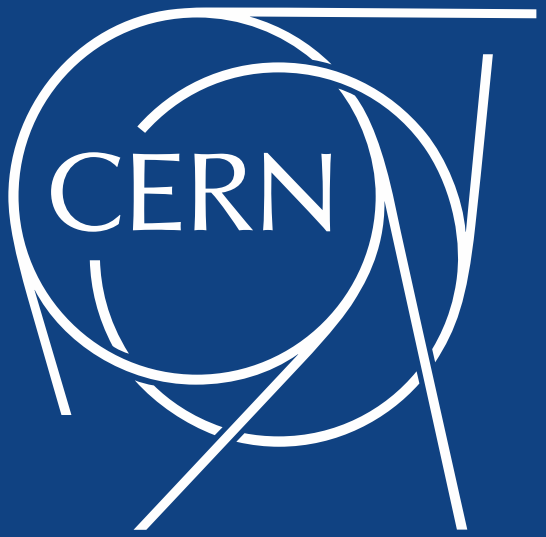

CERN

# Numerical Methods II

Kevin Li

With acknowledgements to:

H. Bartosik, X. Buffat, L.R. Carver, S. Hegglin, G. Iadarola, L. Mether, E. Metral, N. Mounet, A. Oeftiger, A. Romano, G. Rumolo, B. Salvant, M. Schenk

# Outline

Introduction to macroparticle models – implementations, applications and examples

- Part 1 – numerical modelling
  - Initialisation
  - Simple tracking
  - Chromaticity and detuning
  - Wakefields with examples
    - Constant wakes
    - Dipole wakes
    - TMCI & headtail modes
- Part 2 – electron cloud
  - Modelling of e-cloud interactions
  - PIC solvers
  - Application for e-cloud instabilities

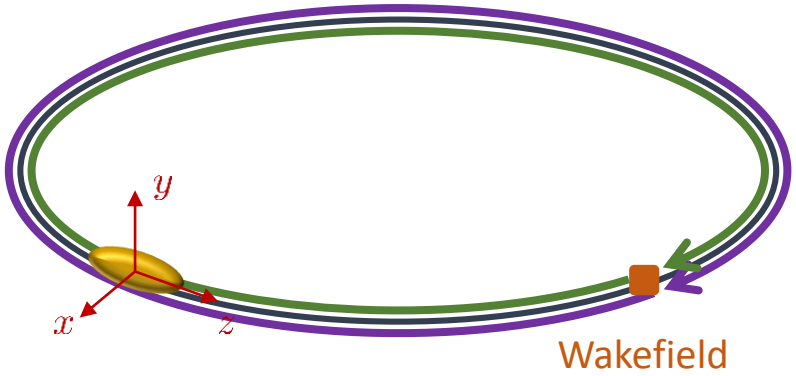
# Outline

Introduction to macroparticle models – implementations, applications and examples

- Part 1 – numerical modelling
  - Initialisation
  - Simple tracking
  - Chromaticity and detuning
  - Wakefields with examples
    - Constant wakes
    - Dipole wakes
    - TMCI & headtail modes
- Part 2 – electron cloud
  - Modelling of e-cloud interactions
  - PIC solvers
  - Application for e-cloud instabilities

# Summary – where are we?

- We are now ready to track a full turn including the interaction with wake fields




$$\left.\begin{pmatrix} x_i \\ x_i' \end{pmatrix}\right|_{k+1} = \mathcal{M}_i \left.\begin{pmatrix} x_i \\ x_i' \end{pmatrix}\right|_{k}$$

$$\left.\begin{pmatrix} z_i \\ \delta_i \end{pmatrix}\right|_{k+1} = \mathcal{I}\left[\left.\begin{pmatrix} z_i \\ \delta_i \end{pmatrix}\right|_{k}\right]$$

$$\left.\left(x_i'\right)\right|_{k+1} = \left.\left(x_i'\right)\right|_{k} + \mathcal{WK}$$

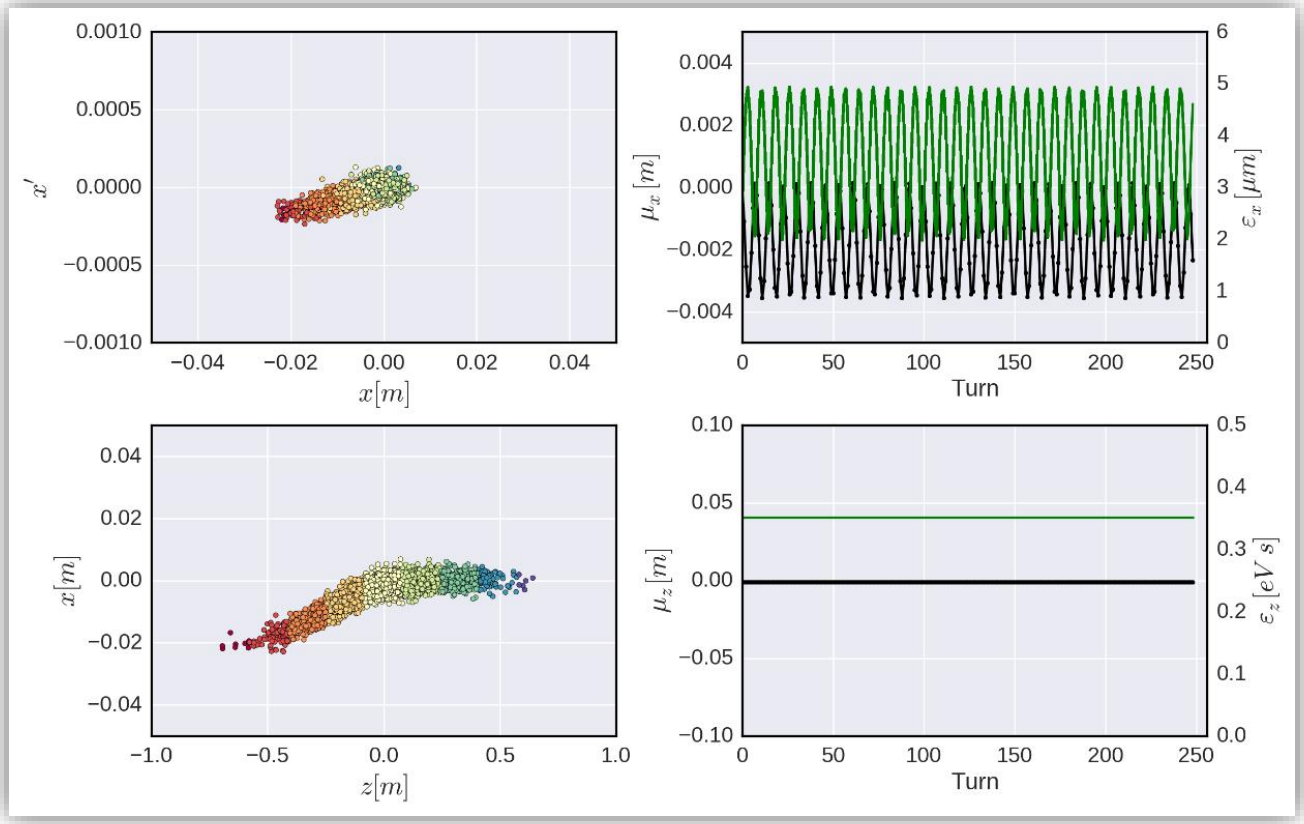
1. Initialise a macroparticle distribution with a given emittance
2. Update transverse coordinates and momenta according to the linear periodic transfer map – adjust the individual phase advance according to chromaticity and detuning with amplitude
3. Update the longitudinal coordinates and momenta according to the leap-frog integration scheme
4. Update momenta only (apply kicks) according to wake field generated kicks
5. Repeat turn-by-turn…

# Examples – constant wakes

$$\Delta x'[i] = \underbrace{-\frac{e^2}{m\gamma\beta^2c^2}\sum_{j=0}^{i} N[j]\cdot W_{Cx}[i-j]}$$

Dipolar term → orbit kick

Slice dependent change of closed orbit (if line density does not change)



# Examples – dipole wakes

$$\Delta x'[i] = -\frac{e^2}{m\gamma\beta^2c^2\,C}\sum_{j=0}^{i} N[j]\,\langle x\rangle[j]\cdot W_{Dx}[i-j]$$

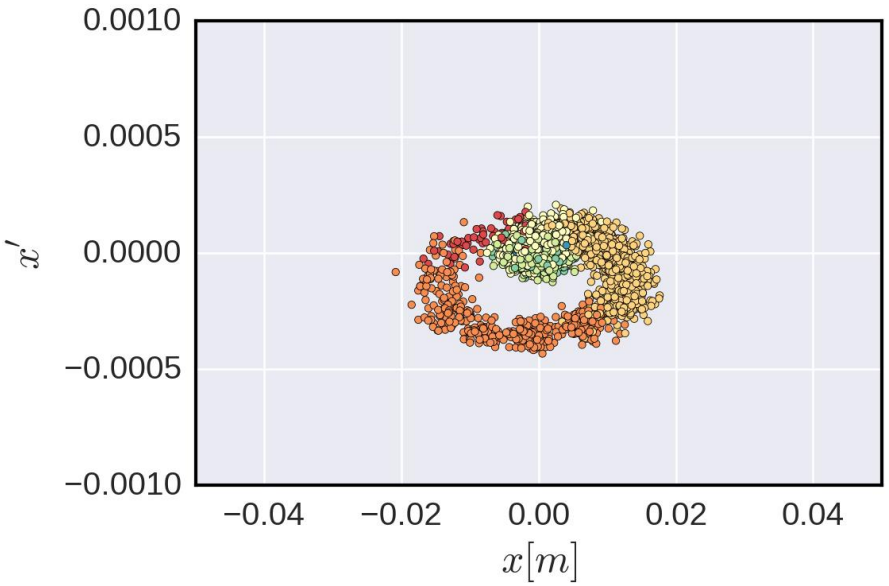
Dipolar term → orbit kick

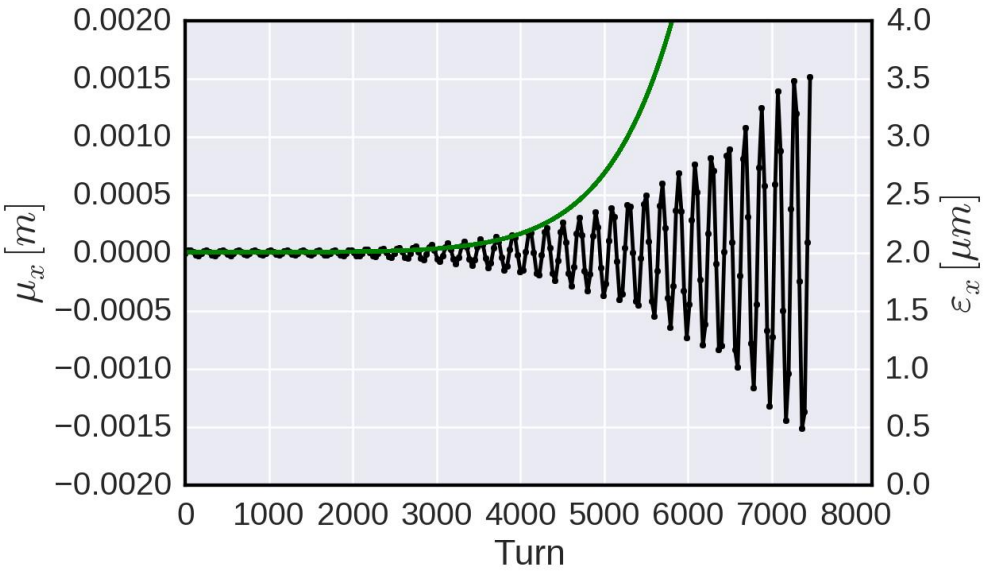
Offset dependent orbit kick
→ kicks can accumulate

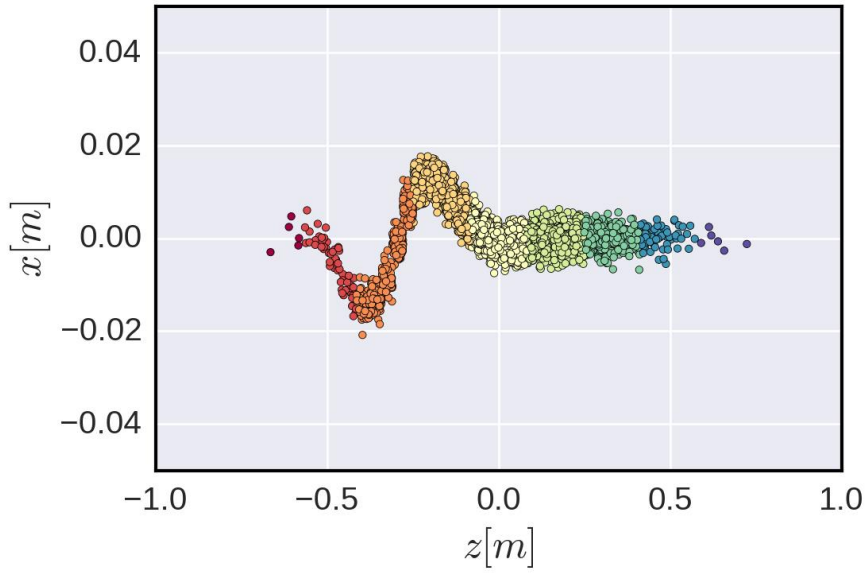
With synchrotron motion we can get into a feedback loop

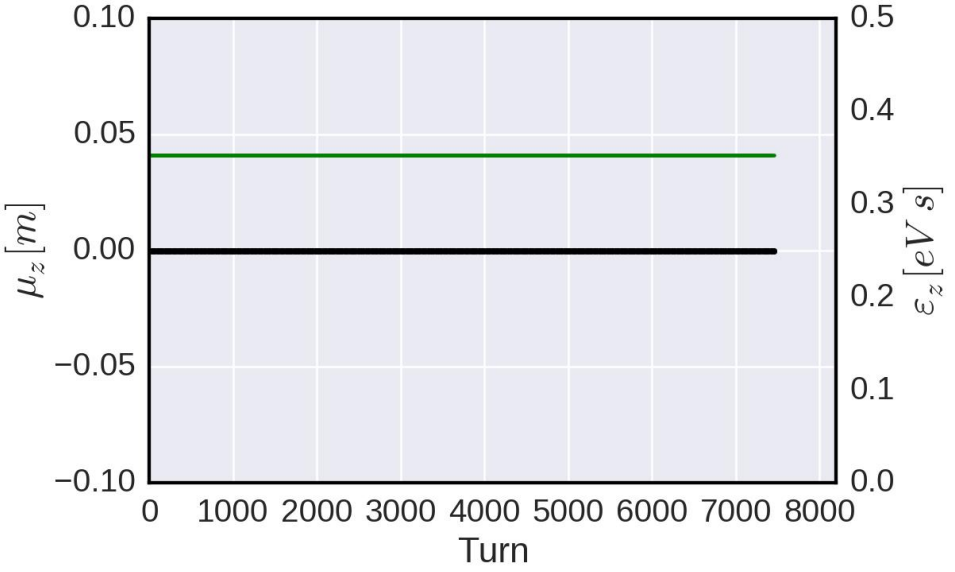
- Without synchrotron motion:
  kicks accumulate turn after turn – the beam is unstable → beam break-up in linacs
- With synchrotron motion:
  - Chromaticity = 0
    - Synchrotron sidebands are well separated → beam is stable
    - Synchrotron sidebands couple → (transverse) mode coupling instability
  - Chromaticity ≠ 0
    - Headtail modes → beam is unstable (can be very weak and often damped by non-linearities)

# Dipole wakes – beam break-up



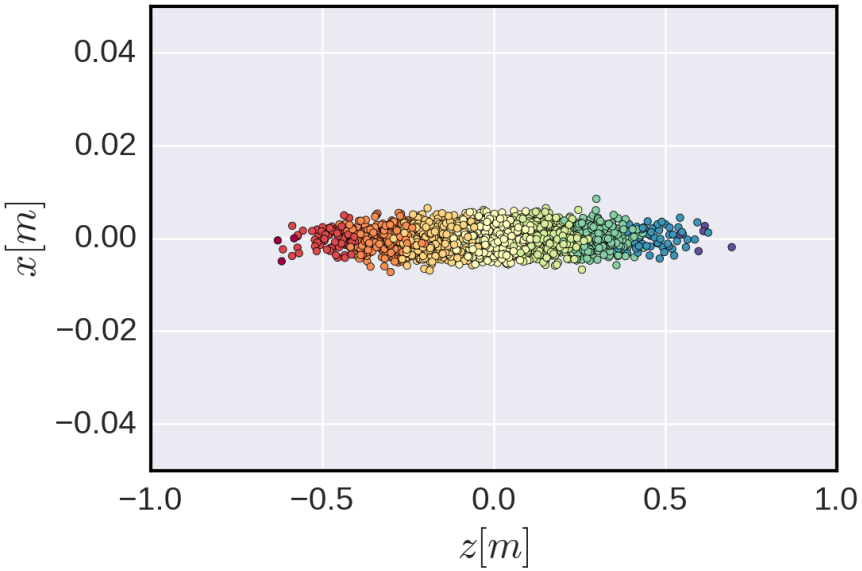



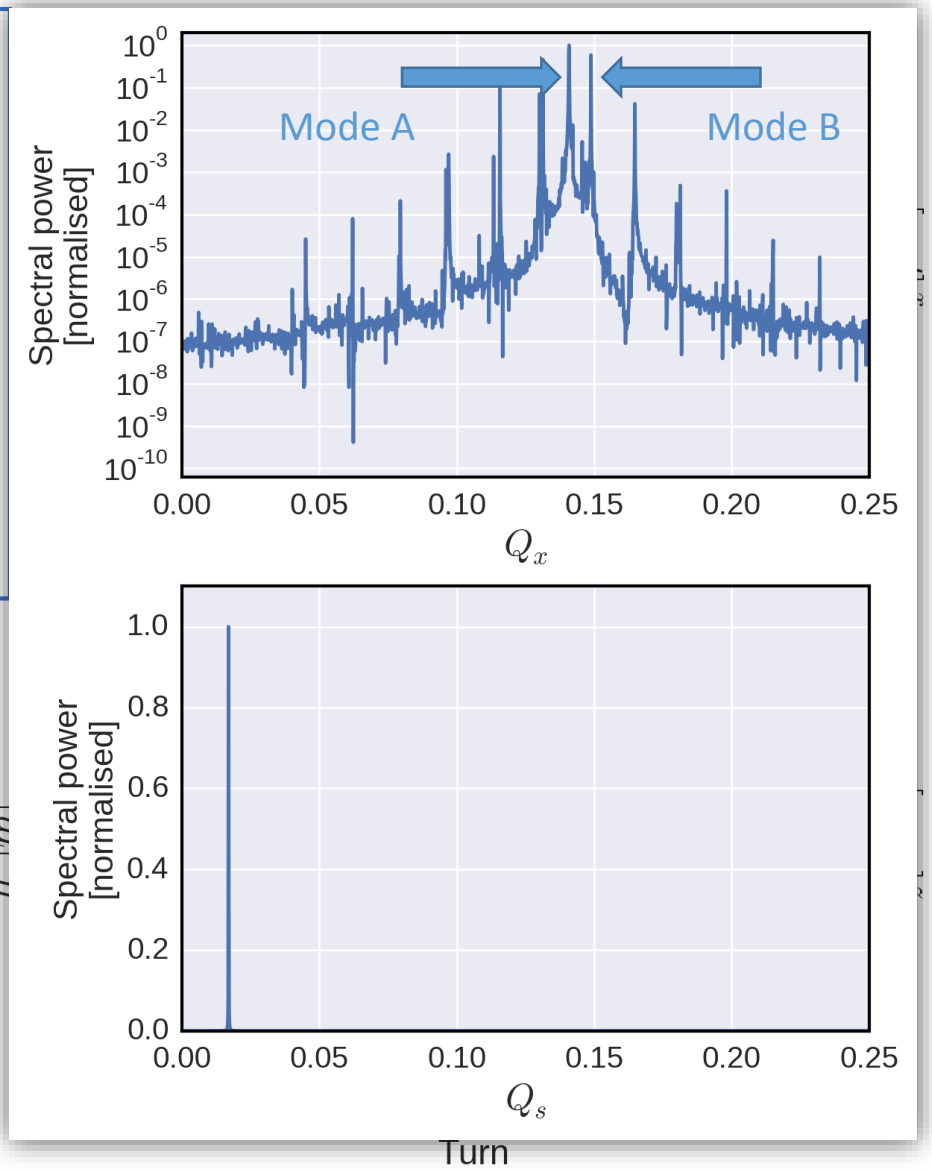



# Dipole wakes – TMCI below threshold

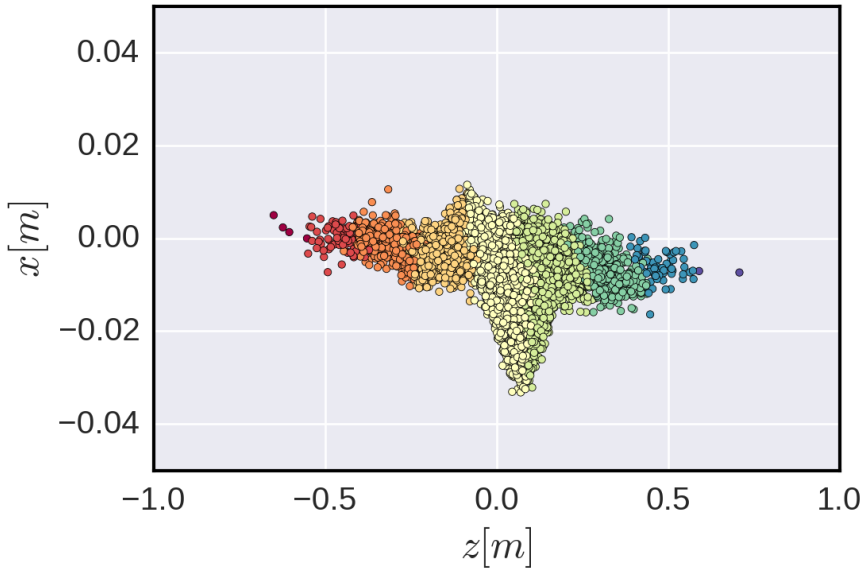
As the intensity increases the coherent modes shift – here, modes A and B are approaching each other

