Modelling & Simulation

Hands on I/II

R. A. Fonseca[1,2], Jorge Vieira[1]

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

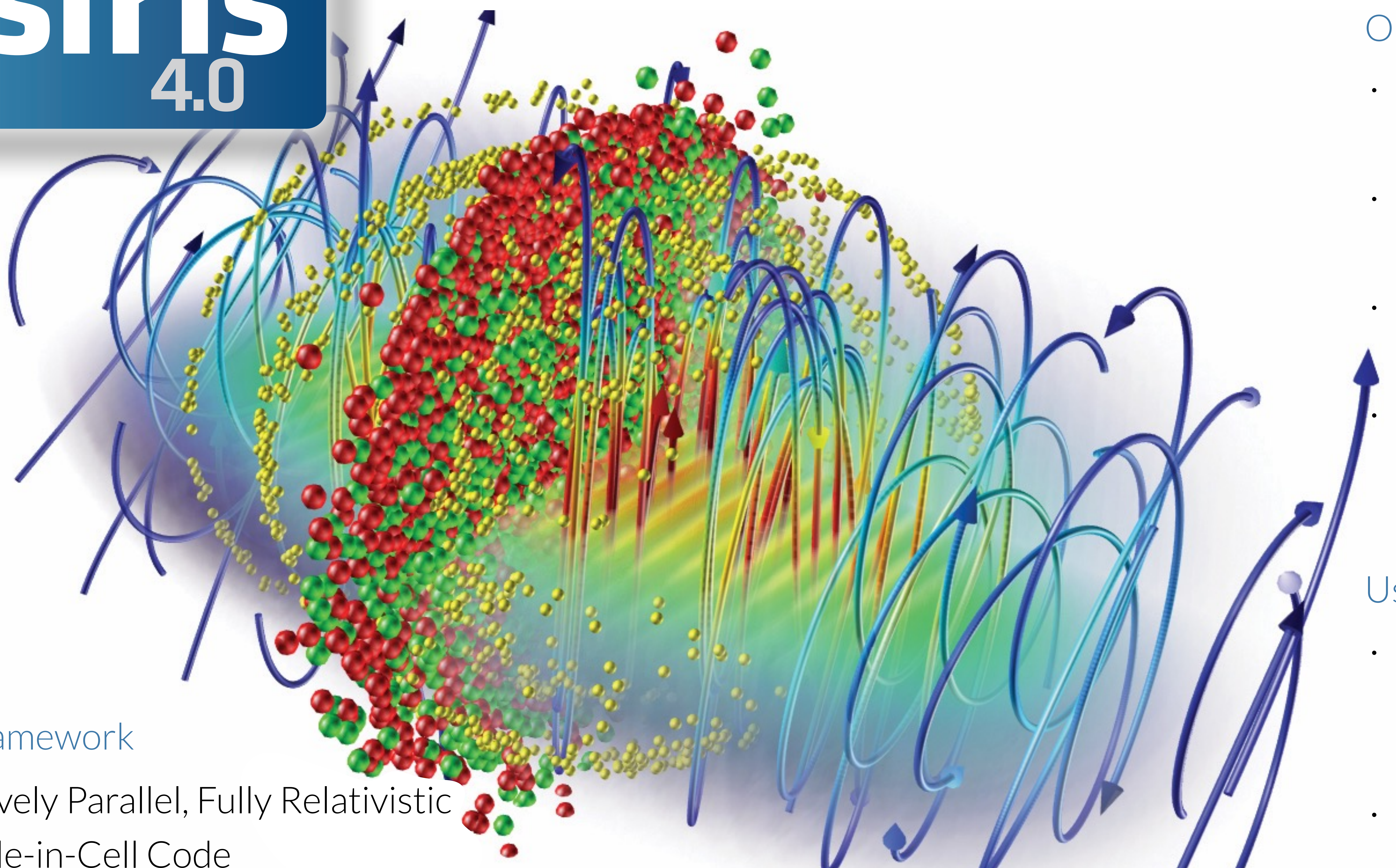
[2] DCTI, ISCTE-Instituto Universitário de Lisboa, Portugal

TÉCNICO LISBOA

ISCTE IUL

Instituto Universitário de Lisboa

Committed to open science



OSIRIS framework

- Massively Parallel, Fully Relativistic Particle-in-Cell Code
- Parallel scalability to 2 M cores
- Explicit SSE / AVX / QPX / Xeon Phi / CUDA support
- Extended physics/simulation models

Open-access model

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

Using OSIRIS 4.0

- The code can be used freely by research institutions after signing an MoU
- Find out more at:

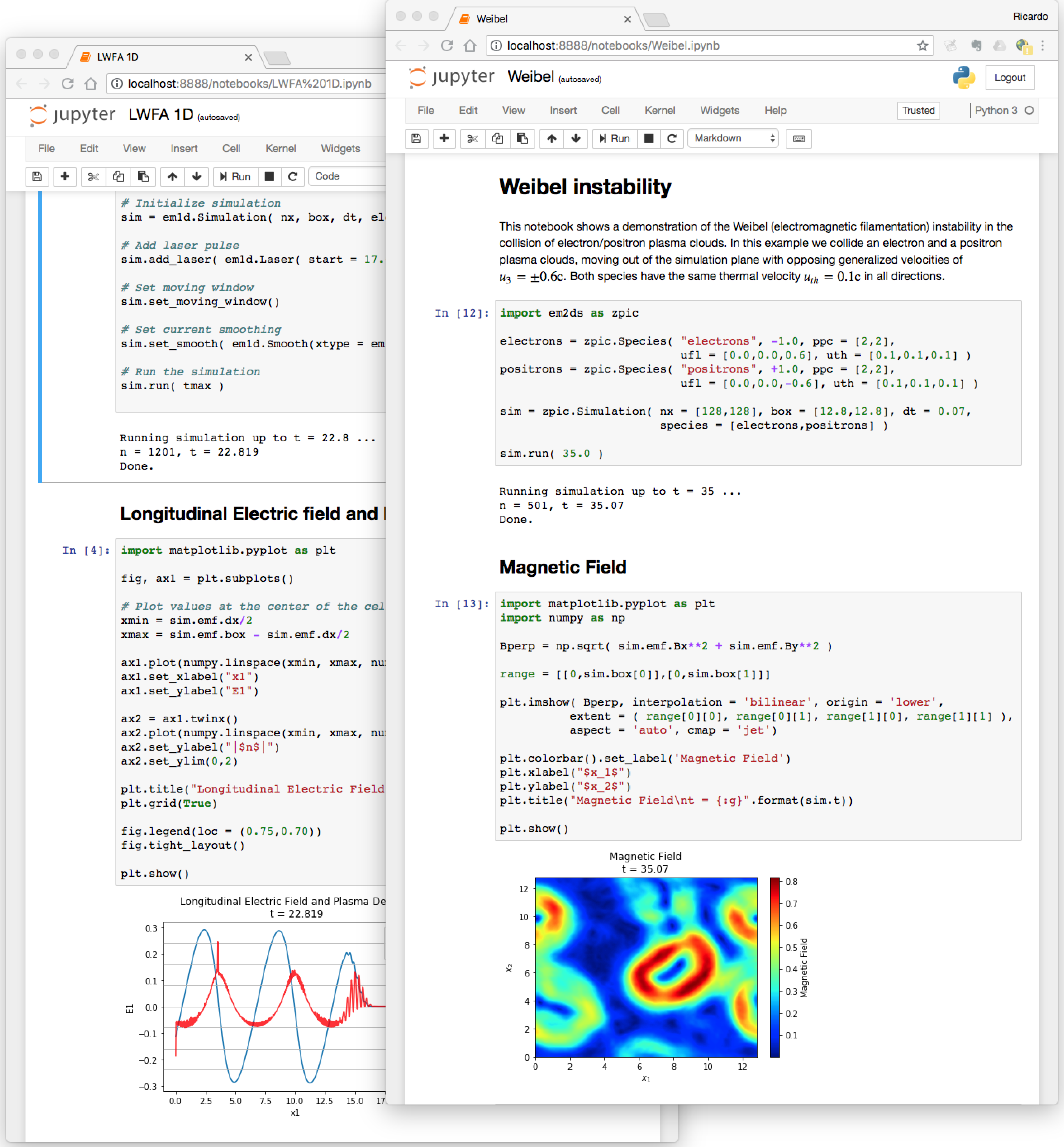
http://epp.tecnico.ulisboa.pt/osiris

UCLA TÉCNICO LISBOA

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

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
- **Jupiter 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/zpic**



zpic@edu

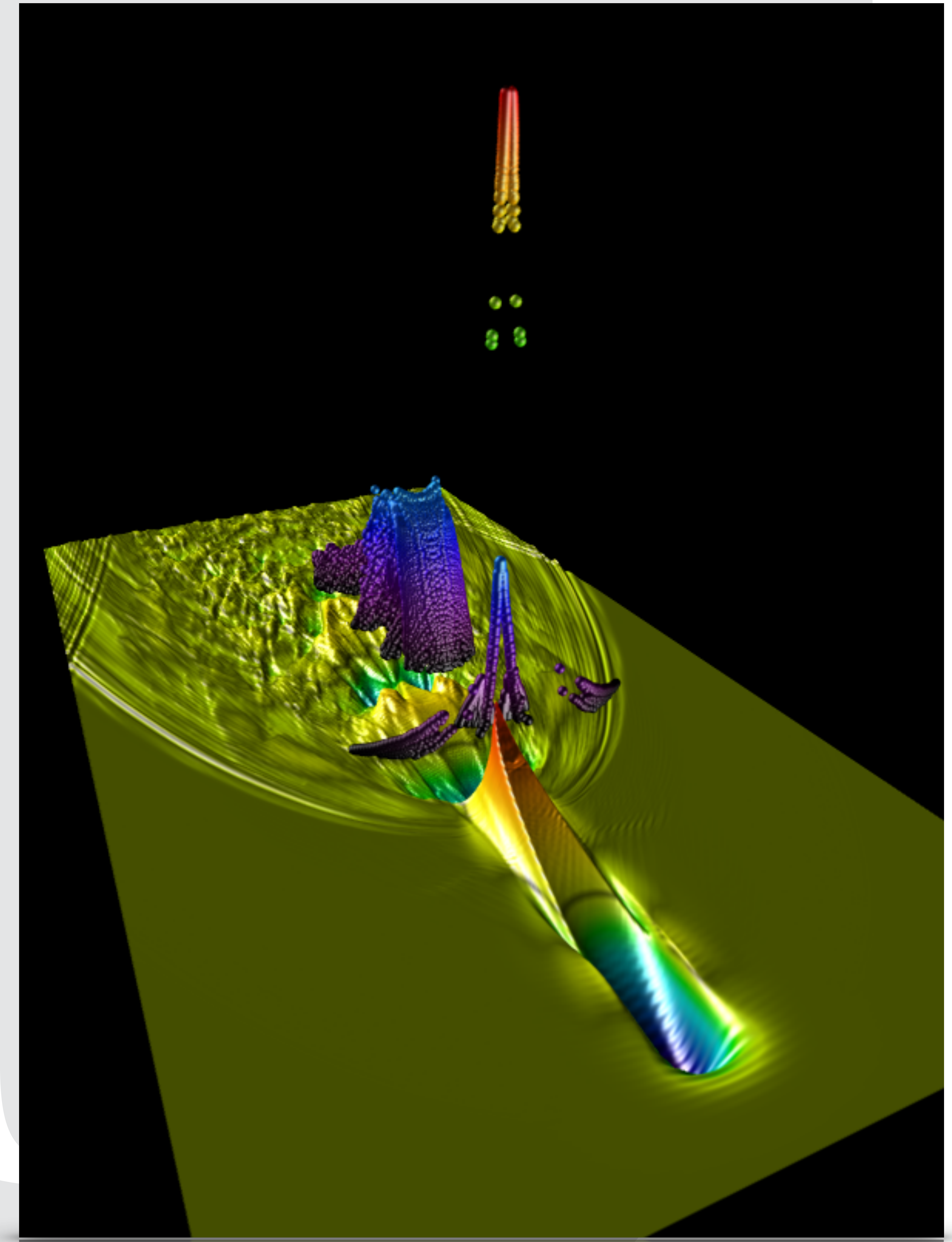
Come find us on GitHub
github.com/zambzamb/zpic

Outline

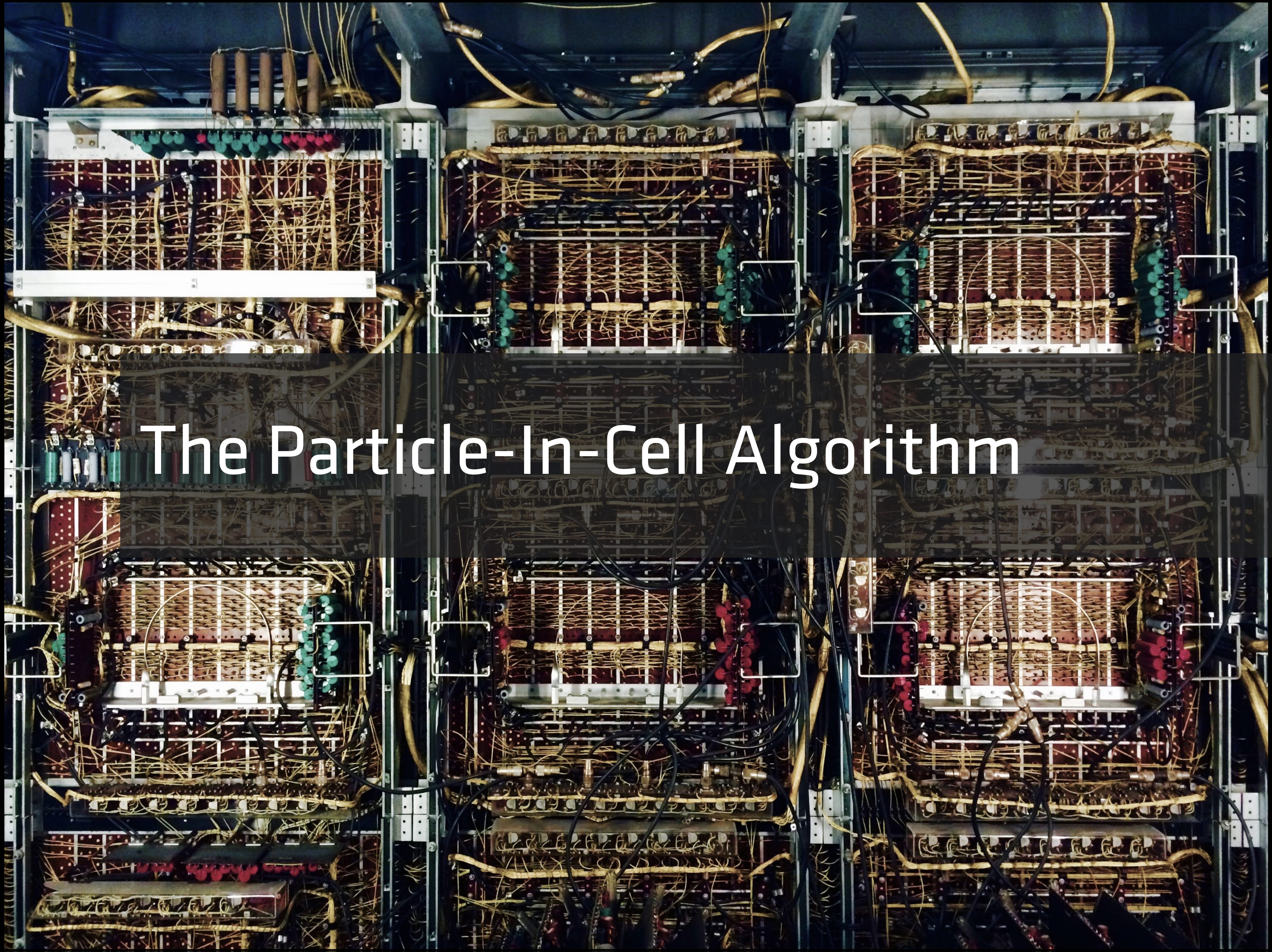
- **Review of the Particle-In Cell Algorithm**
  - Plasma simulation using particles
  - The Particle-In-Cell algorithm
  - Units and Normalization
  - Time-step considerations

