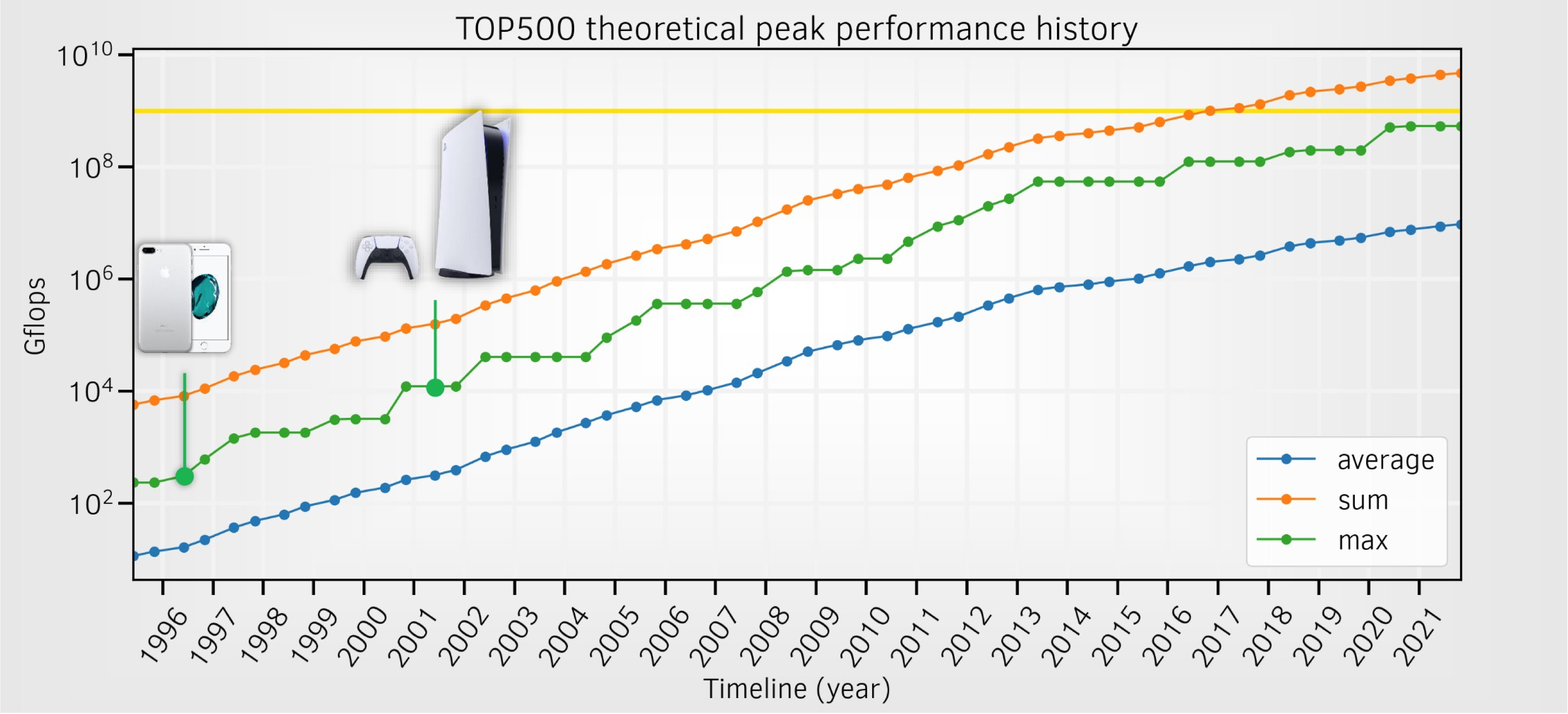
Duration 400 fs, maximum timestep ~ 0.004 fs