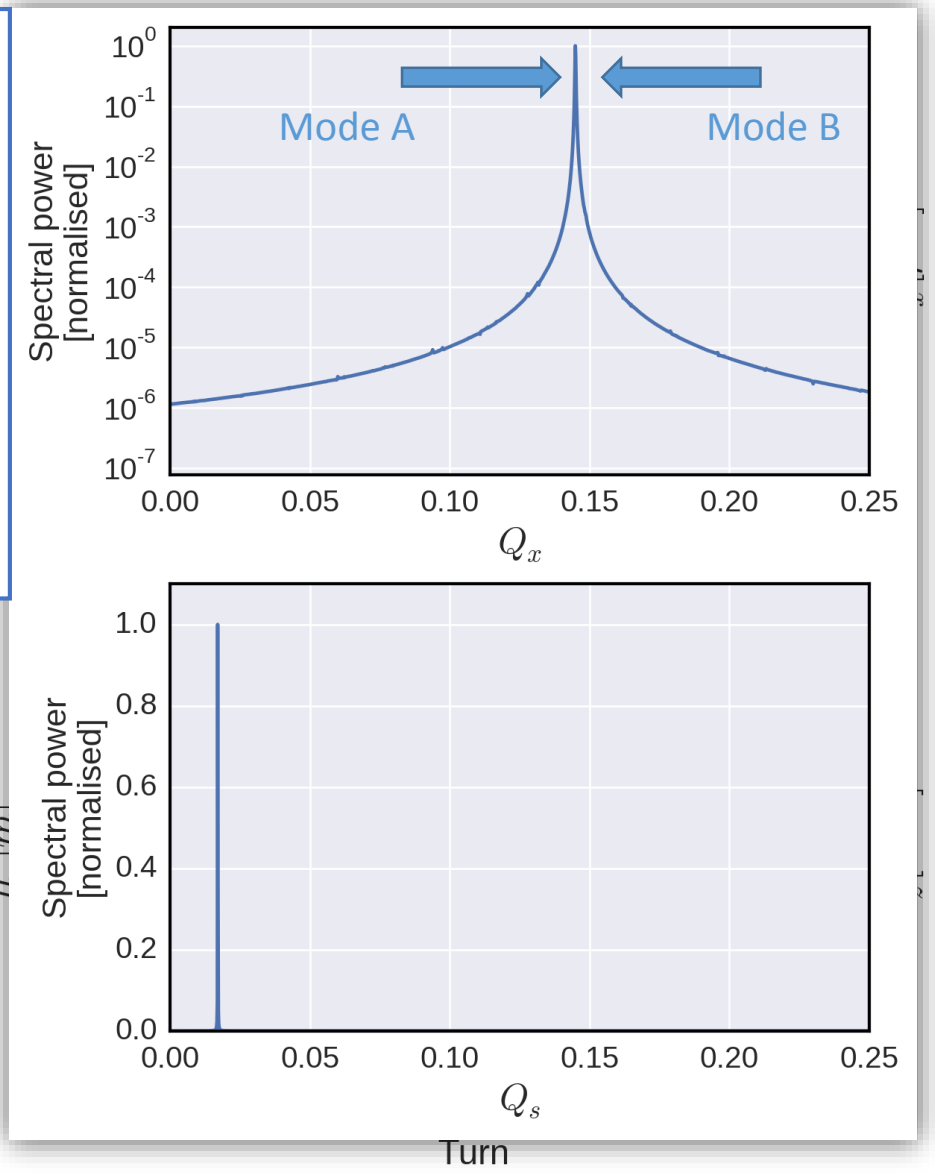


# Dipole wakes – TMCI above threshold

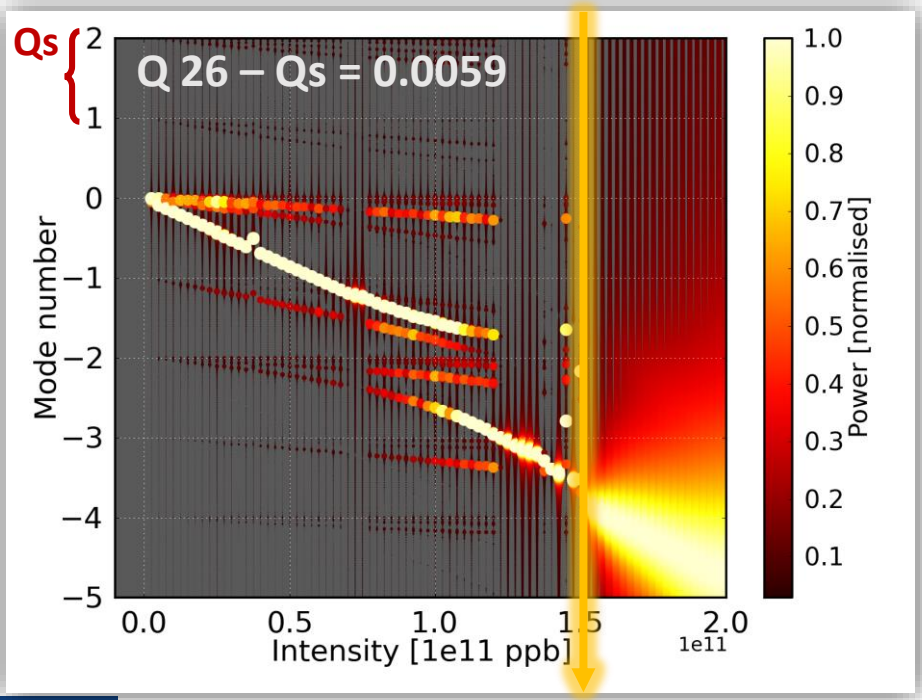
When the two modes merge a fast coherent instability arises – the transverse mode coupling instability (TMCI) which often is a hard intensity limit in many machines



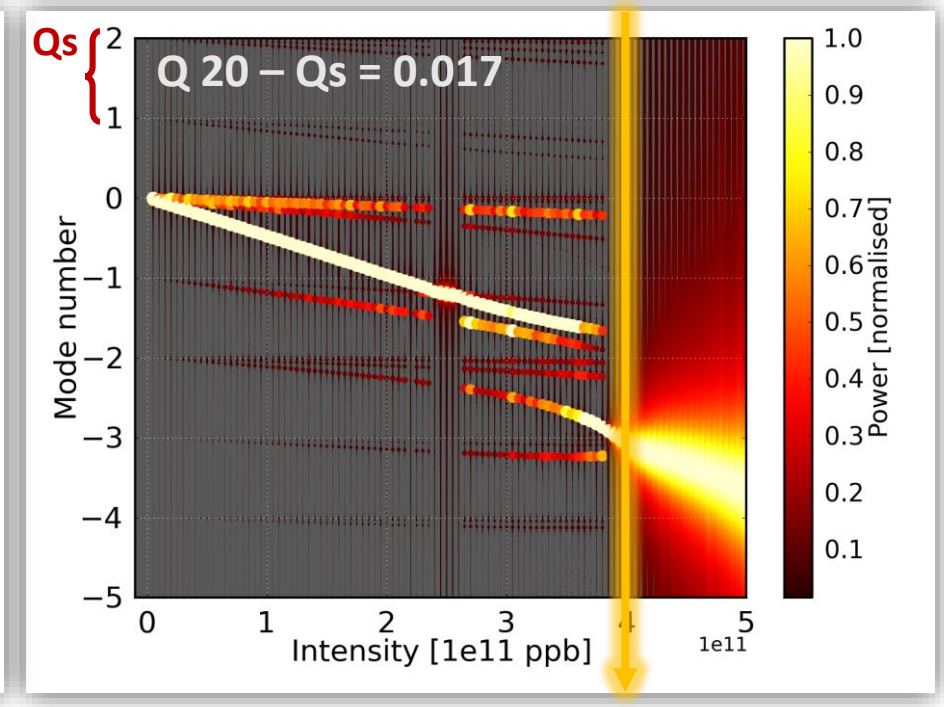



# Raising the TMCI threshold – SPS Q20 optics

- In simulations we have the possibility to perform scans of variables, e.g. we can run 100 simulations in parallel changing the beam intensity
- We can then perform a spectral analysis of each simulation…
- … and stack all obtained plot behind one another to obtain…
- … the typical visualization plots of TMCI



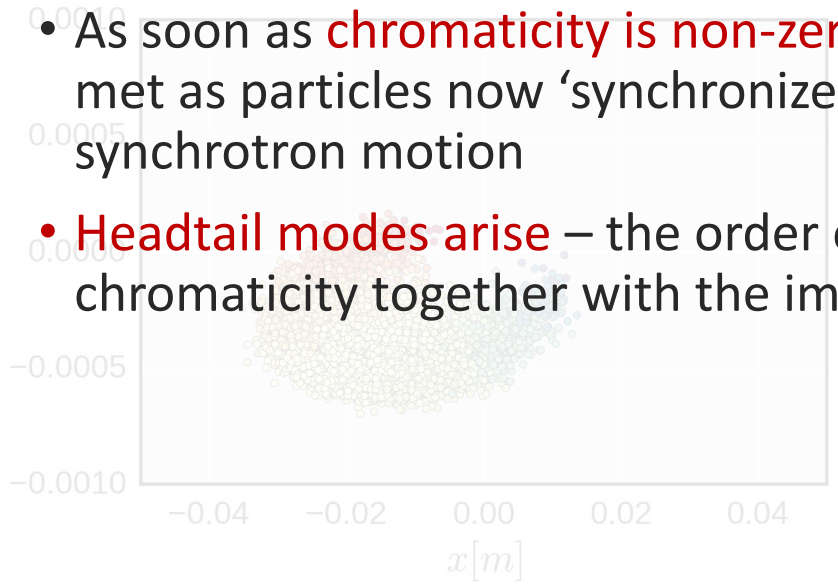

TMCI threshold

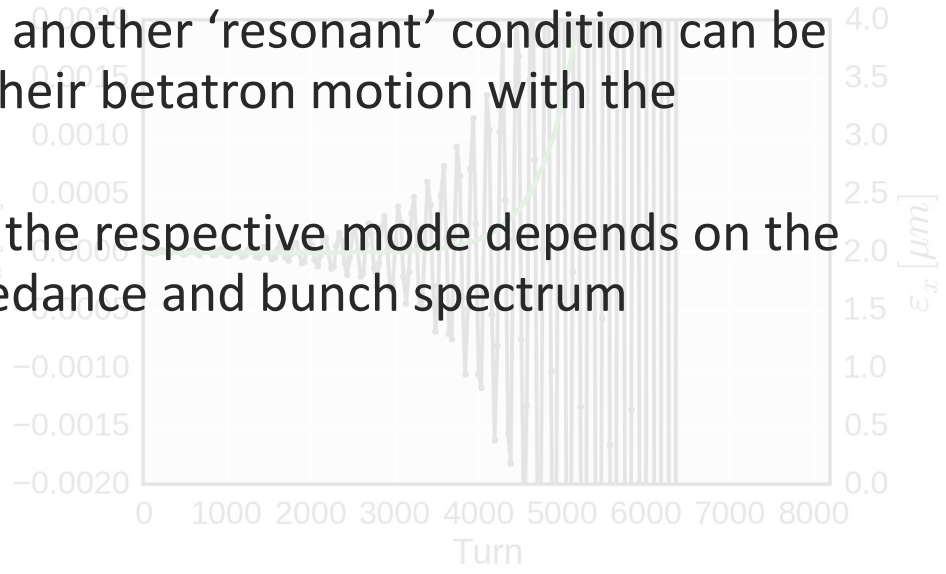



TMCI threshold

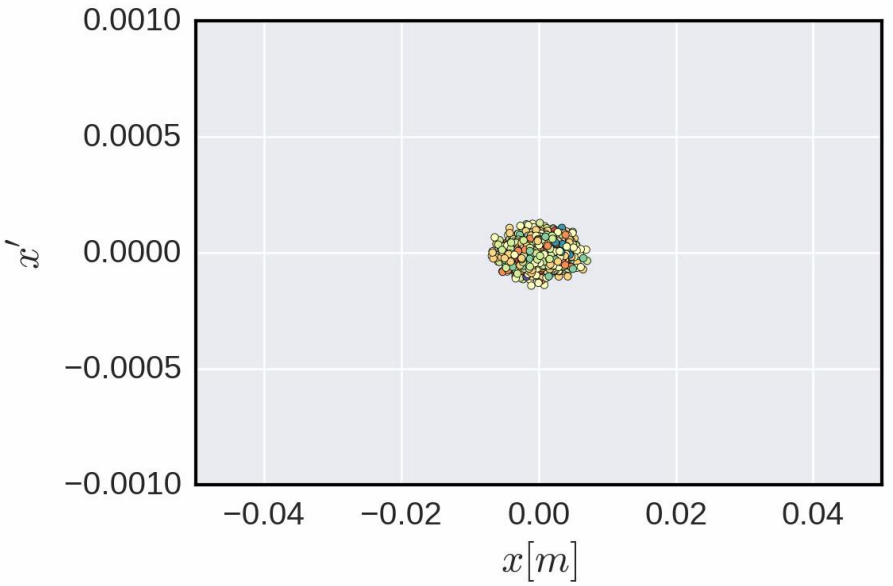
# Dipole wakes – headtail modes

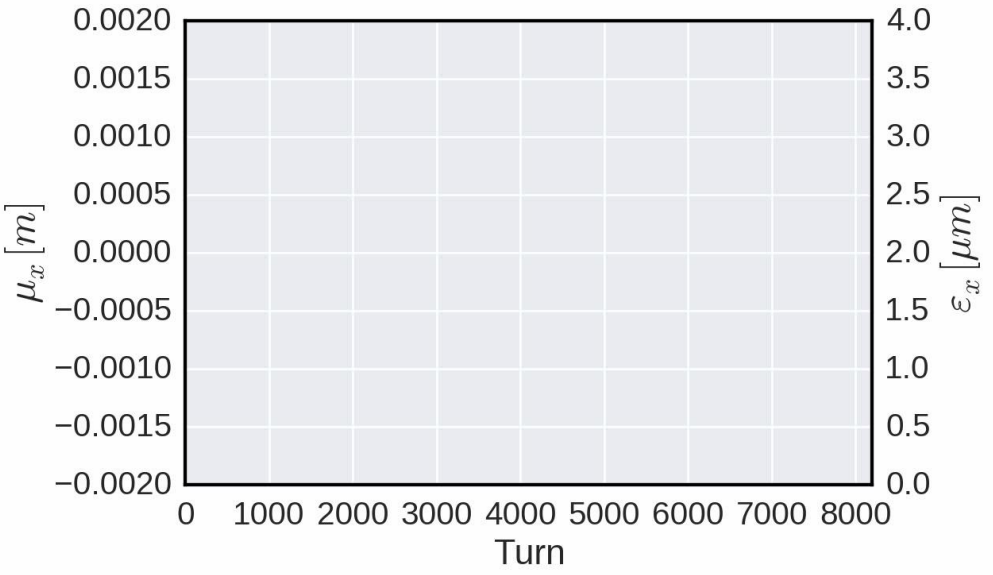
- As soon as chromaticity is non-zero, another ‘resonant’ condition can be met as particles now ‘synchronize’ their betatron motion with the synchrotron motion
- Headtail modes arise – the order of the respective mode depends on the chromaticity together with the impedance and bunch spectrum