- **Modelling LWFA with PIC Codes**
  - Choice of normalization units
  - Resolution and box size
  - Simulation Particles
  - Useful diagnostics

- **Running ZPIC on your Computer**
  - Compiling from source
  - Using a Docker image
  - Hands-on examples



The Particle-In-Cell Algorithm



TÉCNICO LISBOA

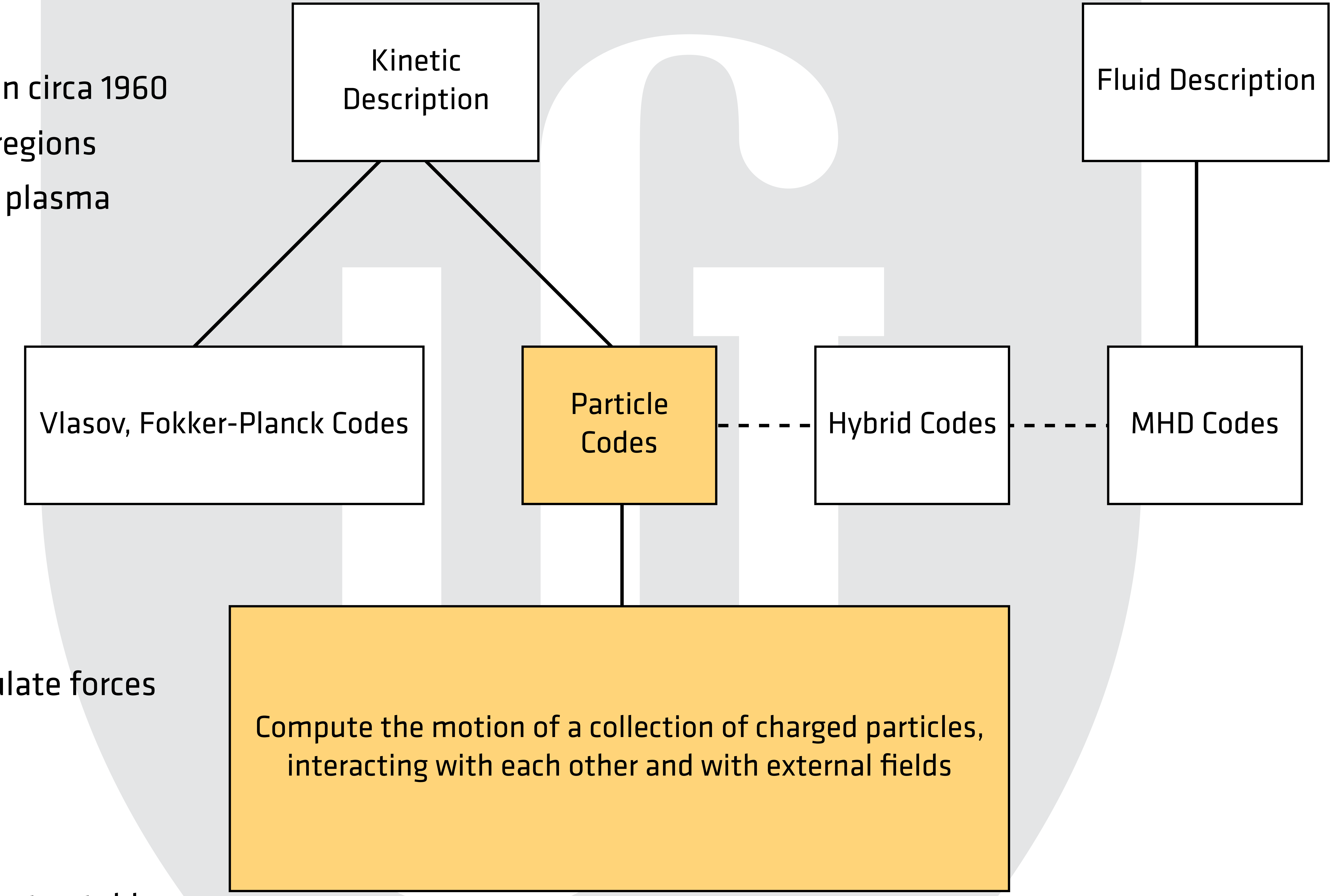
UNIVAC 1 - 1951

Internal view

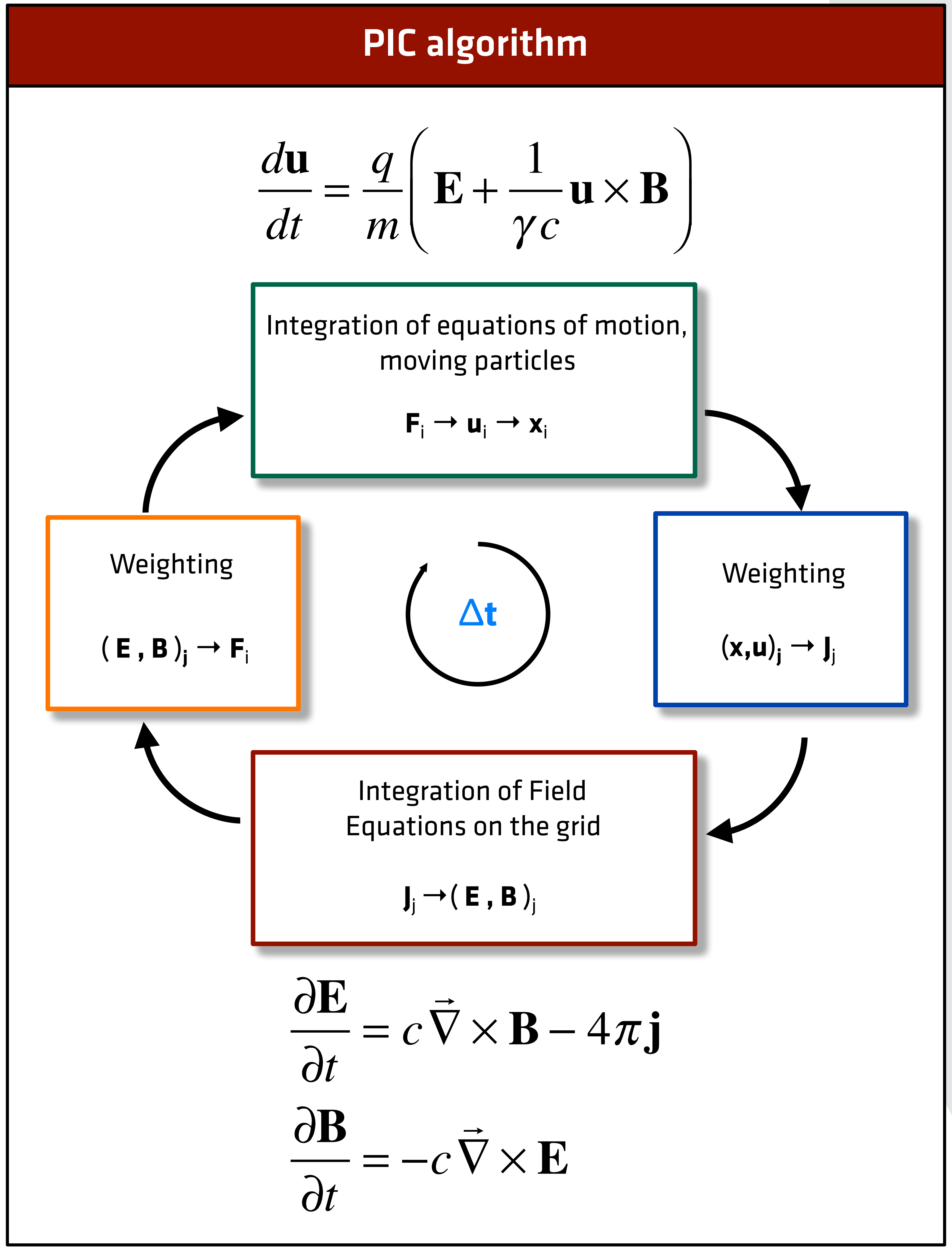
Overview of Plasma Simulation Algorithms

- **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



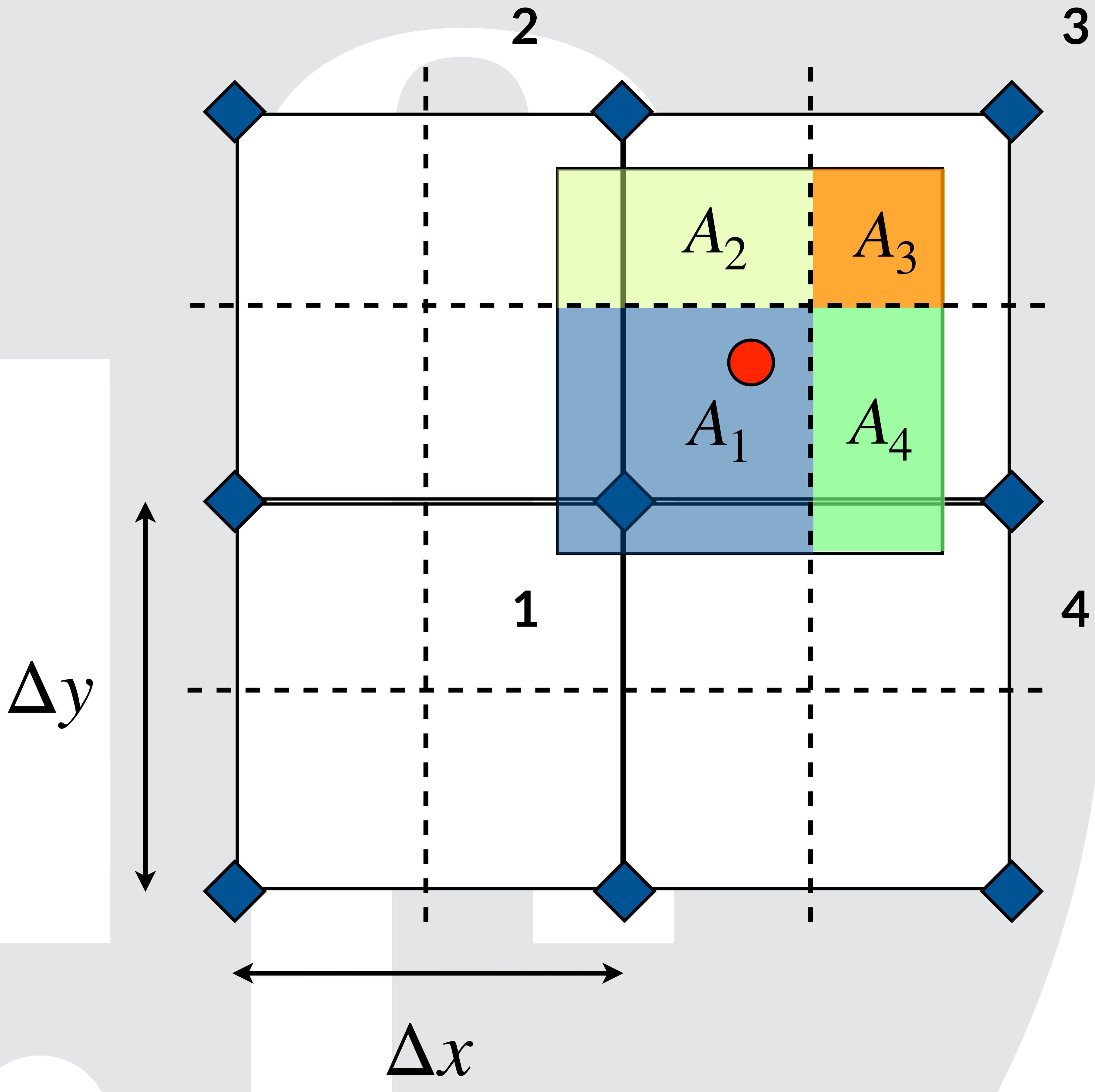
- **Fully Relativistic, Electromagnetic Particle-In-Cell algorithm**
  - Discretize Electric and Magnetic fields on a grid
  - Cell size must resolve shortest relevant lengths in the simulation
    - Typically the laser wavelength or the plasma skin depth
  - Represent plasma particles with simulation macro-particles
    - Free to move in entire *n*D-3V phasespace
    - Each macro-particle represents several plasma particles
  - Must have enough particles per cell to properly resolve velocity distributions
- **Fields and particles don't exist in the same simulation topology**
  - Field quantities are limited to grid points
  - Field interpolation connects fields → particles
  - Current deposition connects particles → fields
- **Four major steps**
  - Field interpolation
  - Particle advance
  - Current deposition
  - Field advance

zpic@edu

Interpolating the fields

- **Particles are free to move to any position**
  - Field are discretized on a grid
  - Field values at particle positions are required to advance particle momenta
- **Interpolate fields at particle positions**
  - ZPIC uses linear interpolation
  - In 2D this can be viewed as area weighting
  - The interpolating scheme must be consistent with charge / current deposition
- **Momentum conserving algorithm**
  - Avoids self-forces
  - $dp/dt = 0$ for single particle

$$A = \Delta x \Delta y$$



$$E_{part} = \frac{A_1 E_1 + A_2 E_2 + A_3 E_3 + A_4 E_4}{A}$$

Pushing the particles

- **Advance generalized velocity and position of individual particles**
  - ZPIC is a fully relativistic code so we work with $\mathbf{u} = \gamma\,\boldsymbol{\beta}$ instead $\mathbf{v}$.
  - We use a leap-frog scheme to integrate particle motion:
    - Positions (**x**) are defined at integral time-steps t^n
    - Velocities (**u**) are defined at half time-steps $t^{n+½}$
  - Second-order accuracy in time
- **Velocities are integrated using a relativistic Boris pusher**
  - Separate **E** and **B** contributions
    - i. Accelerate with ½ electric impulse
    - ii. Full magnetic field rotation
    - iii. Add remaining ½ electric impulse
  - Fully relativistic, second order time accurate
  - By construction, no work from **B** field
- **Position advance is straightforward**
  - ZPIC stores cell index and position inside cell