→ 100000 time steps
→ 10^{7} operations/particle
→ 10^{18} operations in total

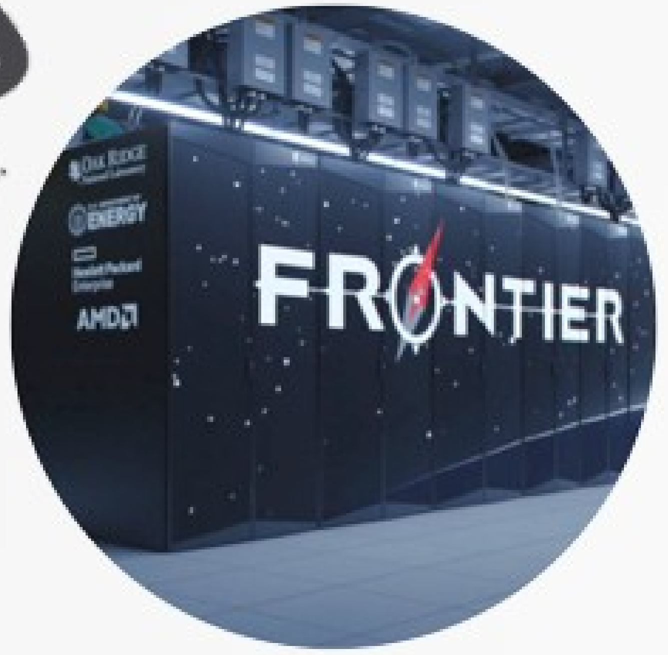
1 CPU @ 2 GHz → 16 **years !**

Solution: use 100 000 CPUs → 12h simulation

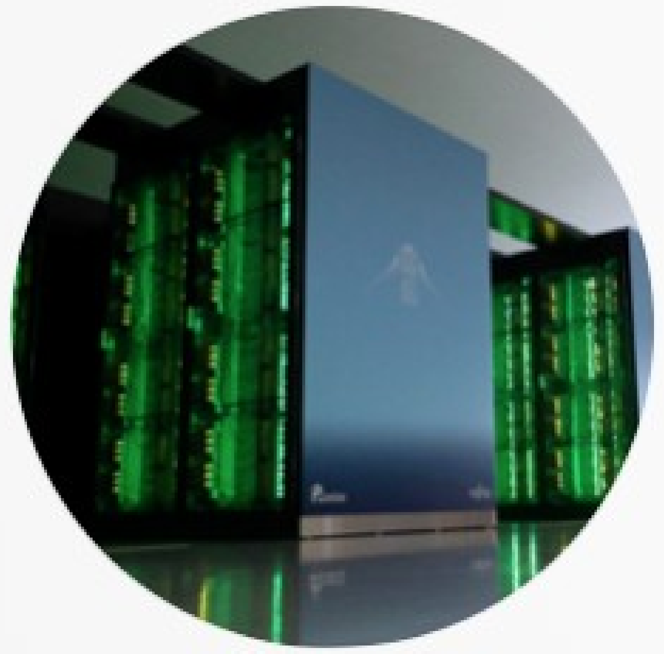
Supercomputers grow exponentially in performance



