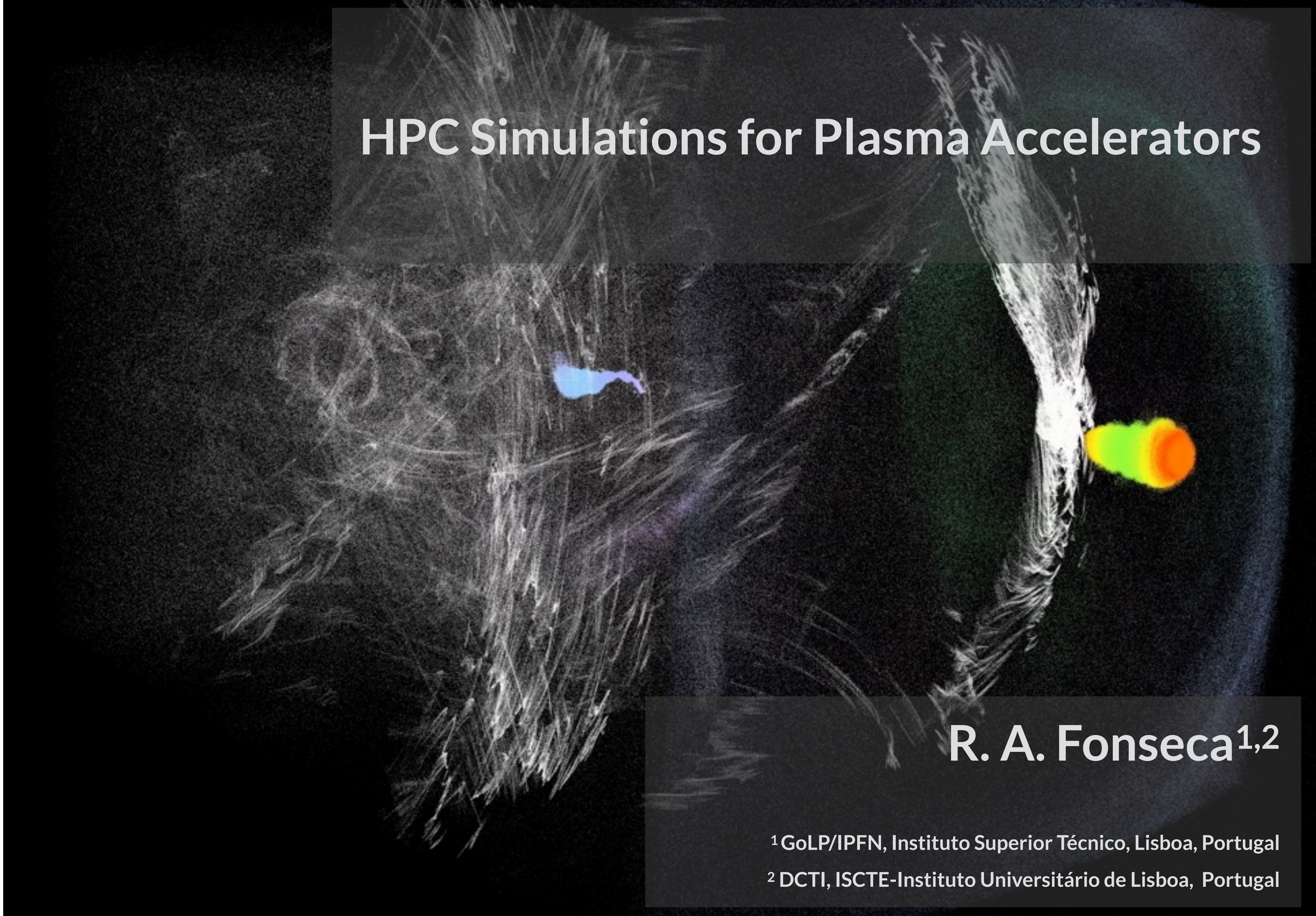


HPC Simulations for Plasma Accelerators

R. A. Fonseca[1,2]

[1] GoLP/IPFN, Instituto Superior Técnico, Lisboa, Portugal

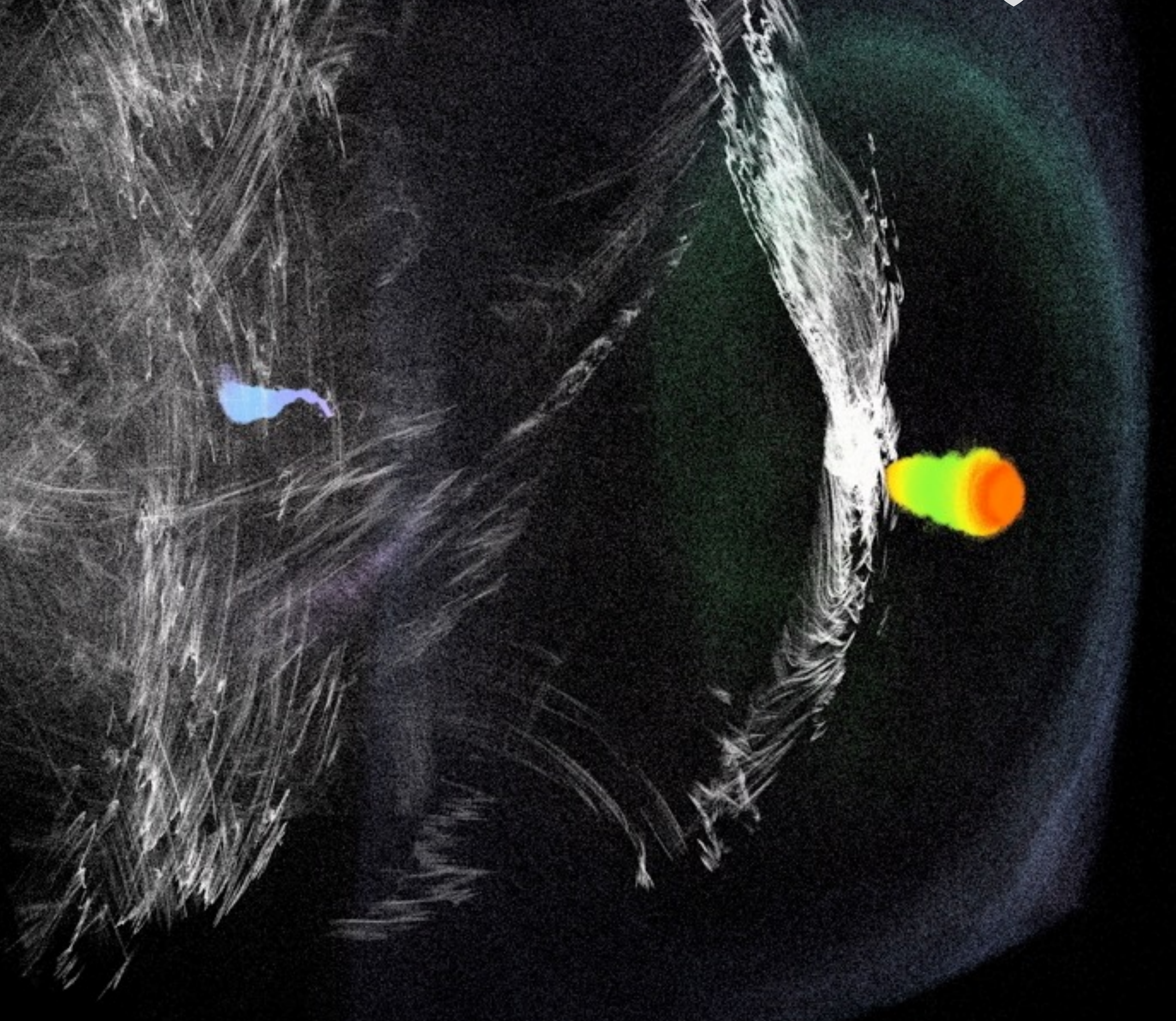
[2] DCTI, ISCTE-Instituto Universitário de Lisboa, Portugal

The need for HPC simulations in plasma accelerators research

- **Plasma accelerators present an extremely challenging scientific problem**
  - Involves multiple scales and phenomena
  - Highly non-linear character of underlying physics
  - Depends on complicated and self-consistent trajectories of individual plasma particles

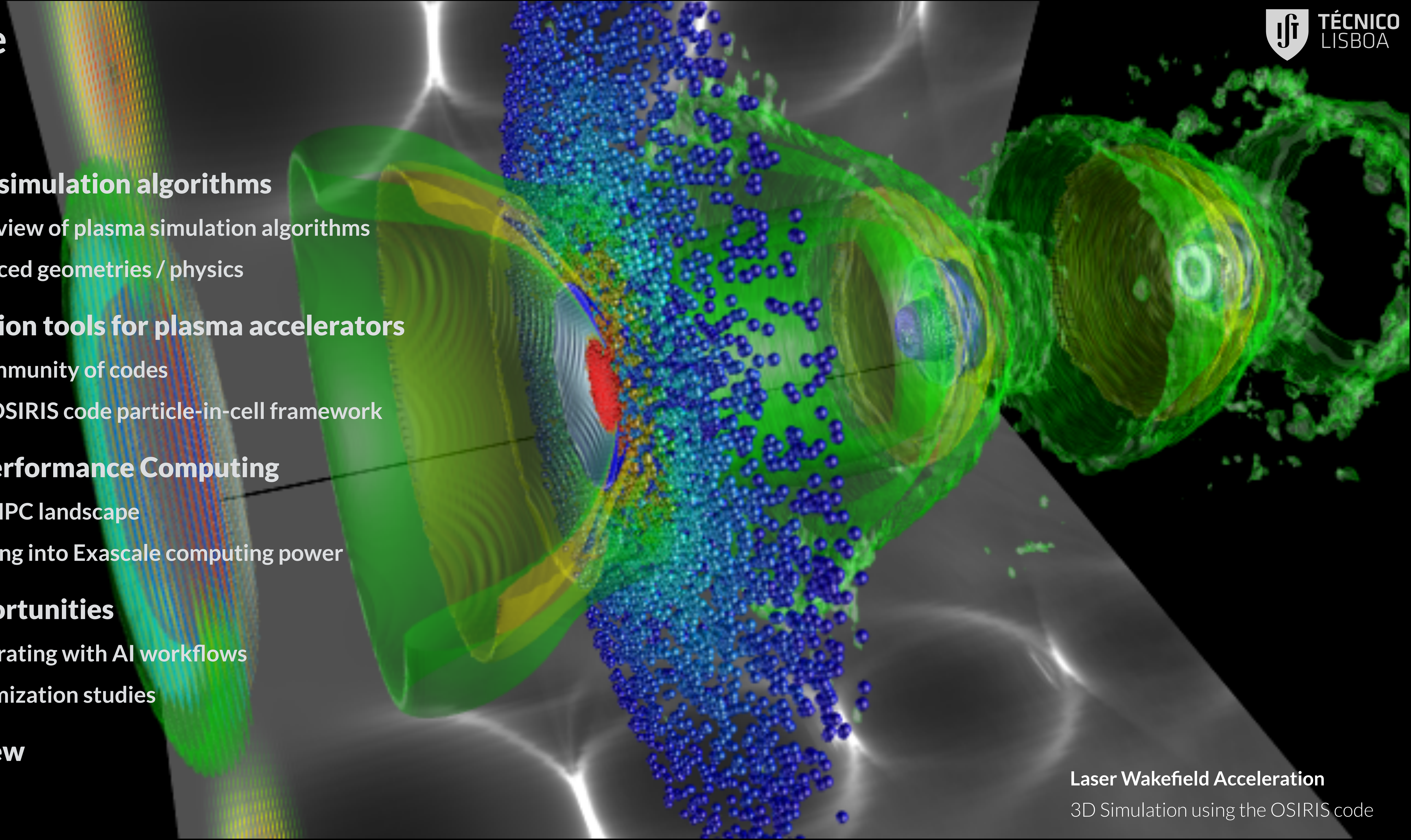
- **Purely analytical treatment impossible**
  - (Mostly) Limited to linear scenarios, and
  - Reduced geometries

- **Furthering knowledge requires advanced numerical tools**
  - Use of state-of-the art numerical codes, and
  - Pushing the envelope of what is achievable with present HPC technology



Outline

- **Plasma simulation algorithms**
  - Overview of plasma simulation algorithms
  - Reduced geometries / physics
- **Simulation tools for plasma accelerators**
  - A community of codes
  - The OSIRIS code particle-in-cell framework
- **High-Performance Computing**
  - The HPC landscape
  - Tapping into Exascale computing power
- **AI Opportunities**
  - Integrating with AI workflows
  - Optimization studies
- **Overview**