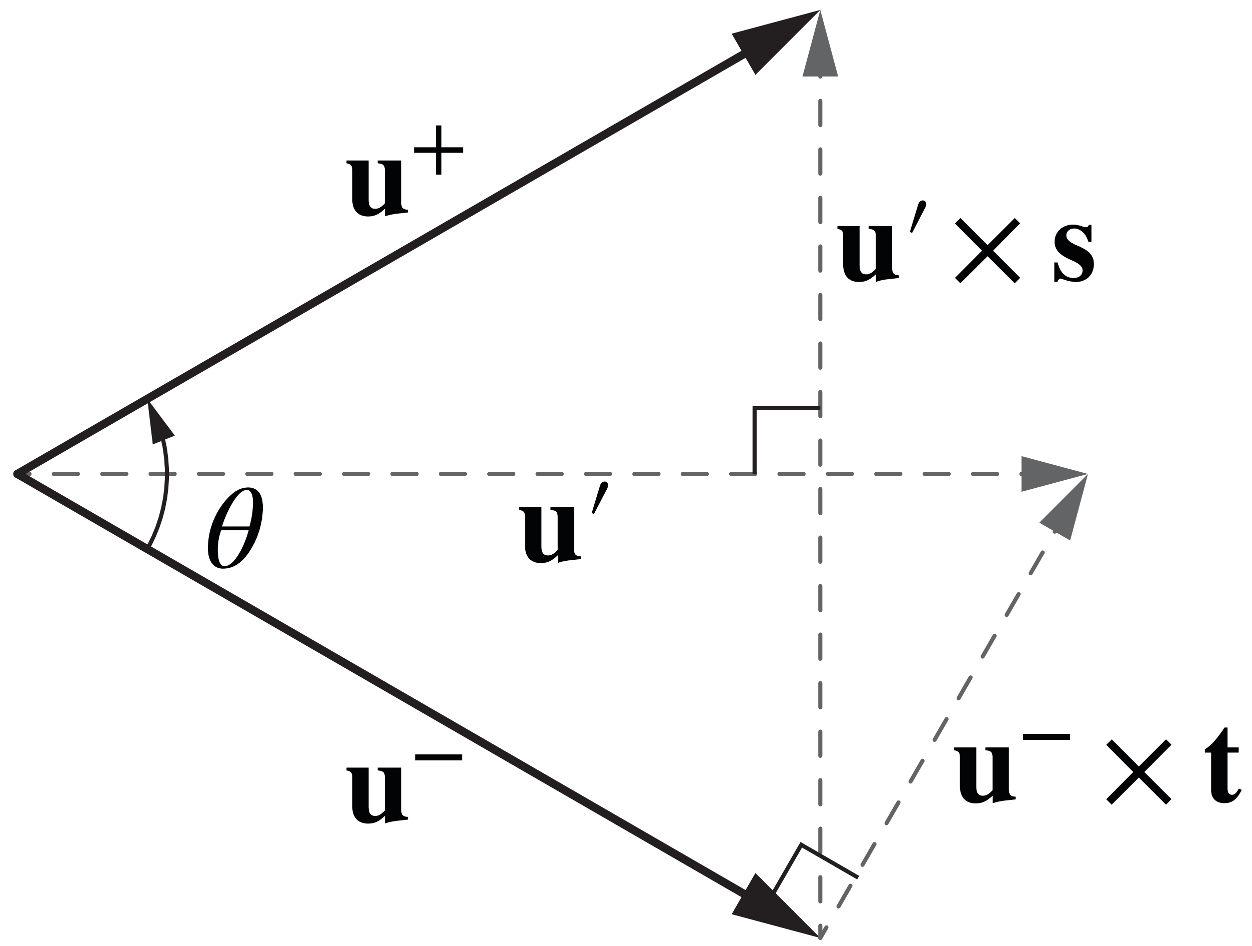
Advance momenta

i. $$\mathbf{u}^- = \mathbf{u}^{n-1/2} + \frac{q\mathbf{E}^n}{m}\frac{\Delta t}{2}$$

ii. $$\mathbf{u}' = \mathbf{u}^- + \mathbf{u}^- \times \mathbf{t}$$

iii. $$\mathbf{u}^+ = \mathbf{u}^- + \mathbf{u}^- \times \mathbf{s}$$

iv. $$\mathbf{u}^{n+1/2} = \mathbf{u}^+ + \frac{q\mathbf{E}^n}{m}\frac{\Delta t}{2}$$



$$\mathbf{t} = -\frac{q\mathbf{B}^n}{\gamma^n mc}\frac{\Delta t}{2}$$

$$\mathbf{s} = \frac{2\mathbf{t}}{1+\mathbf{t}^2}$$

Advance positions

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \frac{\mathbf{u}^{n+1/2}}{\gamma^{n+1/2}}\Delta t$$

Boris, JP Proc. Fourth Conf. Num. Sim. Plasmas (1970) 3-67

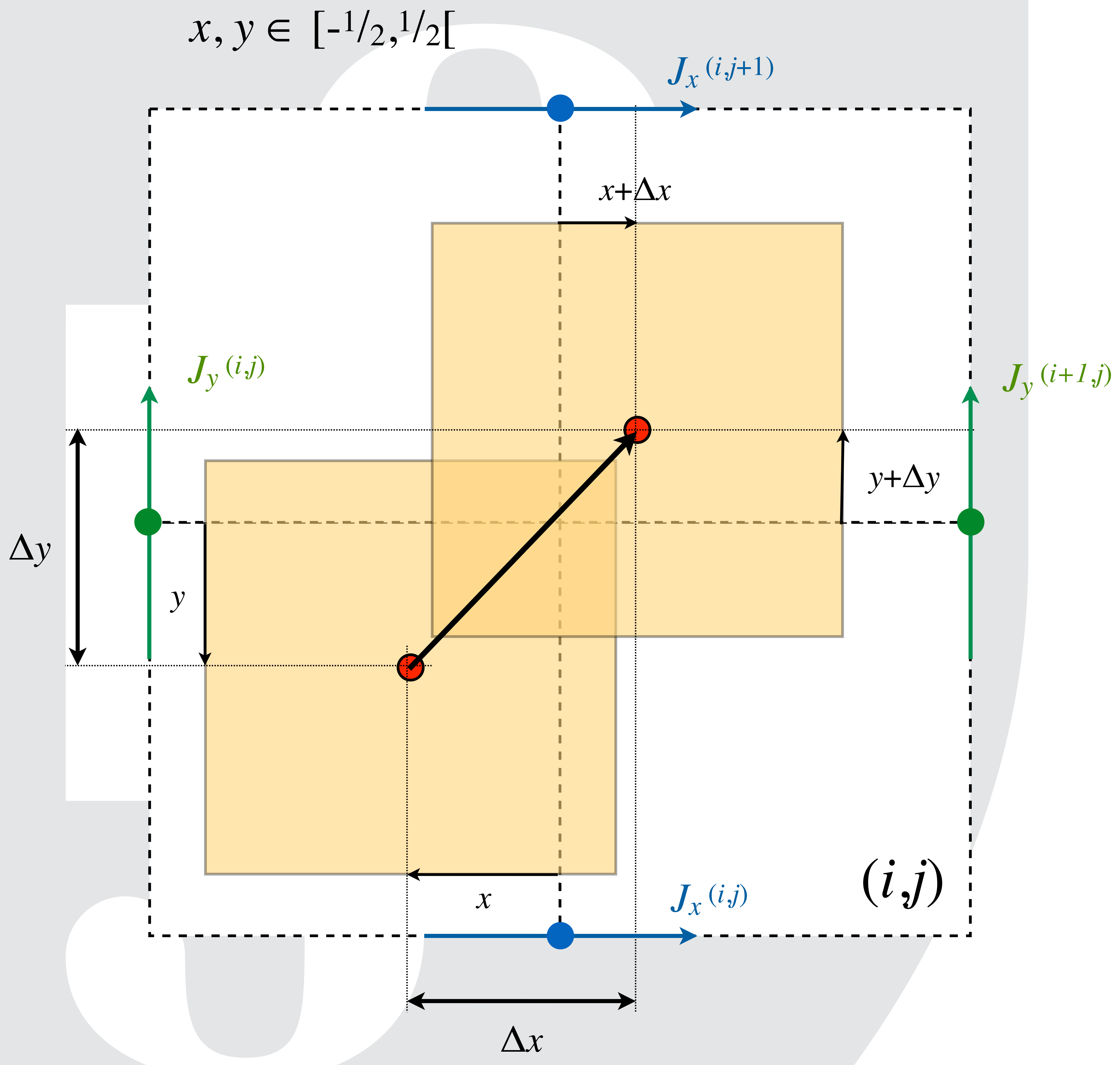
Depositing the current

- **Connects particle motion to field equations**
  - Current deposition must satisfy continuity equation:

$$\frac{\partial \rho}{\partial t} = -\nabla' \cdot \mathbf{j}$$

  - The operator ∇' corresponds to the finite difference approximation
  - Simply depositing $\rho\mathbf{v}$ does not conserve charge
  - Critical to guarantee the solutions to Maxwell's equations are self-consistent

- **Exact charge conserving current deposition scheme**
  - Developed by Villaseñor and Buneman for linear interpolation
  - Looks at particle motion, not velocity
  - Limited to motion inside single cell
    - If particles cross cell boundary, motion is split into segments that don't cross boundaries



Villasenor & Buneman Comp. Phys. Comm. **69** (1992) 306-316

Advancing the EM fields

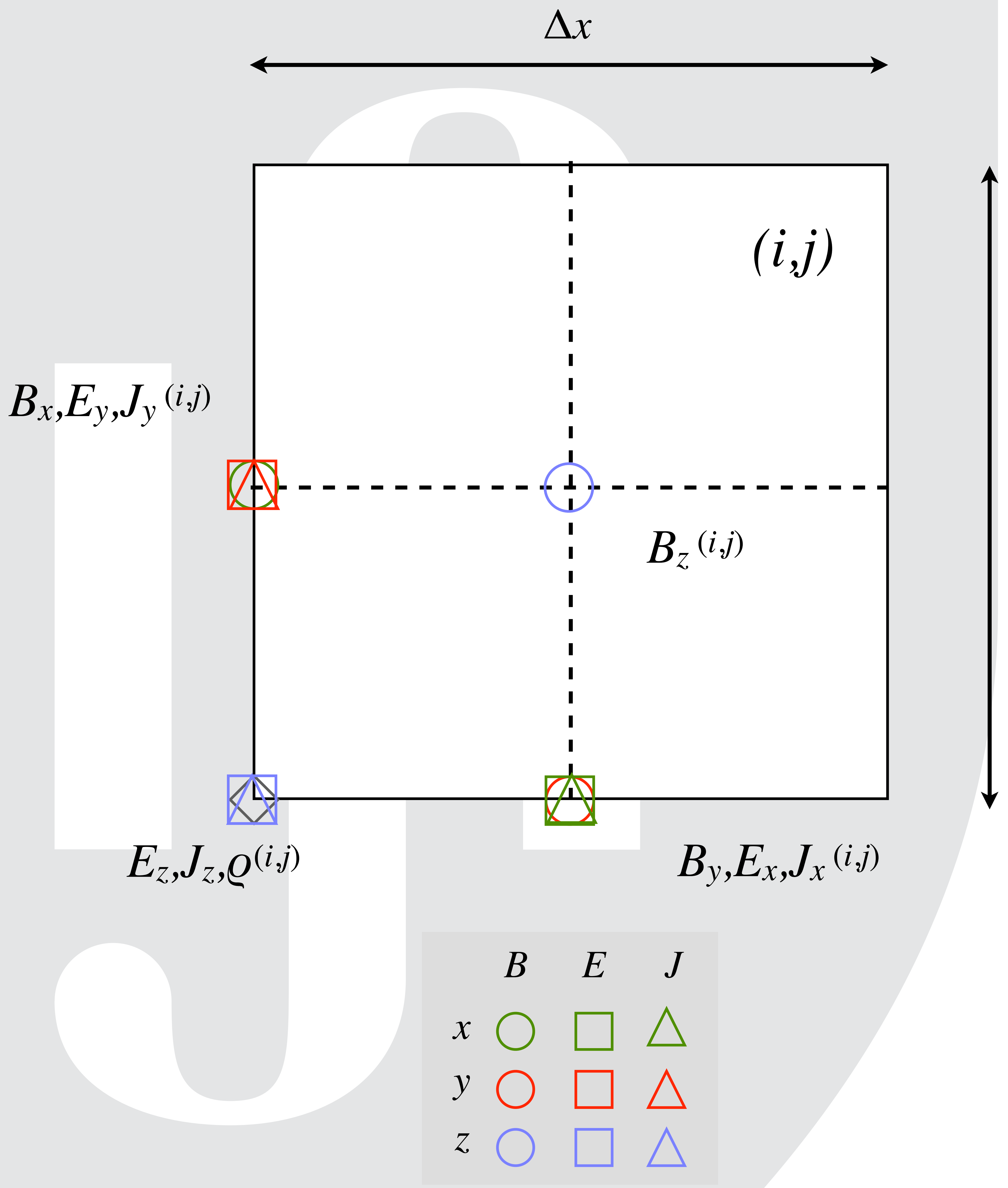
- **EM Fields are advanced in time using Maxwell's equations using the deposited current as source terms**
  - Rearrange Ampére's and Faraday's laws:

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

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

  - Discretize temporal and spatial derivatives using finite differences

- **Careful time and spacial centering of quantities leads to 2nd order accuracy**
  - ZPIC uses the Finite Difference Time Domain (FDTD) algorithm
  - Fields are staggered in time for 2nd order accuracy
    - **E** is defined at times t^n
    - **B** and **j** are defined at times $t^{n+½}$
    - **B** is later time centered for use in particle advance
  - And also in space:
    - Spatial derivatives are also 2nd order accurate



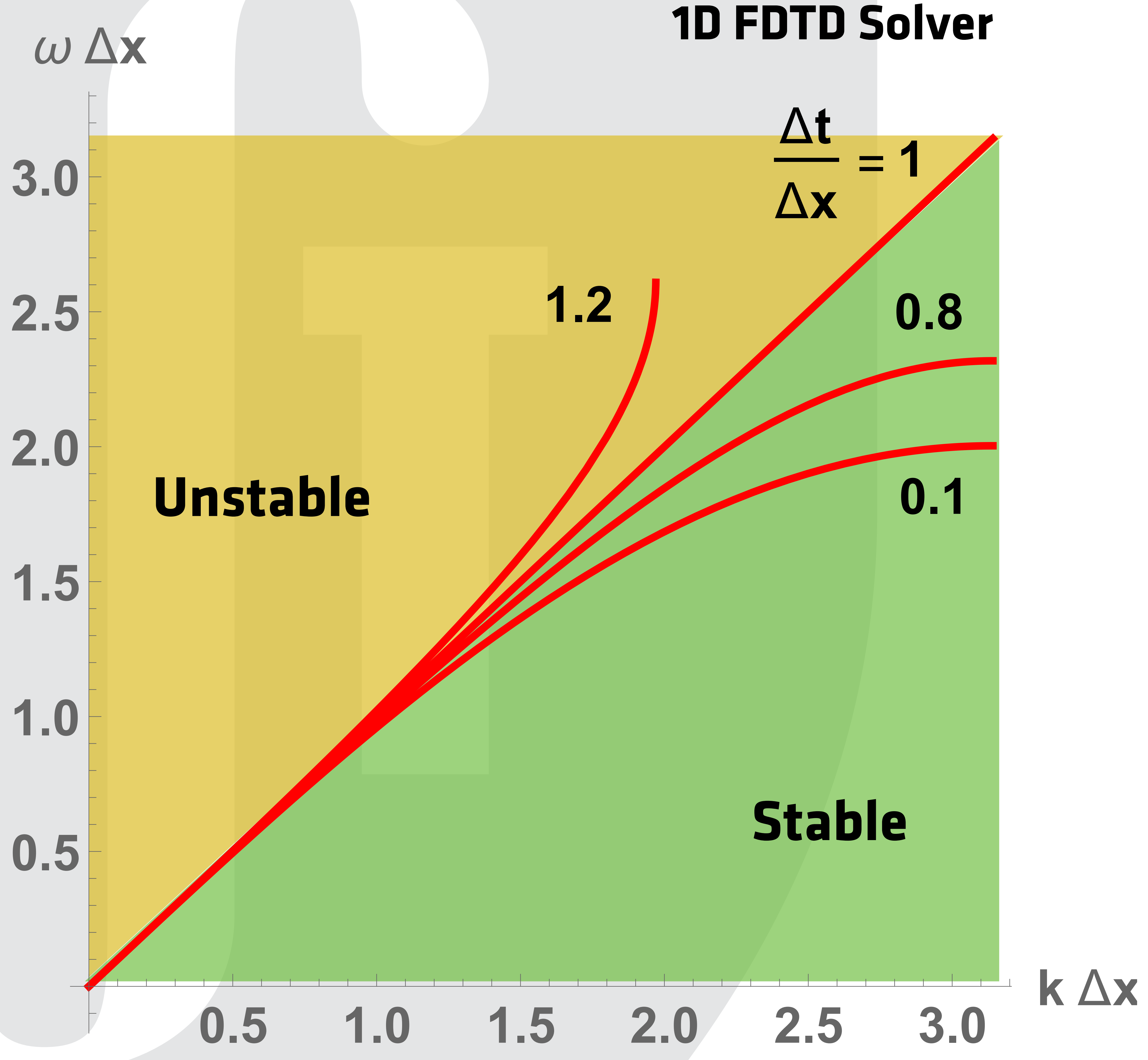
Choice of time-step

- Choice of time-step is dominated by the FDTD solver (in sim. units):

1D $$\Delta t \leq \Delta x$$

2D $$\Delta t \leq (\Delta x^{-2} + \Delta y^{-2})^{-\frac{1}{2}}$$

- If time step is larger than Courant condition the field solver becomes unstable
- If time step is much smaller than courant condition for large k, v_{ph} drops as low as $2/\pi$ = 0.637 c
- Relativistic particles may have $v > v_0$
  - Numerical Cherenkov



Units and Normalization in ZPIC

- **Careful choice of units and normalization is critical**
  - Avoids multiplication by several constants (e.g. m_e, e and c) improving performance and numerical accuracy.
  - By expressing the simulation quantities in terms of fundamental plasma quantities the results are general and not bound to some specific units we may choose