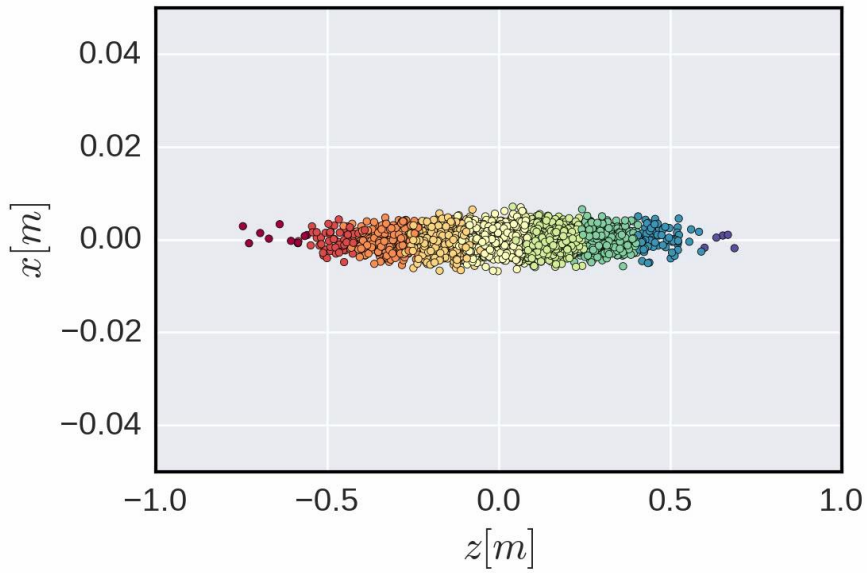


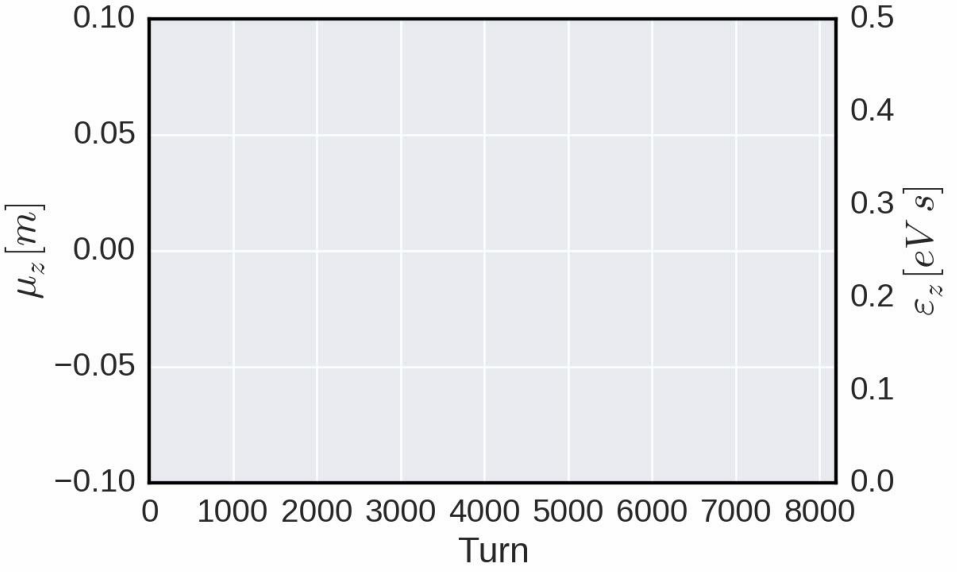



# Dipole wakes – headtail modes

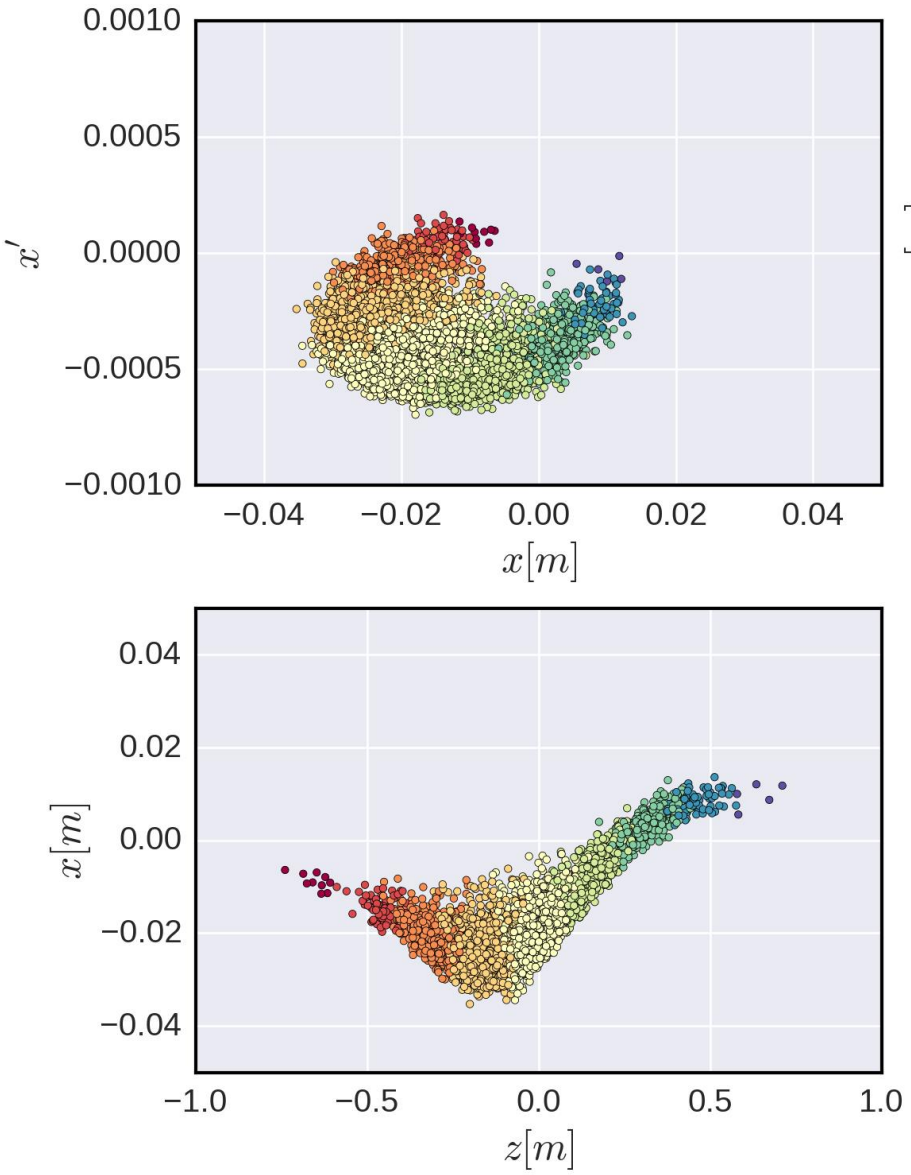


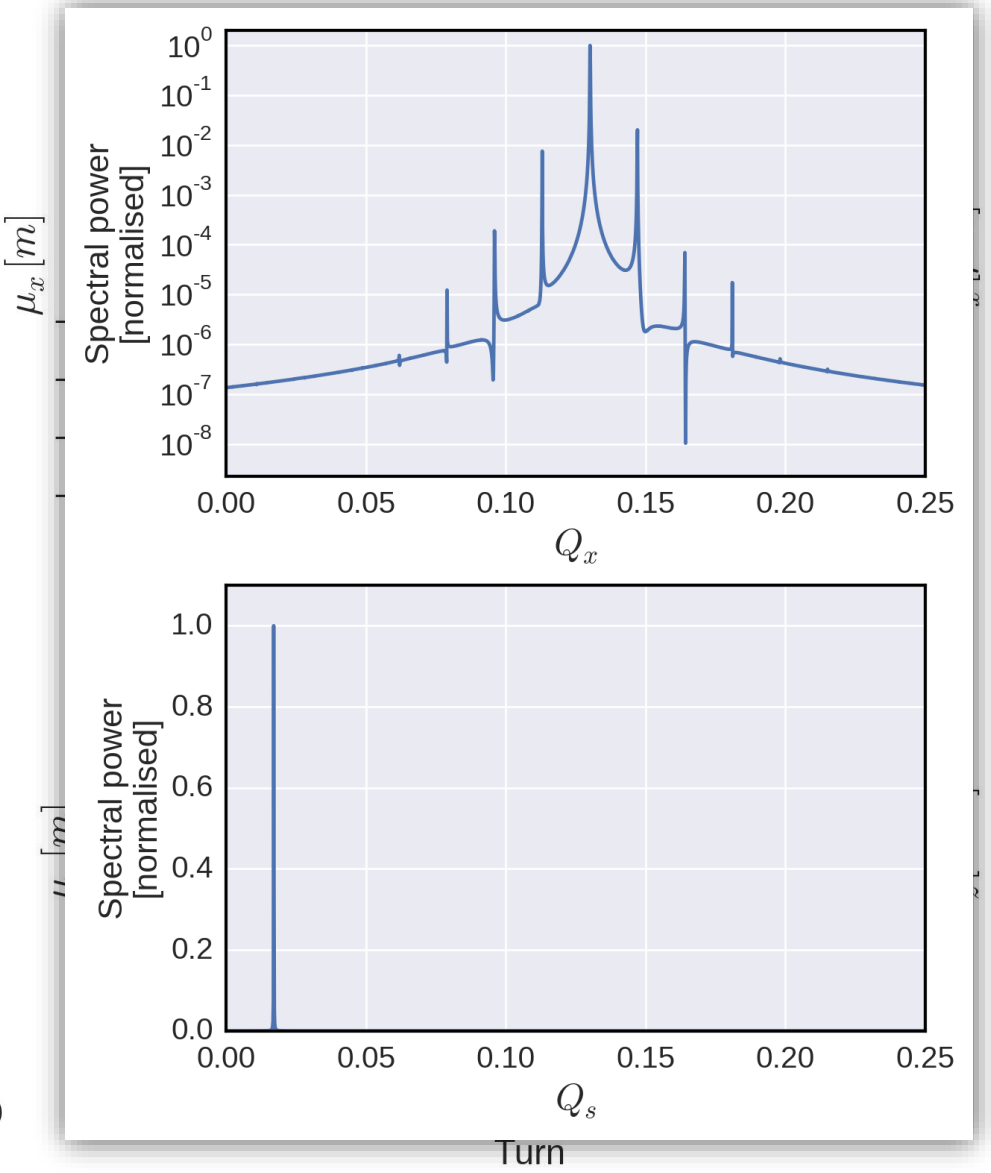



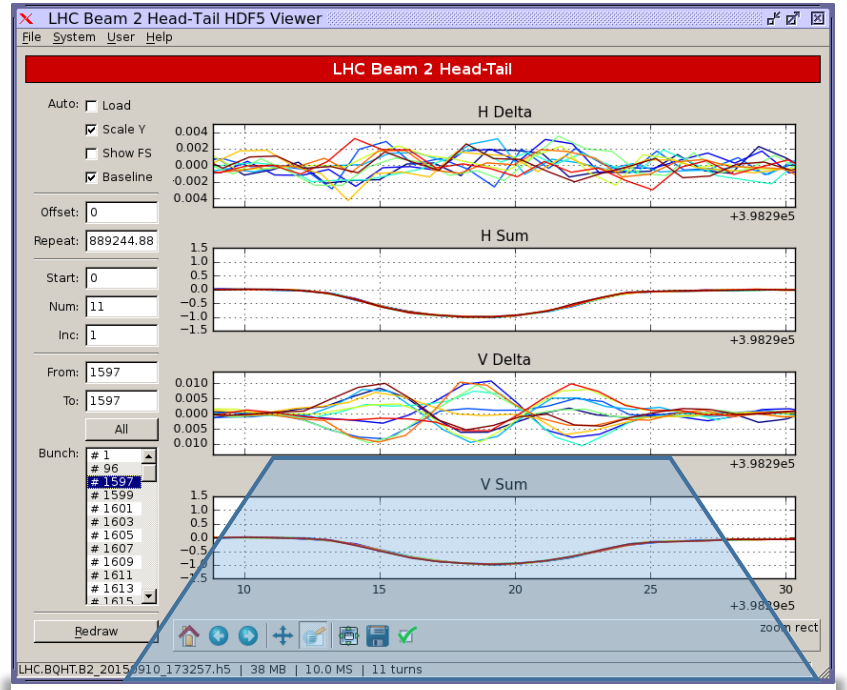



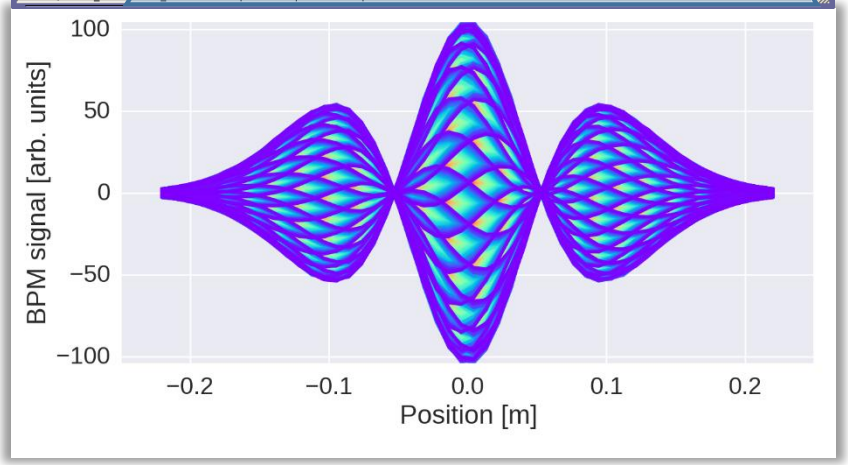
# Dipole wakes – headtail modes

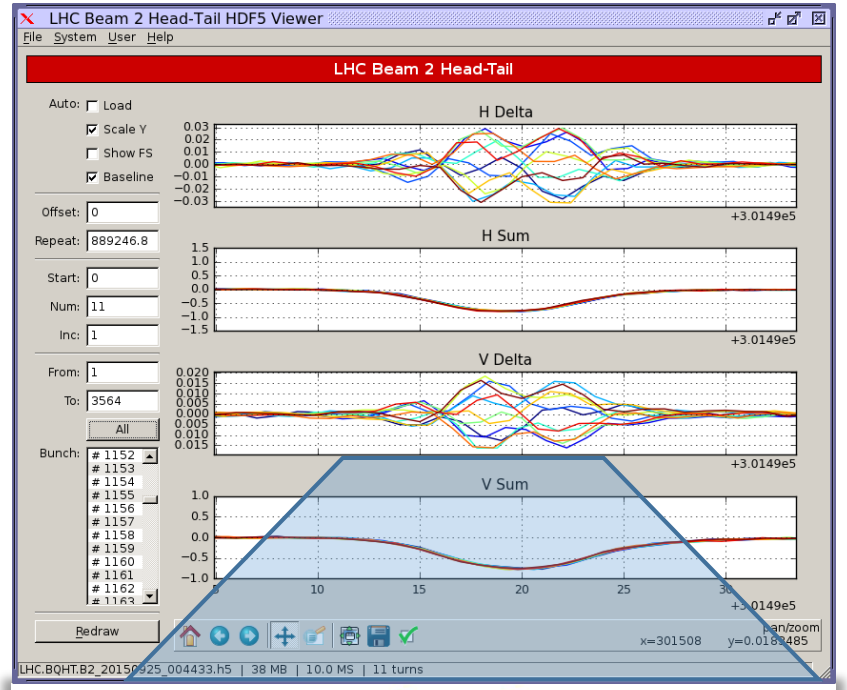


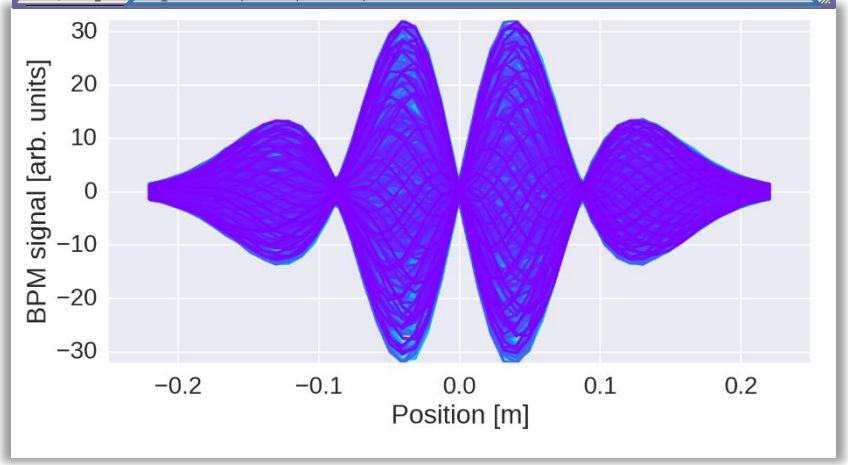



# Example: Headtail modes in the LHC



# End of part I

- Numerical methods allow us
  - to study conditions not realizable in a machine
  - to disentangle effects
  - to use unprecedented analysis tools
- Macroparticle models closely resemble real systems and are relatively easy to implement
- We have learned how to model and implement macroparticle simulations to study intensity effects in circular accelerators

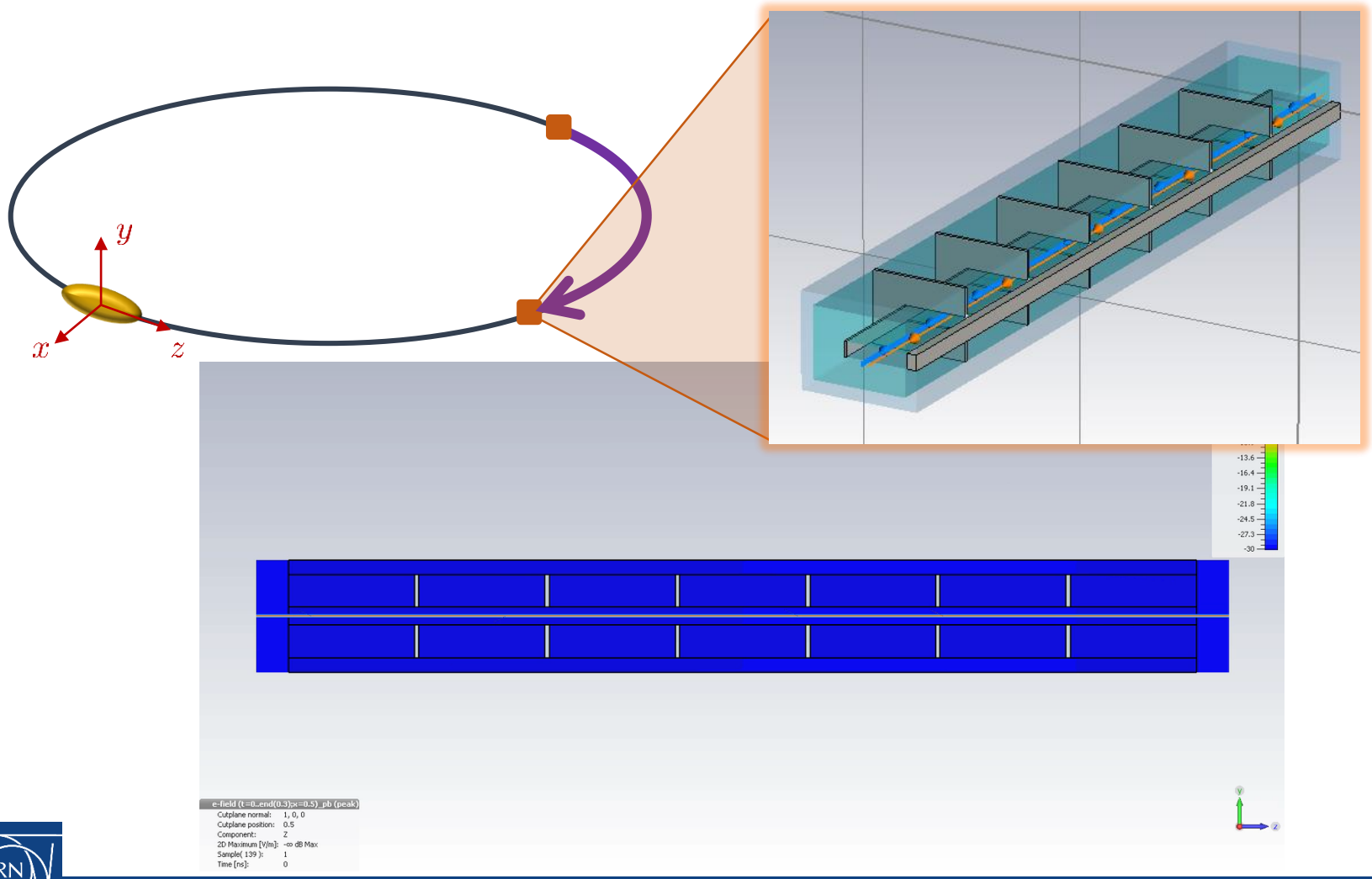
# Outline

Introduction to macroparticle models – implementations, applications and examples

- Part 1 – numerical modelling
  - Initialisation
  - Simple tracking
  - Chromaticity and detuning
  - Wakefields with examples
    - Constant wakes
    - Dipole wakes
    - TMCI & headtail modes
- Part 2 – electron cloud
  - Modelling of e-cloud interactions
  - PIC solvers
  - Application for e-cloud instabilities

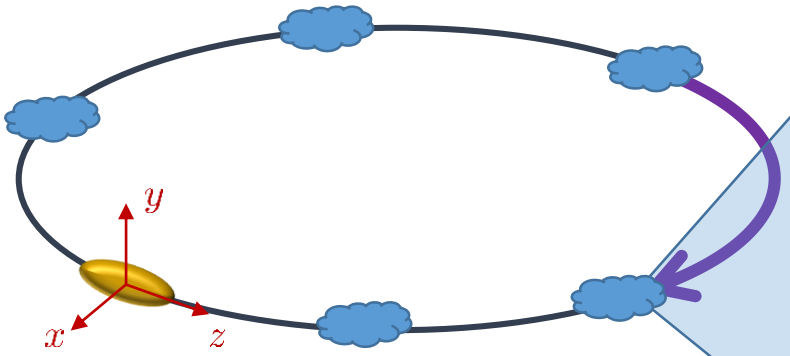
# Accelerator beam system - wakefields

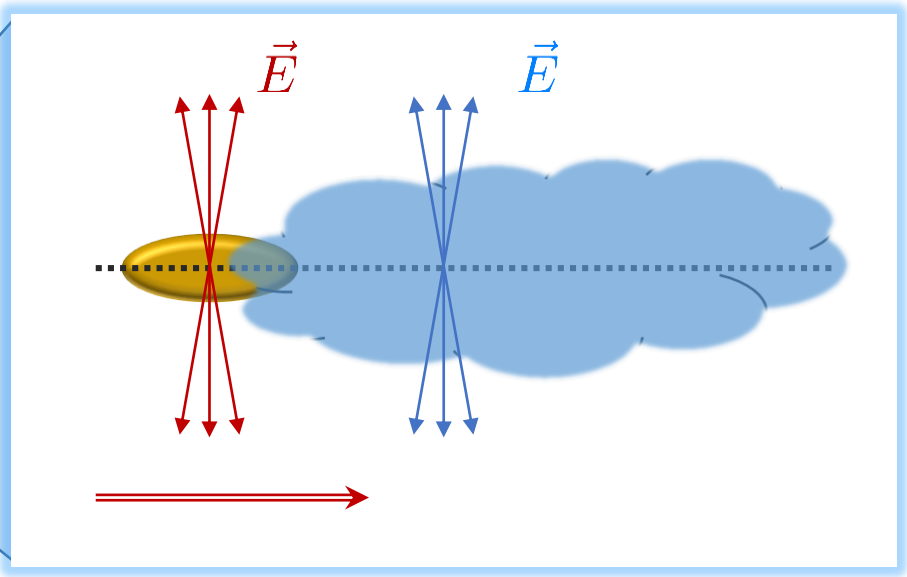
- Our first ‘real’ collective interaction from impedances