The exascale challenge

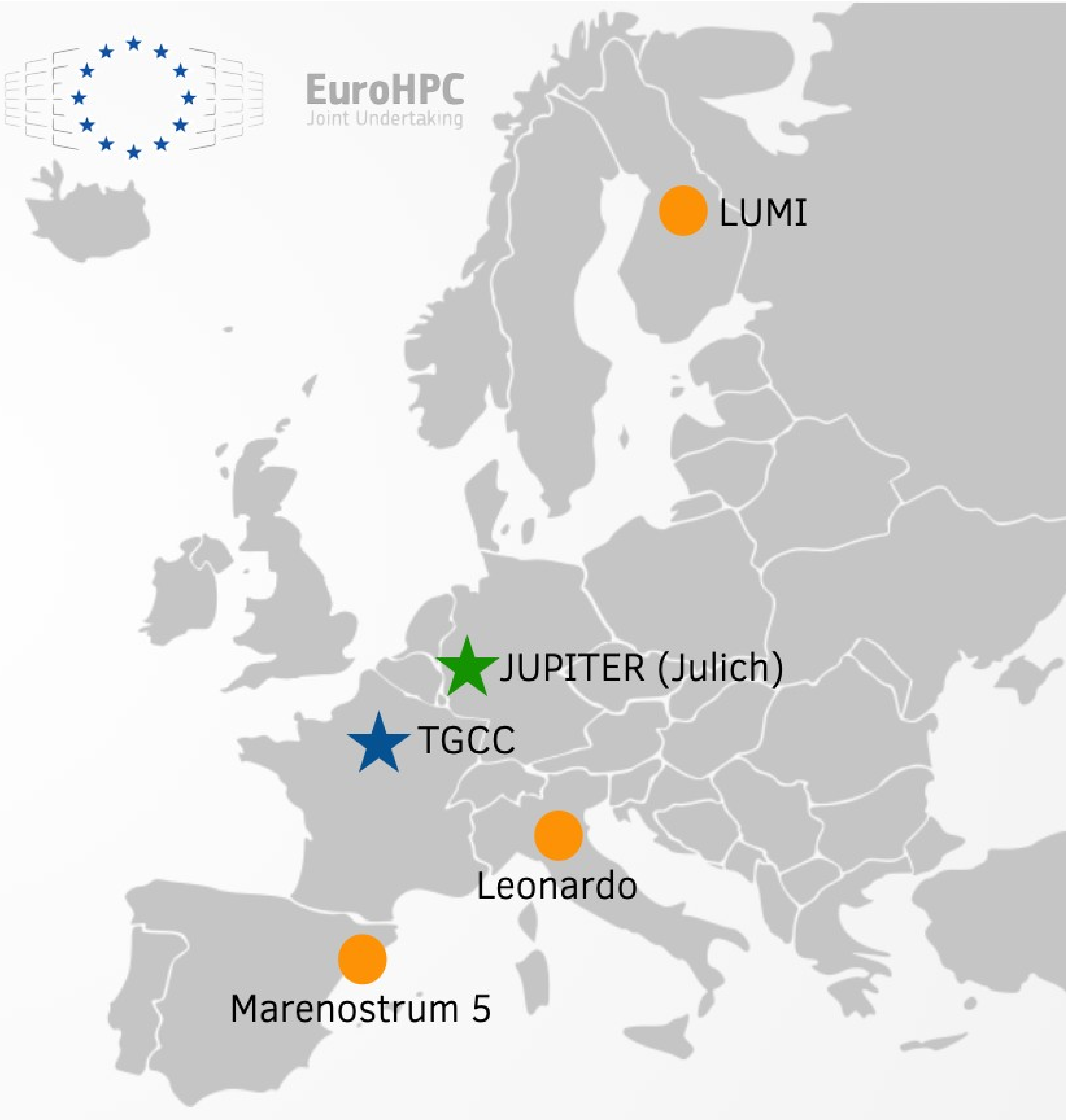


Frontier (US), 1100 Pflops
8.7 Mcores, 21 MW
AMD CPU + accel. (GPU)