- **Units and normalization in ZPIC**
  - The frequencies are normalized to a normalization frequency, ω_n. Time is normalized to ω_n^{-1}.
  - Proper velocities are normalized to the speed of light, c. Space is normalized to c/ω_n.
  - Charge and mass are normalized to the absolute electron charge, e, and the electron mass, m_e. The fields are then normalized appropriately.
  - The density is normalized to ω_n^2 (the normalization frequency squared). So if the density is 1 at a given location then the normalization frequency is the plasma frequency at that location.
  - If the laser frequency is 1, then the normalization frequency is the laser frequency and the density is normalized to the critical densify (for that laser frequency).

zpic units

$$\mathbf{x}' = \frac{\omega_n}{c}\mathbf{x}$$

$$\mathbf{p}' = \frac{\mathbf{p}}{m_{sp}c} = \frac{\gamma \mathbf{v}}{c} = \frac{\mathbf{u}}{c}$$

$$\mathbf{E}' = e\frac{c/\omega_n}{m_e c^2}\mathbf{E}$$

$$\mathbf{B}' = e\frac{c/\omega_n}{m_e c^2}\mathbf{B}$$

m_{sp} is the mass of the species

zpic@edu

Modelling Laser Wakefield Acceleratio

TÉCNICO LISBOA

Laser Wakefield Acceleration

3D Simulation using the OSIRIS code

Reference length and time

Choose the normalization

Plasma sets reference

Plasma density is unity

- Normalize lengths to plasma skin depth and frequency to plasma frequency

Example

- Plasma density $n_p = 10^{18}$ cm^{-3}
- Plasma frequency $\omega_p \sim 5.64 \times 10^{13}$ rad s^{-1}
- Laser wavelength $\lambda_0 = 1$ μm
- Laser frequency $\omega_0 \sim 2.34 \times 10^{15}$ rad s^{-1}
- Normalised laser frequency is $\omega_0/\omega_p \sim 41.5$

Laser sets reference

Reference laser frequency is unity

- Normalize plasma density to critical density; length to inverse laser wavenumber

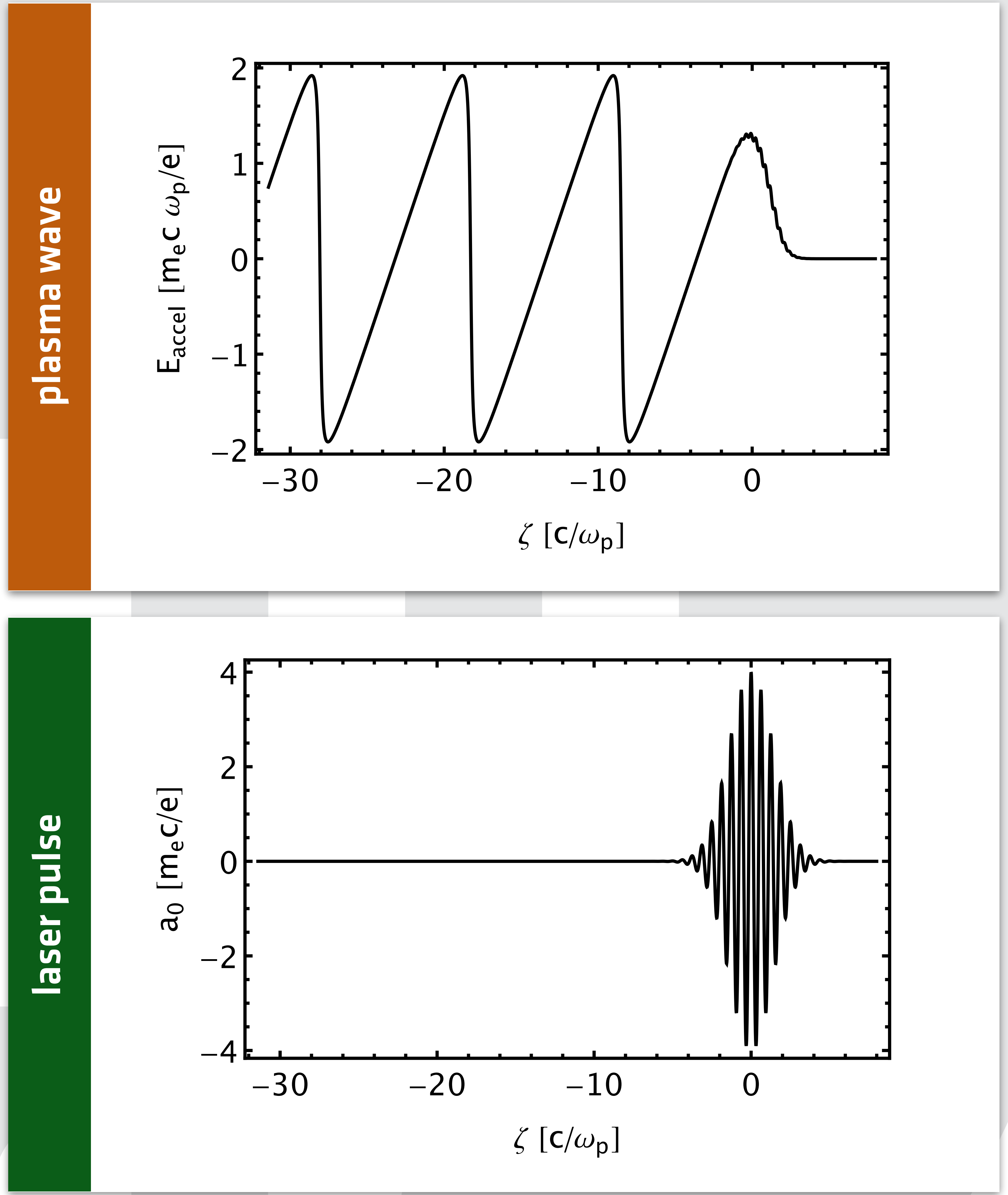
Example

- Laser wavelength λ_0=1μm
- Laser frequency $\omega_0 \sim 2.34 \times 10^{15}$ rad s^{-1}
- Critical frequency $n_{crit} \sim 1.72 \times 10^{21}$ cm^{-3}
- Plasma density $n_p = 10^{18}$ cm^{-3}
- Normalized plasma density $n_p/n_{crit} \sim 5.8 \times 10^{-4}$

Both (and other) normalizations are possible. In this session we will use the plasma as the reference!

Choosing the spatial resolution

- **Laser propagates in an underdense plasma**
  - $n_p \ll n_{crit}$ | $\lambda_0 \ll \lambda_p$ | $\omega_p \ll \omega_0$
- **Need to resolve the smallest scale length**
  - > 20 - 30 cells per wavleng
- **Plasma wave**
  - Skin depth sets the plasma scale length
  - $c/\omega_p \sim 5.3\ \mu m/(\ n_p\ [10^{18}\ cm^{-3}]\ )^{1/2}$
- **Laser**
  - laser wavelength sets the laser scale length
  - $\lambda_0 \sim 1\ \mu m \sim 0.18\ (\ n_p\ [10^{18} cm^{-3}]\ )^{1/2}\ c/\omega_p$

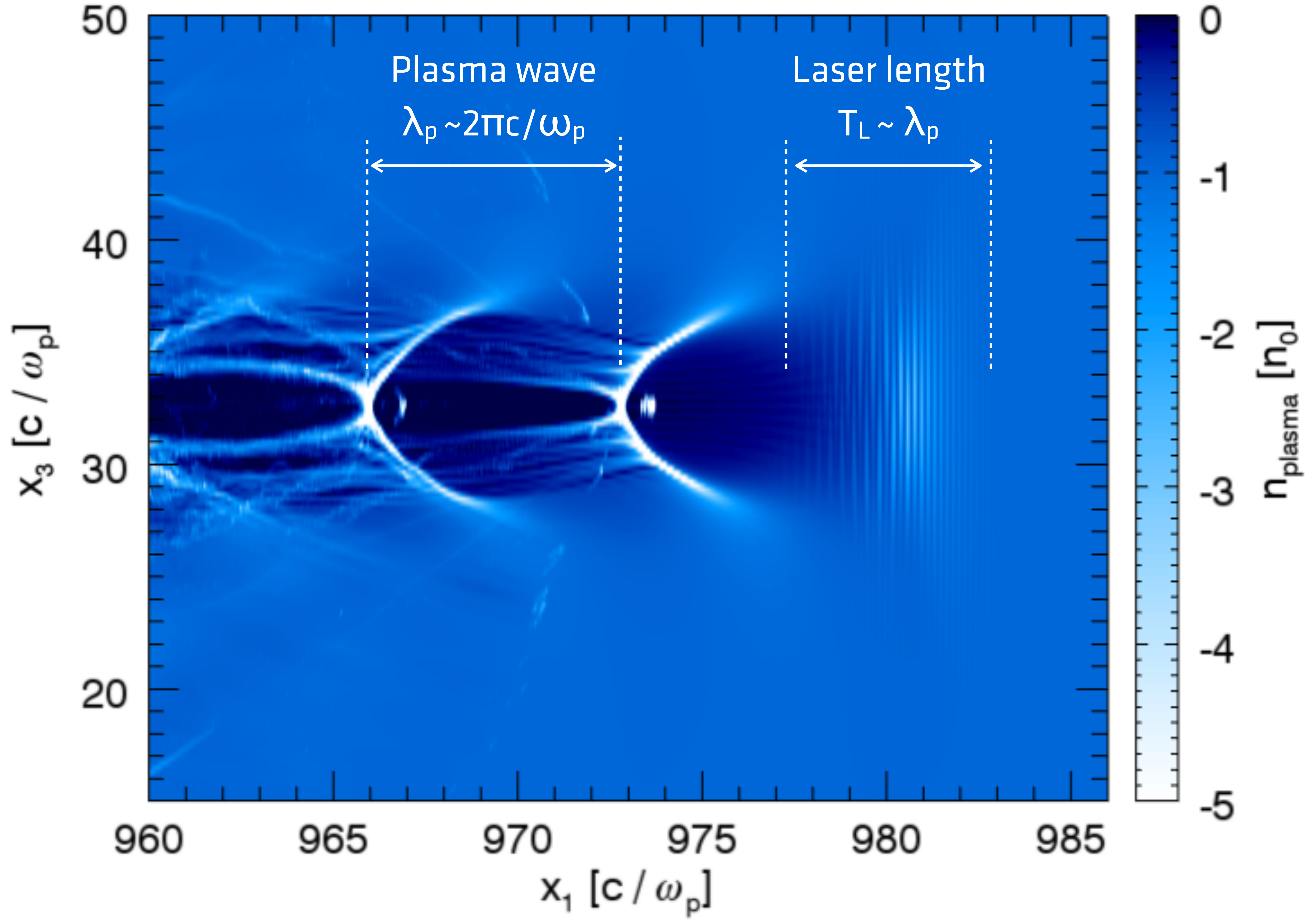


Longitudinal spatial Resolution: $\Delta x \sim \lambda_0/\# \sim 0.18/\#\ (\ n_p\ [10^{18} cm^{-3}]\ )^{1/2}\ c/\omega_p$

> 20-30 (number of cells per laser wavelength)

Simulation box dimensions

- **Simulations are done in a moving window moving at the speed of light**
  - The simulation box does not need to hold the entire propagation length
- **Simulation box needs only to model the relevant structures in the accelerator**
  - Laser driver and initial trailing buckets of accelerating structure
- **Box size determined by largest relevant structures**
  - Longitudinally
    - a few plasma wavelengths long
    - > 4 λp ~ 25 c/ωp
  - Transversely (2D)
    - Laser pulse waist / transverse bubble size
    - > 4 λp ~ 25 c/ωp

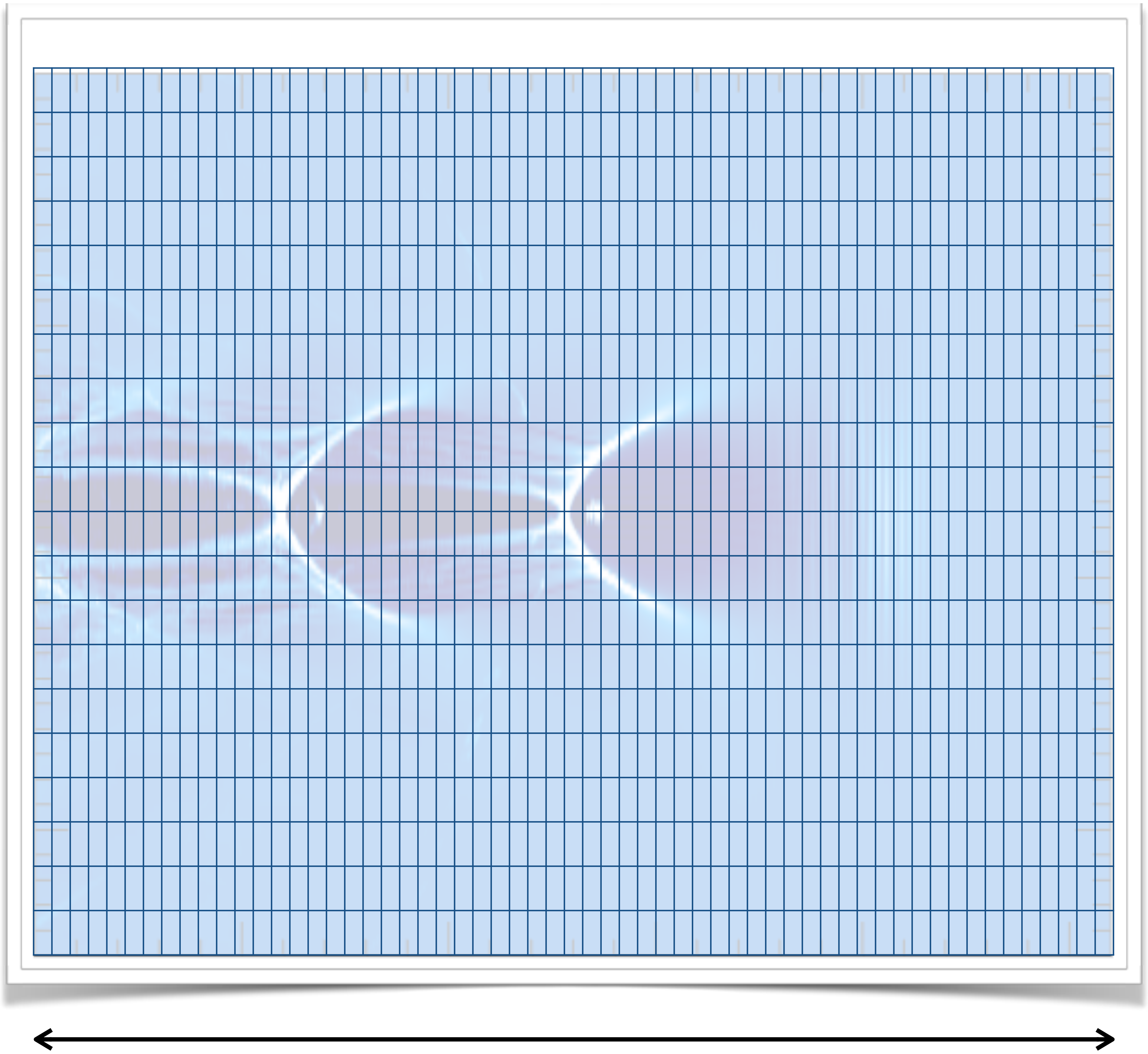


Setting up the simulation: cells, particles

Particles per cell: ≫1 in 1D (e.g. 64) and around 10 in 2D

- **Simulation grid**
  - Box length: $L = 4\ \lambda_p$
  - 20 points per laser wavelength
  - $\Delta x \sim \lambda_0/20 \sim 0.18/20 = 0.009\ c/\omega_p$
  - Number of cells ~ $L\ /\ \Delta x$ ~ 2800 cells