# Accelerator beam system – electron clouds

- Two stream collective interaction – much more involved




For this we need to solve

$$\Delta\,\phi(x,y)_{p^+} = -\frac{\rho_{p^+}(x,y)}{\varepsilon_0}$$

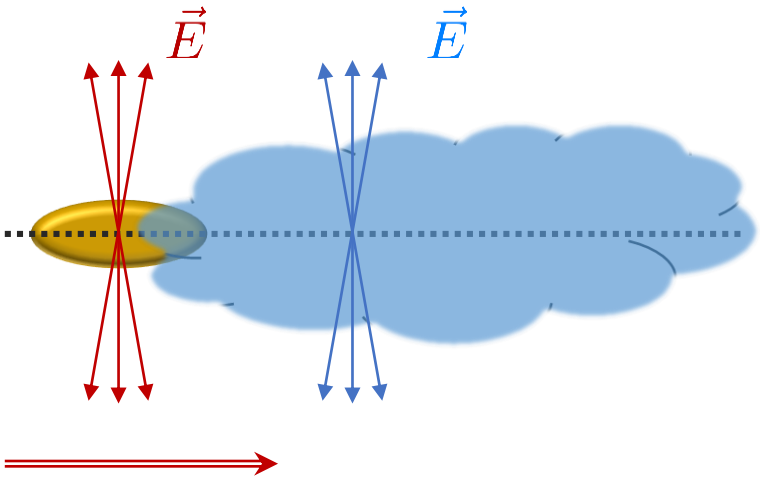
$$\Delta\,\phi(x,y)_{e^-} = -\frac{\rho_{e^-}(x,y)}{\varepsilon_0}$$

and apply the corresponding kicks to the cloud and the beam

Approximations here:

- The beam is ultra-relativistic
- The electron velocity is well below c
- The electron cloud has a low aspect ratio

# Accelerator beam system – electron clouds




| count | x | x' | y | y' |
|---|---|---|---|---|
| 0 | … | … | … | … |
| 1 | … | … | … | … |
| 2 | … | … | … | … |
| 3 | … | … | … | |
| 4 | | | | |

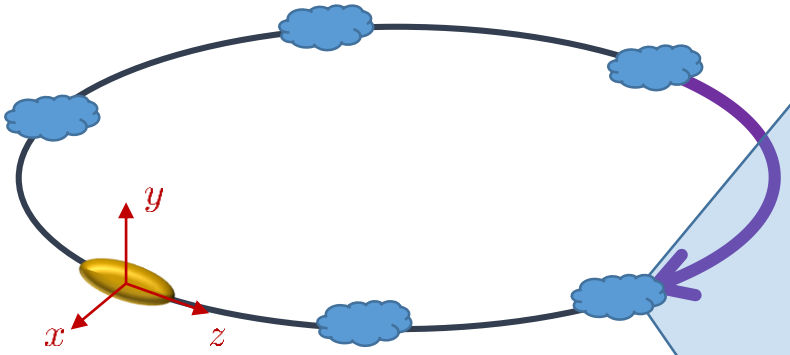
| count | x | y | phi |
|---|---|---|---|
| 0 | … | … | … |
| 1 | … | … | … |
| 2 | … | … | … |
| 3 | … | … | … |
| 4 | … | … | … |
| 5 | … | … | … |

| count | x | x' | y | y' | z | delta |
|---|---|---|---|---|---|---|
| 0 | … | … | … | … | … | … |
| 1 | … | … | … | … | … | … |
| 2 | … | … | … | … | … | … |
| 3 | … | … | … | … | … | … |
| 4 | … | … | … | … | … | … |
| 5 | … | … | … | … | … | … |

More memory, more computation steps – overall more challenging

- Two macroparticle systems now need to be solved simultaneously
- The electric field evaluation usually is the most time-consuming step and should be done efficiently
- Keep track of macroparticle systems and fields

# Electron clouds in a drift section



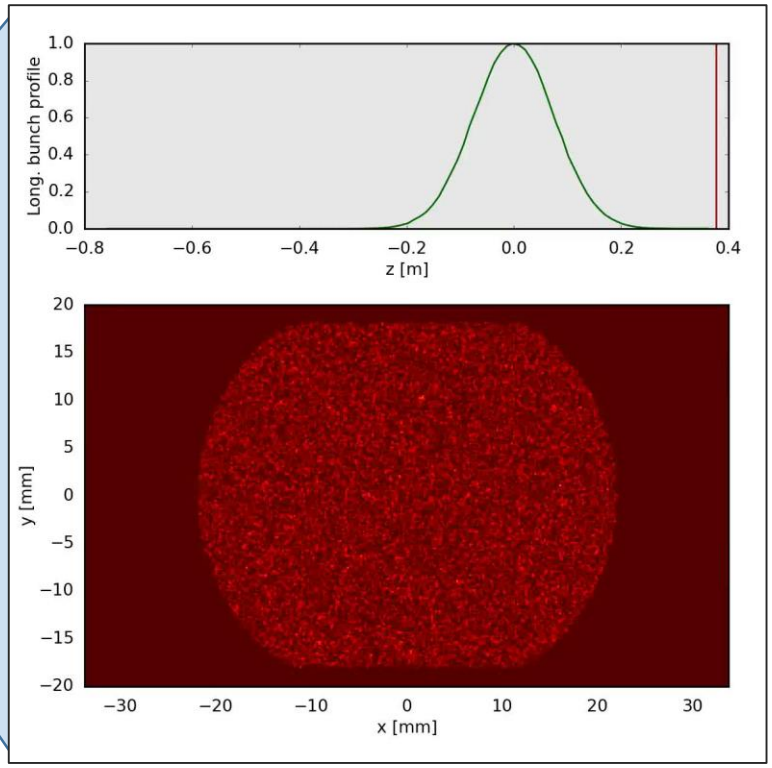

For this we need to solve

$$\Delta\,\phi(x,y)_{p^+} = -\frac{\rho_{p^+}(x,y)}{\varepsilon_0}$$

$$\Delta\,\phi(x,y)_{e^-} = -\frac{\rho_{e^-}(x,y)}{\varepsilon_0}$$

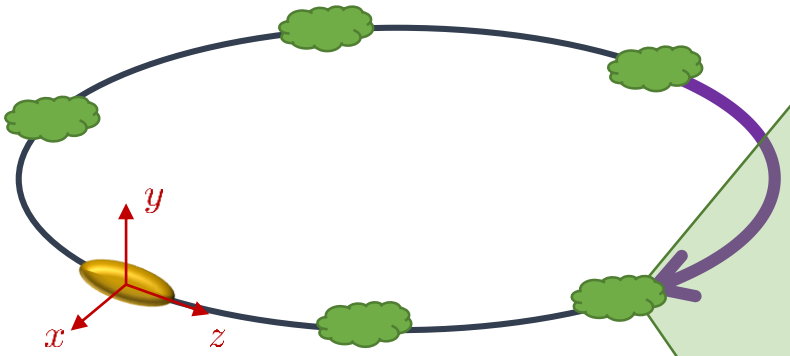
and apply the corresponding kicks to the cloud and the beam

- Two stream collective interaction – much more involved




- Beam passage leads to a pinch of the cloud which in turn acts back on the beam – differently each turn

# Electron clouds in a bending magnet



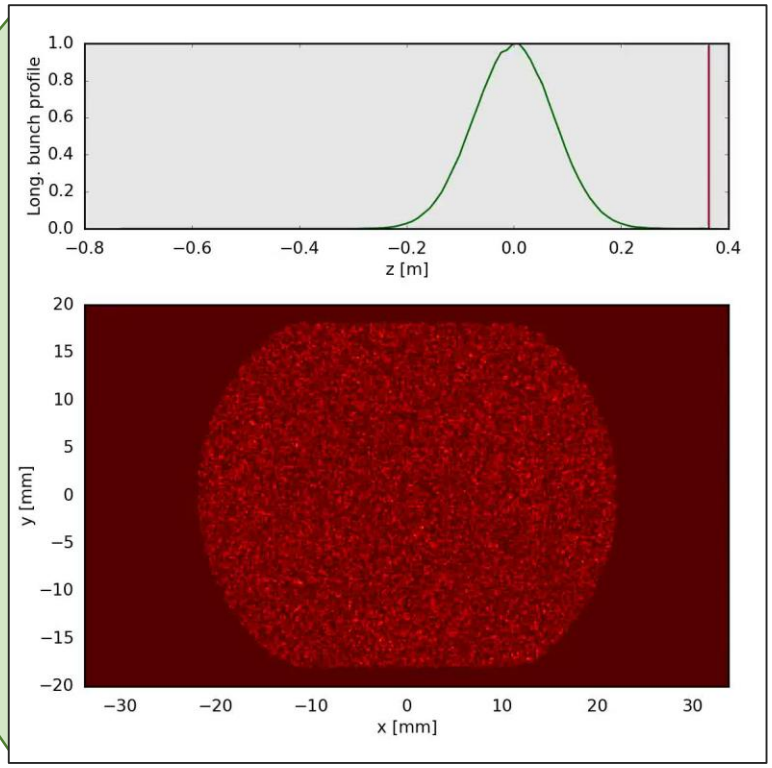

For this we need to solve

$$\Delta\,\phi(x,y)_{p^+} = -\frac{\rho_{p^+}(x,y)}{\varepsilon_0}$$

$$\Delta\,\phi(x,y)_{e^-} = -\frac{\rho_{e^-}(x,y)}{\varepsilon_0}$$

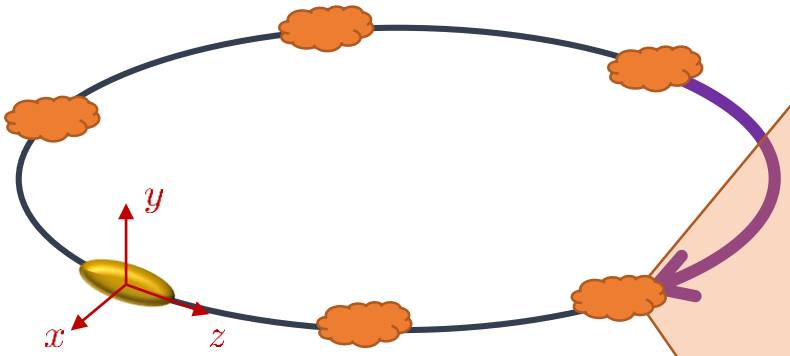
and apply the corresponding kicks to the cloud and the beam

- Two stream collective interaction – much more involved




- Beam passage leads to a pinch of the cloud which in turn acts back on the beam – differently each turn

# Electron clouds in a quadrupole magnet



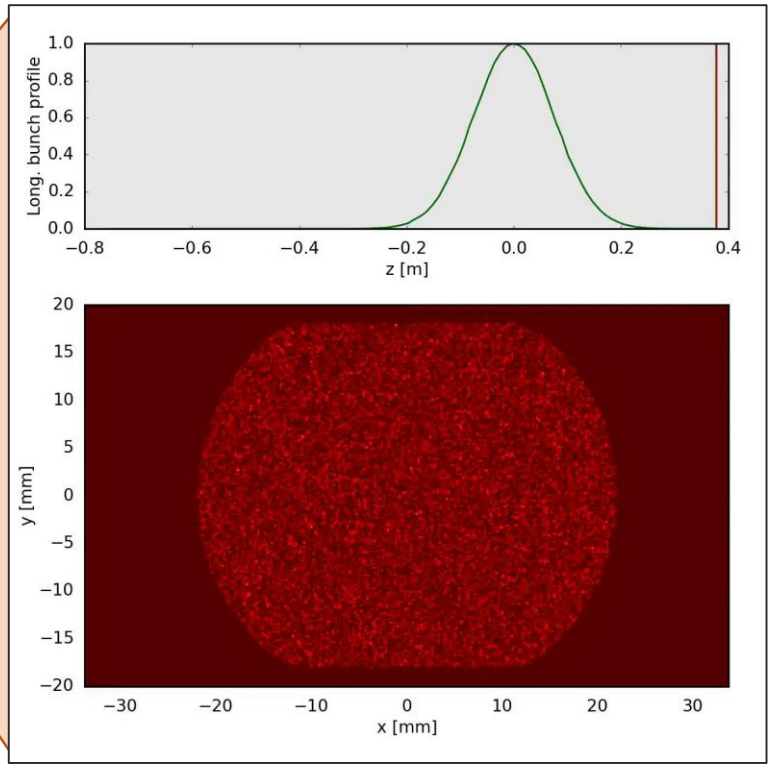

For this we need to solve

$$\Delta\,\phi(x,y)_{p^+} = -\frac{\rho_{p^+}(x,y)}{\varepsilon_0}$$

$$\Delta\,\phi(x,y)_{e^-} = -\frac{\rho_{e^-}(x,y)}{\varepsilon_0}$$

and apply the corresponding kicks to the cloud and the beam

- Two stream collective interaction – much more involved




- Beam passage leads to a pinch of the cloud which in turn acts back on the beam – differently each turn

# Accelerator-beam system – e-cloud

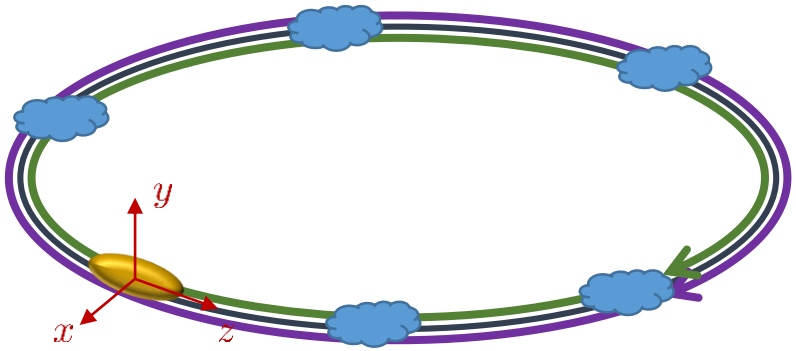
$$\left.\begin{pmatrix} x_i \\ x_i' \end{pmatrix}\right|_{k+1} = \mathcal{M}_i \left.\begin{pmatrix} x_i \\ x_i' \end{pmatrix}\right|_{k}$$

$$\left.\begin{pmatrix} z_i \\ \delta_i \end{pmatrix}\right|_{k+1} = \mathcal{I}\left[\left.\begin{pmatrix} z_i \\ \delta_i \end{pmatrix}\right|_{k}\right]$$

$$\Delta\vec{x}'[i] = -\frac{e^2}{m\gamma\beta^2c^2}\vec{E}_{e^-}\,C$$

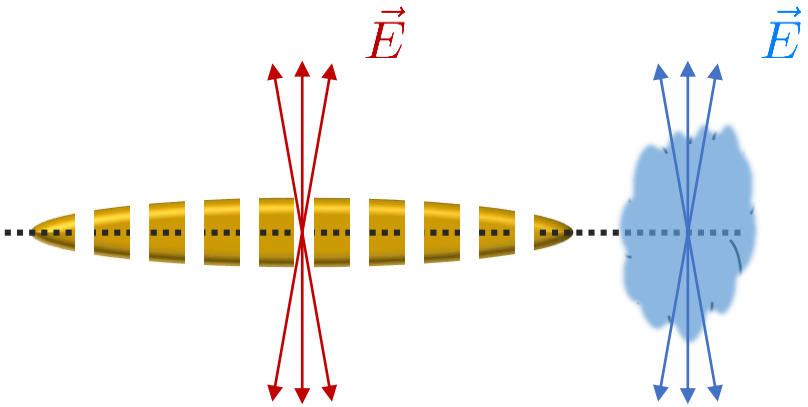
$$\Delta\dot{\vec{x}} = -\frac{e}{m}\left(\vec{E}_{p^+}[i] + \frac{\dot{\vec{x}}\times\vec{B}}{c}\right)\Delta t$$

Particles in/fields from slice i



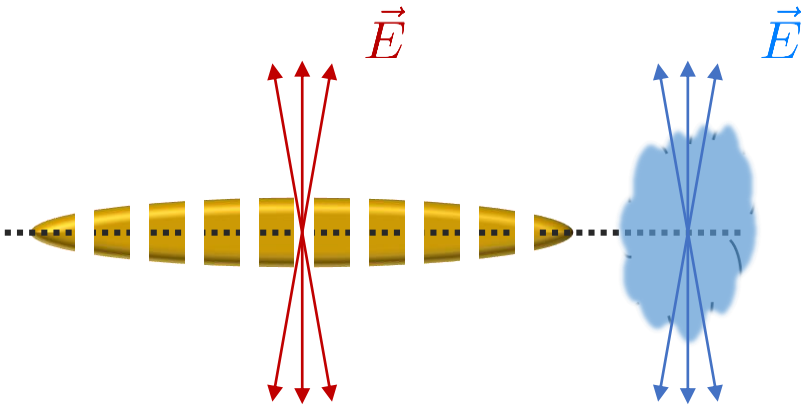

- Basic loop of tracking with wake fields:
  - Transport beam along segment to interaction point
  - Perform e-cloud interaction – requires electric fields from macroparticle distributions
  - Electric field computation in this case conveniently handled with Particle-In-Cell algorithm

# E-cloud beam system




- PIC stands for Particle-In-Cell
- We use this method to compute fields generated by particles to solve e.g. the Poisson equation
- Electron motion occurs at the time scale of a slice of a bunch length → track single slices through the e-cloud and apply integrated kicks

# E-cloud beam system




Solve

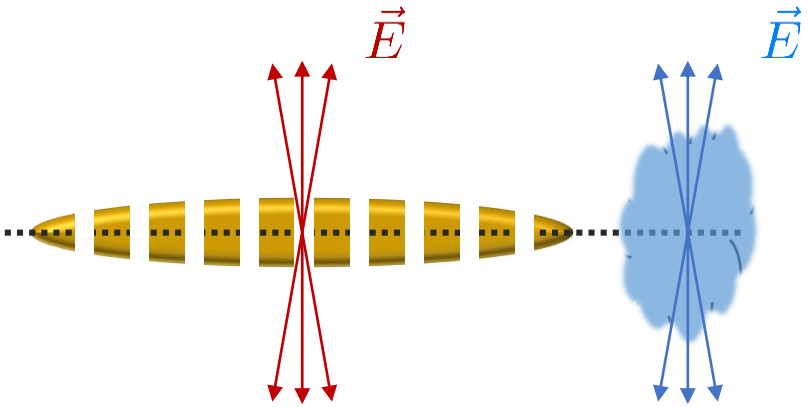
$$\Delta\,\phi(x,y)_{p^+} = -\frac{\rho_{p^+}(x,y)}{\varepsilon_0}$$

$$\Delta\,\phi(x,y)_{e^-} = -\frac{\rho_{e^-}(x,y)}{\varepsilon_0}$$

using PIC method.