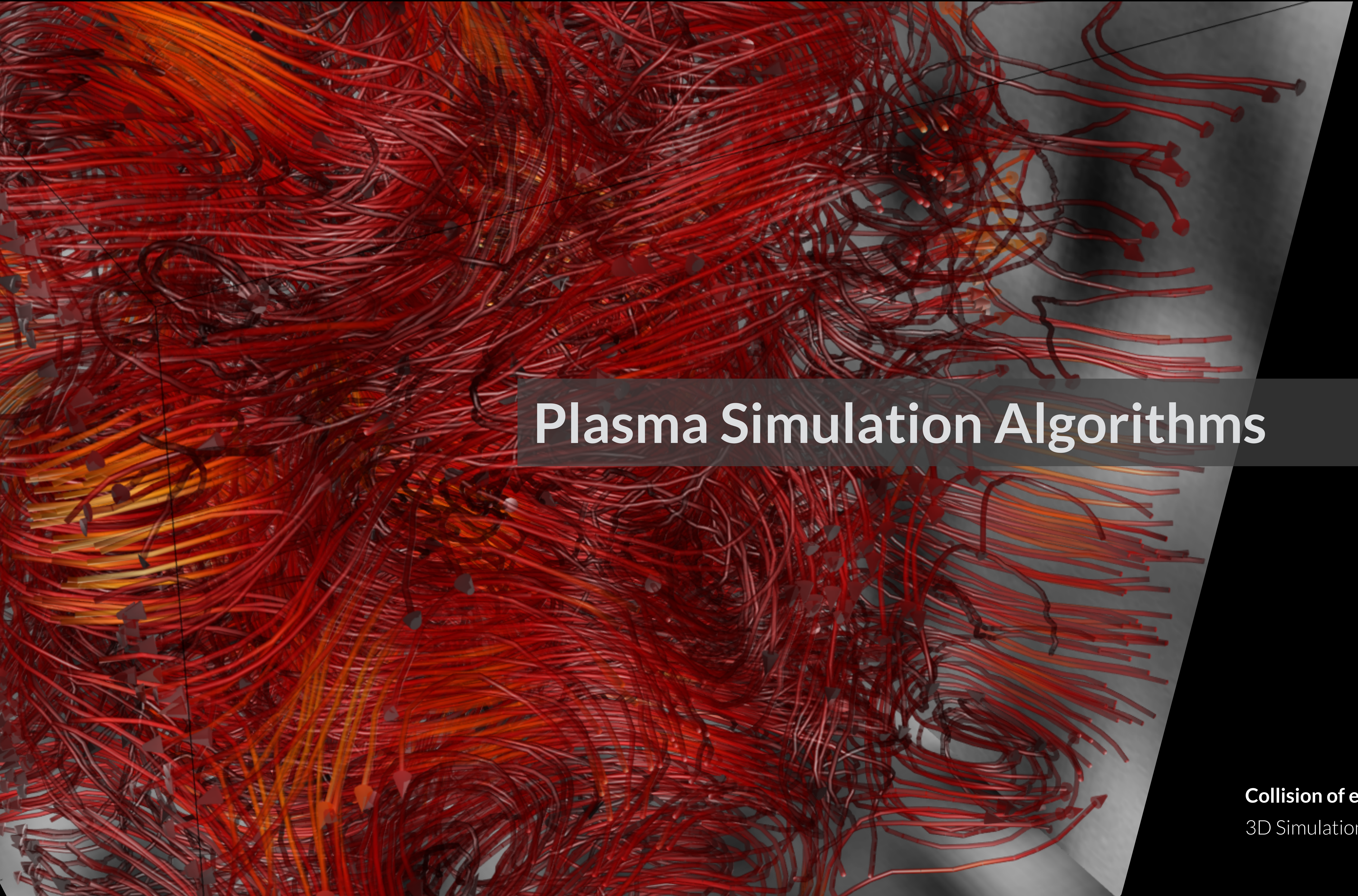


Laser Wakefield Acceleration

3D Simulation using the OSIRIS code

Plasma Simulation Algorithms

TÉCNICO LISBOA



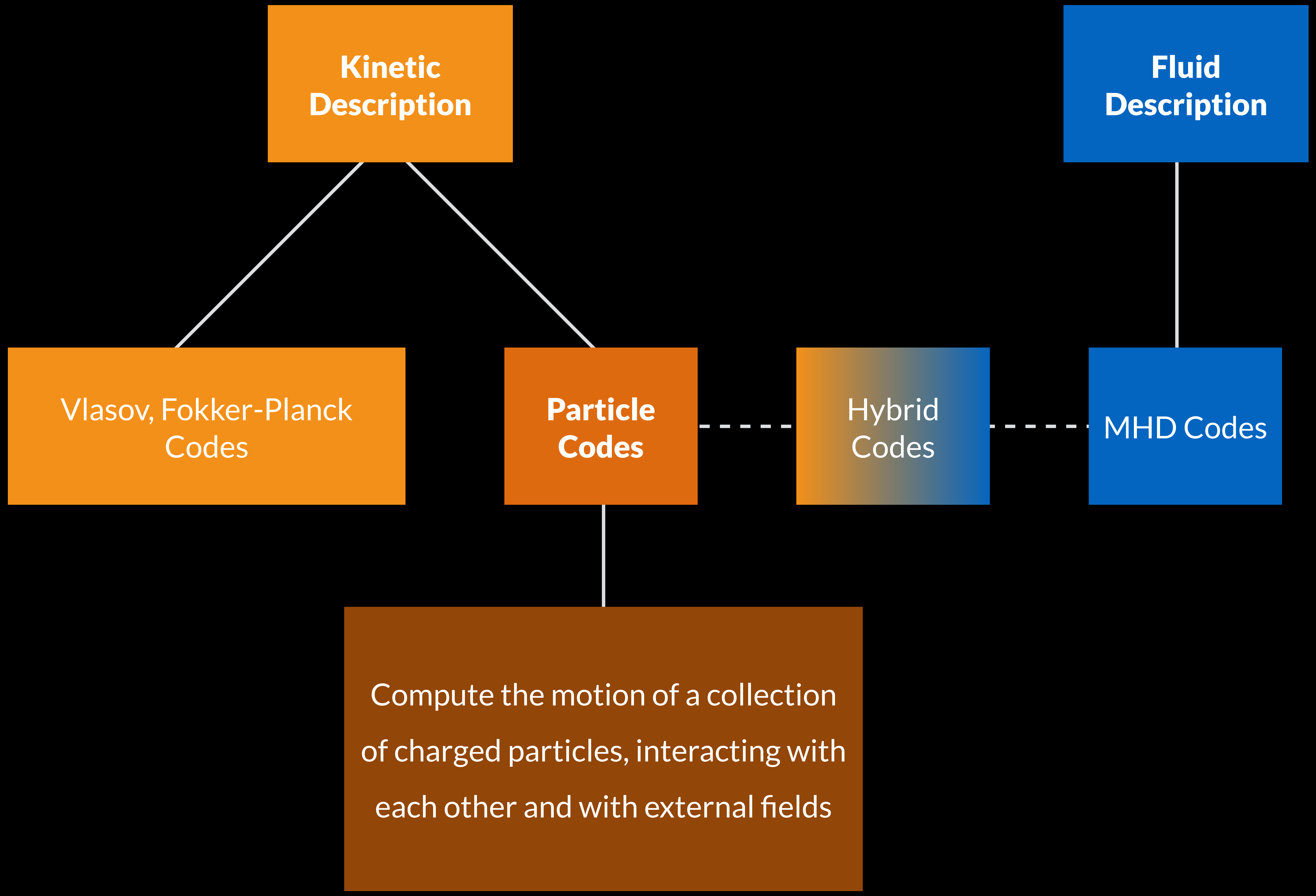
Collision of $e^{-}e^{+}$ plasma shells

3D Simulation using the OSIRIS code

Overview of plasma simulation algorithms

- **Plasma Simulations Using Particles**
  - Pioneered by John Dawson and Oscar Buneman circa 1960
  - Use macro particles to simulate large spatial regions
    - 1 simulation particle corresponds to many plasma particles
  - Particle-Particle simulations
    - Computations go with $O(N_p^2)$
    - Computationally very demanding

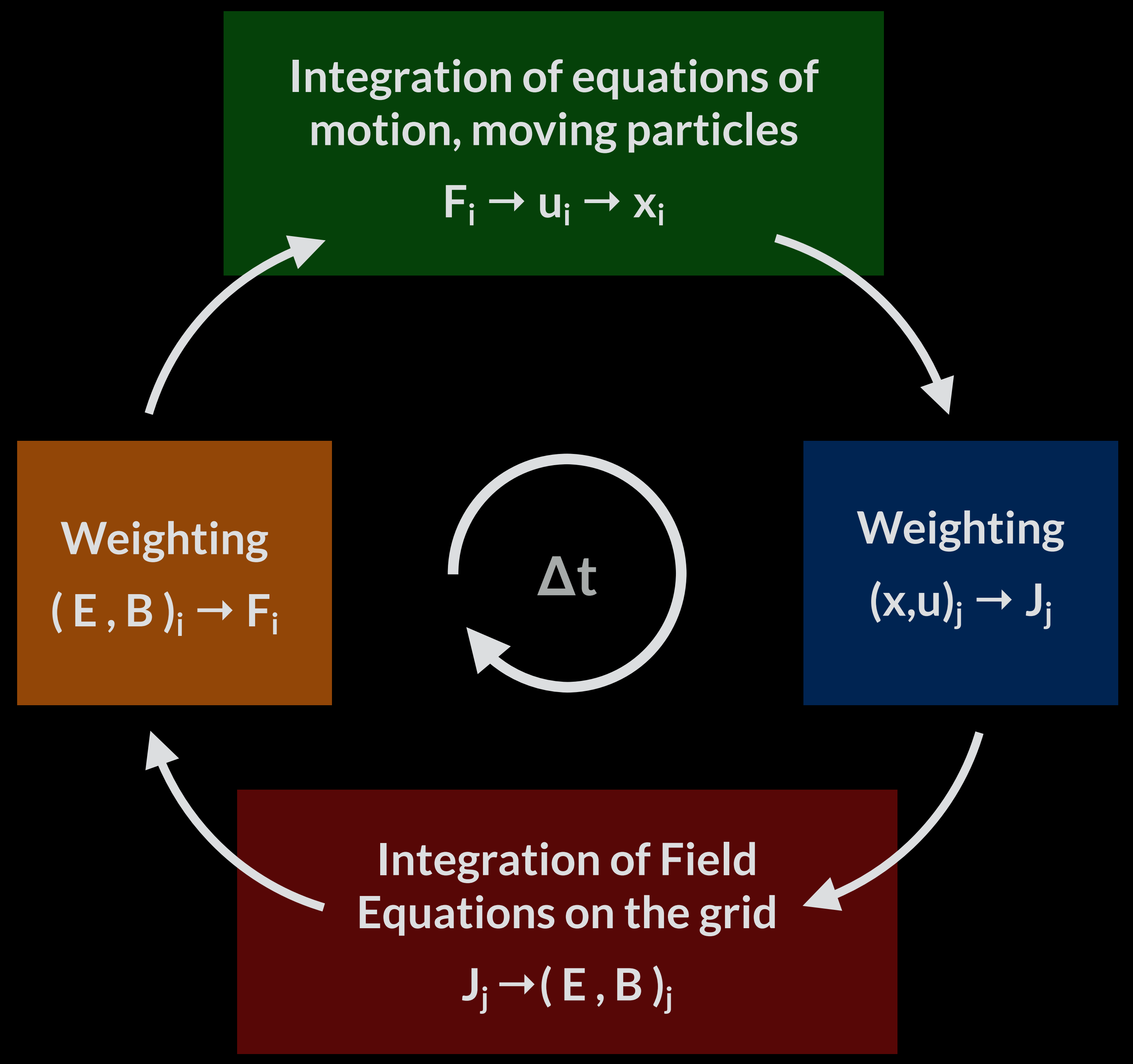
- **Particle-In-Cell algorithms**
  - Interact particles through fields
  - Discretize fields on grids
  - Interpolate fields at particle positions to calculate forces
  - Deposit particle charge/current on a grid
  - Particle-Mesh algorithm
    - Computations go with $O(N_p)$
    - Still computationally heavy but much more tractable



The particle-in-cell (PIC) algorithm

PIC algorithm

$$\frac{d\mathbf{u}}{dt} = \frac{q}{m}\left(\mathbf{E} + \frac{1}{\gamma c}\mathbf{u} \times \mathbf{B}\right)$$



$$\frac{\partial \mathbf{E}}{\partial t} = c\nabla \times \mathbf{B} - 4\pi \mathbf{j}$$

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

PIC codes are computationally demanding

- Models is based on fundamental Physics laws

Multi-scale problems

- Large disparity of spatial/temporal scales

Sample problem: 10 GeV LWFA stage

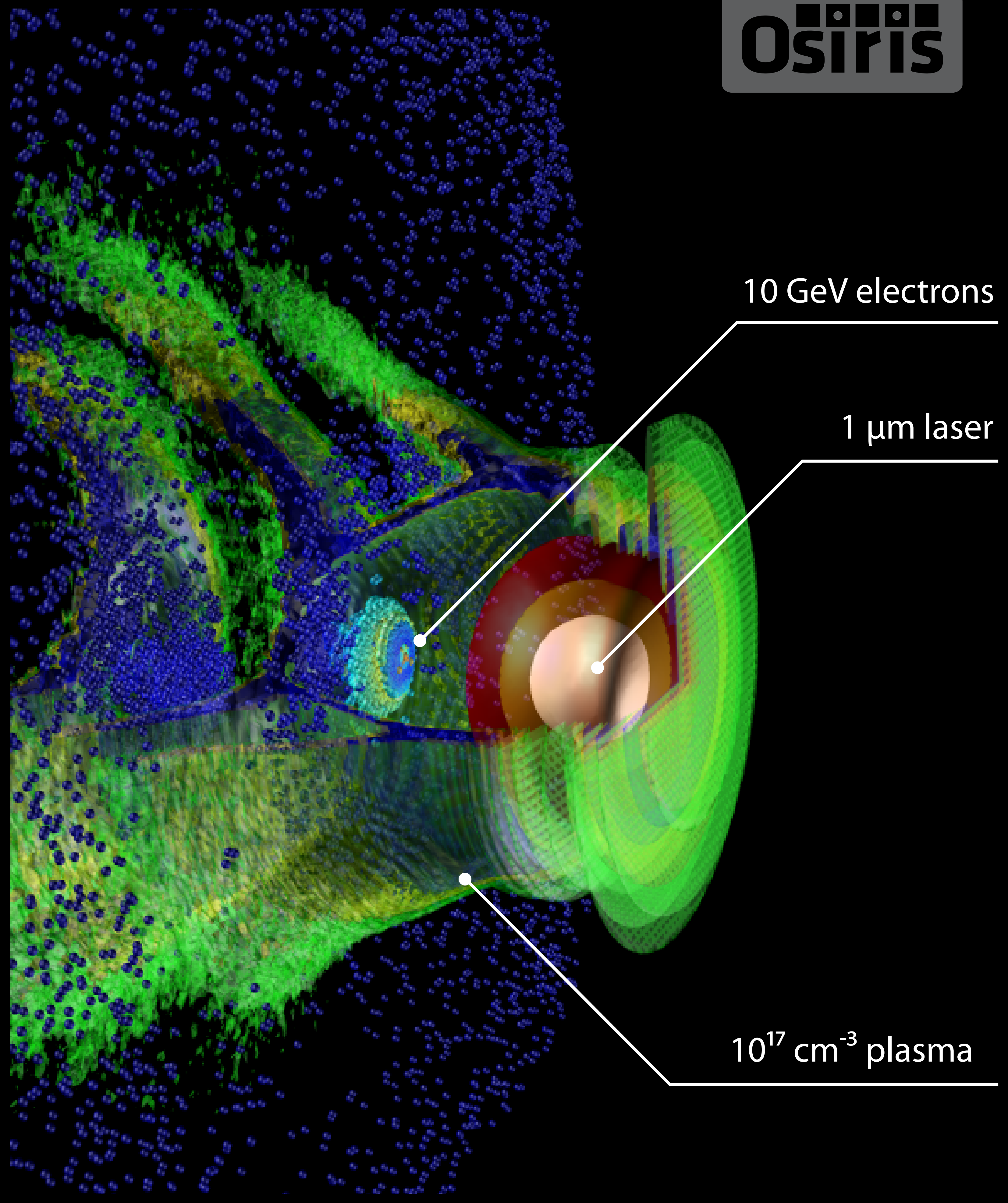
- λ_0 ~ 1 µm
- L ~ 0.5 m

Computational Requirements

- ~ 10^9 grid cells
- ~ 10^{10} particles
- Iterations ~ 10^6 - 10^7
- Memory ~ 1 - 10 TB
- Operations ~ 10^{18} - 10^{19}

Exascale performance

- Simulation time ~ 10s



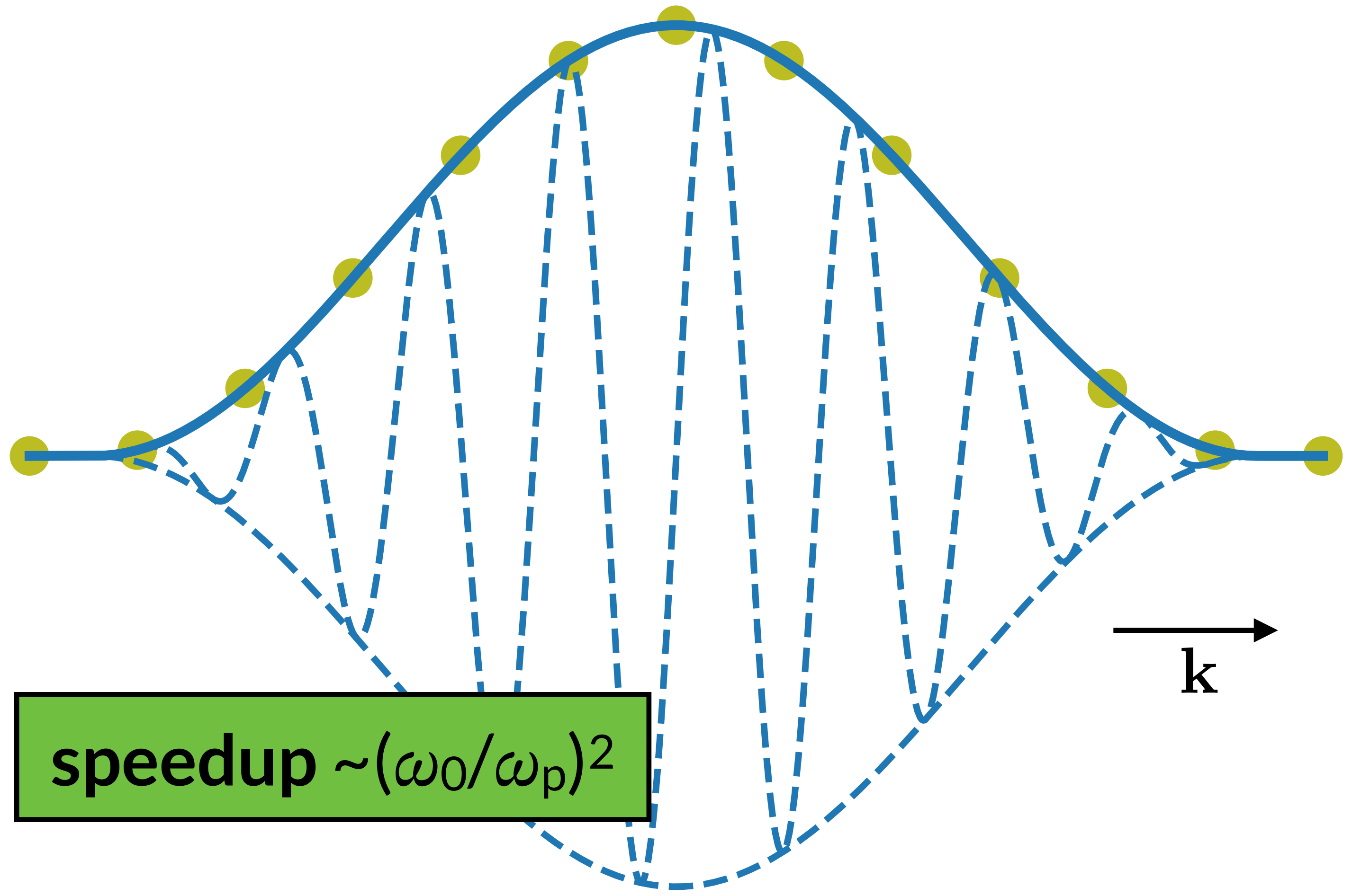
Reduced Physics - Ponderomotive Guiding Center approximation

Ponderomotive Guiding Center* (PGC)

Temporal / Spatial resolution:

PIC: laser frequency/wavelength

PGC: plasma frequency/skin depth



speedup ~$(\omega_0/\omega_p)^2$

- requires model for laser envelope propagation
- push particles using self consistent plasma fields and ponderomotive force