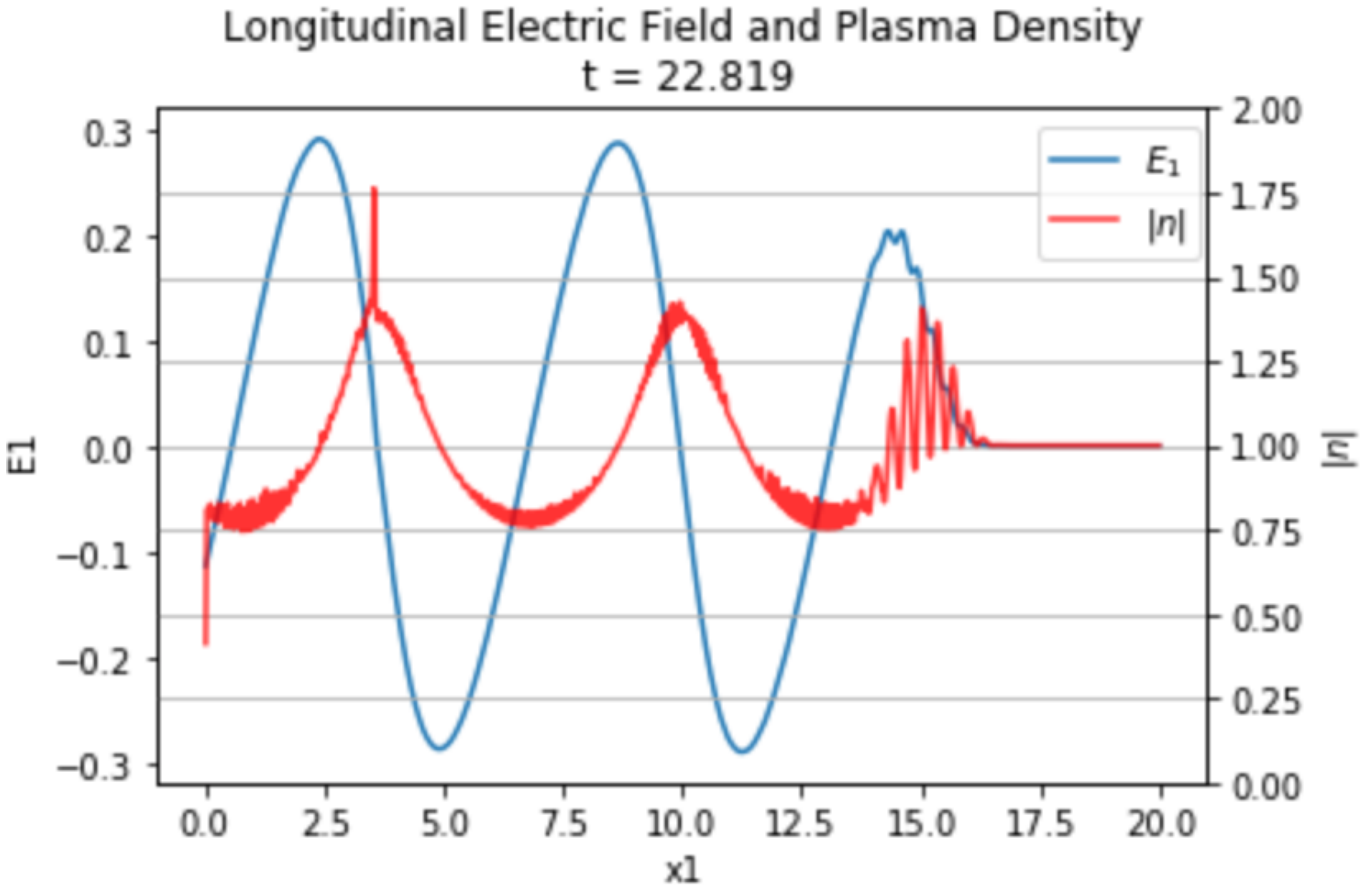
- **Simulation particles**
  - Number of particles per cell must resolve local phasespace
  - ≫1 in 1D (e.g. 64)
  - ~ 10 in 2D
  - Higher numbers improve phasespace resolution (detailed distribution tails)
  - Also reduces simulation noise



Longitudinal cells: $4\ \lambda_p\ /(0.18/20\ c/\omega_p)$ ~ 2800 cells ($n_p = 10^{18}\ cm^{-3}$)

Useful diagnostics

- **Plasma density**
  - Charge density of the background plasma
  - Wave structure and particle loading
- **Longitudinal electric field**
  - Accelerating / decelerating fields
- **Transverse electric field**
  - Laser field
  - Focusing / defocusing fields (2D)
- **Particle phasespace**
  - Show particle momenta as a function of position
  - Most common is u_1/x_1
  - Wave structure and particle acceleration

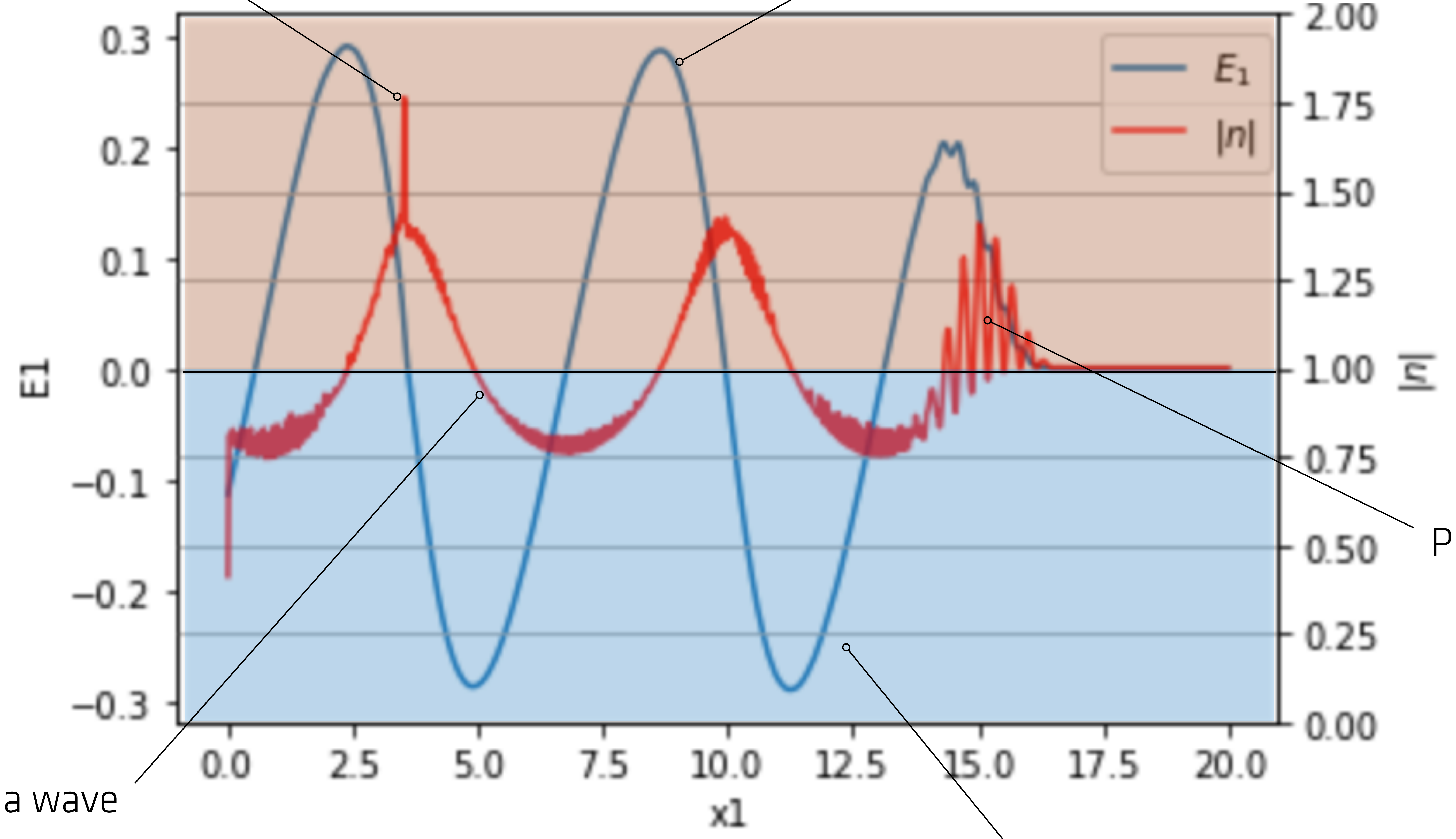


Simulations performed in a moving window that travels at c

Simulations performed in a moving window that travels at c

Particle injection

decelerating field when E>0!



Plasma oscillating in the laser field

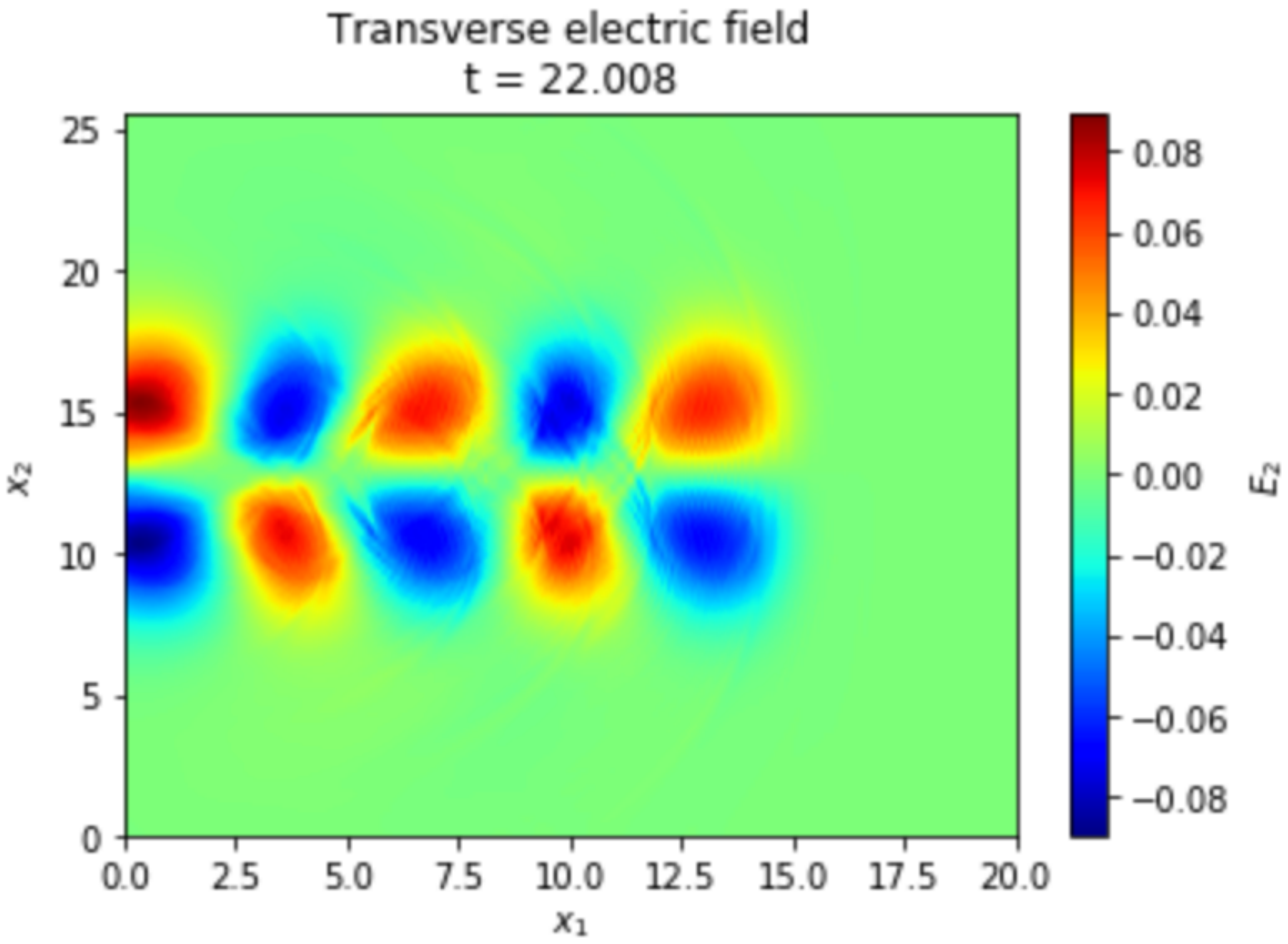
Plasma wave

accelerating field is when E<0!

Transverse electric field (2D/3D)

- **Transverse electric fields**
  - Laser pulse
  - Transverse wave structure

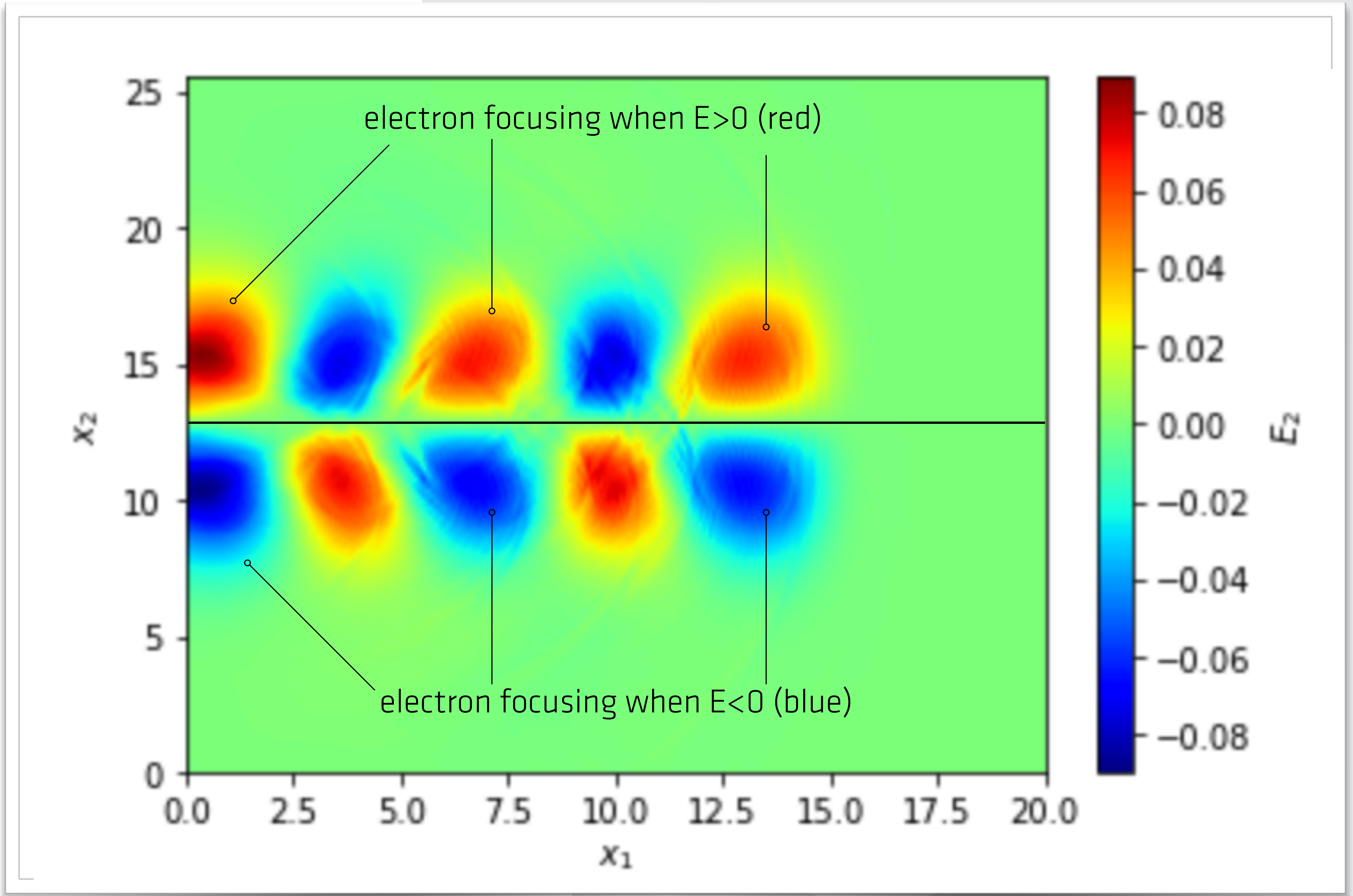
- **Also laser pulse**
  - In this example the laser was polarized out of the plane