- PIC stands for Particle-In-Cell
- We use this method to compute fields generated by particles to solve e.g. the Poisson equation
- Electron motion occurs at the time scale of a slice of a bunch length → track single slices through the e-cloud and apply integrated kicks
  - Compute electric fields from one slice and from e-cloud
  - Apply kicks to protons
  - Advance electrons by one slice length – this is a multi-scale dynamics problem (fast cyclotron motion superposed to slower guiding center drift) → Boris algorithm for tracking (per macroparticle)
  - Track next slice through e-cloud

# E-cloud beam system

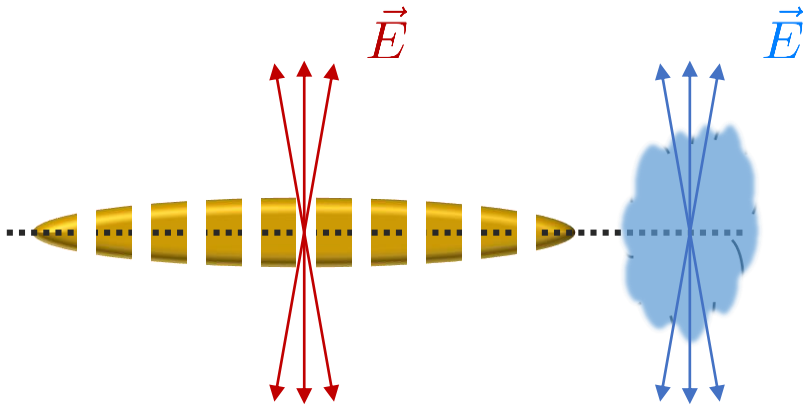



Update momenta in slice $i$ with

$$\Delta\vec{x}'[i] = -\frac{e^2}{m\gamma\beta^2c^2}\vec{E}_{e^-}[i]\,L$$

- PIC stands for Particle-In-Cell
- We use this method to compute fields generated by particles to solve e.g. the Poisson equation
- Electron motion occurs at the time scale of a slice of a bunch length → track single slices through the e-cloud and apply integrated kicks
  - Compute electric fields from one slice and from e-cloud
  - Apply kicks to protons
  - Advance electrons by one slice length – this is a multi-scale dynamics problem (fast cyclotron motion superposed to slower guiding center drift) → Boris algorithm for tracking (per macroparticle)
  - Track next slice through e-cloud

# E-cloud beam system




$$\dot{\vec{x}} = \vec{v}$$

$$\dot{\vec{v}} = \frac{e}{m}\left(\vec{E} + \frac{\vec{v} \times \vec{B}}{c}\right)$$

$$\frac{\mathbf{x}_{k+1} - \mathbf{x}_k}{\Delta t} = \mathbf{v}_{k+1}$$

$$\frac{\mathbf{v}_{k+1} - \mathbf{v}_k}{\Delta t} = \frac{e}{m}\left(\mathbf{E}_k + \frac{(\mathbf{v}_{k+1} + \mathbf{v}_k) \times \mathbf{B}_k}{2c}\right)$$

$$\mathbf{v}^- = \mathbf{v}_k + \frac{e}{m}\mathbf{E}_k\frac{\Delta t}{2}$$

$$\frac{\mathbf{v}^+ - \mathbf{v}^-}{\Delta t} = \frac{e}{2mc}\left((\mathbf{v}^+ + \mathbf{v}^-) \times \mathbf{B}_k\right)$$

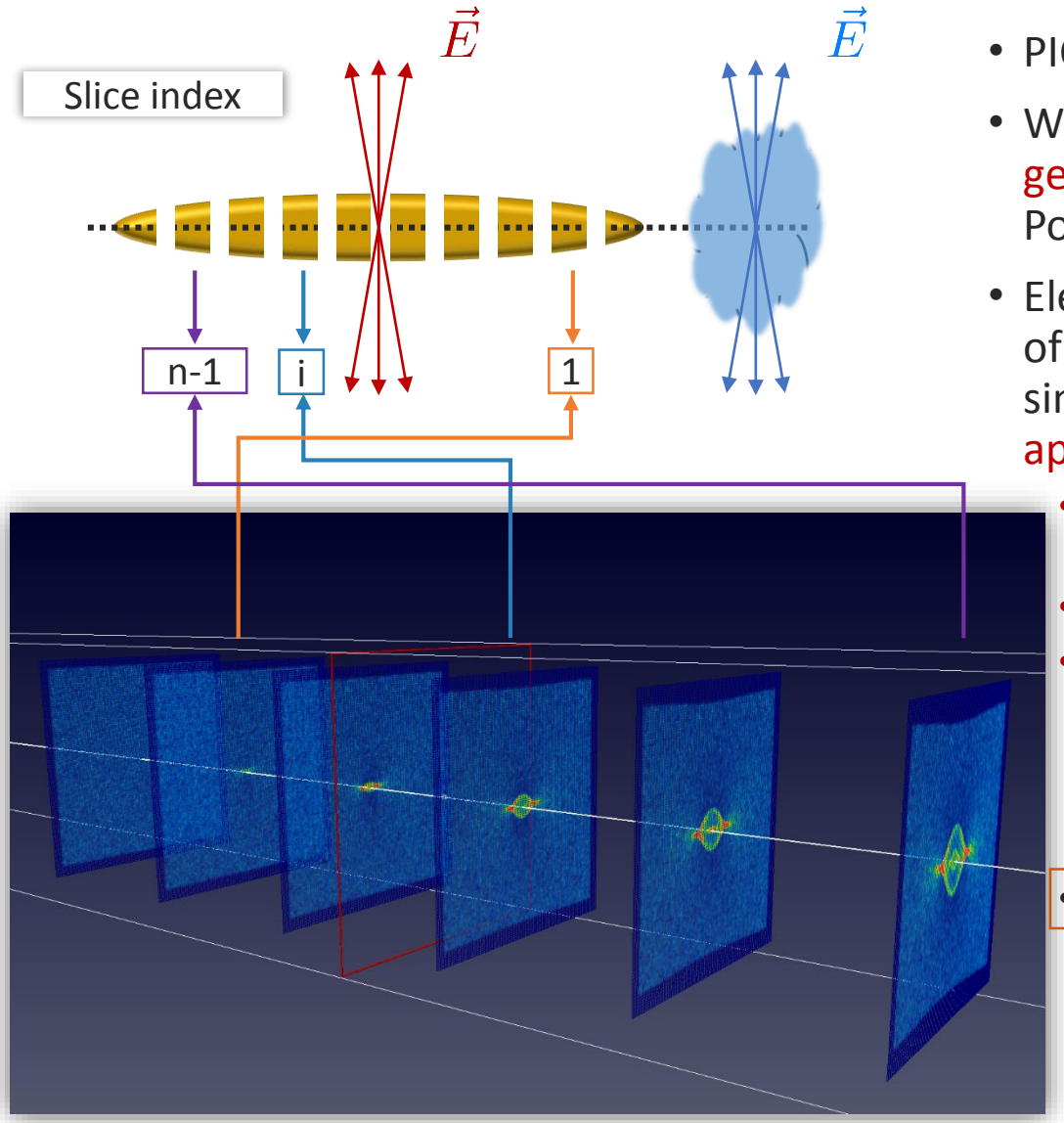
$$\mathbf{v}_{k+1} = \mathbf{v}^+ + \frac{e}{m}\mathbf{E}_k\frac{\Delta t}{2}$$

- PIC stands for Particle-In-Cell
- We use this method to compute fields generated by particles to solve e.g. the Poisson equation
- Electron motion occurs at the time scale of a slice of a bunch length → track single slices through the e-cloud and apply integrated kicks
  - Compute electric fields from one slice and from e-cloud
  - Apply kicks to protons
  - Advance electrons by one slice length – this is a multi-scale dynamics problem (fast cyclotron motion superposed to slower guiding center drift) → Boris algorithm for tracking (per macroparticle)
  - Track next slice through e-cloud

C. Birdsall and A. Langdon, *Plasma Physics Via Computer Simulation* (McGraw-Hill, Inc., New York, 1985)

Hong Qin et al. , *Why is Boris algorithm so good?*, Physics of Plasmas 20, 084503 (2013)

# E-cloud beam system

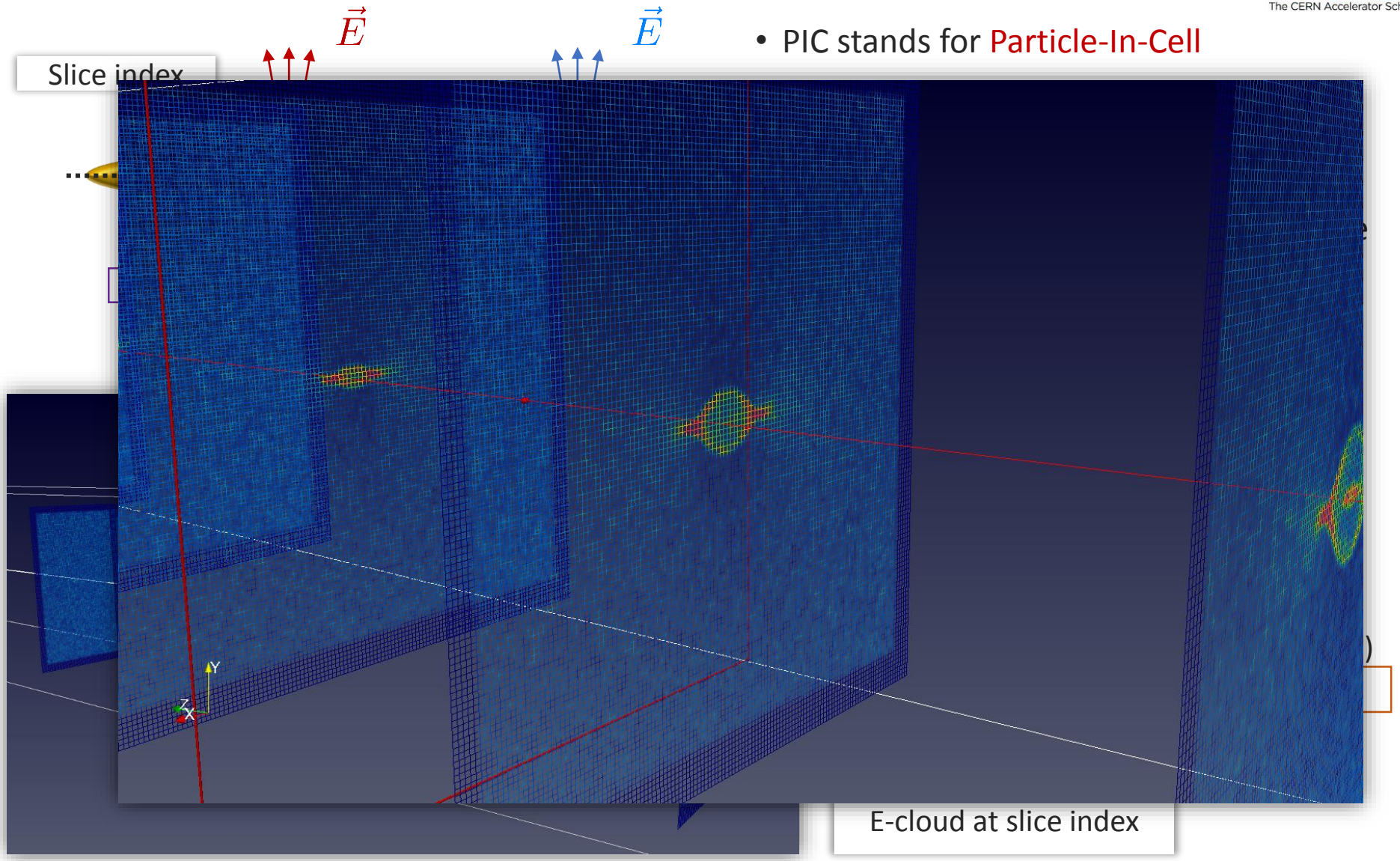



- PIC stands for Particle-In-Cell
- We use this method to compute fields generated by particles to solve e.g. the Poisson equation
- Electron motion occurs at the time scale of a slice of a bunch length → track single slices through the e-cloud and apply integrated kicks
  - Compute electric fields from one slice and from e-cloud
  - Apply kicks to protons
  - Advance electrons by one slice length – this is a multi-scale dynamics problem (fast cyclotron motion superposed to slower guiding center drift) → Boris algorithm for tracking (per macroparticle)
- Track next slice through e-cloud

E-cloud at slice index

# E-cloud beam system

- PIC stands for Particle-In-Cell



# Outline

Introduction to macroparticle models – implementations, applications and examples

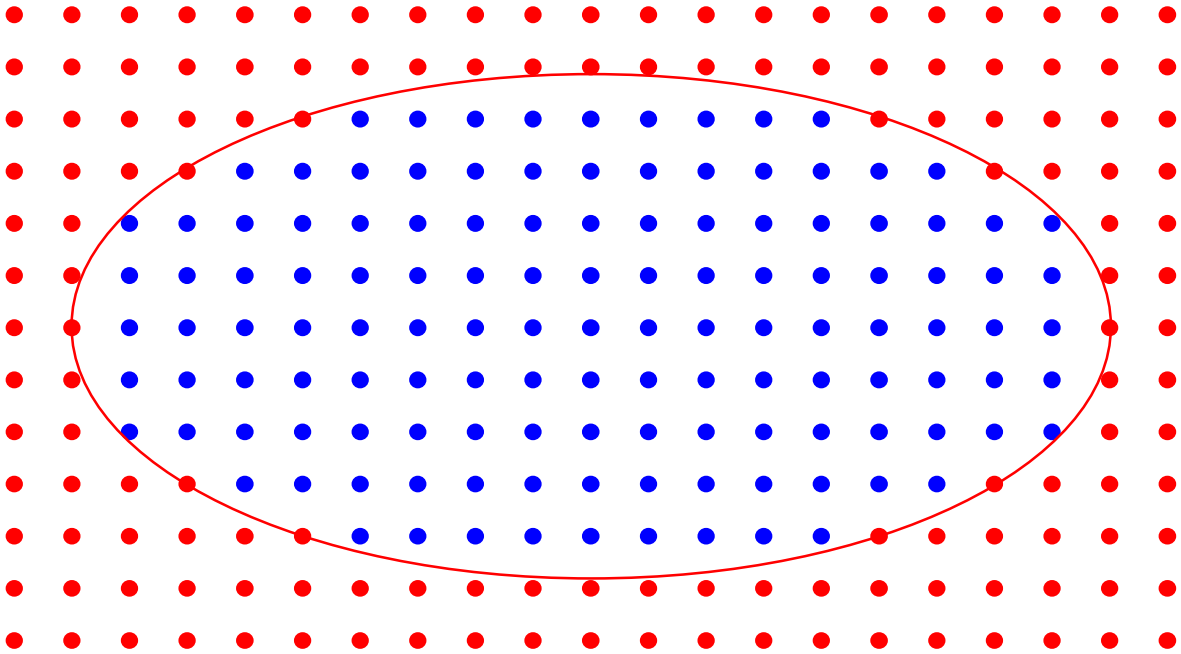
- Part 1 – numerical modelling
  - Initialisation
  - Simple tracking
  - Chromaticity and detuning
  - Wakefields with examples
    - Constant wakes
    - Dipole wakes
    - TMCI & headtail modes
- Part 2 – electron cloud
  - Modelling of e-cloud interactions
  - PIC solvers
  - Application for e-cloud instabilities

# PIC solvers in brief

- In many of our codes, Particle in Cell (PIC) algorithms are used to compute the electric field generated by a set of charged particles in a set of discrete points (can be the locations of the particles themselves, or of another set of particles)
- The solution typically consists of 4 stages:
  1. Charge scatter from macroparticles (MPs) to grid (reduction of macroparticles)
  2. Calculation of the electrostatic potential at the nodes
  3. Calculation of the electric field at the nodes (gradient evaluation)
  4. Field gather from grid to MPs

# PIC solvers in brief

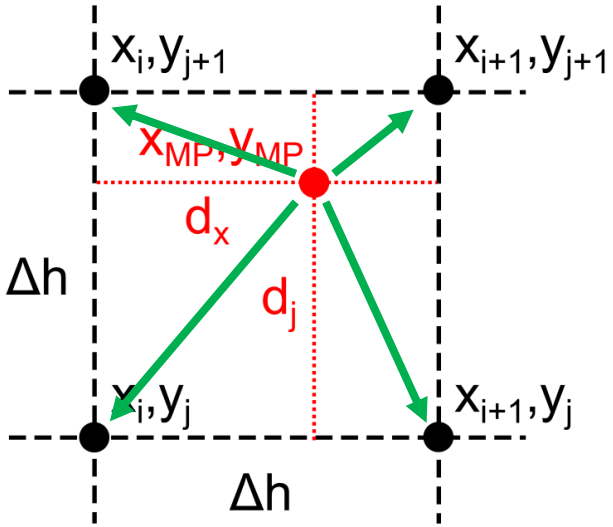
- The solution typically consists of 4 stages:
  1. Charge scatter from macroparticles (MPs) to grid (reduction of macroparticles)
  2. Calculation of the electrostatic potential at the nodes
  3. Calculation of the electric field at the nodes (gradient evaluation)
  4. Field gather from grid to MPs




Uniform square grid

# PIC solvers – basic steps

- The solution typically consists of 4 stages:
  1. Charge scatter from macroparticles (MPs) to grid (reduction of macroparticles)
  2. Calculation of the electrostatic potential at the nodes
  3. Calculation of the electric field at the nodes (gradient evaluation)
  4. Field gather from grid to MPs




$$\rho_{i,j} = \rho_{i,j} + \frac{q\, n_{\mathrm{MP}}}{\Delta h}\left(1 - \frac{d_x}{\Delta h}\right)\left(1 - \frac{d_y}{\Delta h}\right)$$

$$\rho_{i+1,j} = \rho_{i+1,j} + \frac{q\, n_{\mathrm{MP}}}{\Delta h}\left(\frac{d_x}{\Delta h}\right)\left(1 - \frac{d_y}{\Delta h}\right)$$

$$\rho_{i,j+1} = \rho_{i,j+1} + \frac{q\, n_{\mathrm{MP}}}{\Delta h}\left(1 - \frac{d_x}{\Delta h}\right)\left(\frac{d_y}{\Delta h}\right)$$

$$\rho_{i+1,j+1} = \rho_{i+1,j+1} + \frac{q\, n_{\mathrm{MP}}}{\Delta h}\left(\frac{d_x}{\Delta h}\right)\left(\frac{d_y}{\Delta h}\right)$$

# PIC solvers – basic steps

- The solution typically consists of 4 stages:
  1. Charge scatter from macroparticles (MPs) to grid (reduction of macroparticles)
  2. Calculation of the electrostatic potential at the nodes
  3. Calculation of the electric field at the nodes (gradient evaluation)
  4. Field gather from grid to MPs

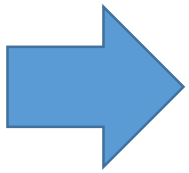
$$\begin{cases} \nabla^2\phi(x,y) = -\dfrac{\rho(x,y)}{\varepsilon_0} \\ \textbf{Boundary conditions} \text{ (e.g., perfectly conducting, open, periodic)} \end{cases}$$

- Different numerical approaches exist to solve these types of equations each with its own advantages and drawbacks:
  - Open space FFT solver (explicit, very fast but open boundaries)
  - Rectangular boundary FFT solver (explicit, very fast but only rectangular boundaries)
  - Finite Difference implicit Poisson solver (arbitrary chamber shape, sparse matrix, possibility to use Shortley Weller boundary refinement, KLU fast routines, computationally more demanding)
  - Dual or multi-grid in combination with direct or iterative solvers