Envelope evolution

laser envelope — spatial coordinate $\xi = t - x$

$$\partial_\tau a = \frac{1}{2i\omega_0}\left(1 + \frac{\partial_\xi}{i\omega_0}\right)\left(\chi a + \Delta_T a\right)$$

time — laser frequency — plasma coupling

Ponderomotive force

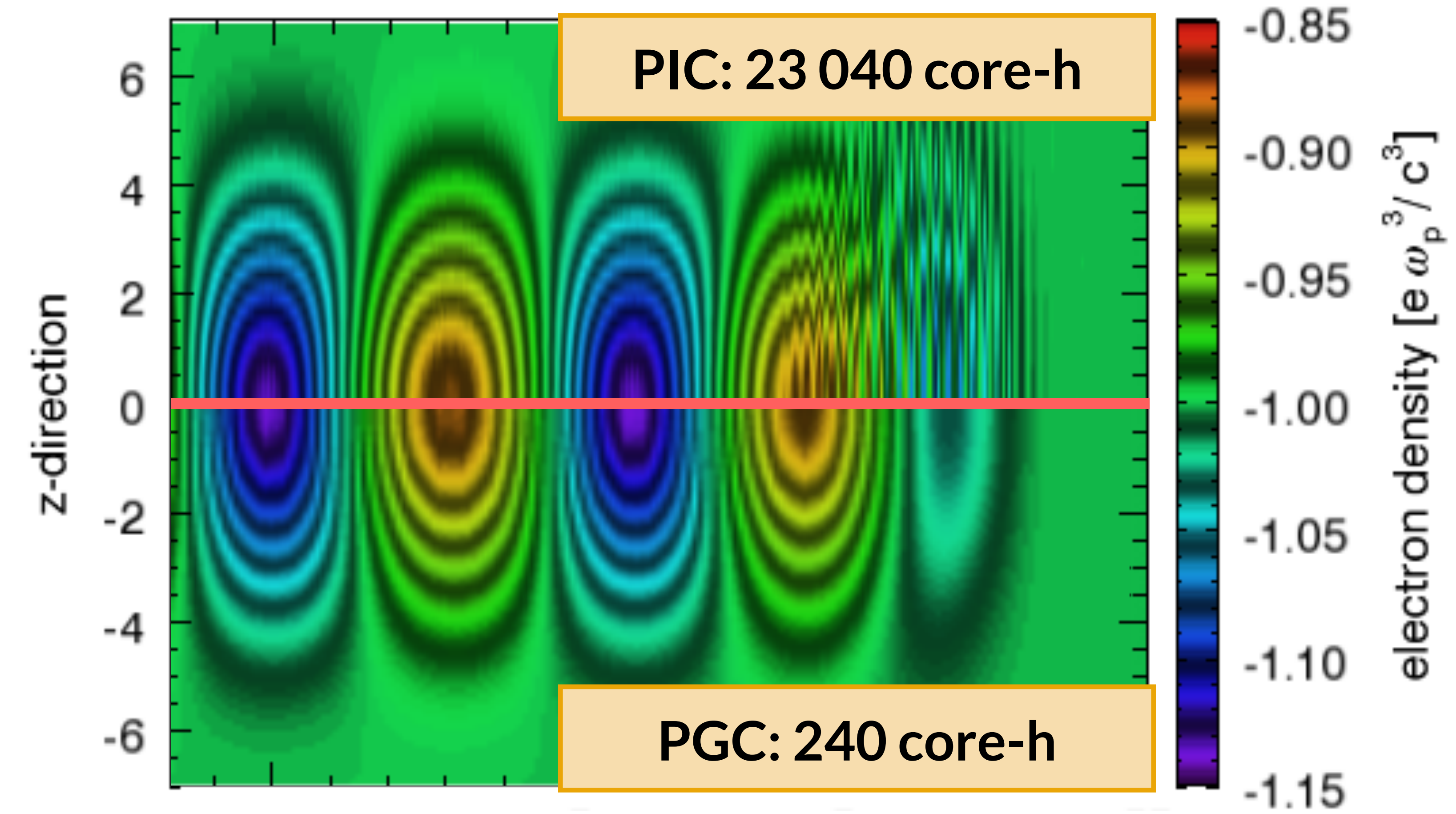
$$\mathbf{F}_p = -\frac{1}{4}\frac{q^2}{\langle m \rangle}\nabla |a|^2$$

Plasma parameters

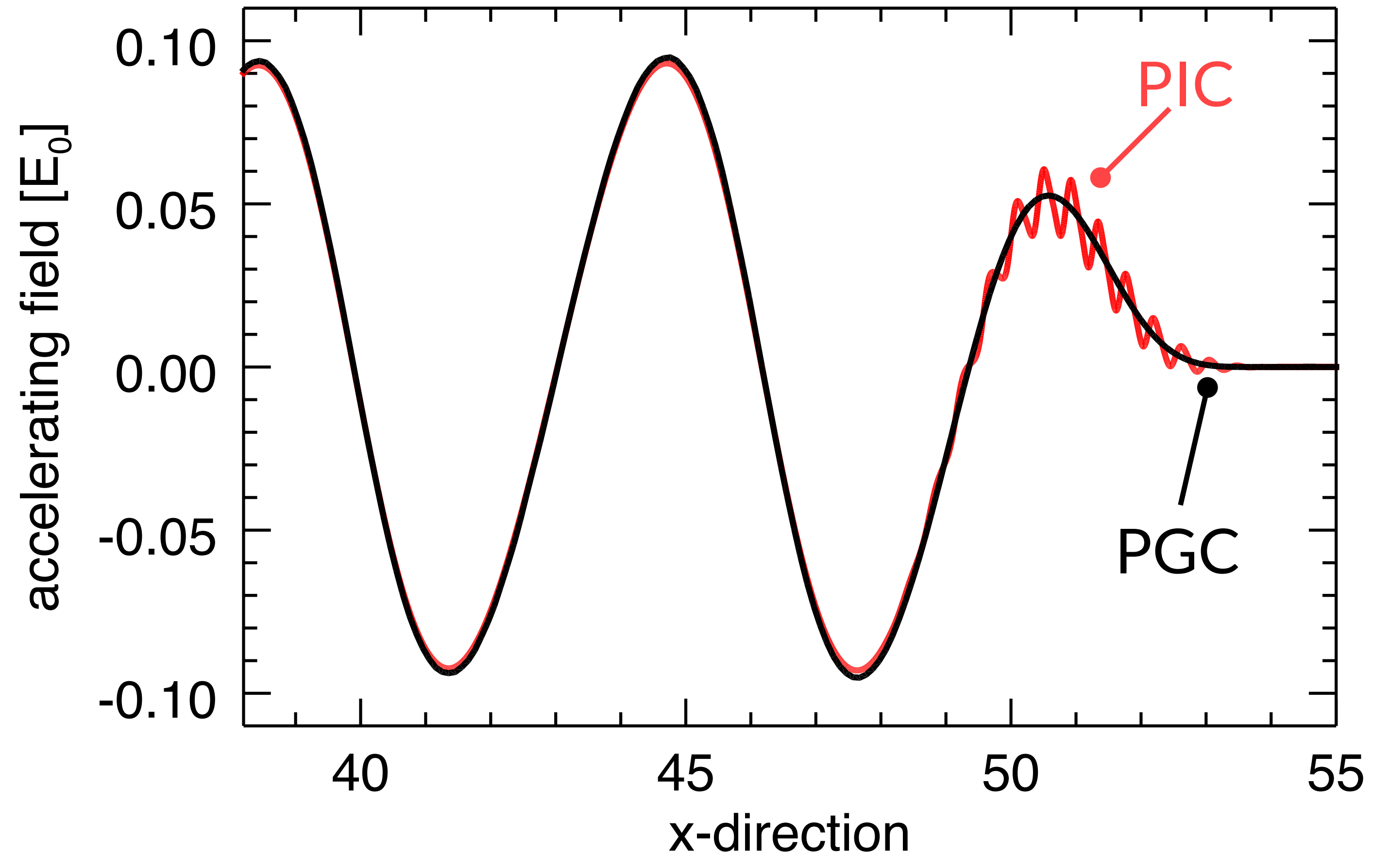
$$\chi = -\sum_i \frac{q_i \rho_i}{\langle m_i \rangle}$$

$$\langle m_i \rangle = \sqrt{m_0^2 + \mathbf{p}^2 + (q|a|)^2/2}$$

electron density



accelerating gradients

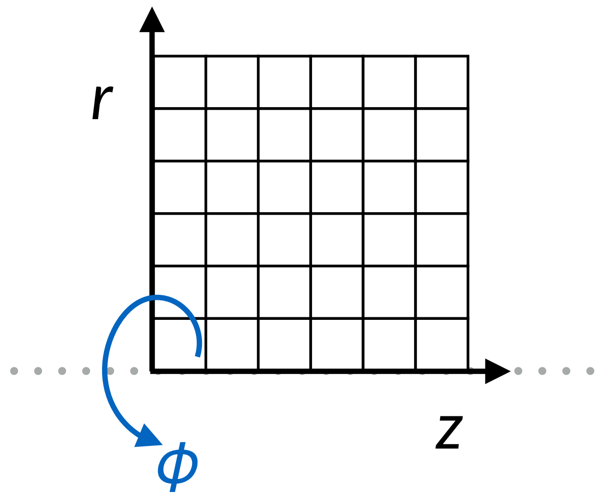


* D. Gordon et. al., **IEEE Trans. Plasma Sci.** 28, 1135 (2000)

Reduced geometries - Cylindrical mode expansion

- **Quasi-3D algorithm**
  - PIC in $r - z$, gridless (spectral) in ϕ
  - Particles in 3D, grids in 2D
- **Retain only relevant cylindrical modes**
  - For particle beam driven accelerator only fundamental mode is required
  - Complex effects (e.g. hosing) require additional modes
  - Laser drivers require at least m = 1
- **Capture 3D effects with smaller computational cost**
  - Speedup from 3D on the order of ~10[2]

Azimuthal Mode Expansion[1]



$$\mathbf{F}(r,z,\phi) = \Re\left\{\sum_{m=0} \mathbf{F}^m(r,z)e^{im\phi}\right\}$$
$$= \mathbf{F}^0(r,z) + \Re\{\mathbf{F}^1\}\cos(\phi) - \Im\{\mathbf{F}^1\}\sin(\phi) + \Re\{\mathbf{F}^2\}\cos(2\phi) - \Im\{\mathbf{F}^2\}\sin(2\phi) + \cdots.$$

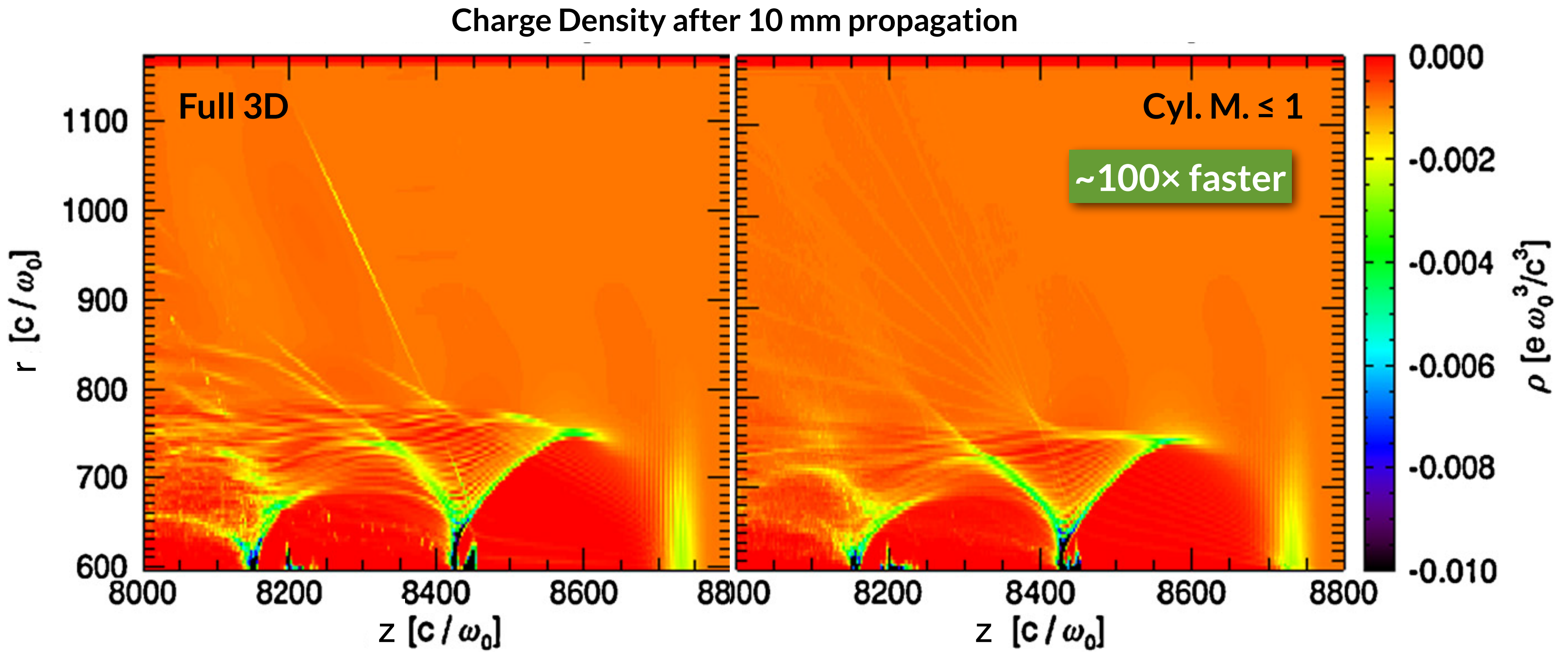
Charge-conserving Field Advance [2]

$$\partial_t B_r^m = \frac{im}{r}E_z^m + \partial_z E_\theta^m \qquad \partial_t E_r^m = -\frac{im}{r}E_z^m - \partial_z E_\theta^m - J_r^m$$

$$\partial_t B_\theta^m = \partial_z E_r^m + \partial_r E_z^m \qquad \partial_t E_\theta^m = \partial_z B_r^m - \partial_r E_z^m - -J_\theta^m$$

$$\partial_t B_z^m = -\frac{1}{r}\partial_r(rE_\theta^m) + \frac{im}{r}E_r^m \qquad \partial_t E_z^m = \frac{1}{r}\partial_r(rB_\theta^m) - \frac{im}{r}B_r^m - -J_z^m$$

LWFA simulation



[1] A. Lifschitz *et al.*, **JCP** 228, 1803 (2009)

[2] A. Davidson *et al.*, **JCP** 281, 1063 (2015)

Simulation tools for plasma accelerators

Laser Wakefield Acceleration

3D Simulation using the OSIRIS code

A large community effort

Computer simulations have played a key role in the field of plasma accelerators

- In particular, PIC codes have been instrumental in demonstrating the feasibility of the concept
- There would not be a plasma accelerator field without PIC simulations

There is a large community of Particle-in-Cell codes supporting the field

- Test and explore new ideas
- Support experimental efforts
- Optimize accelerator design

This is an ever evolving ecosystem of codes and algorithms*

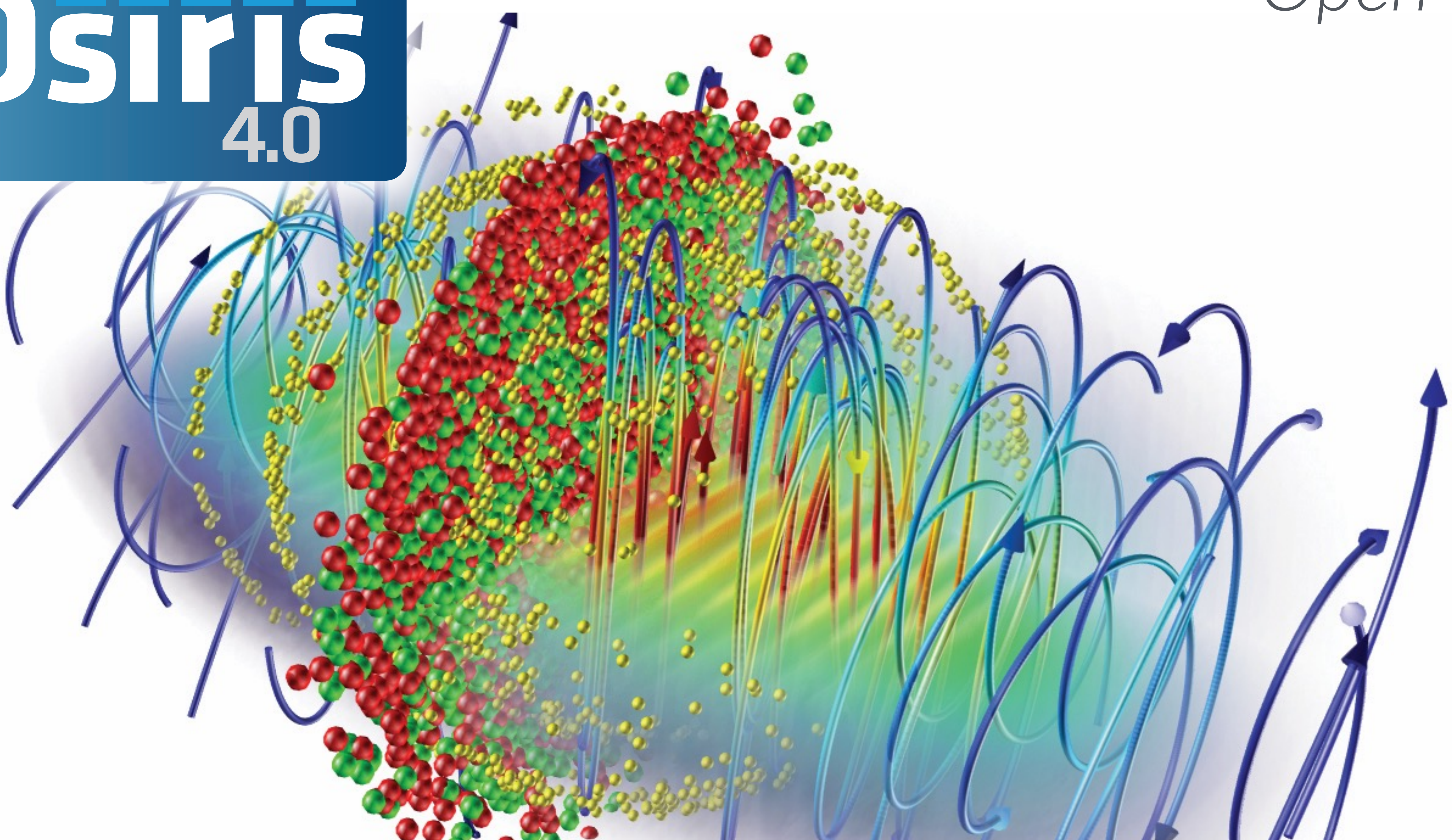
- The different implementations and techniques are critical to ensure reliability
- Collaboration has been instrumental in the developments so far