Simulations performed in a moving window that travels at c

Focusing fields: non relativistic particles

$E_\perp$ is the focusing force for non-relativistic particles $[(\mathbf{v} \times \mathbf{B} / c)_\perp \ll \mathbf{E}_\perp]$

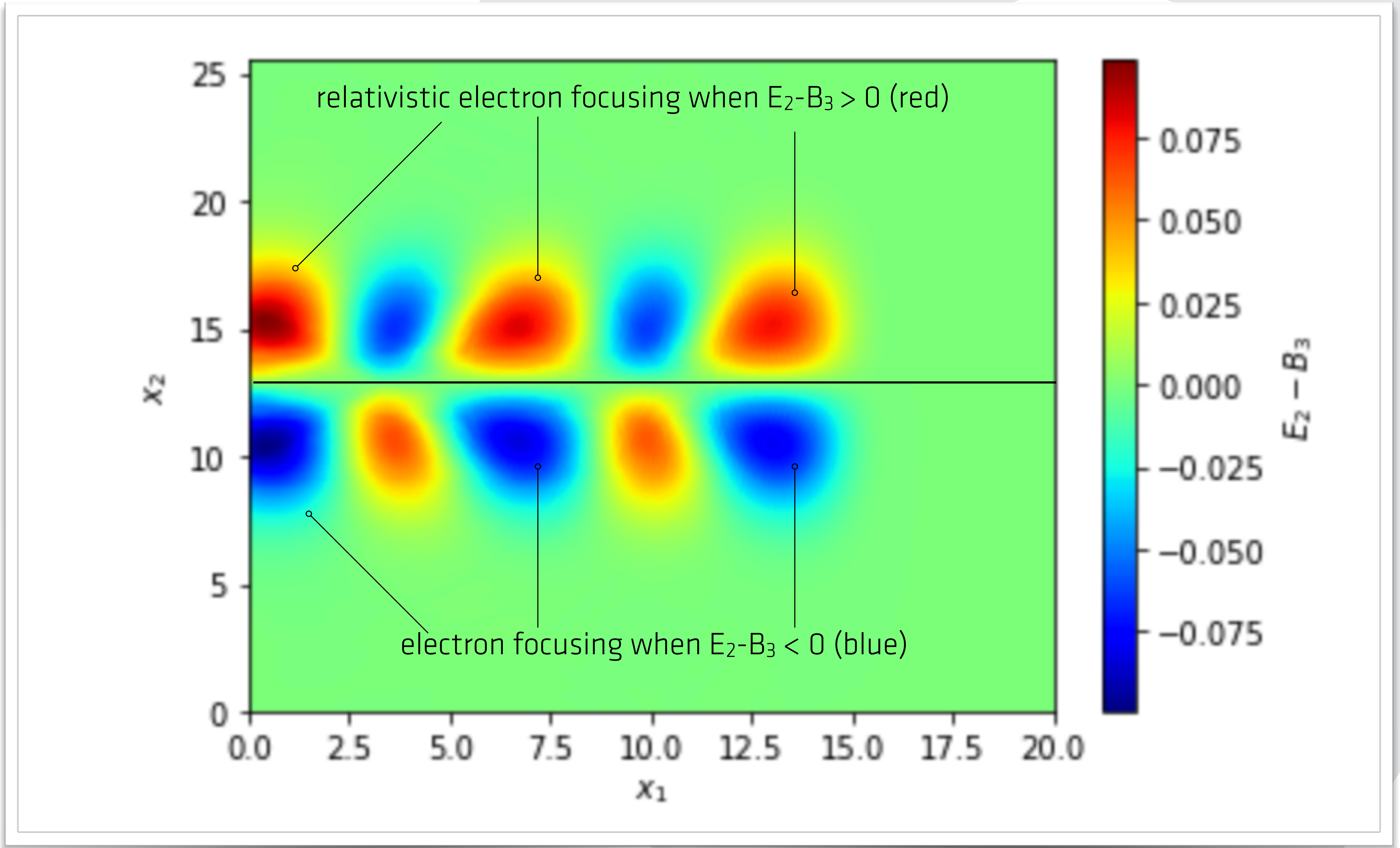


Simulations performed in a moving window that travels at c

Focusing fields: Ultra-relativistic particles

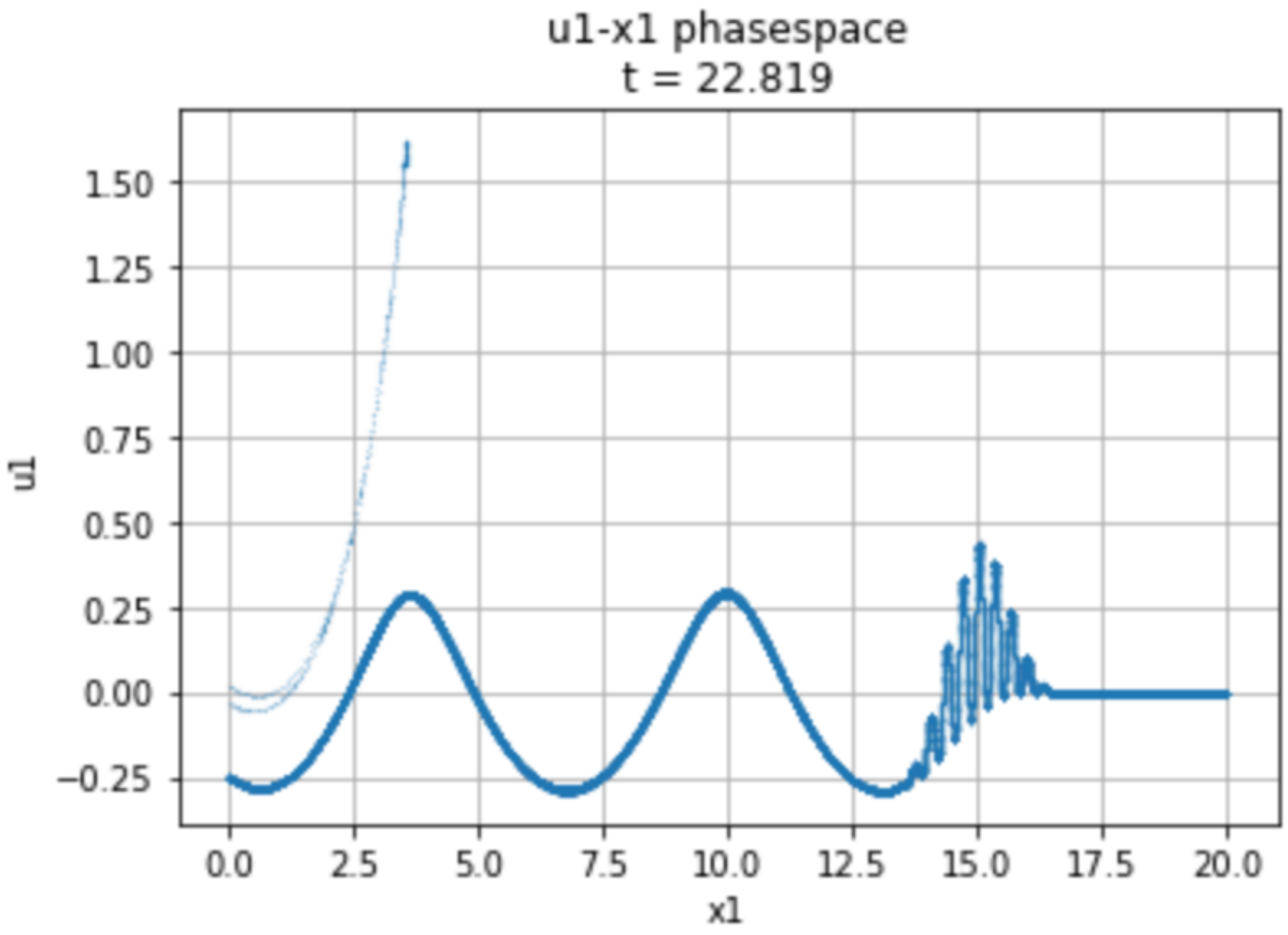
Focusing force for a ultra-relativistic particle:

$$E_r + v_{||} \times B_\Theta/c \simeq E_r - B_\Theta$$



Longitudinal phase-space

- **Longitudinal particle phase-space**
  - Plots momenta vs. position
  - Phasespace density or 1 point per particle
  - Most common is longitudinal momenta vs. longitudinal position i.e. $\underline{u}_1$-x_1

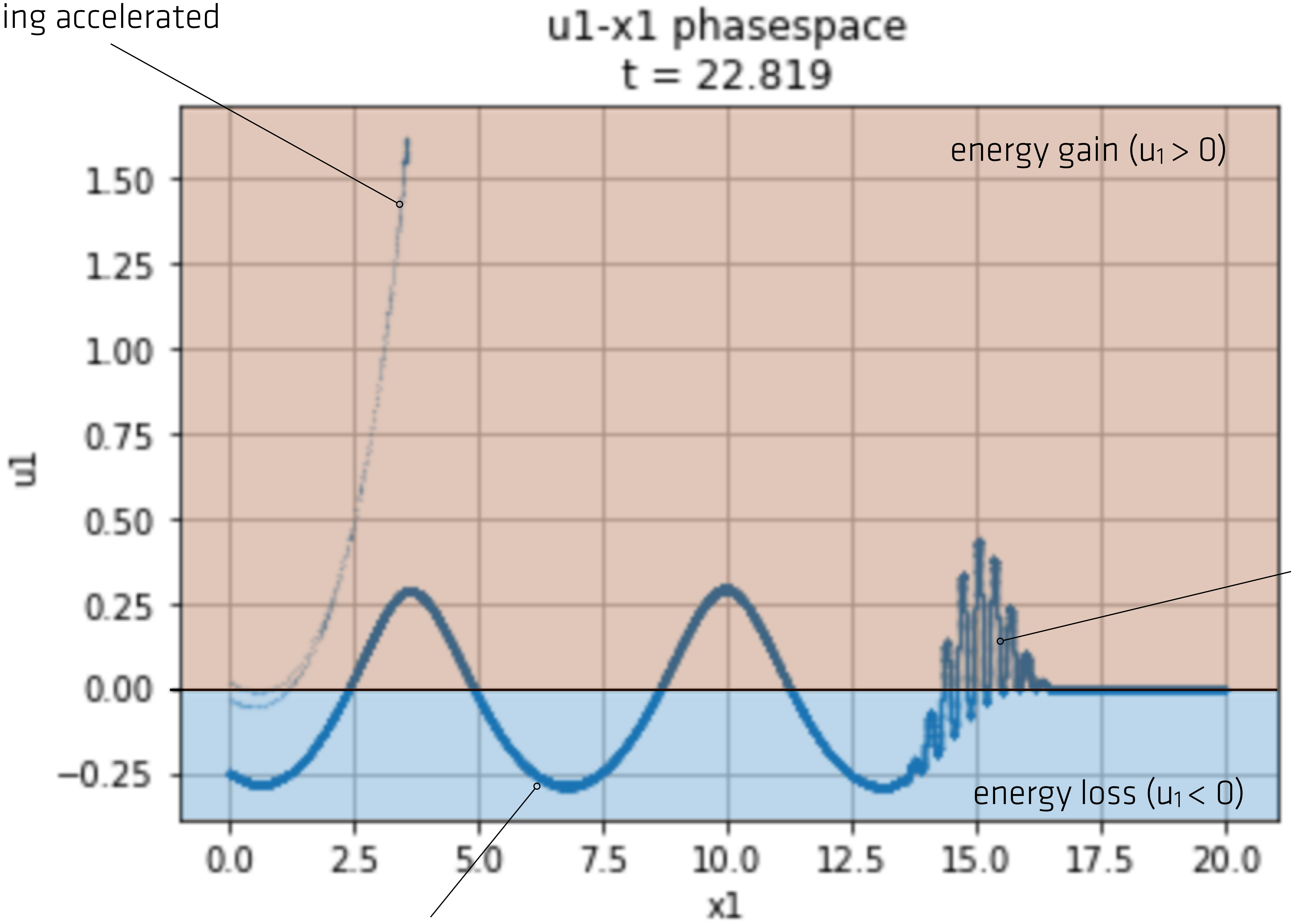


Simulations performed in a moving window that travels at c

Particle acceleration and deceleration

In plasma based acceleration: Energy $[m_e c^2] = \gamma - 1 \simeq \gamma \simeq u_1$ [me c]