# PIC solvers – basic steps

- The solution typically consists of 4 stages:
  1. Charge scatter from macroparticles (MPs) to grid (reduction of macroparticles)
  2. Calculation of the electrostatic potential at the nodes
  3. Calculation of the electric field at the nodes (gradient evaluation)
  4. Field gather from grid to MPs

$$\mathbf{E} = -\nabla\phi$$

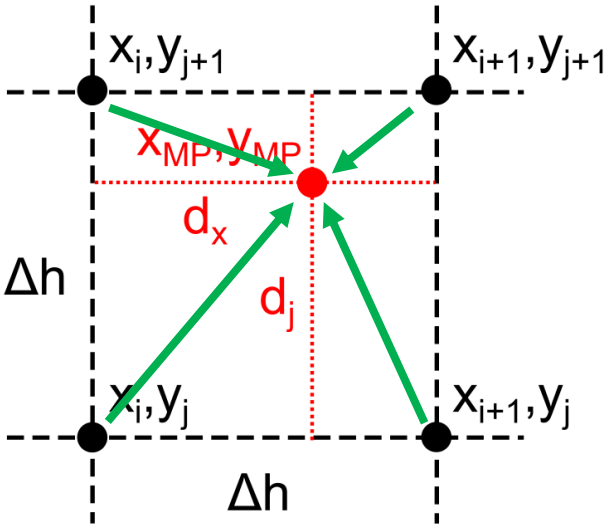


$$(E_x)_{i,j} = -\frac{\phi_{i+1,j} - \phi_{i-1,j}}{2\Delta h}$$

$$(E_y)_{i,j} = -\frac{\phi_{i,j+1} - \phi_{i,j-1}}{2\Delta h}$$

# PIC solvers – basic steps

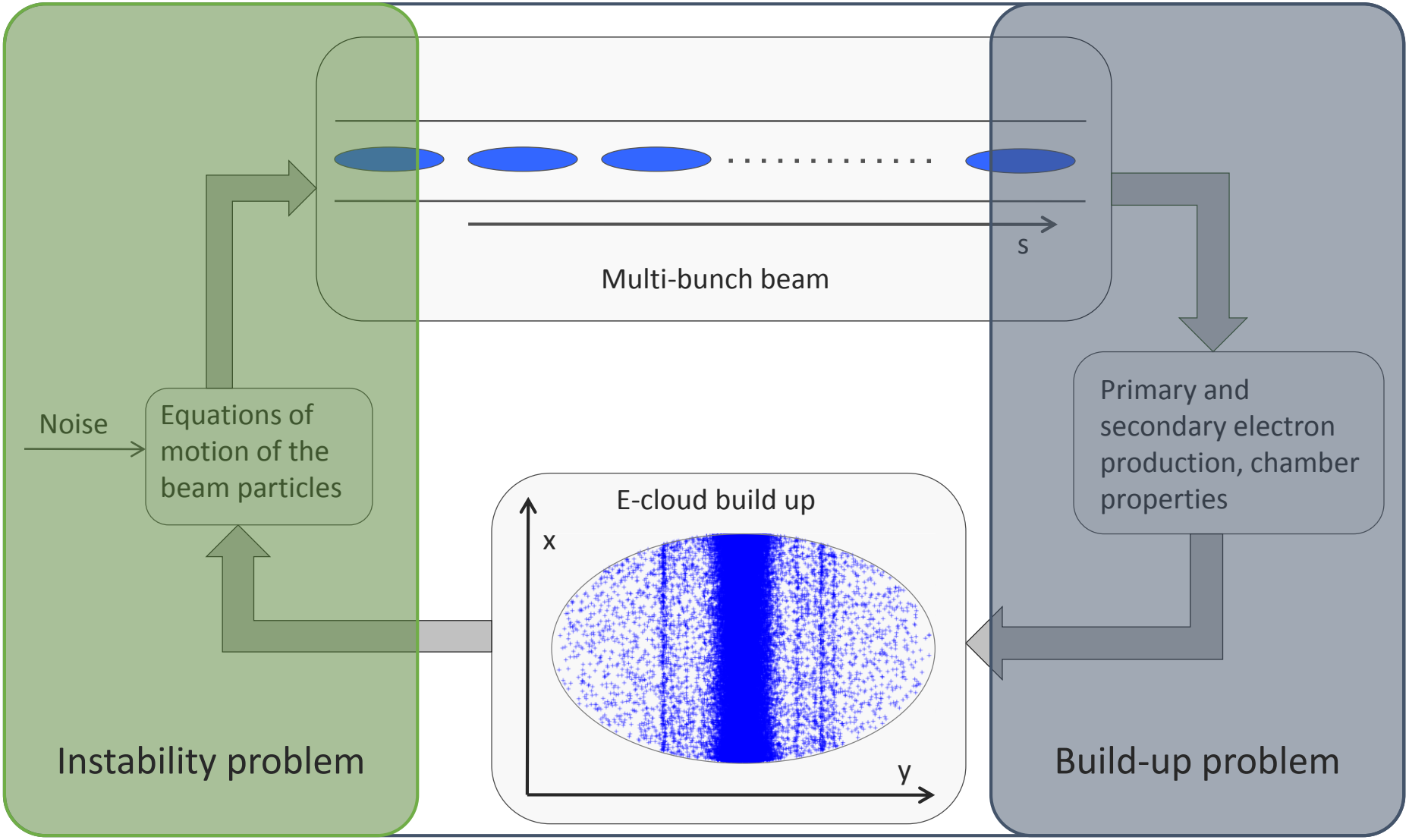
- The solution typically consists of 4 stages:
  1. Charge scatter from macroparticles (MPs) to grid (reduction of macroparticles)
  2. Calculation of the electrostatic potential at the nodes
  3. Calculation of the electric field at the nodes (gradient evaluation)
  4. Field gather from grid to MPs




$$\mathbf{E}(x_{\mathrm{MP}}, y_{\mathrm{MP}}) =$$

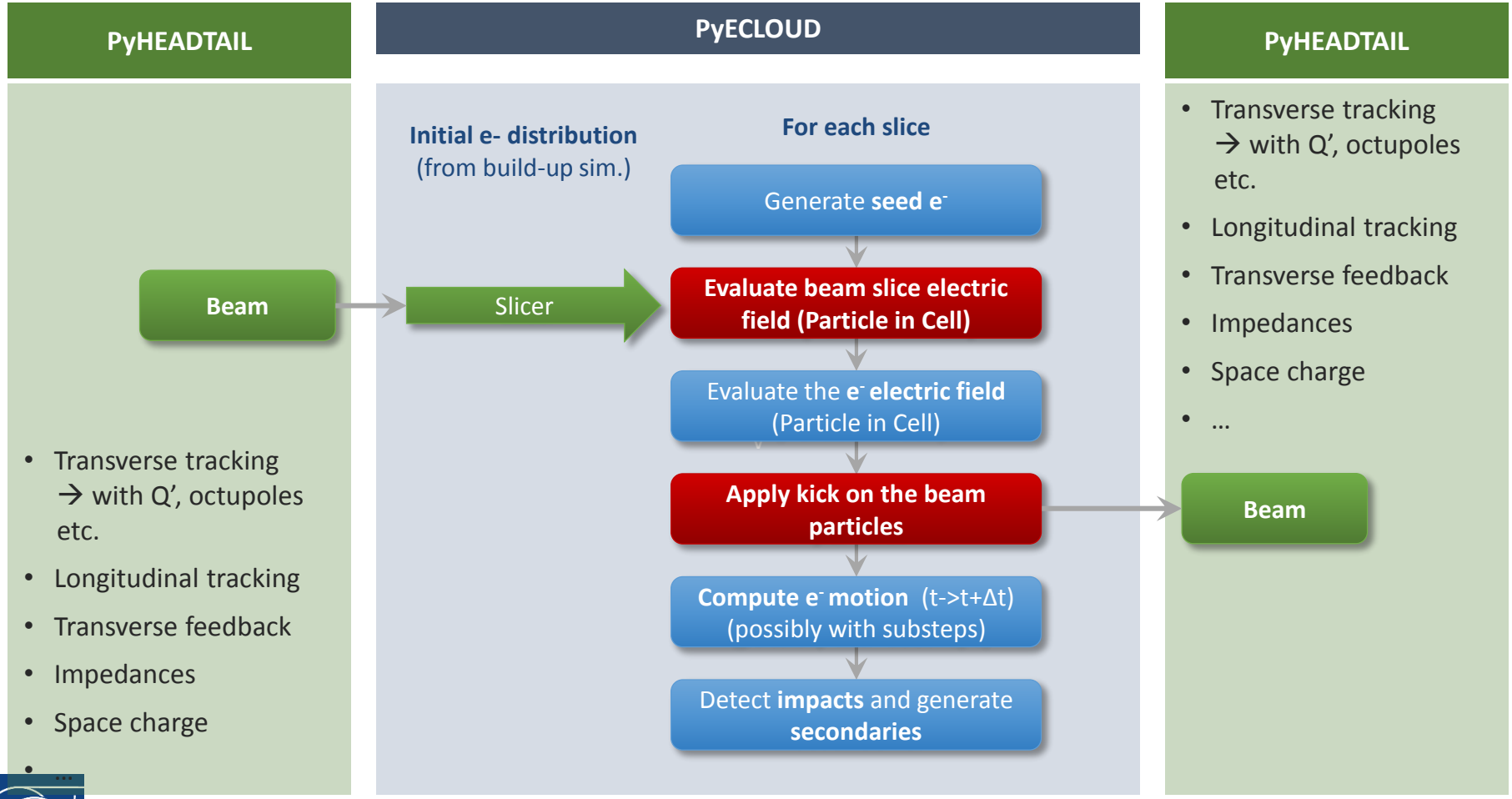
$$\mathbf{E}_{i,j}\left(1-\frac{d_x}{\Delta h}\right)\left(1-\frac{d_y}{\Delta h}\right)+\mathbf{E}_{i+1,j}\left(\frac{d_x}{\Delta h}\right)\left(1-\frac{d_y}{\Delta h}\right)$$
$$+\mathbf{E}_{i,j+1}\left(1-\frac{d_x}{\Delta h}\right)\left(\frac{d_y}{\Delta h}\right)+\mathbf{E}_{i+1,j+1}\left(\frac{d_x}{\Delta h}\right)\left(\frac{d_y}{\Delta h}\right)$$

# Numerical model of electron cloud effects



# Numerical model of electron cloud effects

- A self-consistent treatment requires the combination of an instability and a build-up code
- Becomes easily possible with modular structure and good design of codes (e.g. object orientation)

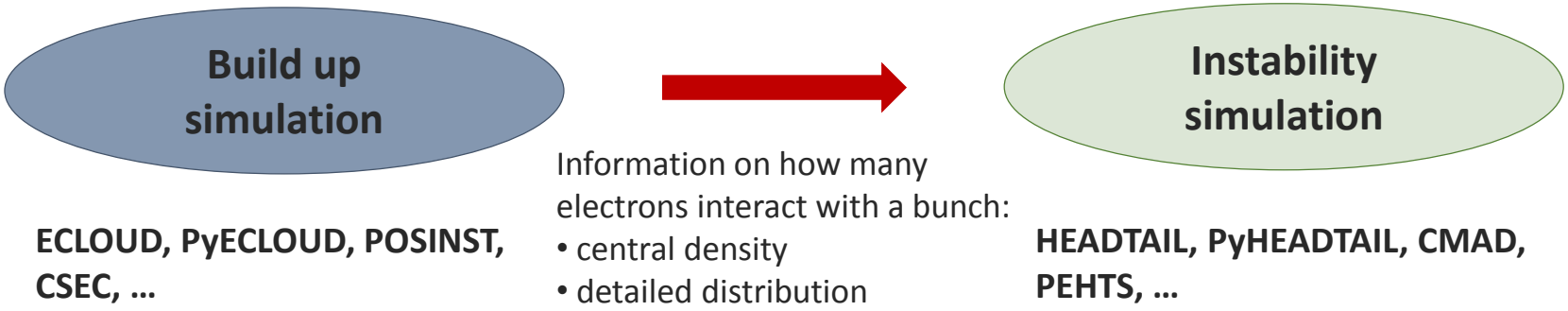



**Legend**: From instability code – From build-up code – Interaction between the two codes

# Numerical model of electron cloud effects

- Coupled bunch electron cloud instability naturally needs a self-consistent solution of the electron cloud problem
  - A broad time scale to cover, currently working on the problem

- For the moment we simulate the two branches separately (similar to what is done for impedances):
  - **Electron cloud build up**
    - ✓ Multi-bunch
    - ✓ Usually single passage, single turn or just few turns
  - **Electron cloud instability**
    - ✓ Single bunch
    - ✓ Multi-turn, or even multi-kick multi-turn



# Numerical model of electron cloud effects

- In principle both coherent instability and incoherent emittance growth could be predicted by these simulations
- Evolution of a beam interacting with an electron cloud depends on a significant number of parameters in a non-trivial way
  - Bunch length (longitudinal emittance)
  - Beam transverse sizes (emittances and beta functions at the electron cloud location)
  - Beam energy
  - Beam current (number of particles per bunch)
  - Chromaticity
  - Magnetic field (field-free, dipole, quadrupole)
  - Electron cloud density and distribution (in reality determined by many of the above parameters, but can be set independently in simulations)

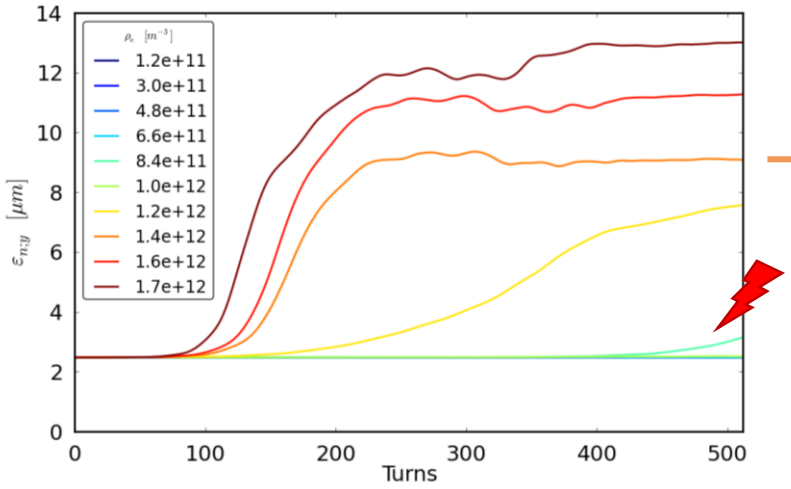
# Outline

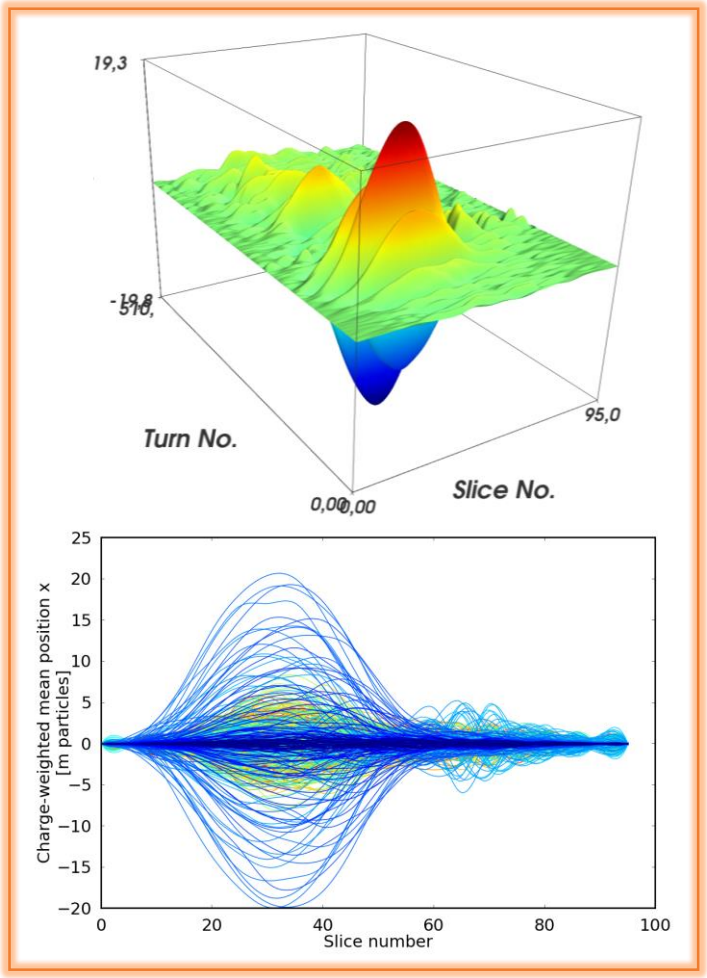
Introduction to macroparticle models – implementations, applications and examples

- Part 1 – numerical modelling
  - Initialisation
  - Simple tracking
  - Chromaticity and detuning
  - Wakefields with examples
    - Constant wakes
    - Dipole wakes
    - TMCI & headtail modes
- Part 2 – electron cloud
  - Modelling of e-cloud interactions
  - PIC solvers
  - Application for e-cloud instabilities

# Electron cloud induced instabilities

- Typical e-cloud simulation try to identify the e-cloud central density threshold for an instability
- Scans in the central density are performed until an exponential growth can be observed in the emittance

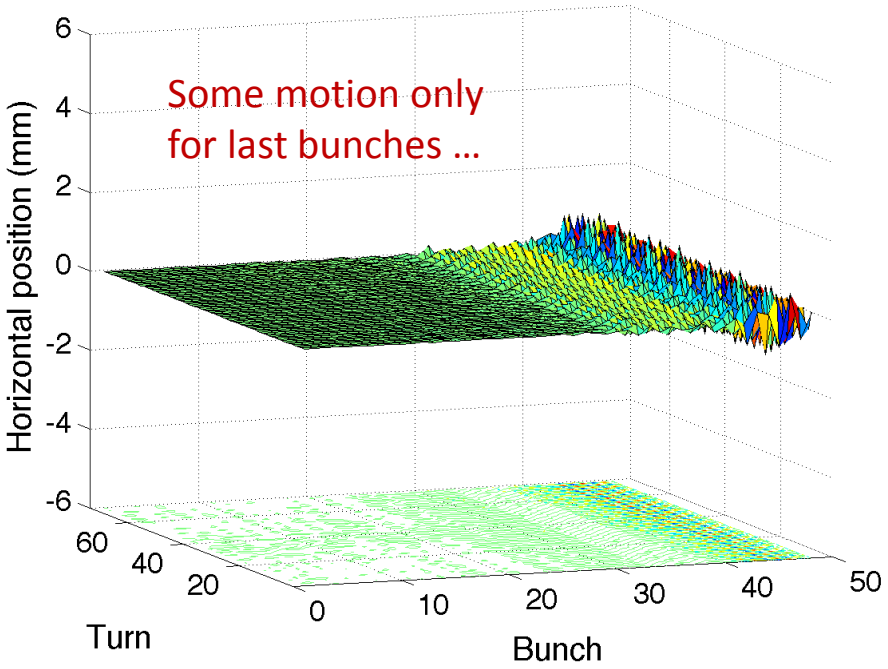


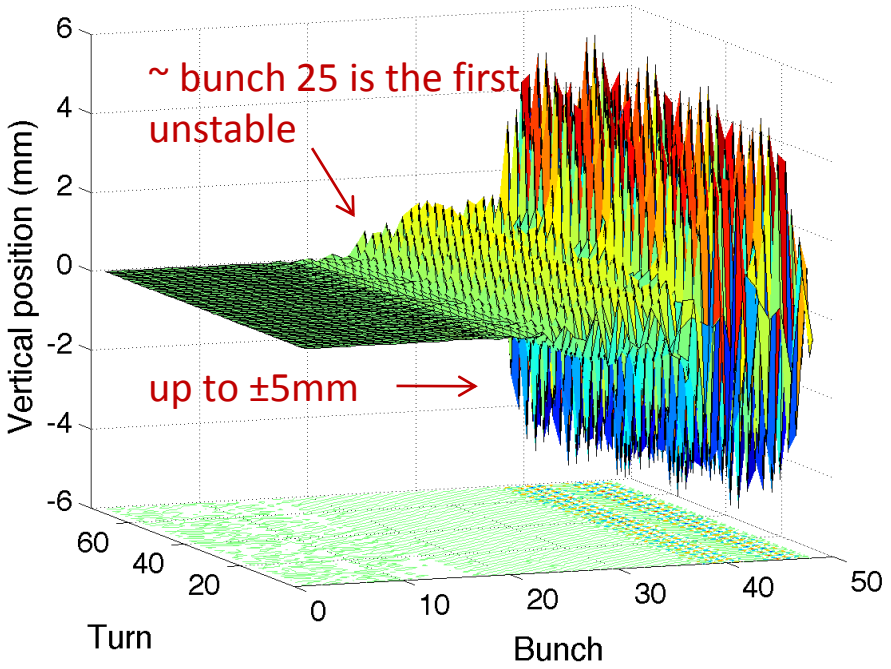




- Coherent instabilities occur when a certain central cloud density threshold is breached
- This leads to coherent intra bunch motion which grows exponentially
- A consequence is emittance blow-up and losses

# Ex. of coherent e-cloud effects in the LHC

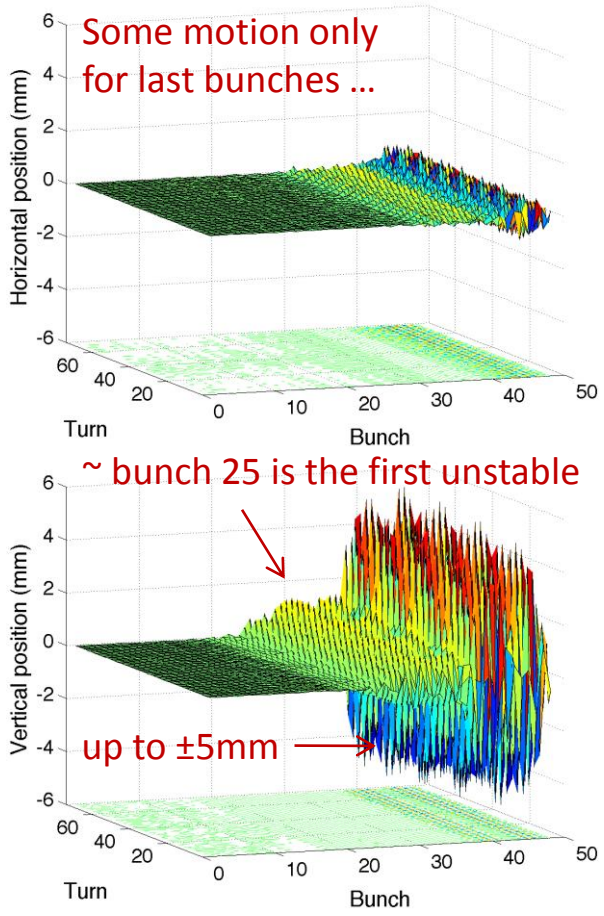
- First injection of 48 bunches of 25 ns beam into the LHC in 2011
- Beam was dumped twice due to a violent instability in the vertical plane, causing losses above the interlock threshold