ALaDyn
CALDER
EPOCH
FBPIC
HiPACE
INF&RNO
Osiris 4.0
PICADOR
PIConGPU
PSC
QuickPIC
Smilei
turboWAVE
UPIC-EMMA
VLPL
VORPAL / VSIM
VPIC
WarpX

* 🙁 Sorry if your favorite code is not listed

Osiris 4.0

Open-source version available



Open-access model

- 40+ research groups worldwide are using OSIRIS
- 400+ publications in leading scientific journals
- Large developer and user community
- Detailed documentation and sample inputs files available
- Support for education and training

Using OSIRIS 4.0

- The code can be used freely by research institutions after signing an MoU
- Open-source version at:

https://osiris-code.github.io/

OSIRIS framework

- Massively Parallel, Fully Relativistic Particle-in-Cell Code
- Support for advanced CPU / GPU architectures
- Extended physics/simulation models
- AI/ML surrogate models and data-driven discovery

visXD

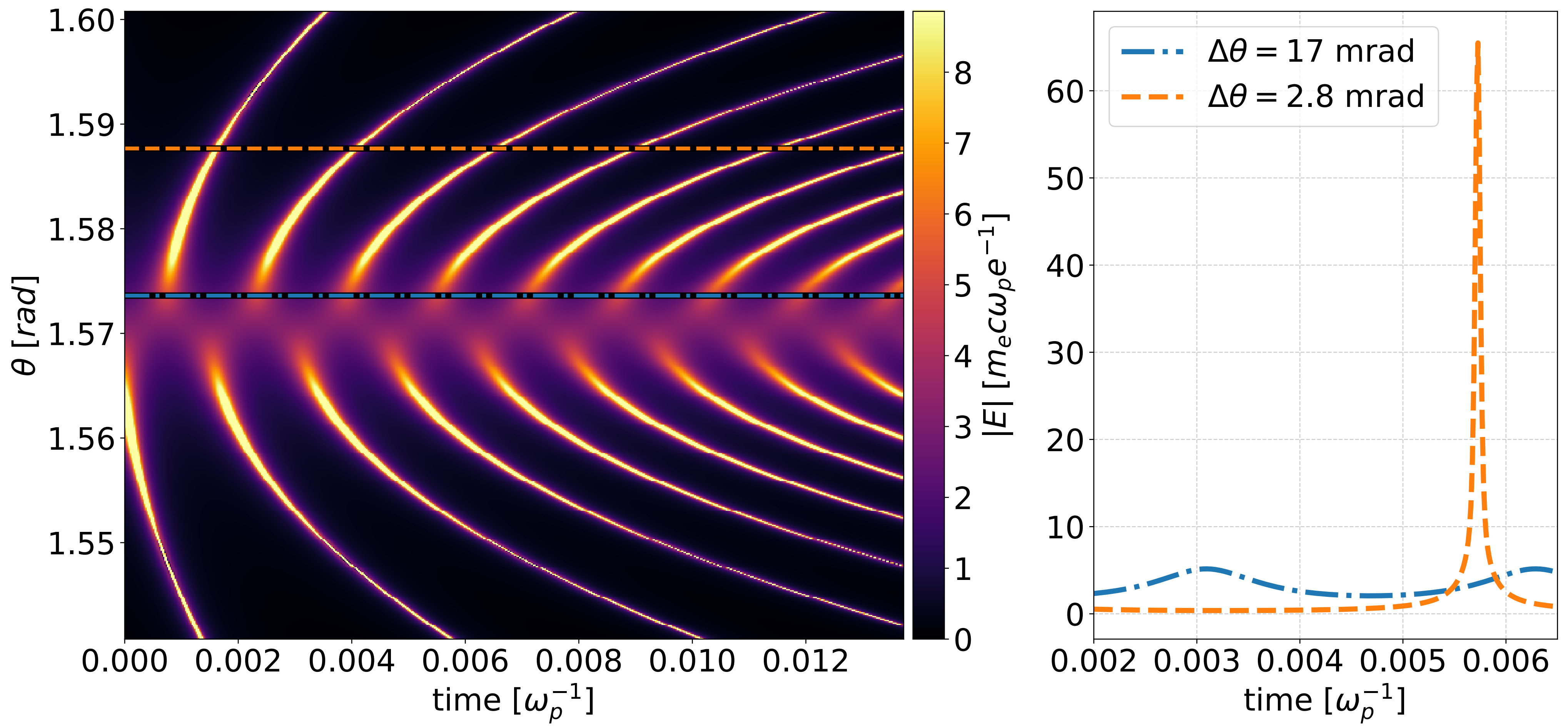
UNIVERSITY OF MICHIGAN · UCLA · TÉCNICO LISBOA

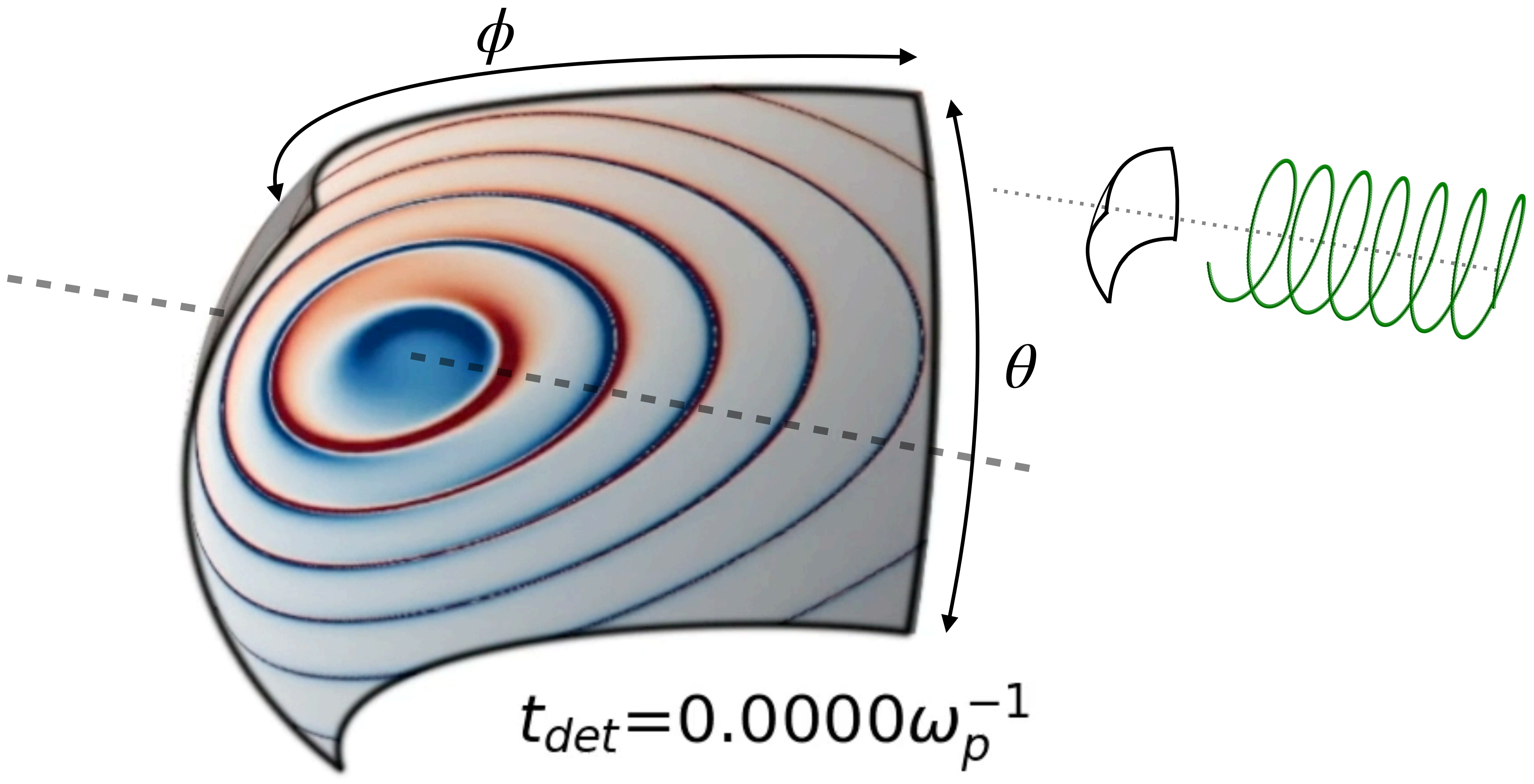
Ricardo Fonseca: ricardo.fonseca@tecnico.ulisboa.pt

Extended physics / simulation models & diagnostics

TÉCNICO LISBOA

Radiation from 3D Helical motion*





*M. Pardal et. al. **Comp. Phys. Comms.** 285, 108634 (2023)

Extending the PIC algorithm

- Field / impact ionization
- Classical radiation damping
- Particle merging
- **QED package** (pair creation, radiation reaction, non-linear Compton scattering)
- **Customizable EM field solver**
- **Alternative particle pushers**

Supported geometries and Reduced models

- 1D, 2D, 3D cartesian and 2D cylindrical
- **Quasi-3D with azimuthal mode expansion**
- Ponderomotive guiding center (PGC) in 2D (x-y), 2D (r-z) and 3D
- Modified Spherical coordinates
- 2D shearing co-rotating framework

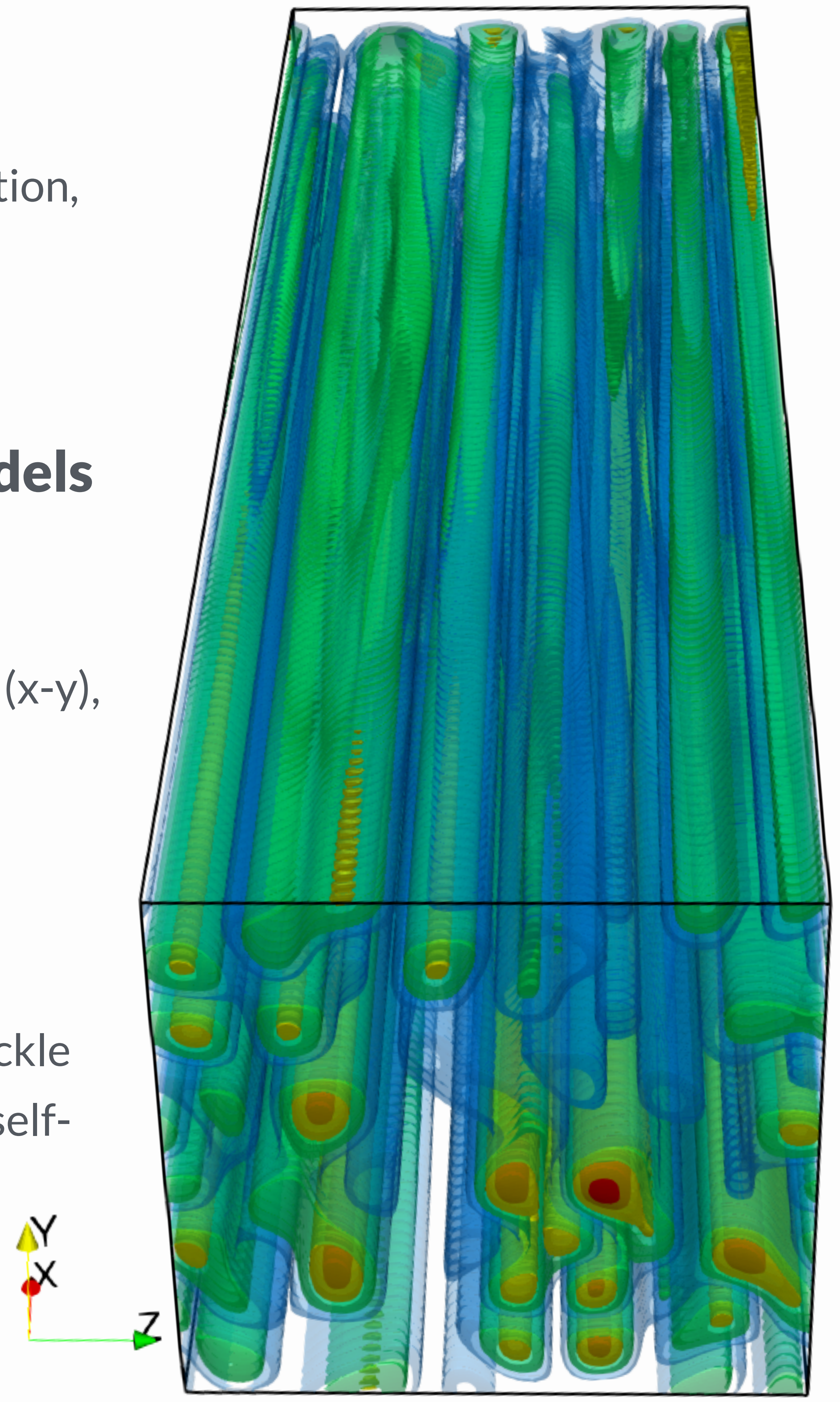
Advanced initialization routines

- Laser Pulses with OAM / sliding focus / speckle
- Particle beams with Twiss parameters and self-consistent initial fields

Advanced diagnostics

- Particle tracking
- Short-wavelength radiation

Speckle module



QED effects for high intensity laser-plasma interactions

TÉCNICO LISBOA

Nonlinear Compton scattering

$$\chi_e = \frac{\sqrt{\left(p_\mu F^{\mu\nu}\right)^2}}{E_s mc} \quad \chi_\gamma = \frac{\sqrt{\left(\hbar k_m u F^{\mu\nu}\right)^2}}{E_s mc} \qquad \frac{d^2\mathcal{P}}{dt\, d\chi_\gamma} = \frac{\alpha m c^2}{\sqrt{3}\pi\hbar\gamma\chi_e}\left[\left(1-\xi+\frac{1}{1-\xi}\right)K_{2/3}\left(\tilde{\chi}\right) - \int_{\tilde{\chi}}^{\infty} dx\; K_{1/3}(x)\right]$$

$$\frac{d\mathbf{p}}{dt} = q\left(\mathbf{E} + \frac{\mathbf{v}\times\mathbf{B}}{c}\right) + \frac{d\mathcal{P}_\gamma}{dt\; d\chi}$$

Probabilistic

Breit–Wheeler process

Integration of equations of motion: moving particles
$\mathbf{F}_p \rightarrow \mathbf{u}_p \rightarrow \mathbf{x}_p$

Emission of photons
Probability of pair creation
➡ new particles

Interpolation: evaluating force on particles
$(\mathbf{E},\mathbf{B})_i \rightarrow \mathbf{F}_p$

Δt

Deposition: calculating current on grid
$(\mathbf{x},\mathbf{u})_p \rightarrow \mathbf{j}_i$

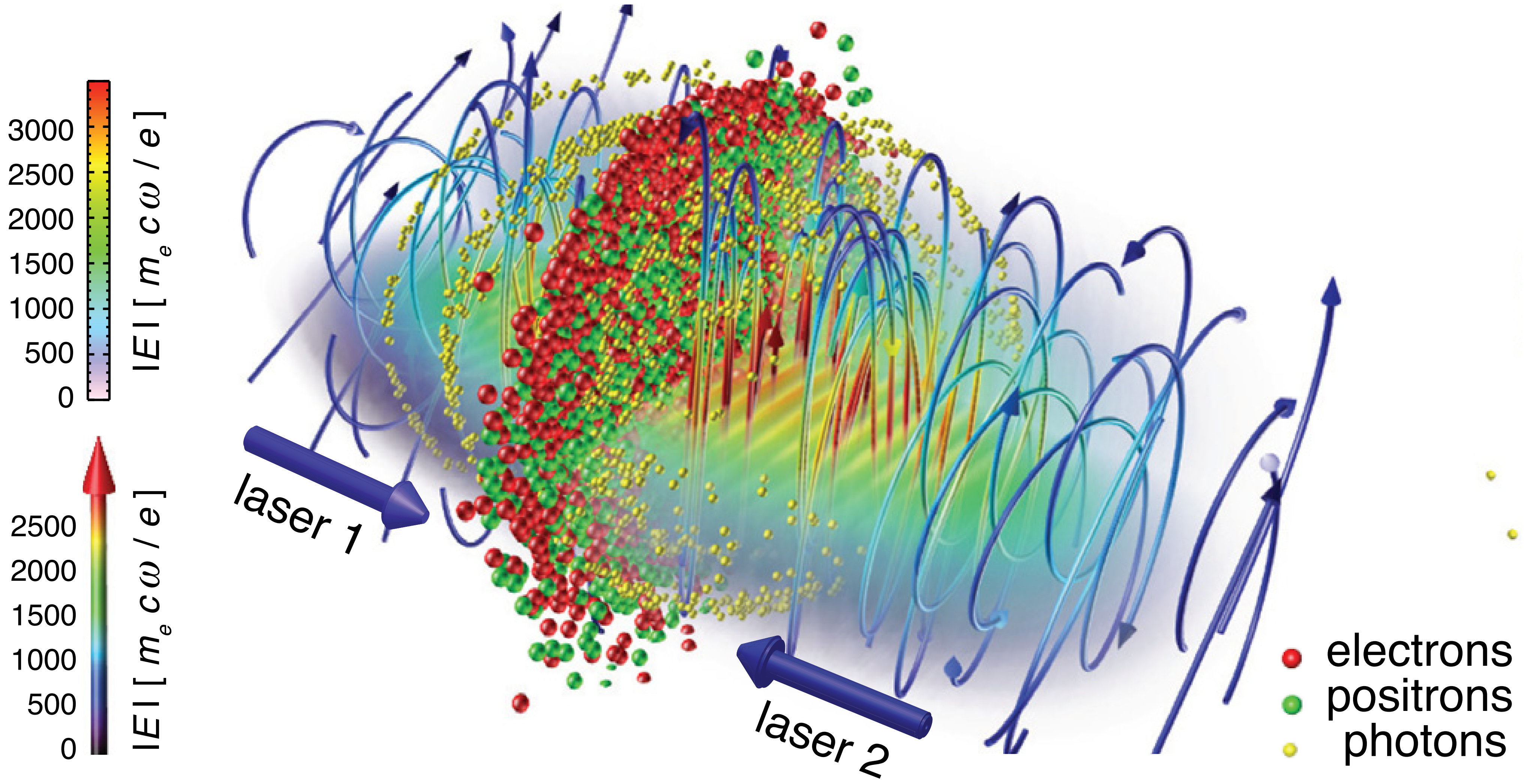
Particle Merging

Integration of field equations: updating fields
$(\mathbf{E},\mathbf{B})_i \leftarrow \mathbf{J}_i$

$$\frac{\partial \mathbf{E}}{\partial t} = c\nabla\times\mathbf{B} - 4\pi\mathbf{j} \qquad \frac{\partial \mathbf{B}}{\partial t} = -c\nabla\times\mathbf{E}$$