Trapped particles being accelerated



Particles oscillating in the laser field

Plasma oscillating in the wakefield

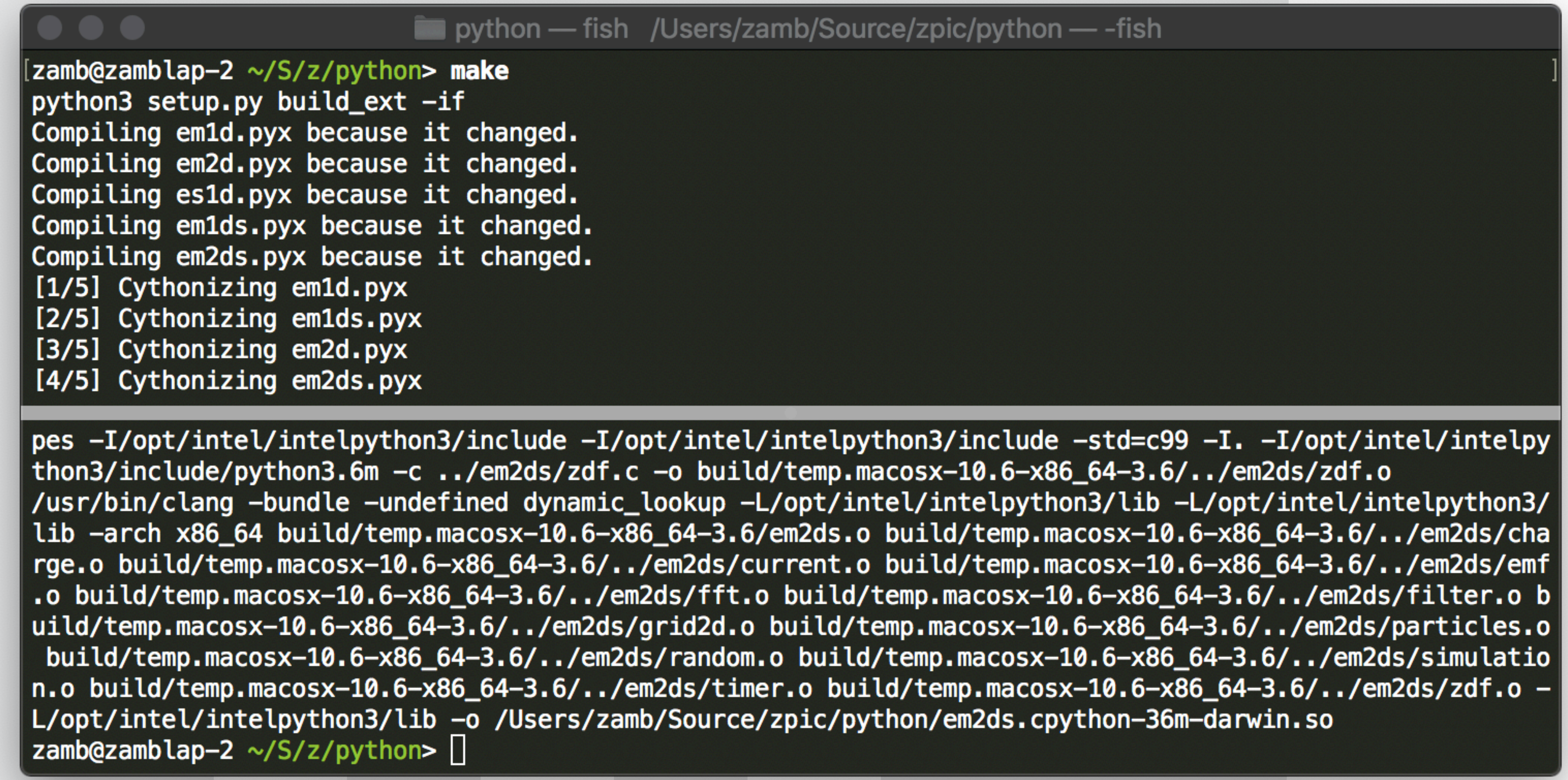
Running ZPIC on your computer

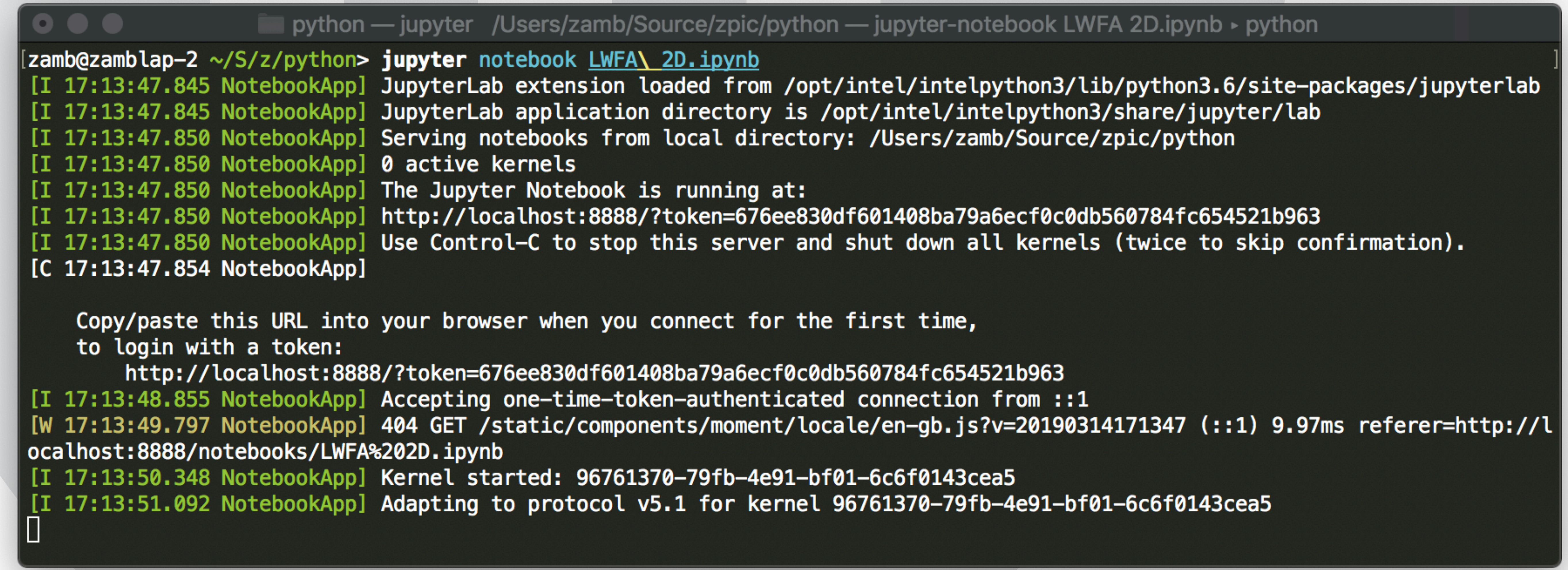
Harvard Mark I - 1944

Rear view of Computing Section

Running ZPIC - Option 1 - compile from source

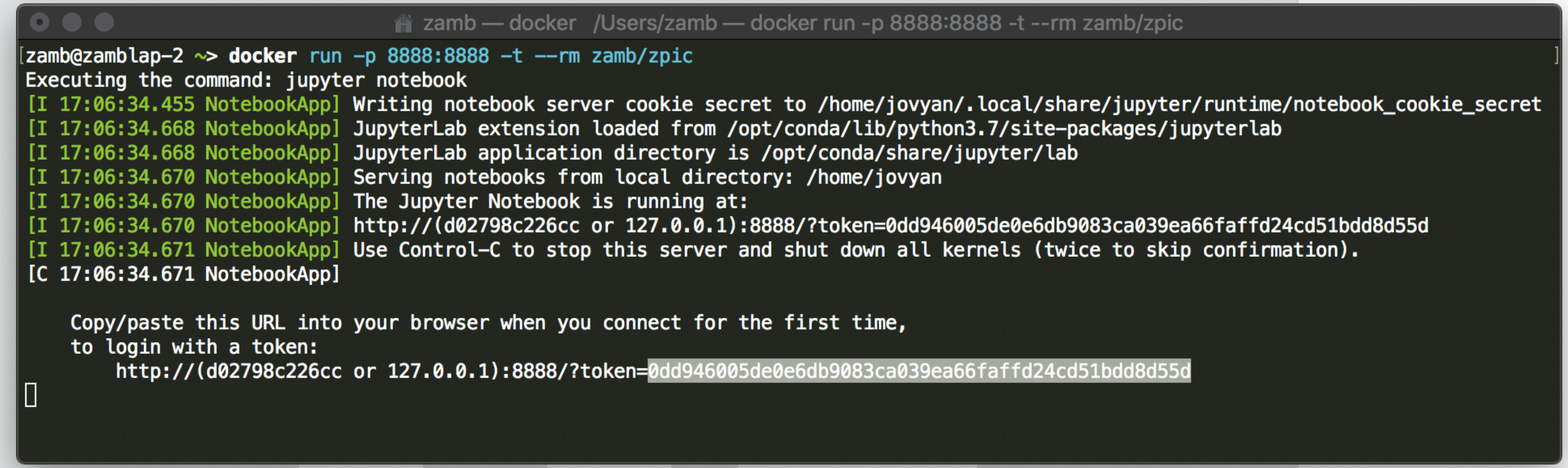
- **Build from ZPIC source**
  - ZPIC itself has no external dependencies, and requires only a C99 compliant C compiler
    - **gcc**, **clang** and **intel** tested
  - The code is open-source and hosted on GitHub
    - https://github.com/zambzamb/zpic
- **Build Python interface**
  - The Python interface requires a Python3 installation
    - We recommend the Intel Python distribution
  - The interface also requires NumPy and Cython packages to be installed
  - Just use the Makefile in the python subfolder of the ZPIC distribution
    - This will also compile all of the ZPIC codes
- **Using the Jupyter notebooks**
  - Requires a working Jupyter + Python installation
    - Again, we recommend the Intel Python distribution
  - Launch Jupyter and open one of the example notebooks

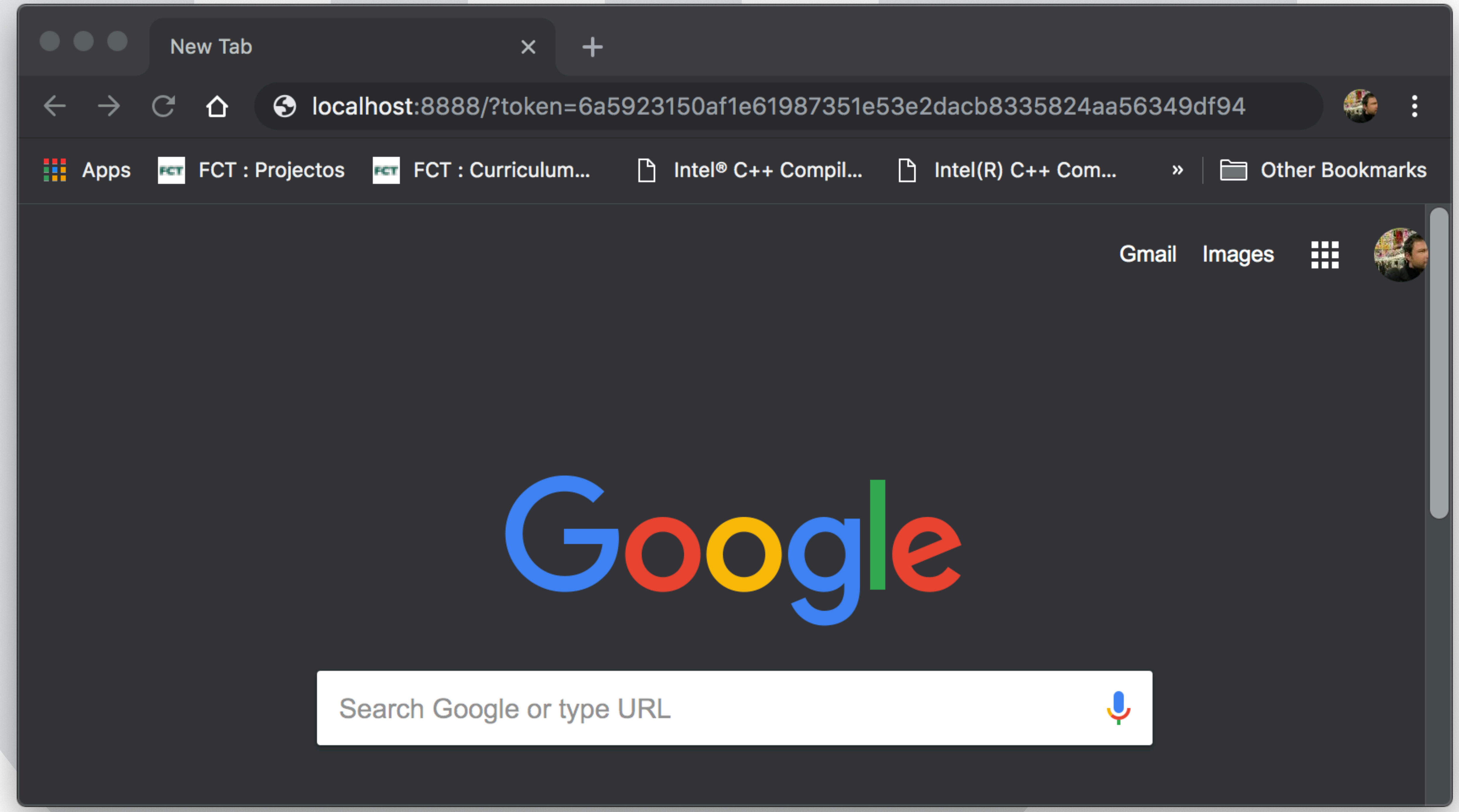




Running ZPIC - Option 2 - use a Docker container

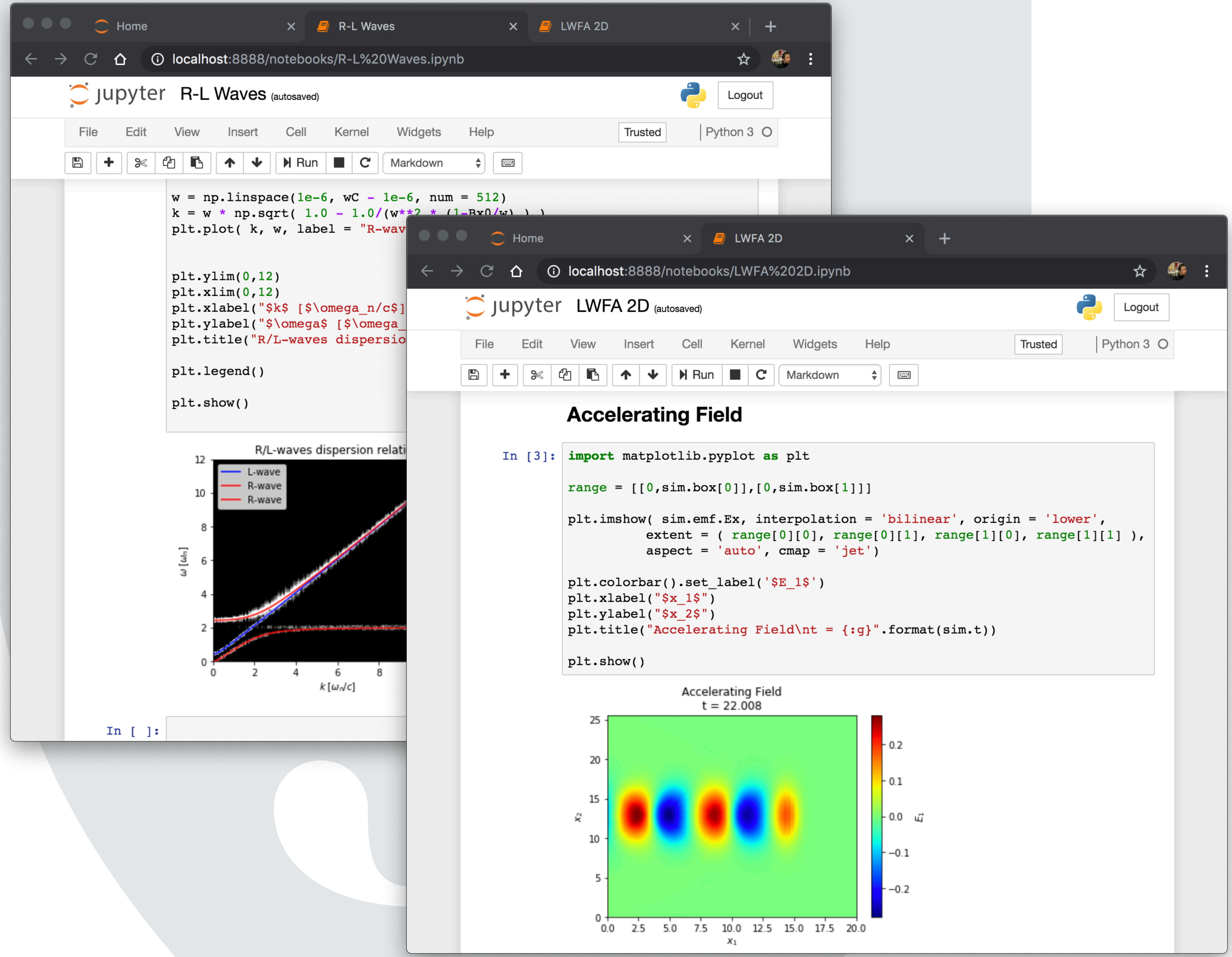
- **Install Docker desktop on your computer**
  - Available for free at:
    - https://www.docker.com/products/docker-desktop
- **Run the ZPIC image**
  - The ZPIC container image is hosted on DockerHub
  - Open a terminal window and type the following command
    - `> docker run -p 8888:8888 -t zamb/zpic`
  - The first time you do it, it will download the ZPIC container image. This can take a little time.
- **Open a web browser on your computer and point it to the appropriate port**
  - Type in the following as the address
    - `localhost:8888/?token=[TOKEN]`
  - Get the [TOKEN] value from the output of the docker run command
  - The port number must match the docker run command





Using ZPIC Notebooks

- **Jupyter notebooks**
  - Similar to Mathematica notebooks but for Python
  - Run in a web browser
  - Organized in a sequence of cells
  - Each cell can contain Python code or annotations
- **The code is runs inside the notebook**
  - Initialize the simulation
  - Run to specified time
  - Access simulation data directly to visualize output
  - Several examples provided
- **Saving simulation output not necessary**
  - Example simulations run in ~ 1 minute
  - Visualize results in the notebook
  - Interactively modify simulation parameters
  - If required (e.g. for longer simulations) the code can save simulation results to disk
    - Files are saved in the ZDF format
    - a Python module is provided to read these files



Hands-on

CERN Large Hadron Collider

Accelerator Tunnel

Launch a ZPIC notebook

- **Option 1 - Compile from source**
  i. Compile the code
  ii. Launch the Jupyter notebook from the source folder:

```
> jupyter notebook LWFA1D.ipynb
```

- **Option 2 - Use a Docker Container**
  i. Install Docker
  ii. Launch the zpic container

```
> docker run -p 8888:8888 -t -v $PWD:/home/jovyan/work zamb/zpic
```

  – This mounts the directory **$PWD** on the directory **work** on your container so you can save changes to the existing notebooks or create new ones