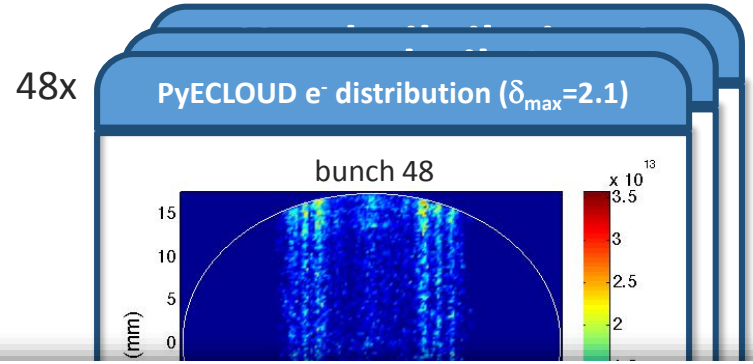


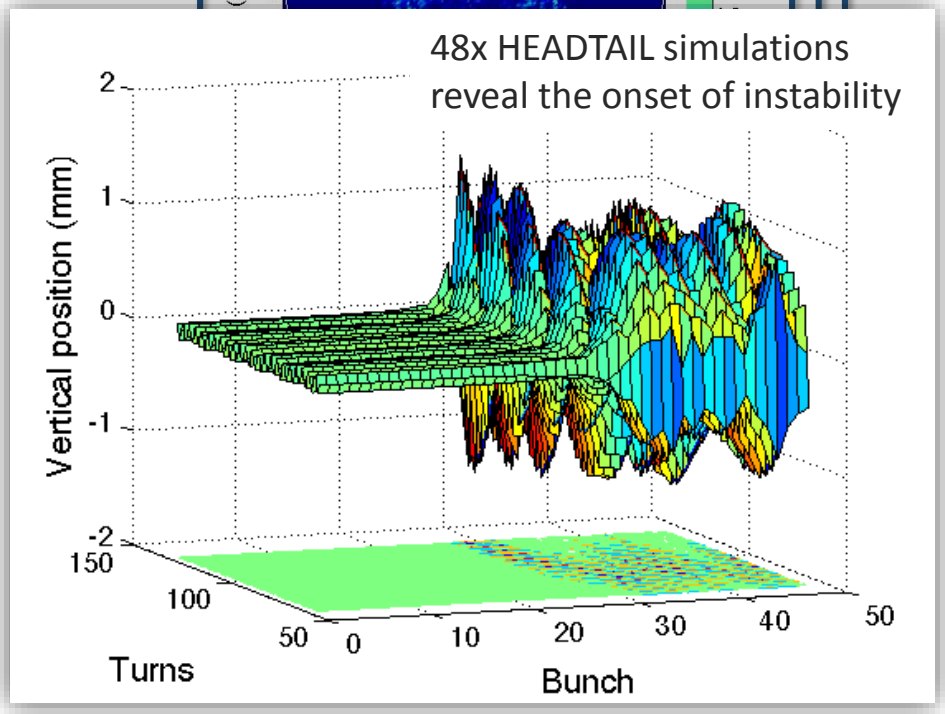



# Ex. of coherent e-cloud effects in the LHC

## 48b injection test (26/08/11)

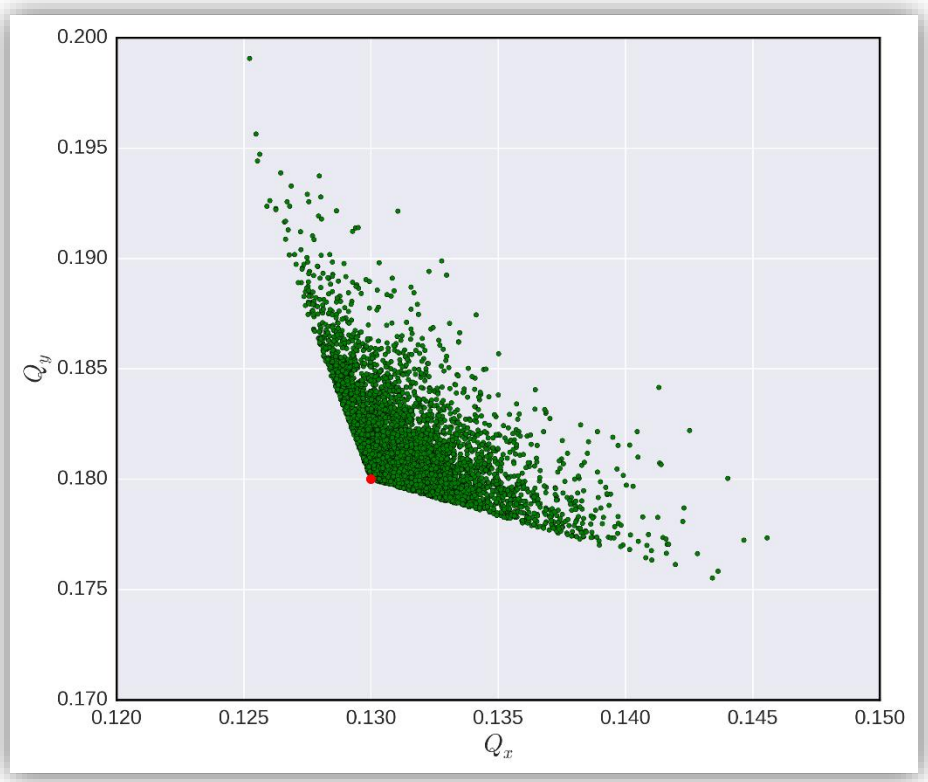


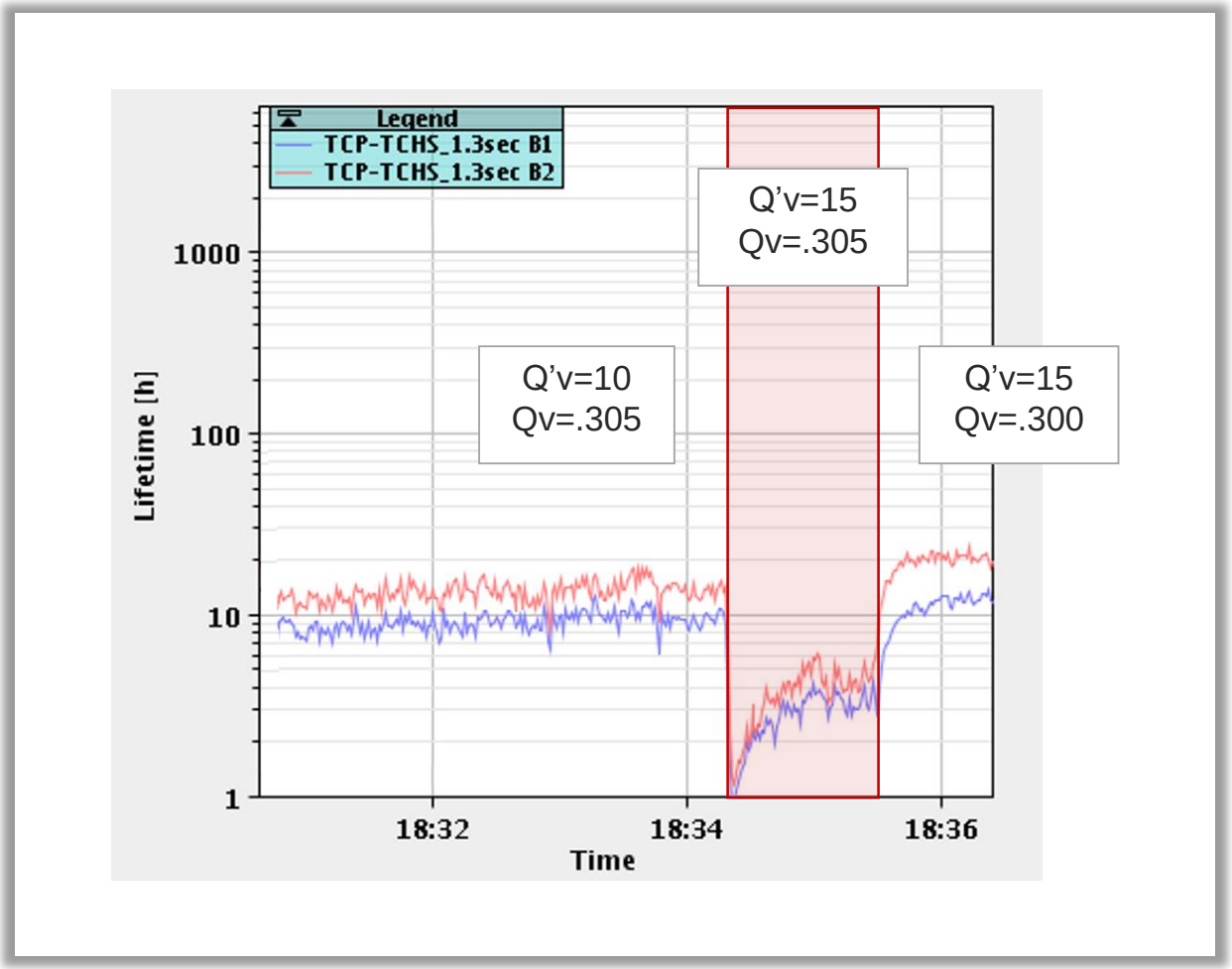



# Ex. of incoherent e-cloud effects in the LHC

- Remember tune footprint from octupoles in Part I

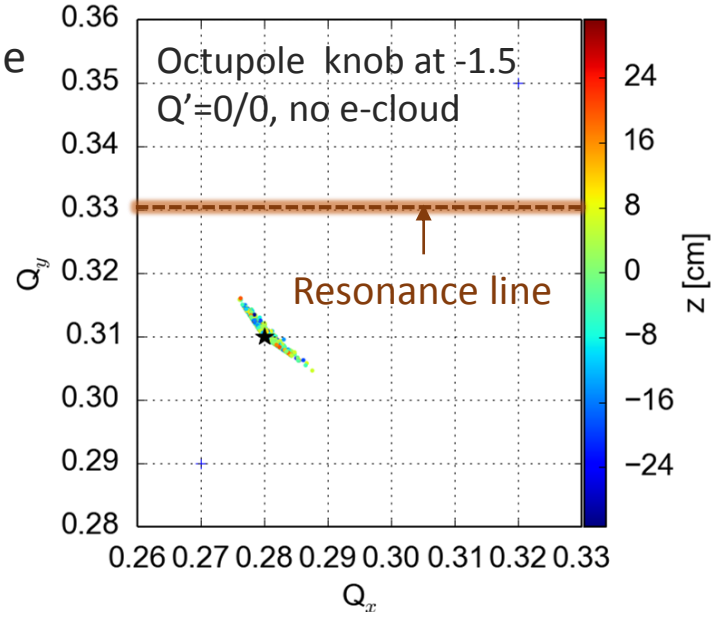


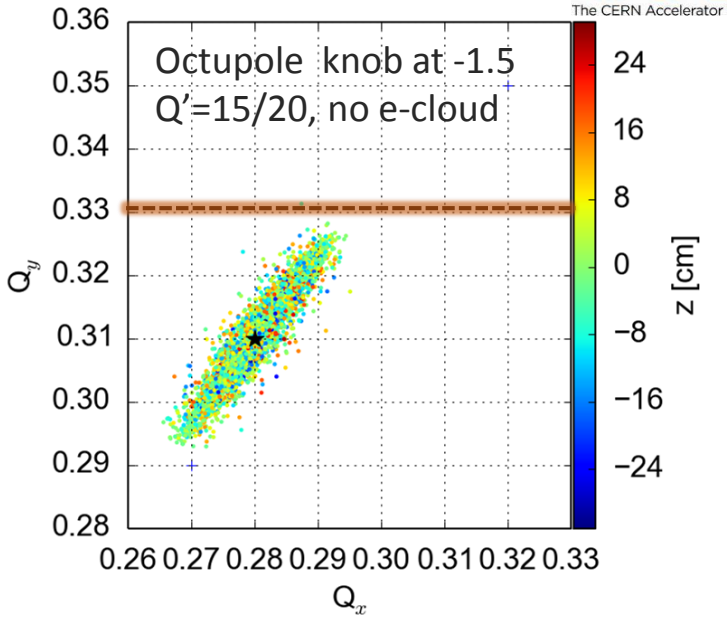
# Ex. of incoherent e-cloud effects in the LHC

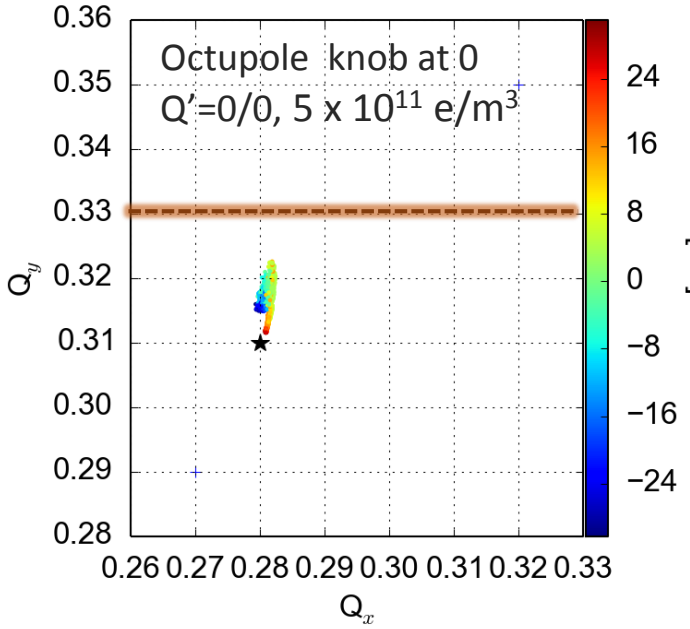


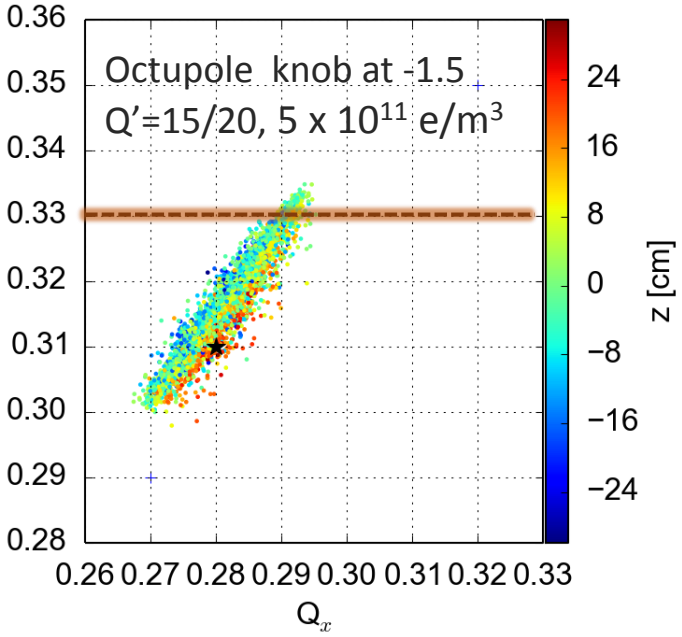
# Ex. of incoherent e-cloud effects in the LHC

- Macroparticle simulations allow to obtain tune footprint from all effects separated
- … as well as from all effects combined



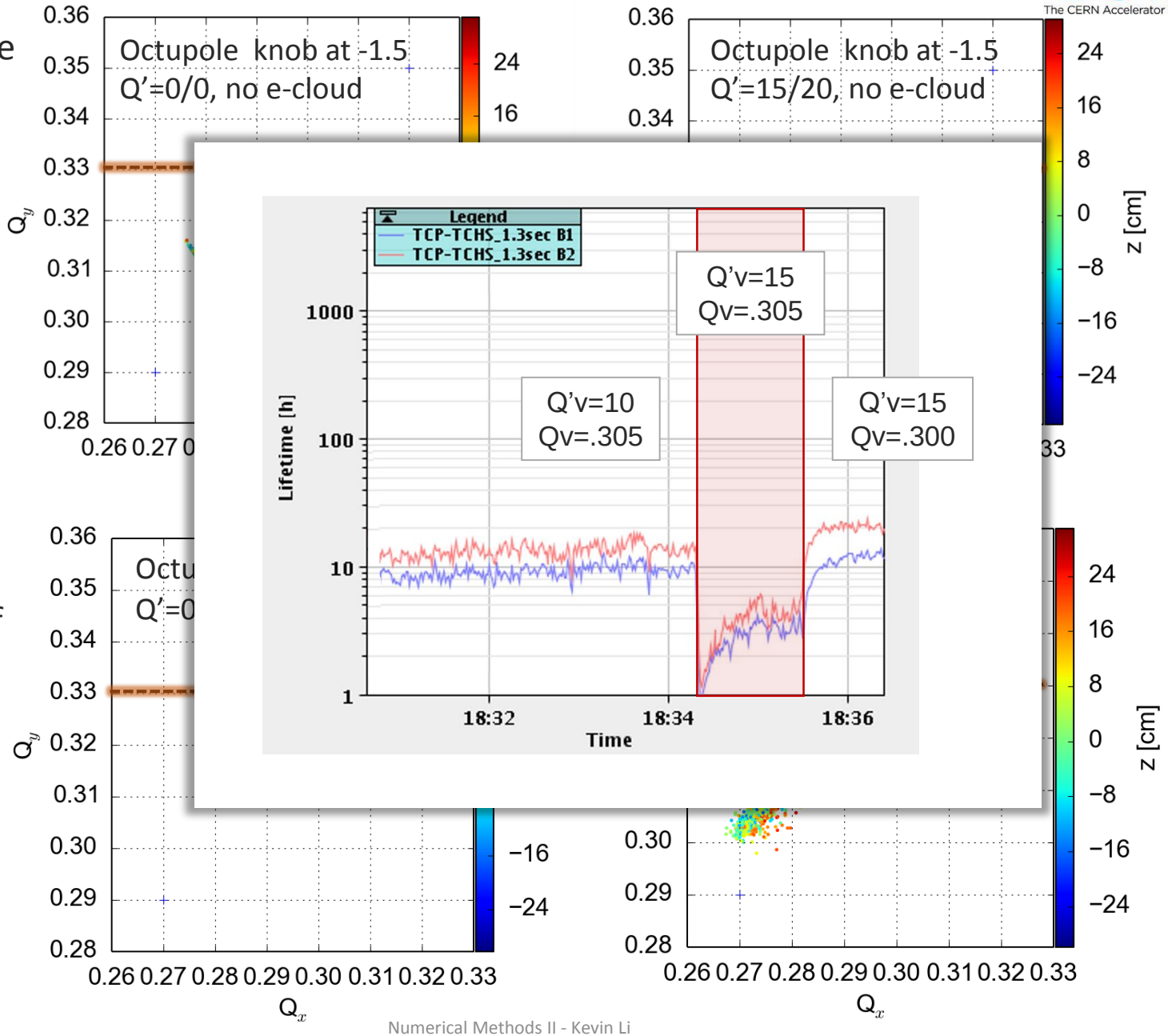




A. Romano et al

# Ex. of incoherent e-cloud effects in the LHC

- Macroparticle simulations allow to obtain tune footprint from all effects separated
- … as well as from all effects combined
- … to identify the source of incoherent losses in the LHC