Simulation Parameters

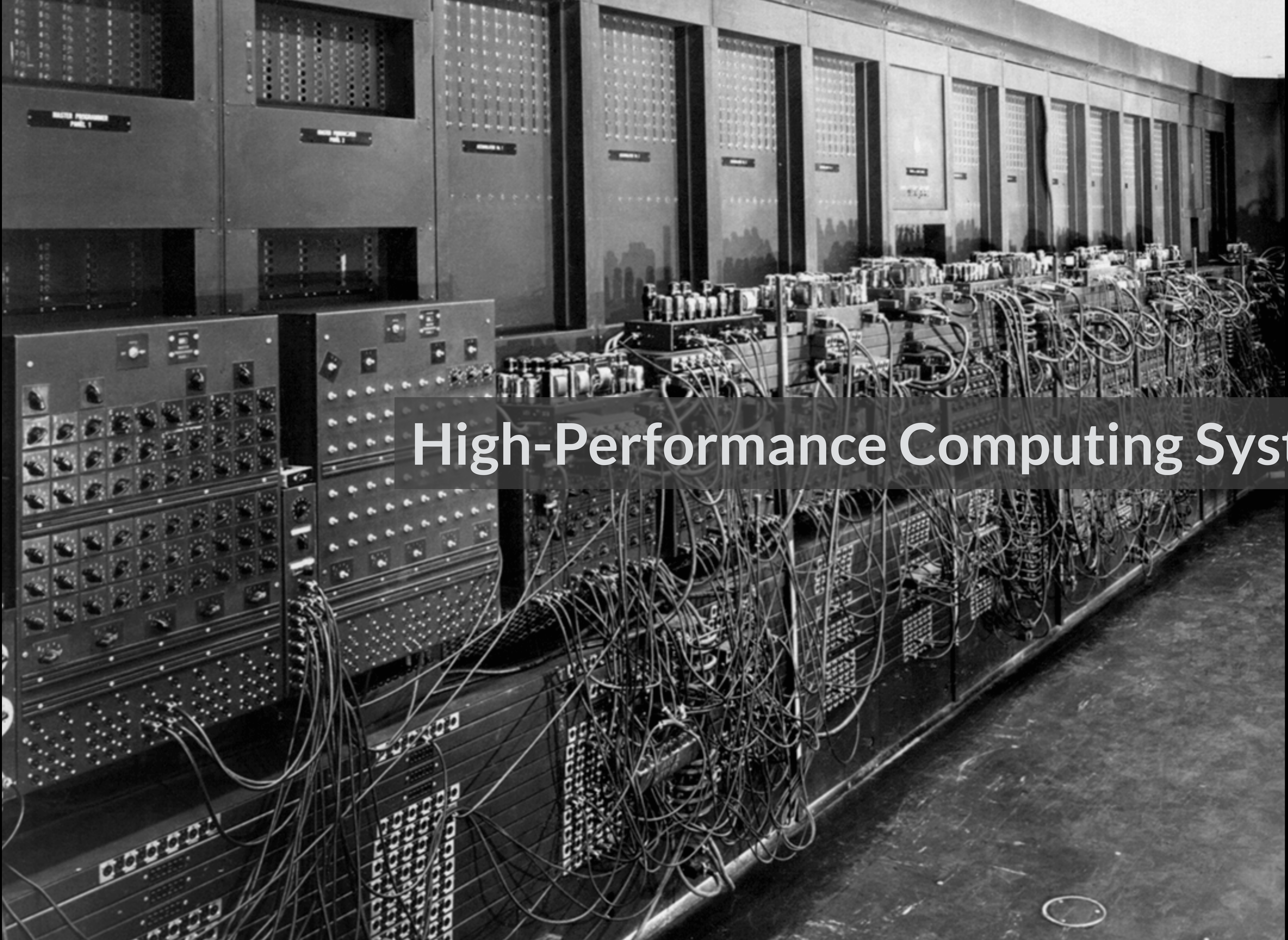
- 2 counter propagating laser pulses
  - λ_0 = 1 µm
  - a_0 = 2000
  - τ_L = 30 fs
  - W_0 = 3.2 µm
- Simulation box
  - 300 × 120 × 120 $(c/\omega_p)^3$
  - 3000 × 1200 × 1200 cells^3



M. Vranic et al., **CPC** (2015); T. Grismayer et al., **POP** (2016); T. Grismayer et al., **PRE** (2017)

High-Performance Computing Systems

TÉCNICO LISBOA



ENIAC - 1946

Evolution of Computing power

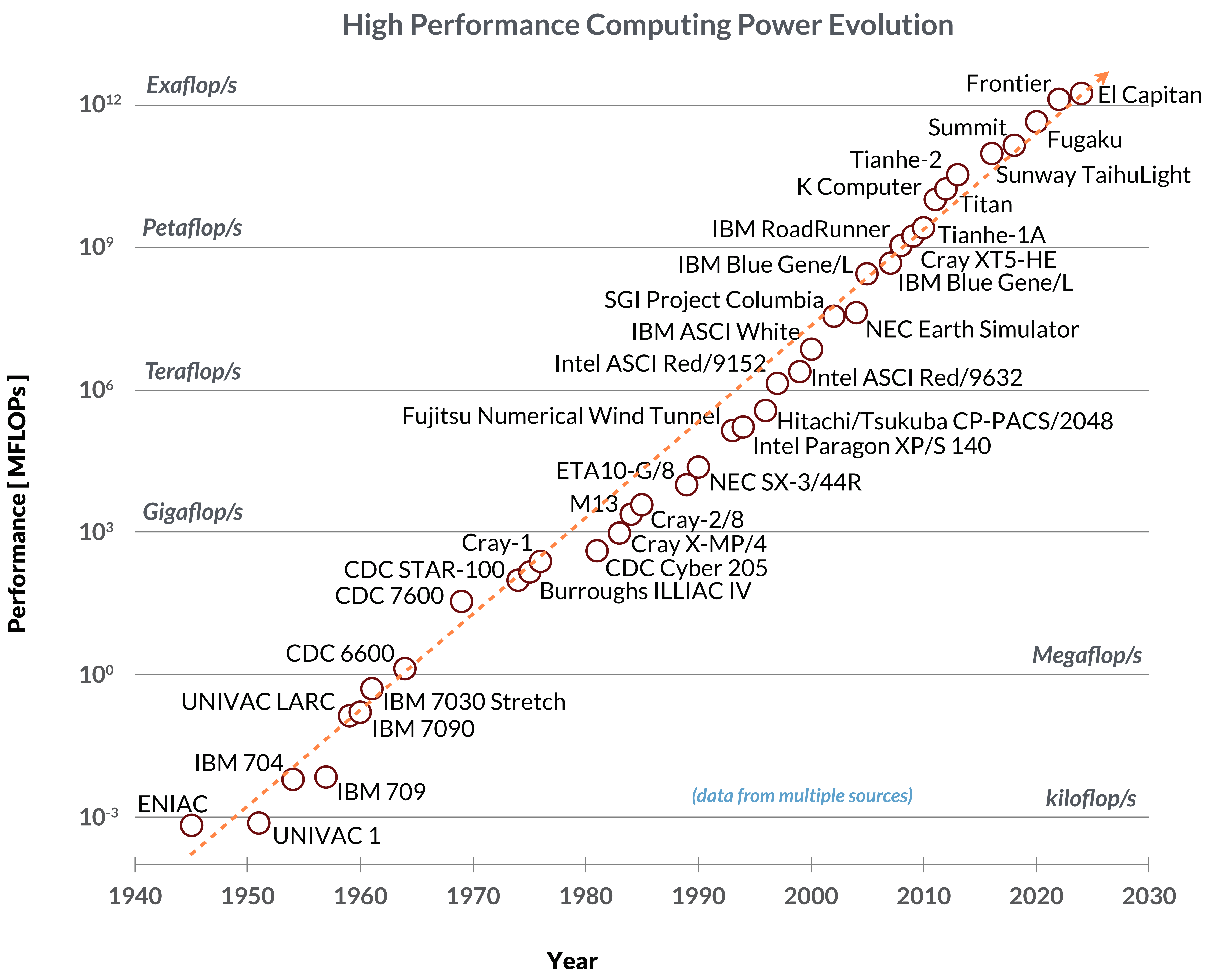
TÉCNICO LISBOA

To Exaflop and beyond

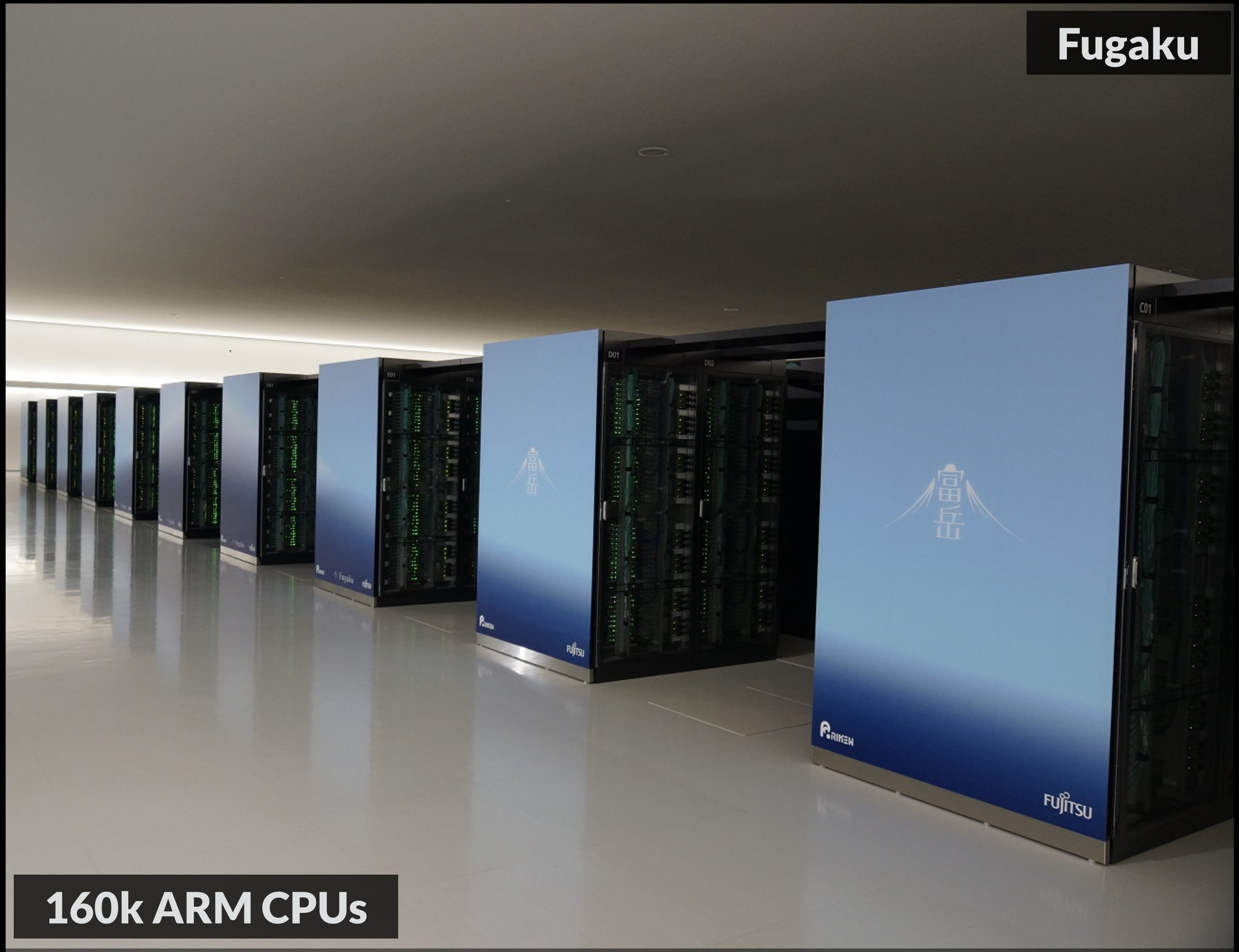
- Steady progress for over 75 years
- 15 orders of magnitude improvement since 1949
- Top system performs at 1.74 EFlop/s, other exaflop systems being deployed

Supported by many computing paradigm evolutions

- From electronic relays to vacuum tubes, transistors and integrated circuits
- Memory went from mercury delay lines to electrostatic vacuum tubes, magnetic storage and solid state RAM cells
- System architecture went from scalar, to super-scalar, to vector systems, massively parallel systems, and accelerator boards