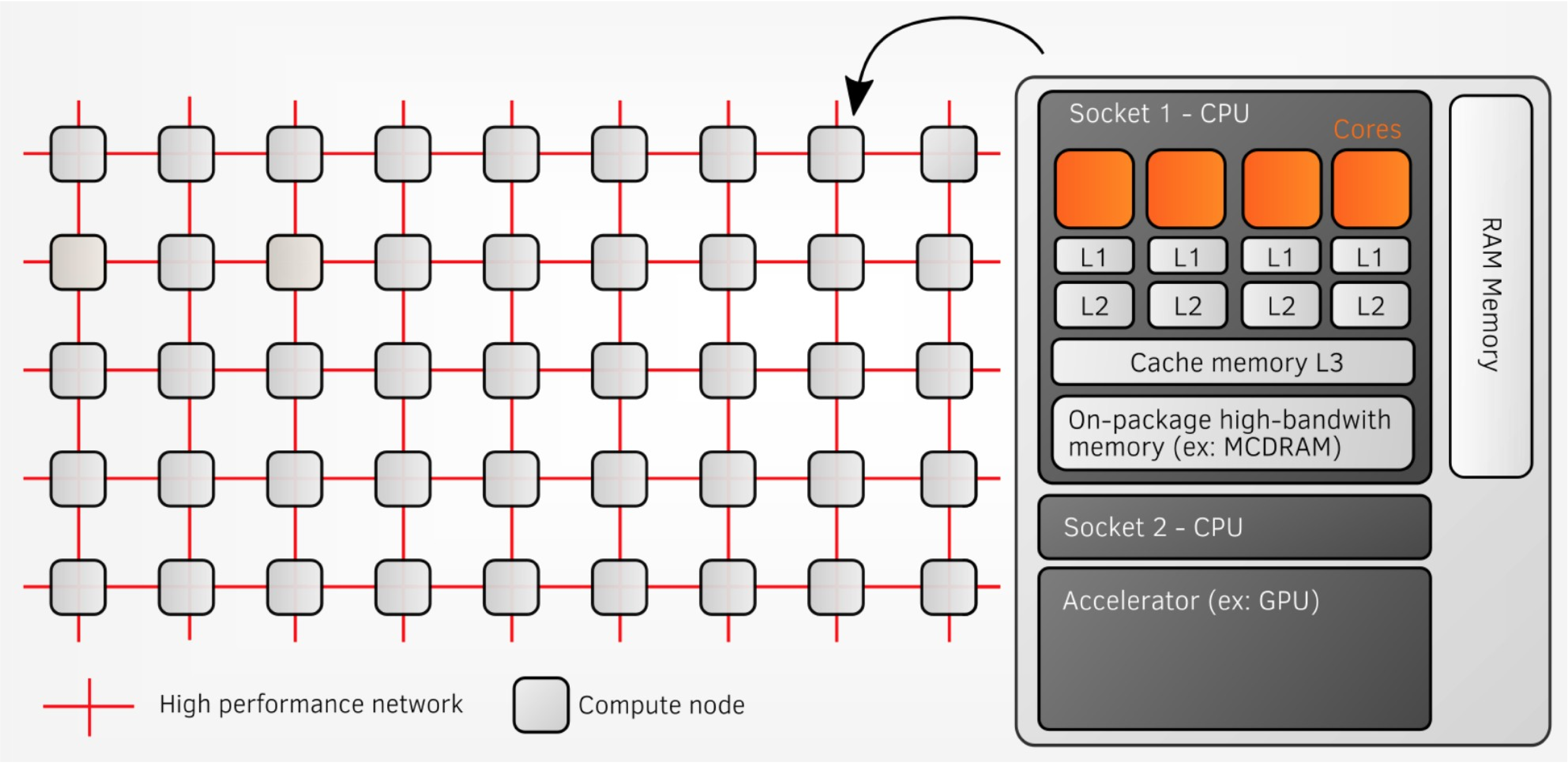


Fugaku (Japan), 442 Pflops
7.6 Mcores, 21 MW
Fujitsu ARM A64X

LUMI (Finland, EuroHPC), 152 Pflops
1.1 Mcores, 3 MW
AMD CPU + accel. (GPU)



Supercomputers are complex machines



It is extremely challenging to use efficiently this kind of computer

PIC users must know how to carefully setup supercomputers

- Choose the decomposition of the simulation box
- Distribute the workload across CPUs / GPUs
- Test many possible configurations