A. Romano et al

# Vlasov solvers

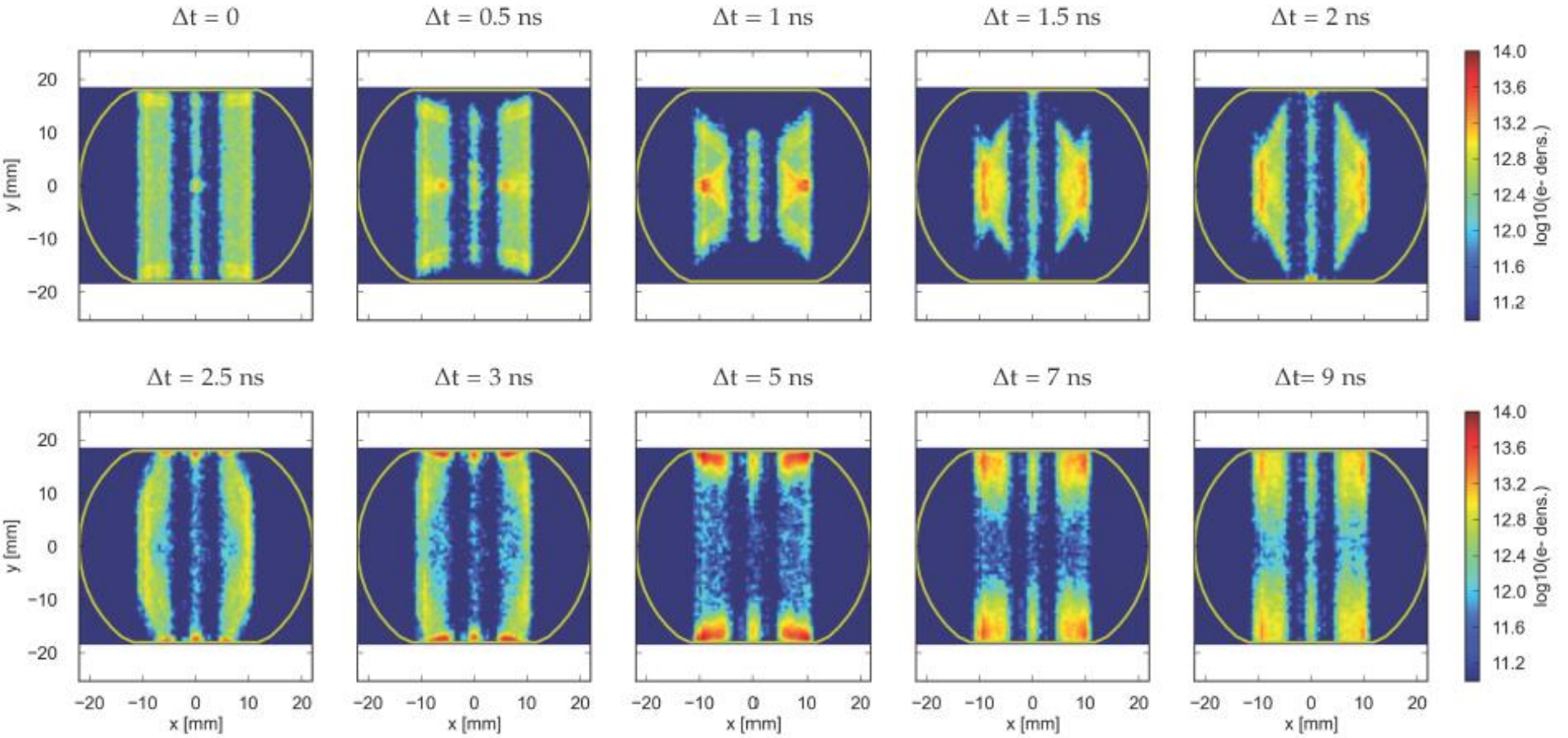
- No time this time…

# End part II

# Backup

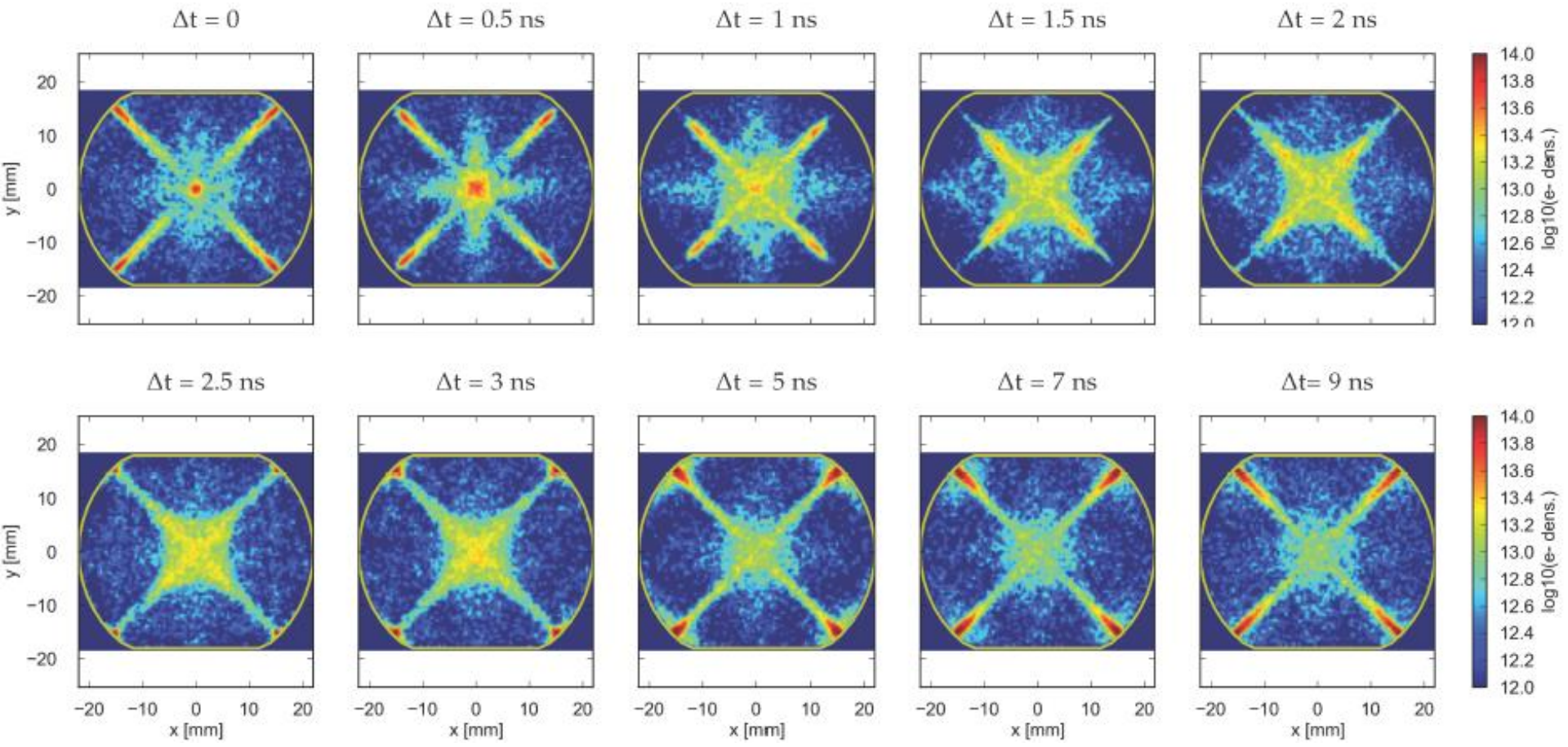
# Electron clouds in a bending magnet

- The electrons exhibit different transverse (x,y) distributions, according to the type of region in which the electron cloud is formed
  - In dipole regions, the electron motion is confined along the lines of the magnetic field. Example: snaposhots of multipacting in the dipole of an LHC arc cell during bunch passage and including secondary production.



# Electron clouds in a quadrupole magnet

- The electrons exhibit different transverse (x,y) distributions, according to the type of region in which the electron cloud is formed
  - In quadrupole regions, the electrons tend to multipact along the pole-to-pole lines of the cross section (example: snapshots of multipacting in an LHC arc quadrupole). Multipacting thresholds are usually lower in quadrupoles because electrons survive long thanks to trapping due to the magnetic gradient.



# Basic stages of a PIC algorithm

**Standard Particle In Cell (PIC) → 4 stages:**

1. Charge scatter from macroparticles (MPs) to grid
2. Calculation of the electrostatic potential at the nodes
3. Calculation of the electric field at the nodes (gradient evaluation)
4. Field gather from grid to MPs

$$\begin{cases} \nabla^2\phi(x,y) = -\dfrac{\rho(x,y)}{\varepsilon_0} \\ \phi(x,y) = 0 \qquad \text{on the boundary} \end{cases}$$

**Internal nodes:**

$$\frac{\phi_{i-1,j} + \phi_{i,j-1} - 4\phi_{i,j} + \phi_{i+1,j} + \phi_{i,j+1}}{\Delta h^2} = -\frac{\rho_{i,j}}{\varepsilon_0}$$

**External nodes:**

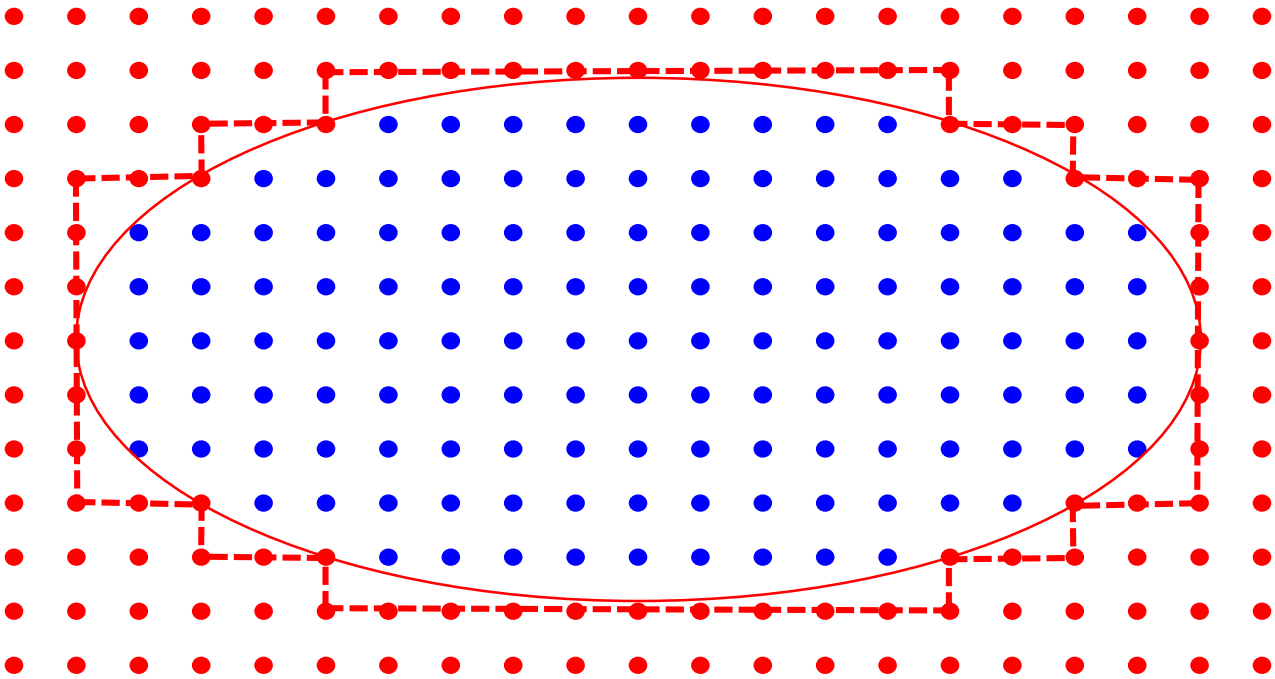
$$\phi_{i,j} = 0$$

Can be written in matrix form:

$$\underline{\underline{A}}\,\underline{\phi} = \frac{1}{\varepsilon_0}\underline{\rho}$$

**A is sparse and depends only on chamber geometry and grid size**
→ It can be computed and LU factorized in the initialization stage to speed up calculation

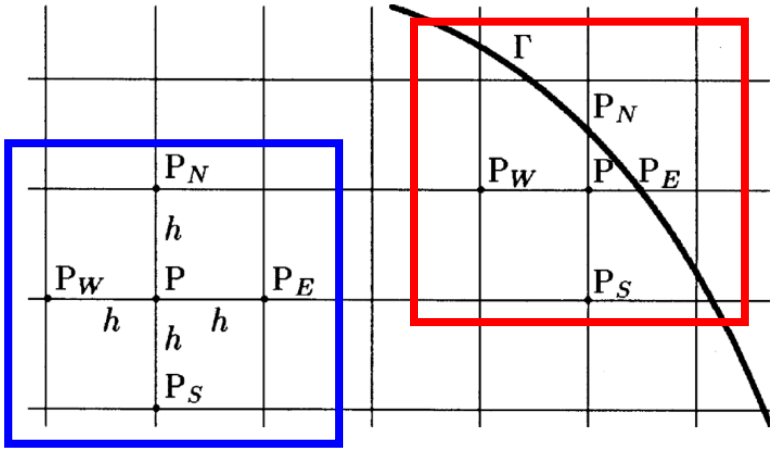
# Electron space charge evaluation in PyECLOUD



With this approach a curved boundary is **approximated with a staircase**

**Can we do better?**

# The Shortley - Weller method




Refined approximation of Laplace operator at boundary nodes:

$$-\Delta_h^{(SW)}U(P)=\left(\frac{2}{h_E h_W}+\frac{2}{h_S h_N}\right)U(P)$$
$$-\frac{2}{h_E(h_E+h_W)}U(P_E)-\frac{2}{h_W(h_E+h_W)}U(P_W)$$
$$-\frac{2}{h_S(h_S+h_N)}U(P_S)-\frac{2}{h_N(h_S+h_N)}U(P_N)$$

Usual 5-points formula at internal nodes:

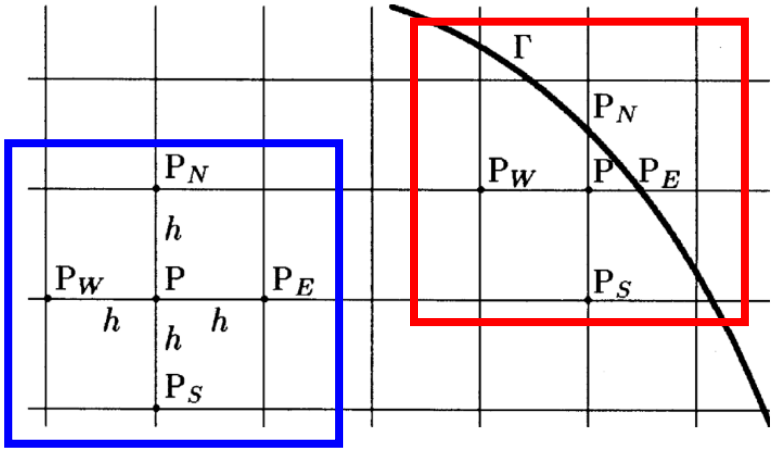
$$-\Delta_h^{(SW)}U(P)=\frac{4}{h^2}U(P)-\frac{1}{h^2}U(P_E)-\frac{1}{h^2}U(P_W)$$
$$-\frac{1}{h^2}U(P_S)-\frac{1}{h^2}U(P_N)$$

$O(h^2)$ truncation error is preserved

(see: N. Matsunaga and T. Yamamoto, Journal of Computational and Applied Mathematics 116 – 2000, pp. 263–273)

# The Shortley - Weller method

Sorry for the change of notation…




Refined gradient evaluation at boundary nodes:

$$E_x(P) = -\frac{1}{2}\left(\frac{U(P_E) - U(P)}{h_E} + \frac{U(P) - U(P_W)}{h_W}\right)$$

$$E_y(P) = -\frac{1}{2}\left(\frac{U(P_N) - U(P)}{h_N} + \frac{U(P) - U(P_S)}{h_S}\right)$$

Usual central difference for gradient evaluation at internal nodes:

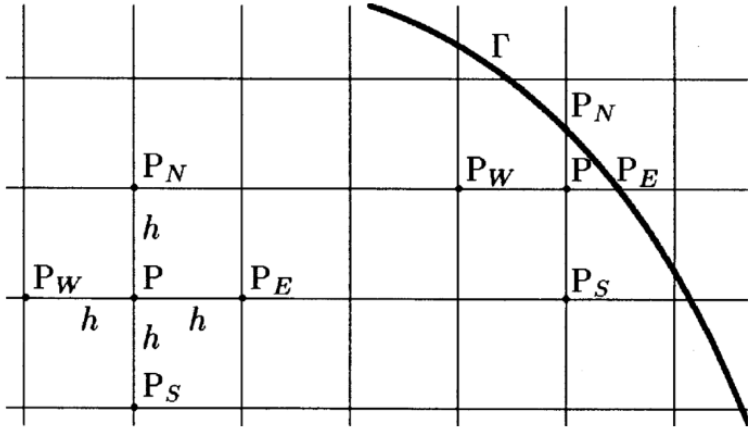
$$E_x(P) = -\frac{1}{2}\left(\frac{U(P_E) - U(P)}{h} + \frac{U(P) - U(P_W)}{h}\right)$$

$$E_y(P) = -\frac{1}{2}\left(\frac{U(P_N) - U(P)}{h} + \frac{U(P) - U(P_S)}{h}\right)$$

# The Shortley - Weller method

**Tricky implementation:**

- **Boundary nodes need to be identified, distances from the curved boundary** need to be evaluated
  - **PyECLOUD impact routines** have been employed (some refinement was required since they are optimized for robustness while here we need accuracy)
- **Nodes too close to the boundary** can lead to **ill conditioned A matrix** → we identify them and impose U=0
  - **Special treatment for gradient evaluation** is needed at these nodes
- Since chamber geometry and grid size stay constant along the simulation **most of the boundary treatment can be handled in the initialization stage**




$$\underline{\underline{A}}\,\underline{U} = \frac{1}{\varepsilon_0}\underline{\rho}$$

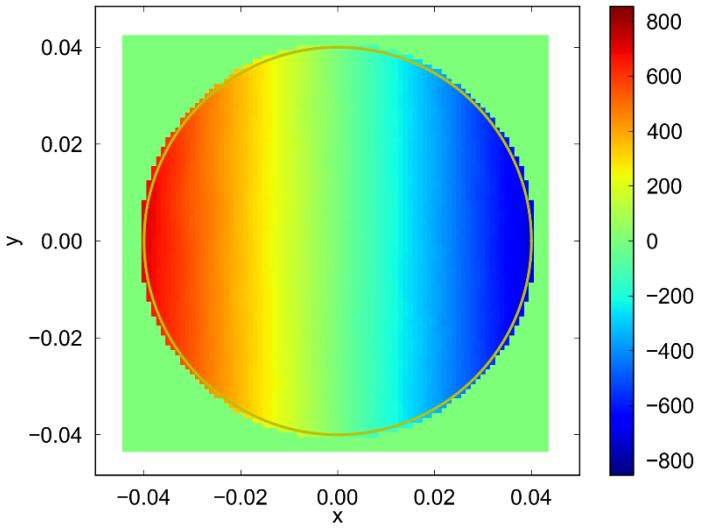
$$\underline{E_x} = \underline{\underline{D_x}}\,\underline{U}$$

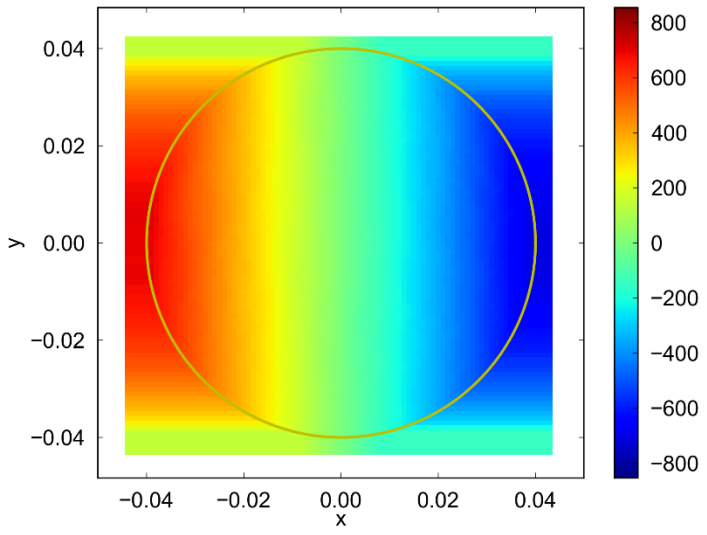
$$\underline{E_y} = \underline{\underline{D_y}}\,\underline{U}$$

# The Shortley - Weller method

**Tricky implementation:**

- **Boundary nodes need to be identified, distances from the curved boundary** need to be evaluated
  - **PyECLOUD impact routines** have been employed (some refinement was required since they are optimized for robustness while here we need accuracy)
- **Nodes too close to the boundary** can lead to **ill conditioned A matrix** → we identify them and impose U=0
  - **Special treatment for gradient evaluation** is needed at these nodes
- Since chamber geometry and grid size stay constant along the simulation **most of the boundary treatment can be handled in the initialization stage**
- **Field map extrapolated outside the chamber** to simplify field gather for particle close to the chamber's wall



# Dual grid

- Alternative approach: use a dual grid solver