Modern HPC systems - CPU vs GPU



RIKEN Fugaku

Fujitsu

A64FX 48c 2.2 GHz

#1 - TOP500 Nov/21

R_{max} 0.44 EFlop/s

LLNL El Capitan

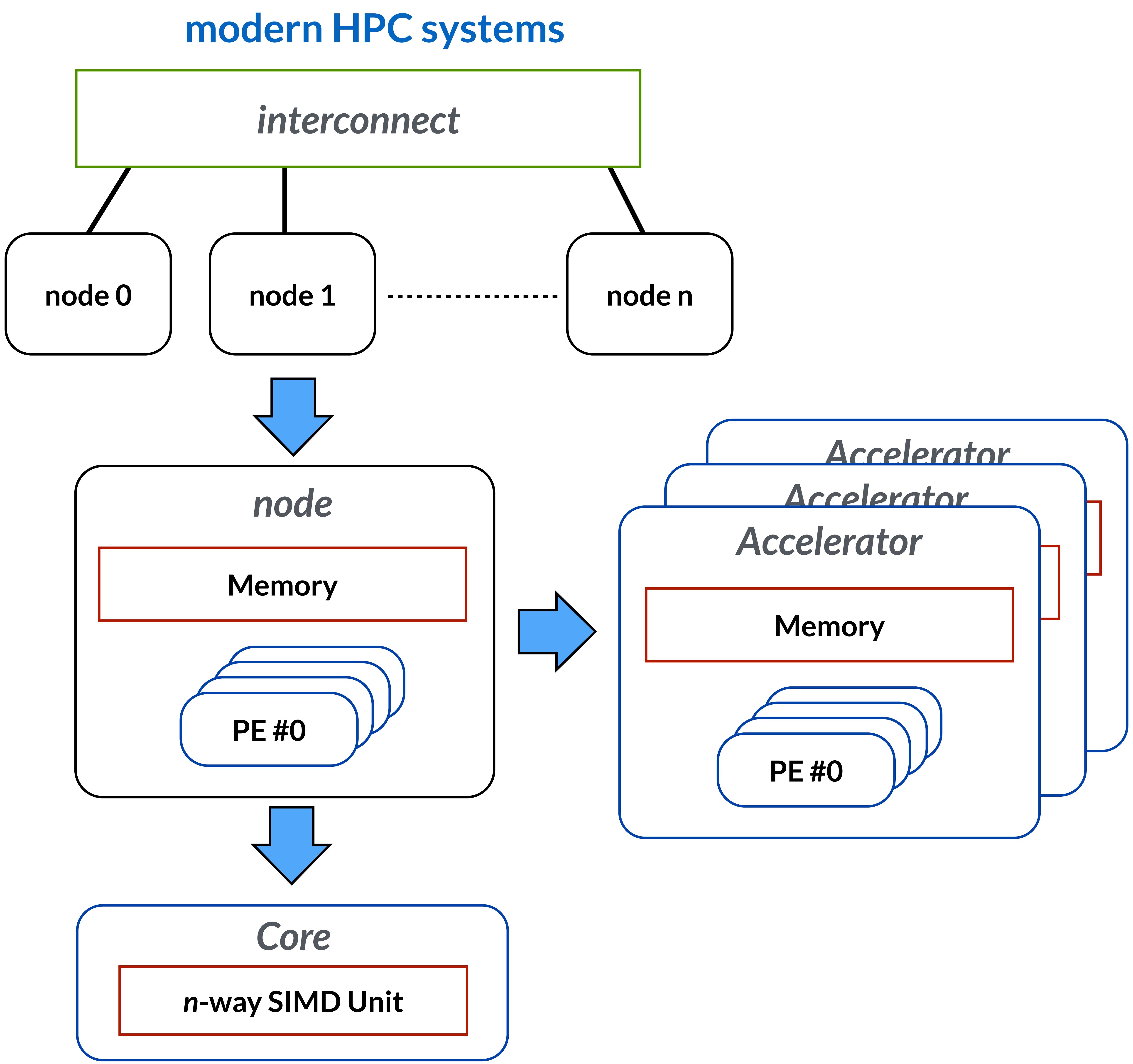
HPE Cray

AMD Instinct MI300A

#1 - TOP500 Nov/24

R_{max} 1.74 EFlop/s

High Performance Computing Systems

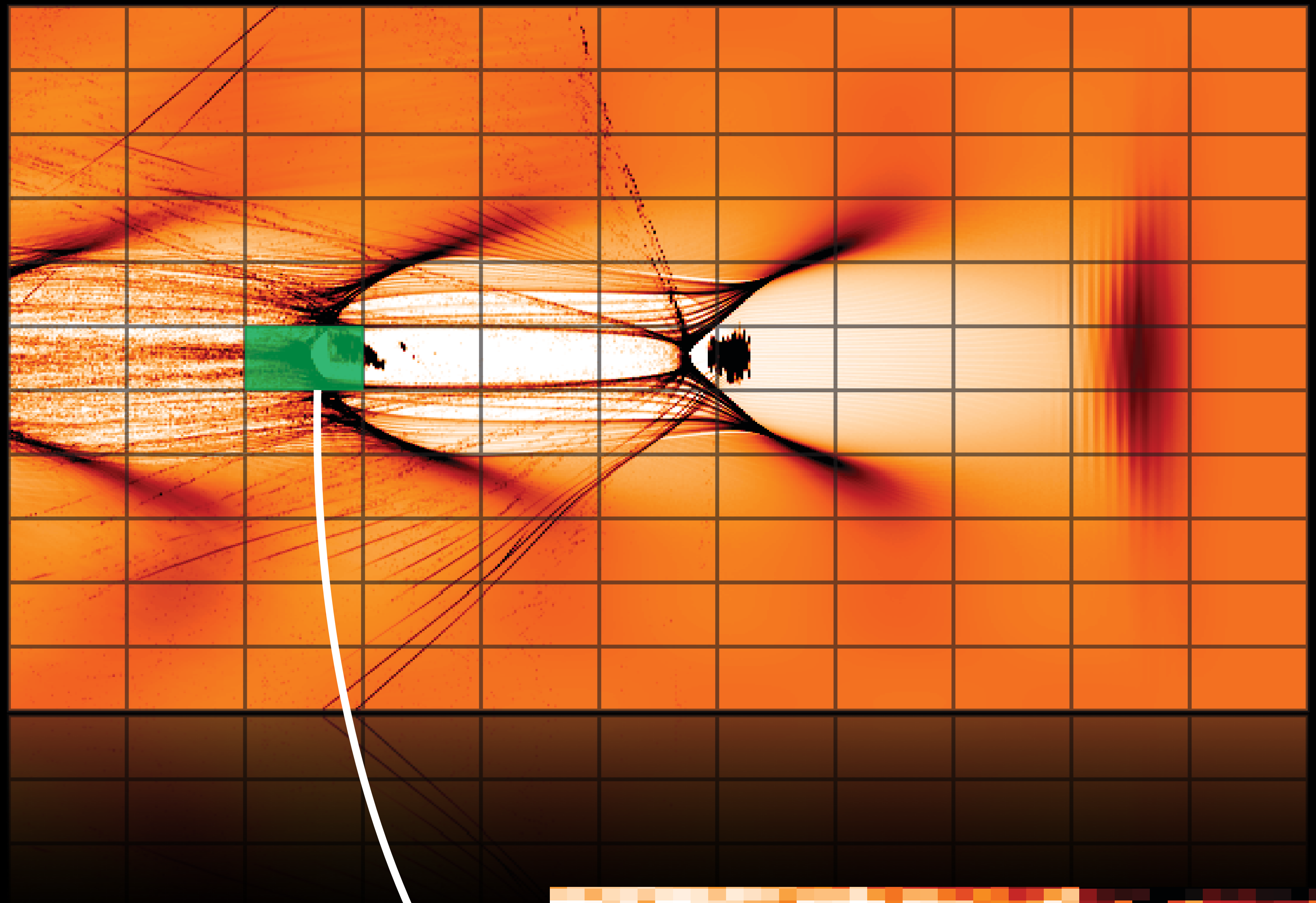


- **Modern HPC systems share a common anatomy, presenting a hierarchy of parallelism**
  - Optimal performance requires exploiting all levels of parallelism and performance hardware
  - Shared anatomy means reusable strategies, but matching your algorithm to the hardware gives best performance
  - Efficient deployment on a wide array of systems requires a flexible approach, combining code reusability and specialization.

- **OSIRIS development strategy focuses on optimal performance targeting current and future systems**
  - Address large scale parallelism issues, in particular focusing on parallel load balance
  - Optimize for specialized hardware, such SIMD units on modern CPUs, or GPGPU accelerators
  - Explore new and emerging programming models and paradigms, ensure readiness for future systems

Micro-spatial domain decomposition (tiles)

- **Divide simulation data over small spatial domains**
  - Domains can be as small as ~ 10 - 100 cells
  - Each domain (tile) holds all particle and grid data for that spatial region
- **Assign tiles to computing elements (CPU core, GPU, etc.)**
  - Multiple tiles may be assigned to the same computing element
  - Tiles may be reassigned dynamically to ensure an even computational load
- **Computation in each tile is (mostly) independent**
  - All tiles may be processed in parallel
  - After advancing the particles these may need to be assigned to different tiles
  - After advancing the fields edge values must be updated from neighboring tiles

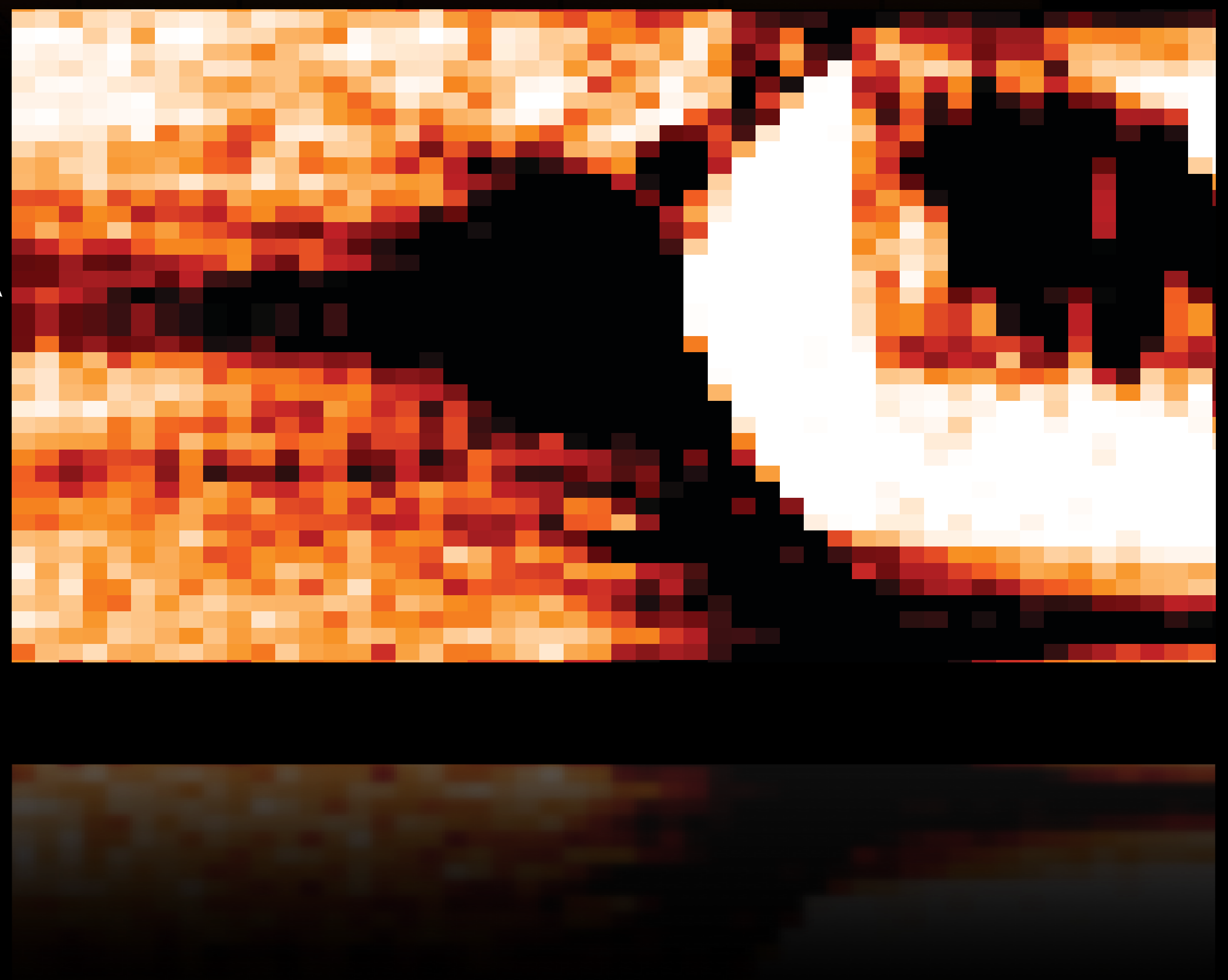


Simulation data

- Particles
- Grids
- Organized by tiles

Each tile is processed in parallel

- Most computations strictly local



Tile data

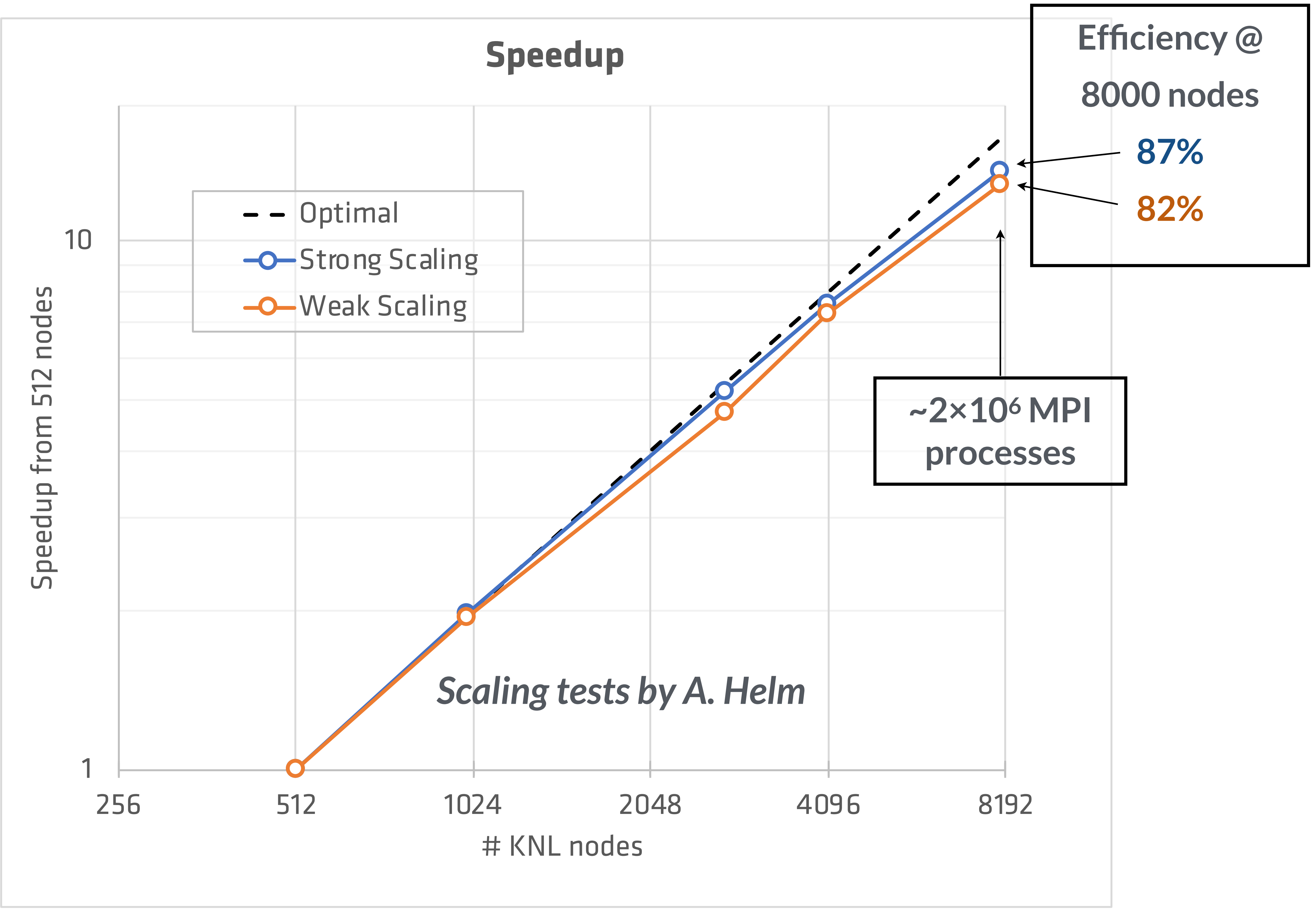
- Particles
- Grids

Inter-node communication

- **Current HPC systems are a collection of independent nodes**
  - These nodes are connected using a high-speed computer network
- **Tiles may be neighbours to other tiles on different nodes**
  - Grid and particle data must be communicated between different nodes

- **Inter-node communication is done using the Message Passing Interface (MPI)**
  - Standard since late 1990's
  - Explicit parallelization:
    - programmer must write MPI instructions for sending / receiving messages between nodes
  - Works very well also inside computing nodes
    - Benefits from shared memory
    - Message passing inside a node is very efficient
  - Very efficient for uniform plasmas

- **Efficient even for extreme process counts**
  - Parallel scaling demonstrated up to ~ 2 M processes



Full-scale modeling of the AWAKE* experiment

TÉCNICO LISBOA

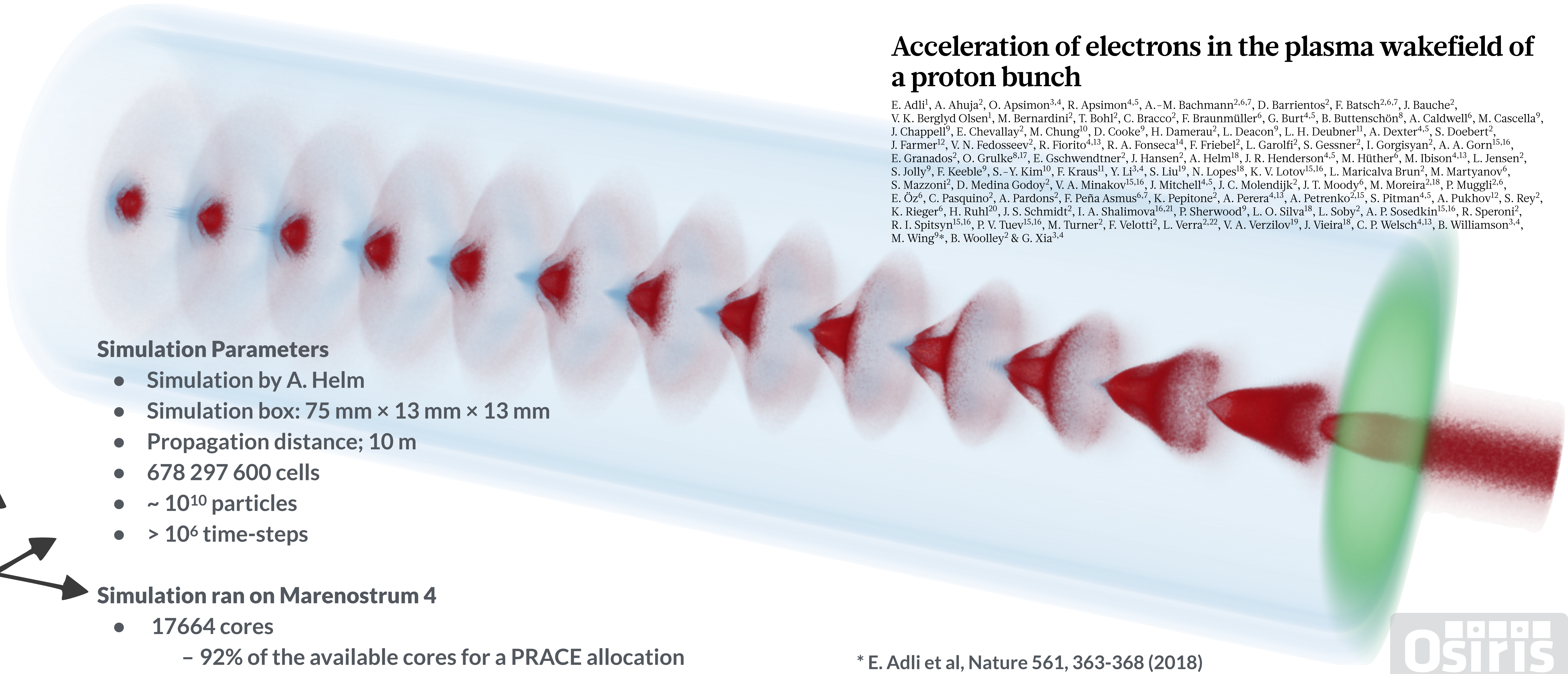
LETTER

OPEN

https://doi.org/10.1038/s41586-018-0485-4

Acceleration of electrons in the plasma wakefield of a proton bunch

E. Adli[1], A. Ahuja[2], O. Apsimon[3,4], R. Apsimon[4,5], A.-M. Bachmann[2,6,7], D. Barrientos[2], F. Batsch[2,6,7], J. Bauche[2], V. K. Berglyd Olsen[1], M. Bernardini[2], T. Bohl[2], C. Bracco[2], F. Braunmüller[6], G. Burt[4,5], B. Buttenschön[8], A. Caldwell[6], M. Cascella[9], J. Chappell[9], E. Chevallay[2], M. Chung[10], D. Cooke[9], H. Damerau[2], L. Deacon[9], L. H. Deubner[11], A. Dexter[4,5], S. Doebert[2], J. Farmer[12], V. N. Fedosseev[2], R. Fiorito[4,13], R. A. Fonseca[14], F. Friebel[2], L. Garolfi[2], S. Gessner[2], I. Gorgisyan[2], A. A. Gorn[15,16], E. Granados[2], O. Grulke[8,17], E. Gschwendtner[2], J. Hansen[2], A. Helm[18], J. R. Henderson[4,5], M. Hüther[6], M. Ibison[4,13], L. Jensen[2], S. Jolly[9], F. Keeble[9], S.-Y. Kim[10], F. Kraus[11], Y. Li[3,4], S. Liu[19], N. Lopes[18], K. V. Lotov[15,16], L. Maricalva Brun[2], M. Martyanov[6], S. Mazzoni[2], D. Medina Godoy[2], V. A. Minakov[15,16], J. Mitchell[4,5], J. C. Molendijk[2], J. T. Moody[6], M. Moreira[2,18], P. Muggli[2,6], E. Öz[6], C. Pasquino[2], A. Pardons[2], F. Peña Asmus[6,7], K. Pepitone[2], A. Perera[4,13], A. Petrenko[2,15], S. Pitman[4,5], A. Pukhov[12], S. Rey[2], K. Rieger[6], H. Ruhl[20], J. S. Schmidt[2], I. A. Shalimova[16,21], P. Sherwood[9], L. O. Silva[18], L. Soby[2], A. P. Sosedkin[15,16], R. Speroni[2], R. I. Spitsyn[15,16], P. V. Tuev[15,16], M. Turner[2], F. Velotti[2], L. Verra[2,22], V. A. Verzilov[19], J. Vieira[18], C. P. Welsch[4,13], B. Williamson[3,4], M. Wing[9]*, B. Woolley[2] & G. Xia[3,4]