Summary

PIC codes are popular and can address many plasma physics problems

The electromagnetic PIC method solves the Vlasov+Maxwell system

It has important limitations and stability requirements

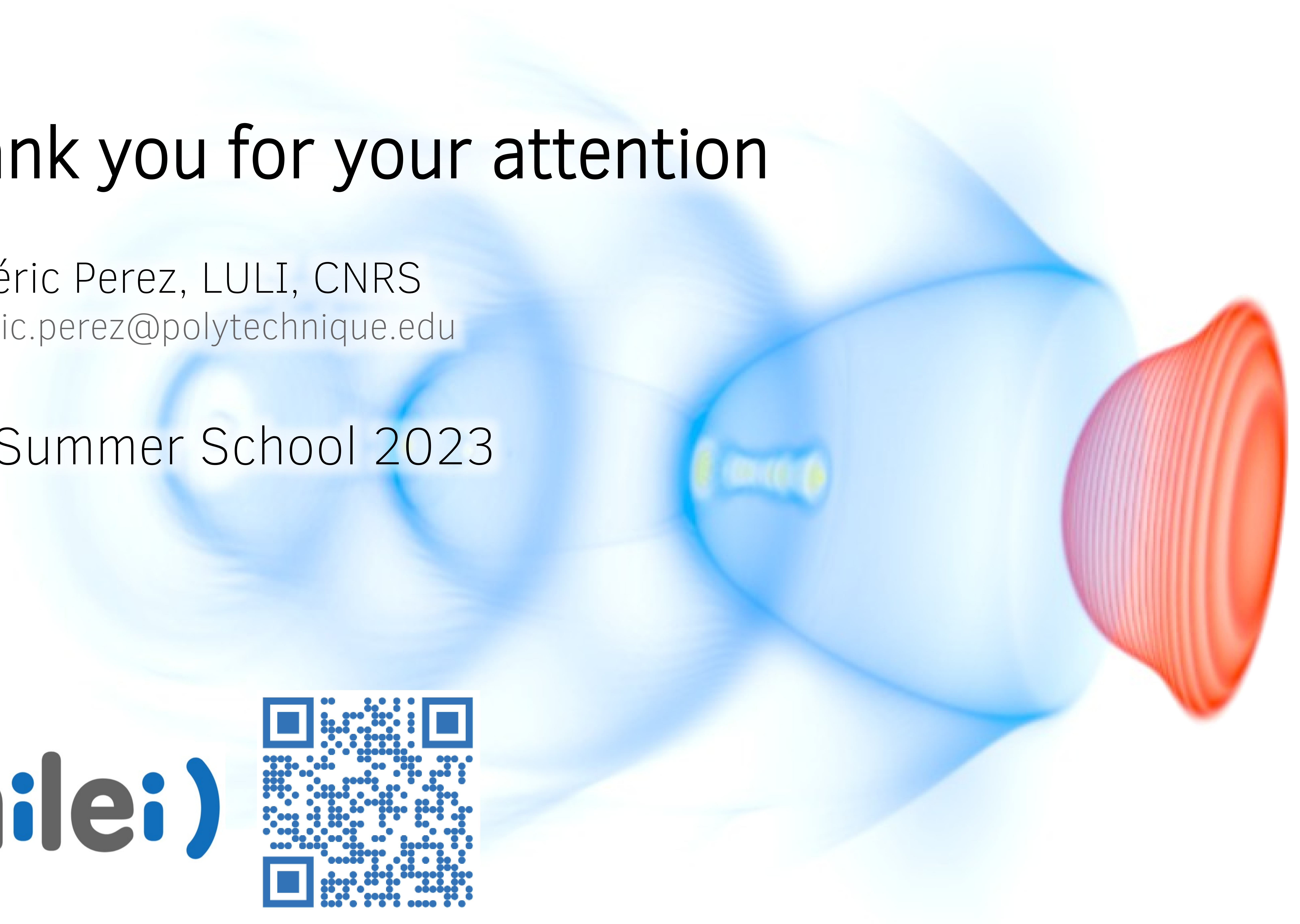
Other physics can be added (collisions, ionization, QED, ...)

Supercomputers are often needed, and only modern codes can tackle recent development in computer architectures

Thank you for your attention

Frédéric Perez, LULI, CNRS
frederic.perez@polytechnique.edu

ELI Summer School 2023



Smilei)