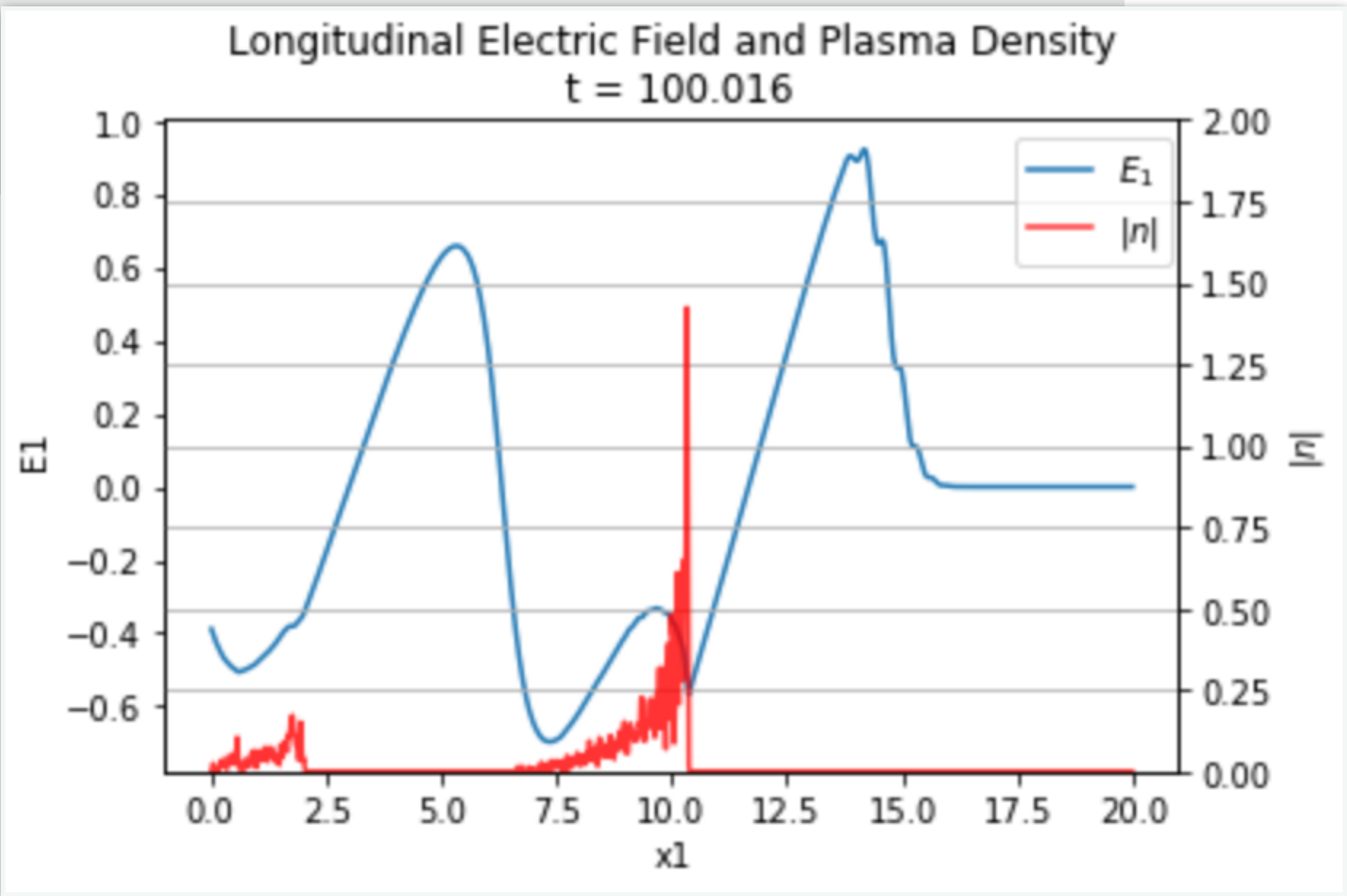
Example: Laser Wakefield Accelerator

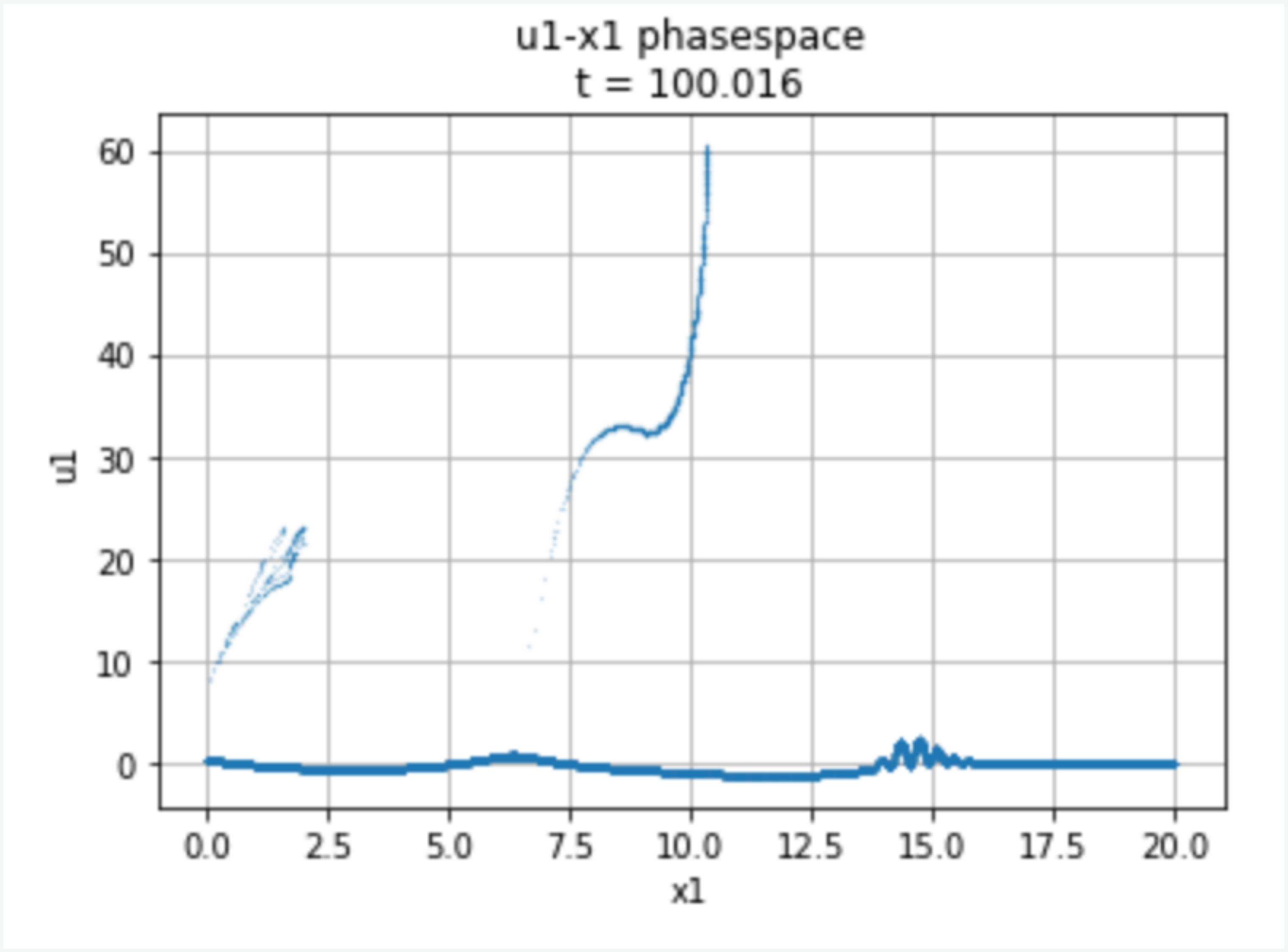
Simulate a laser wakefield accelerator:

- Add an ultra-intense laser beam as a driver (a_0~2)
- Choose laser length smaller than λ_p

Questions:

1. can you observe particle injection and trapping?
2. is the energy gain consistent with the longitudinal electric field values?
3. describe and justify the shape for the plasma electric field in the region where particles accelerate
4. could you accelerate positrons in this plasma wave? where would you place them and with what initial velocity/energy? try to simulate!

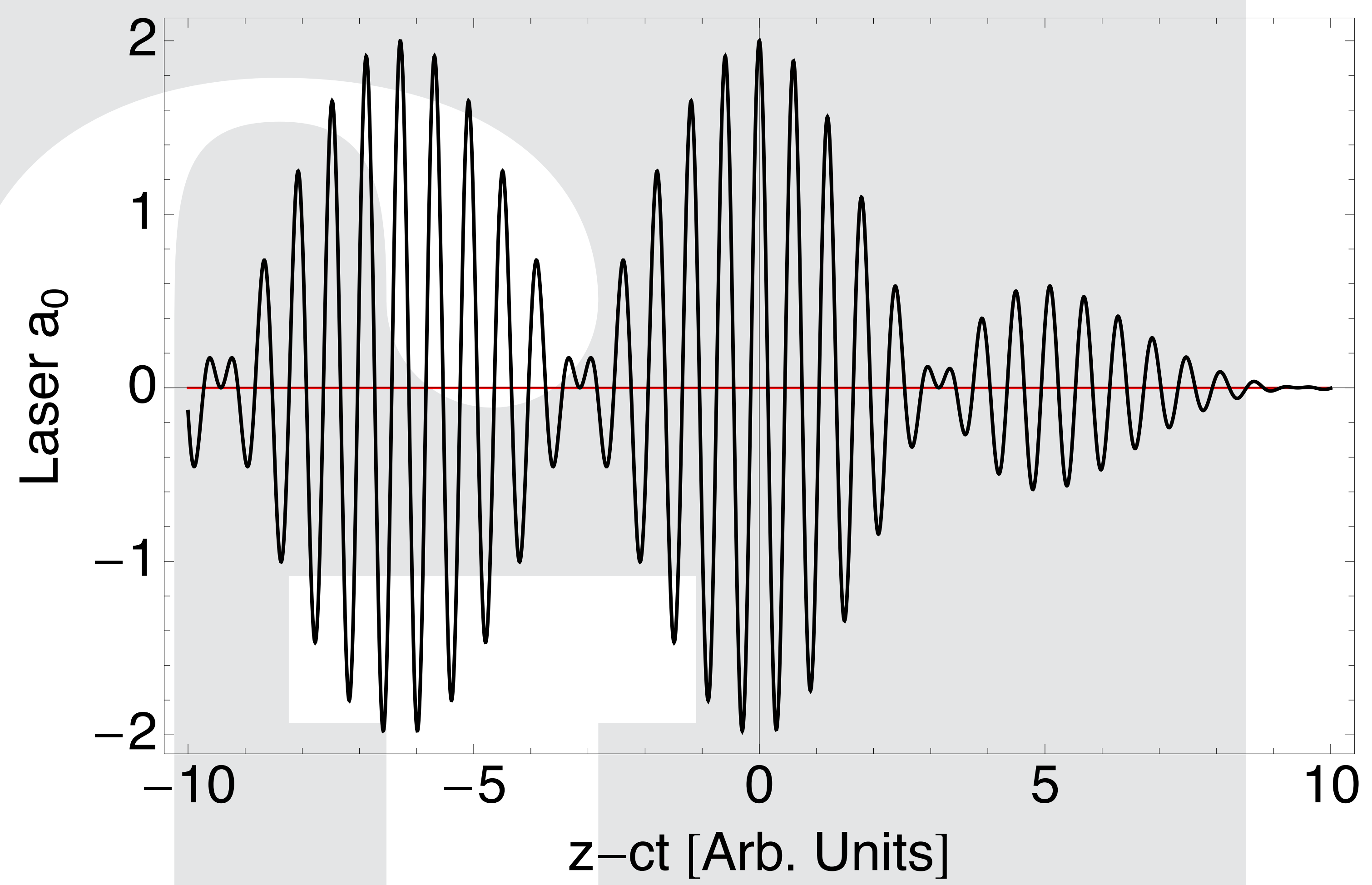




Example: plasma beat wave accelerator

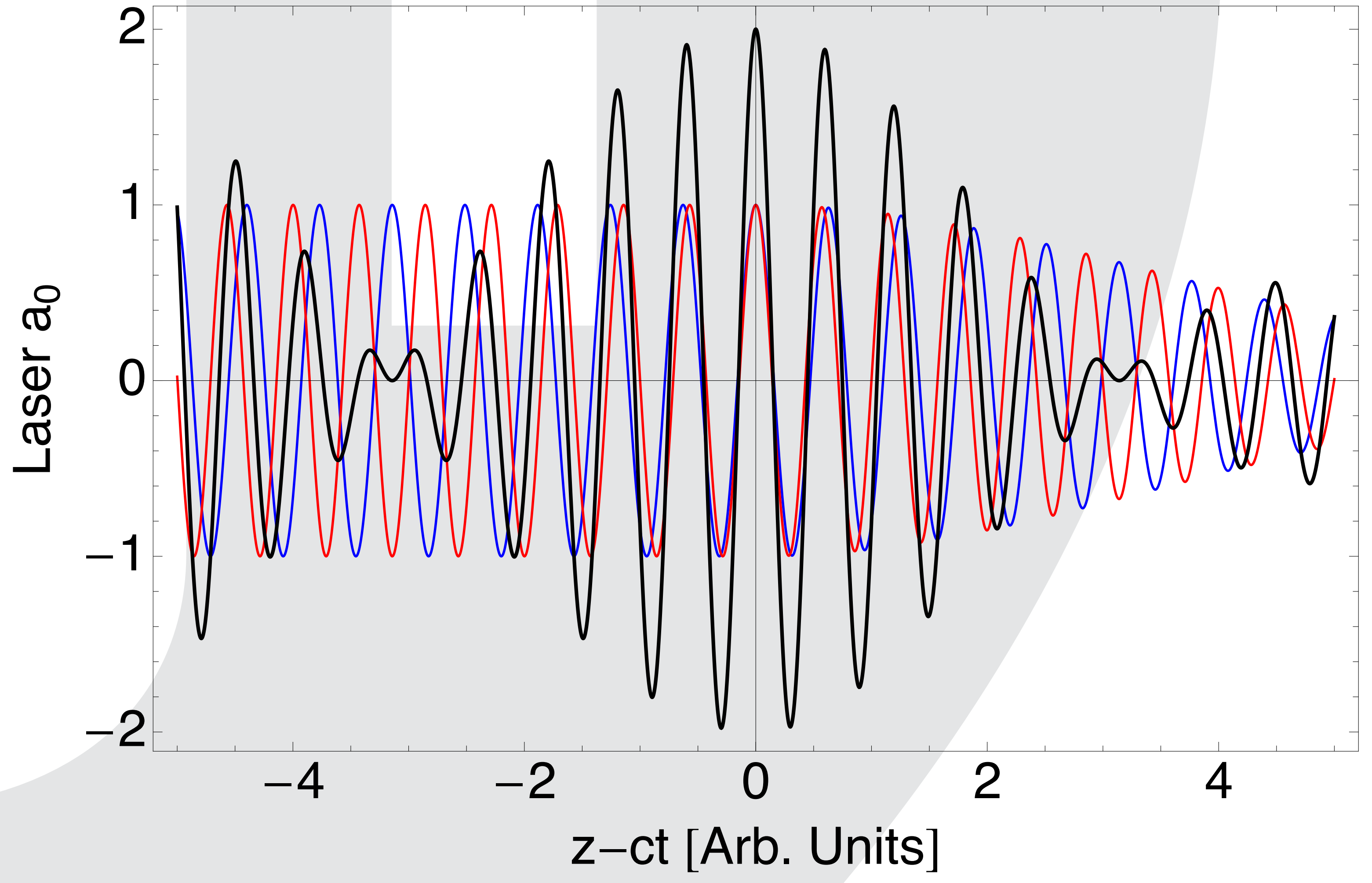
Simulate a plasma beat-wave accelerator:

- Super-impose three laser modes with frequencies differing by ω_p (e.g. $\omega_0 = 10, 11\ \omega_p$)
- Choose Laser length $\gg \lambda_p$



Questions:

1. why does the plasma wave amplitude increase along the pulse?
2. what happens if the the initial frequencies of the lasers are not separated by the plasma frequency? Why?
3. compare the trapping threshold, as a function of the peak laser a_0, for a standard LWFA (with pulse length smaller than λ_p) with the beat-wave accelerator. Which one is the lowest?
4. Decrease the amplitude of the laser side bands and run the simulation for longer times. What happens to the laser?



Example: plasma wakefield accelerator

Simulate a plasma wakefield accelerator:

- Use an ultra-relativistic particle beam as a driver
  - e.g. u_{fl} = 100, length ~ 10 c/ω_p, density ~ 0.3
- Choose plasma length $\gg \lambda_p$

Questions:

1. Why does the head of the driver loose energy?
2. What happens to the energy of the driver if it has a length comparable to λ_p?
3. What is the phase velocity of the plasma wave?
4. Can you observe plasma electron trapping and acceleration?