Simulation Parameters

- Simulation by A. Helm
- Simulation box: 75 mm × 13 mm × 13 mm
- Propagation distance; 10 m
- 678 297 600 cells
- ~ 10^{10} particles
- > 10^{6} time-steps

Simulation ran on Marenostrum 4

- 17664 cores
  - – 92% of the available cores for a PRACE allocation
- ~ 3M core×h

* E. Adli et al, Nature 561, 363-368 (2018)

Osiris

AI Opportunities

Cray-2 - 1985

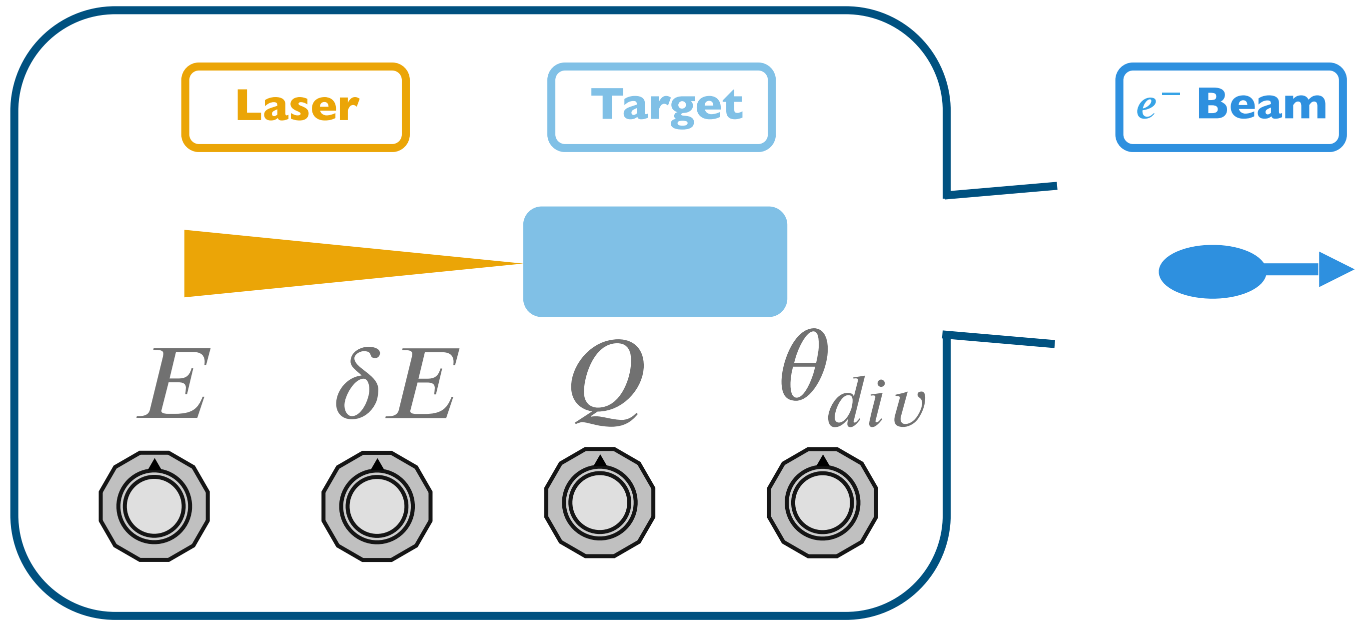
Central unit (left) &

Fluorinet cooling waterfall (right)

Optimizing LWFA targets with Machine Learning*

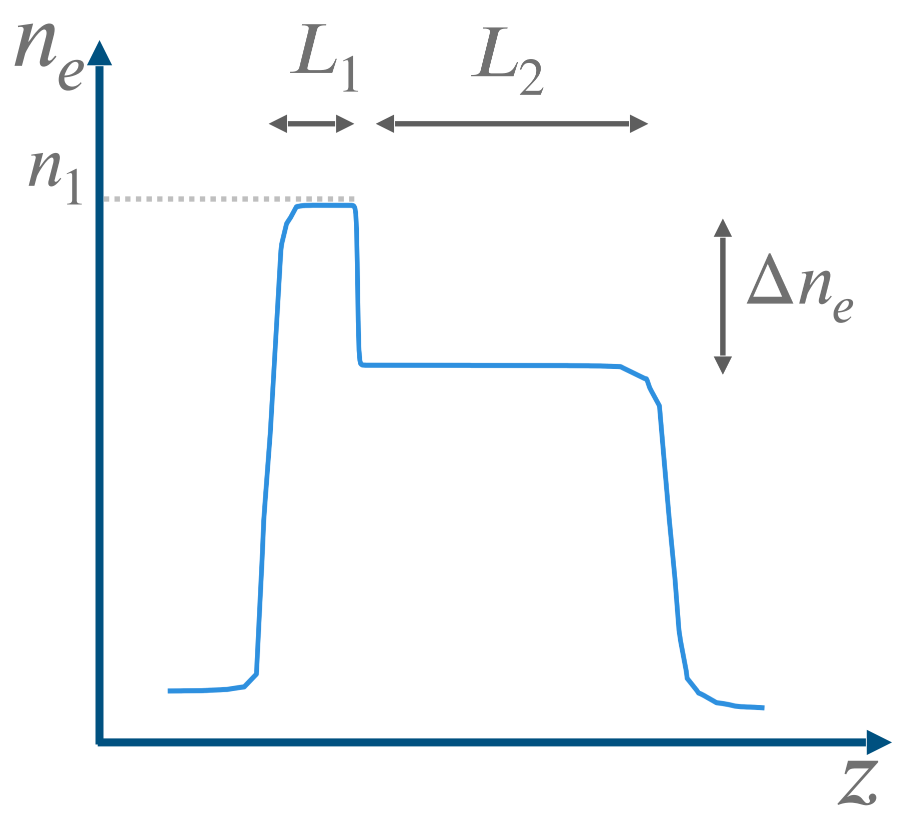
A tunable LWFA target

Goal: electron beams with tunable parameters, compatible with applications

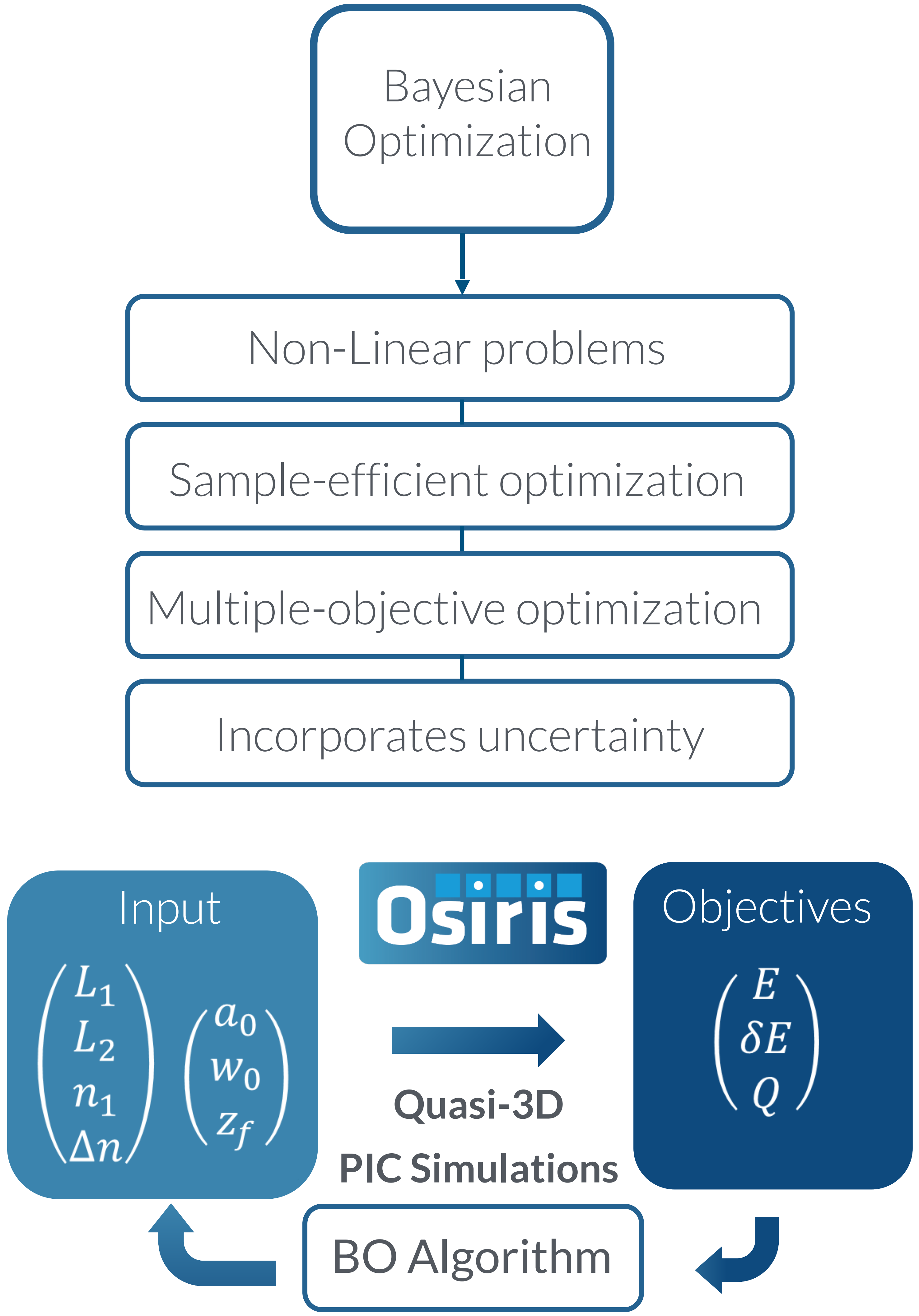


How to achieve this?

Plasma **density down-ramp** injection scheme

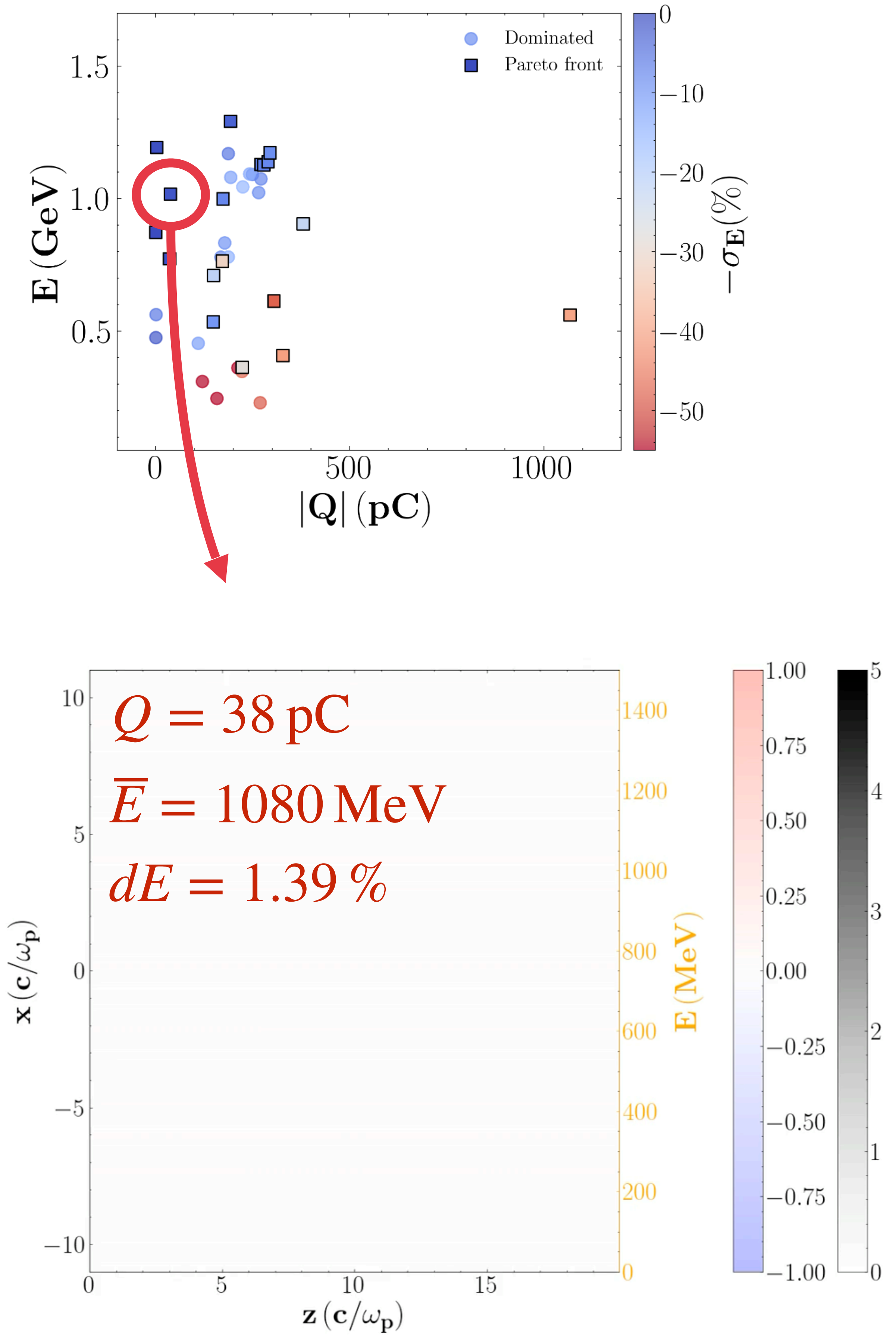


Optimization Framework



Monoenergetic bunches

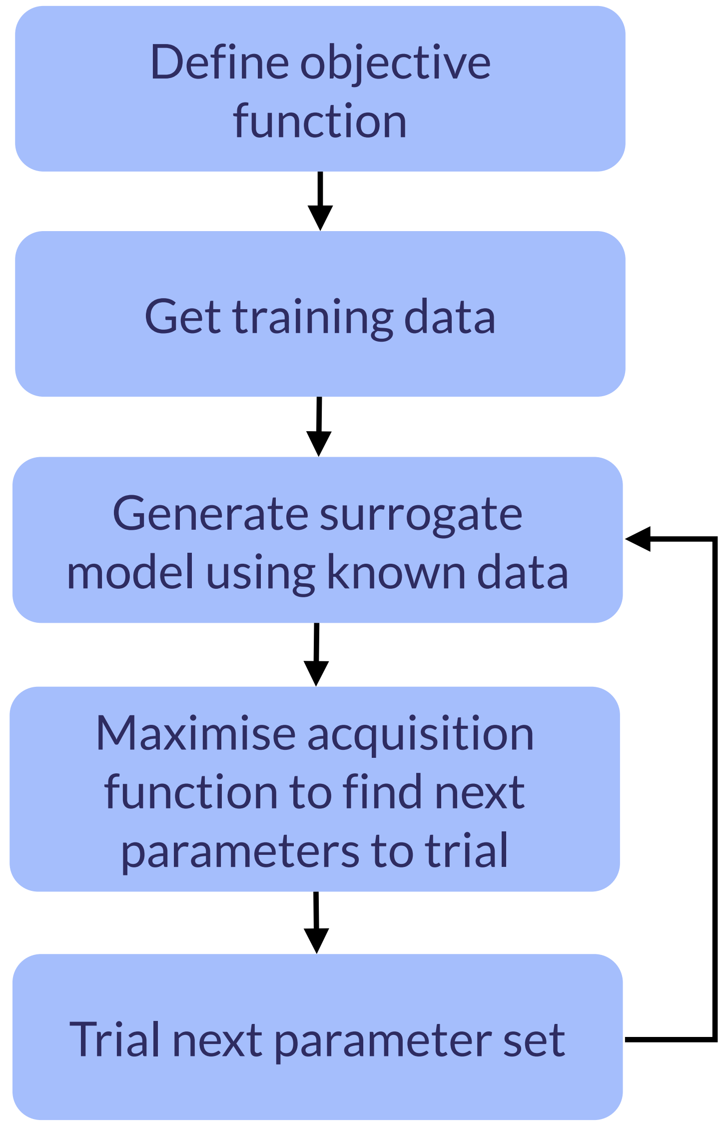
Pareto Front (set of optimal solutions)



* **Work by Diogo Lemos @ GoLP**

Optimization of PIC Simulations for EPAC*

Bayesian Optimization



- **In-house optimization code**
  - Surrogate model using Gaussian Process Regression
  - Implementation using Scikit-Learn and SciPy libraries
- **Other optimization libraries available**
  - e.g Optimas, developed at DESY

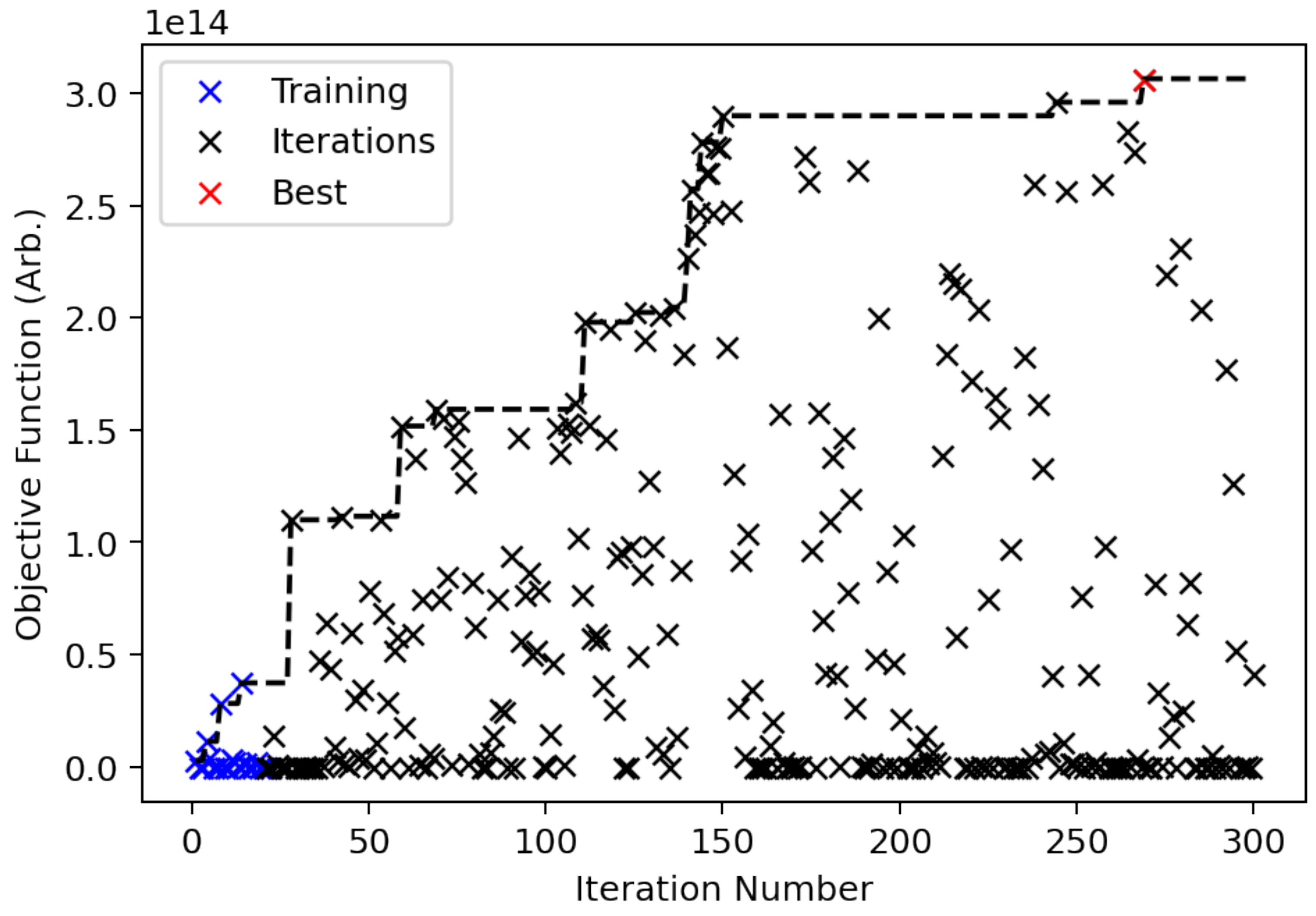
https://github.com/optimas-org/optimas

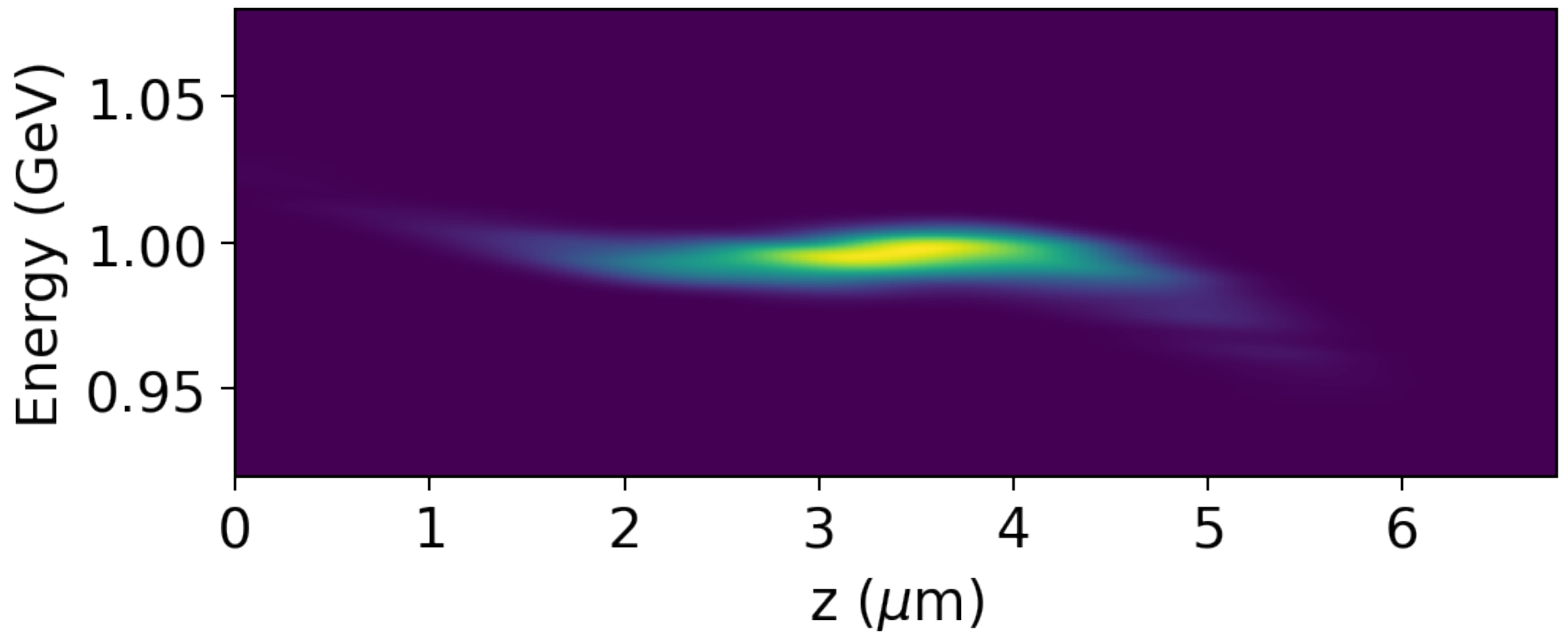
Application to EPAC

- **Parameters scanned were**
  - Laser focal plane.
  - Gas density in injector region.
  - Gas density in accelerator region.
  - Gas dopant percentage.
  - Accelerator length.
- **Objective function prioritized high brightness and proximity to 1 GeV**
- **Output Bunch Parameters**
  - Mean energy = 0.99 GeV
  - Energy spread rms = 0.6 %.
  - Charge = 25 pC.
- **Best setting found after ~ 270 iterations here for large, 5d parameter space.**

FBPIC

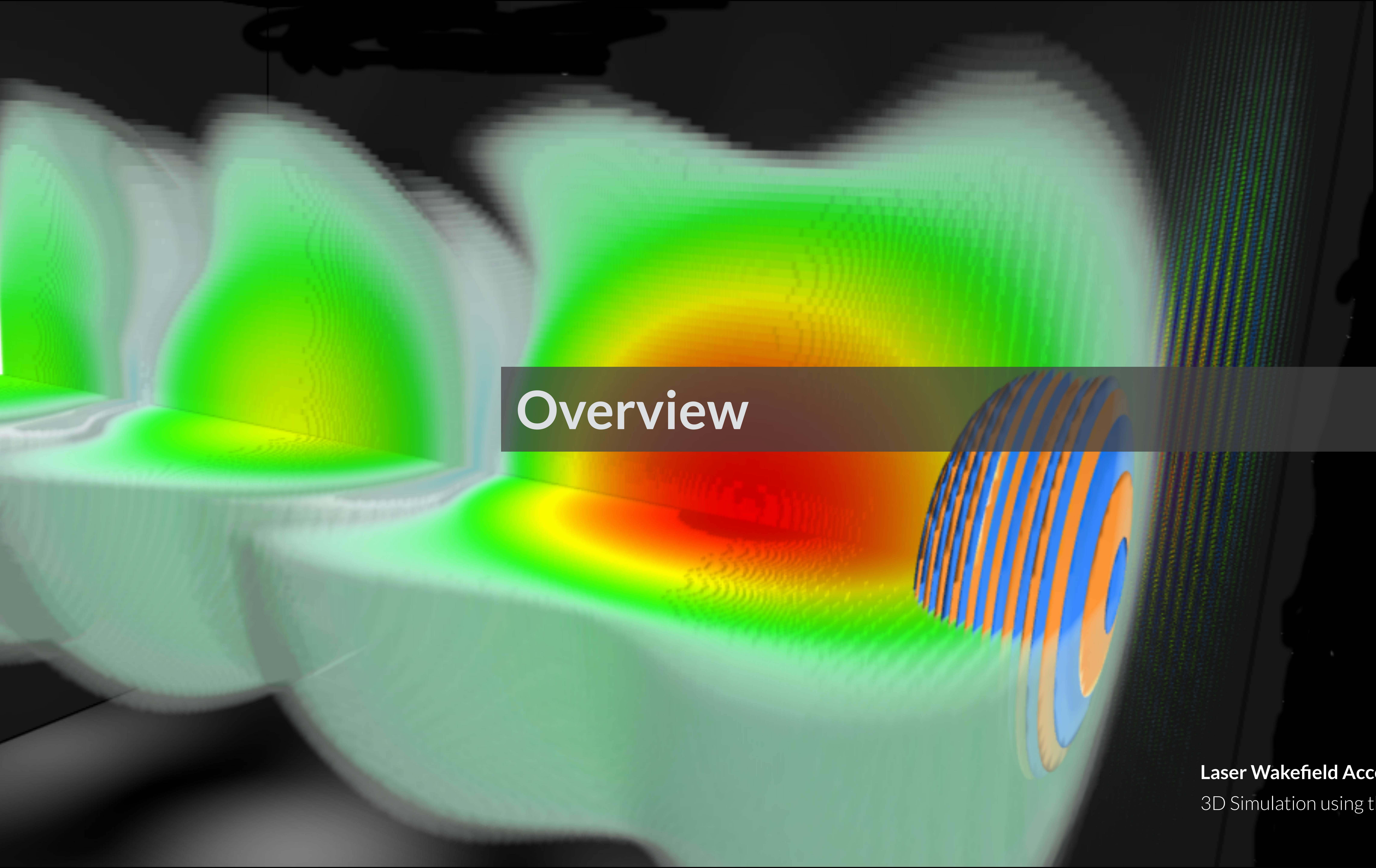
https://fbpic.github.io/





*** Work by Oliver Finlay @ EPAC**

Overview



Laser Wakefield Acceleration

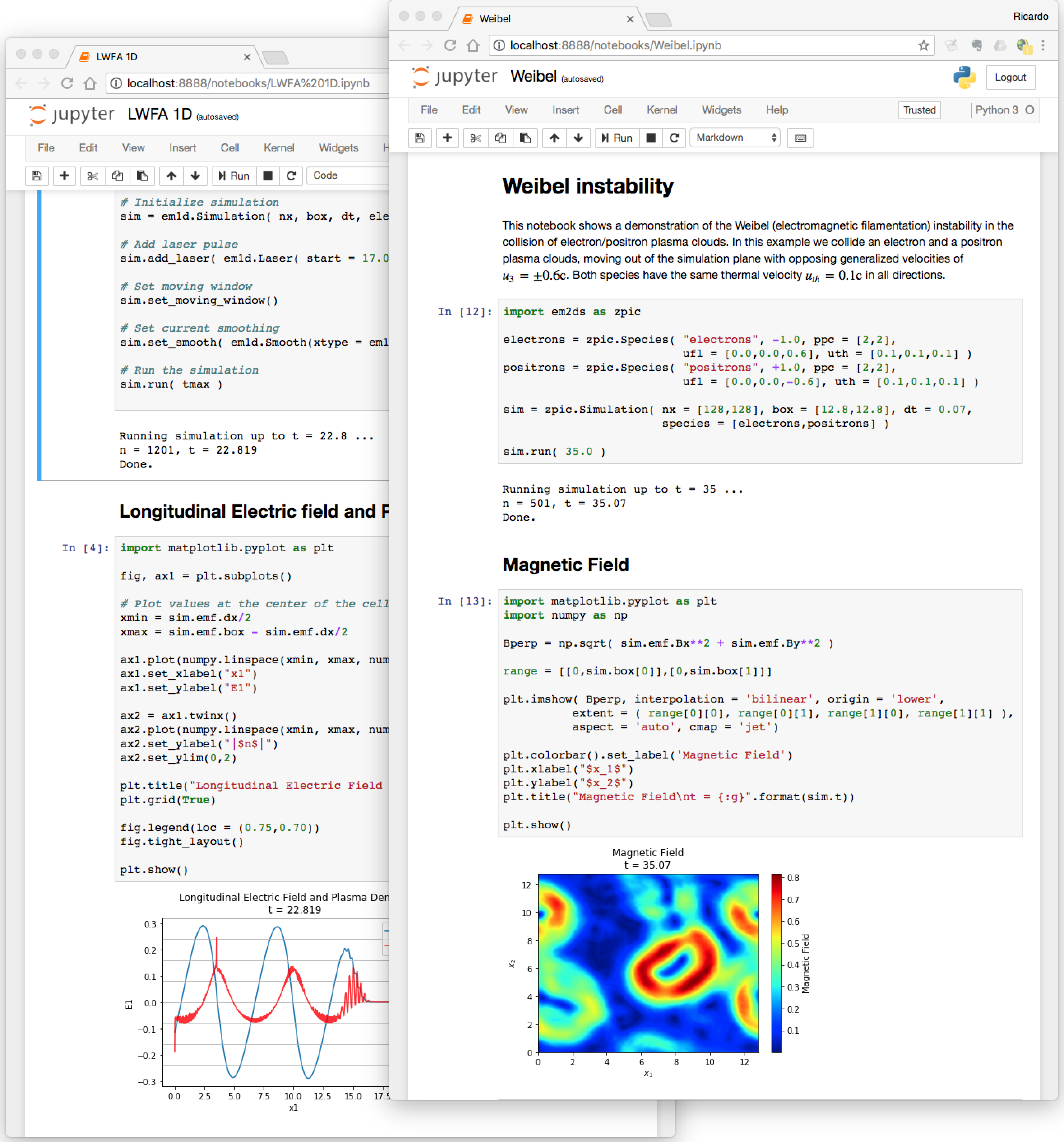
3D Simulation using the OSIRIS code

Overview

- **Simulations played a critical role in the development of the field of plasma based accelerators**
  - In particular, particle-in-cell codes were instrumental in demonstrating the feasibility of this technology
  - Remain as invaluable tools for testing new ideas and concepts, supporting experiments and optimizing designs
- **PIC codes are computationally heavy…**
  - The core algorithm makes very little approximations, leading to high computational costs
  - Reduced models and/or geometries may be used to offset this cost
- **… but, fortunately, exascale simulations are within reach**
  - The existing HPC landscape can offer the required computational power
  - Increasing quality and quantitative fidelity of simulations
  - Continuously evolve algorithms and codes to efficiently use new generations of computing hardware
- **The AI revolution opens exciting prospects**
  - Combining numeric plasma models with AI workflows enables a new generation of optimization and prototyping studies

The zpic educational code suite

- **ZPIC code suite**
  - Open-source PIC code suit for plasma physics education
  - Fully relativistic 1D and 2D EM-PIC algorithm
  - Eletrostatic 1D/2D PIC algorithm
- **Requirements**
  - No external dependencies, requires only C99 compiler
  - Optional Python interface
- **Jupyter Notebooks**
  - Includes set of Python notebooks with example problems
  - Detailed explanations of code use and physics
- **Also available through Docker**
  - If you just want to run the notebooks you can use a Docker image available on DockerHub: **zamb/epic**



zpic@edu

https://zpic-plasma.github.io/