Progress on macro-particle simulations of fast beam-ion instabilities in the framework of the PyHEADTAIL code

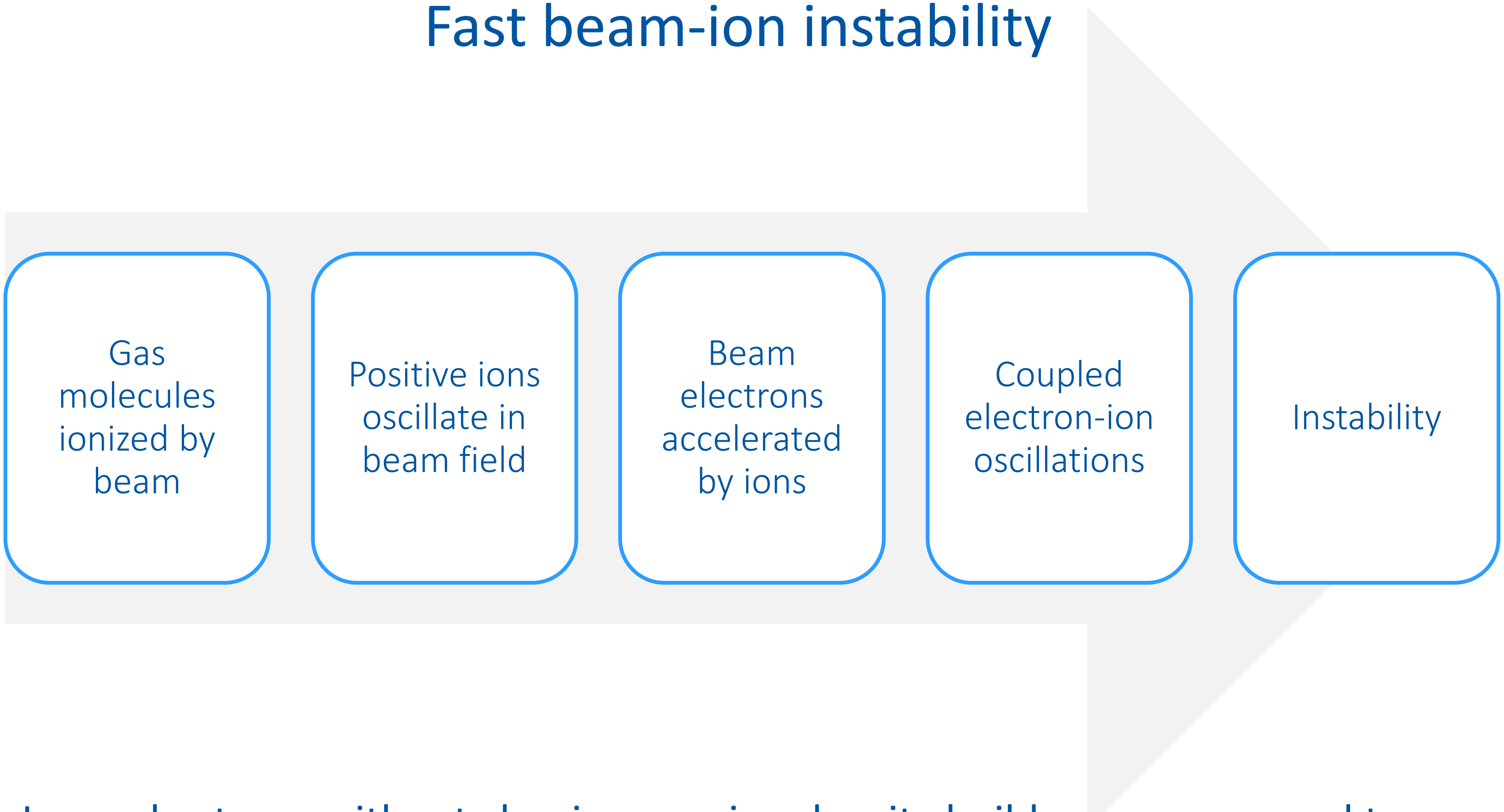
L. Mether, G. Rumolo

TWIICE 2 February 8th 2016

Outline

- Fast beam-ion instability
- Simulating the fast beam-ion instability
- Simulation codes
  - PyECLOUD
  - PyHEADTAIL
  - Ions in PyECLOUD and PyHEADTAIL
- Outlook
- Summary

Fast beam-ion instability



Gas molecules ionized by beam

Positive ions oscillate in beam field

Beam electrons accelerated by ions

Coupled electron-ion oscillations

Instability

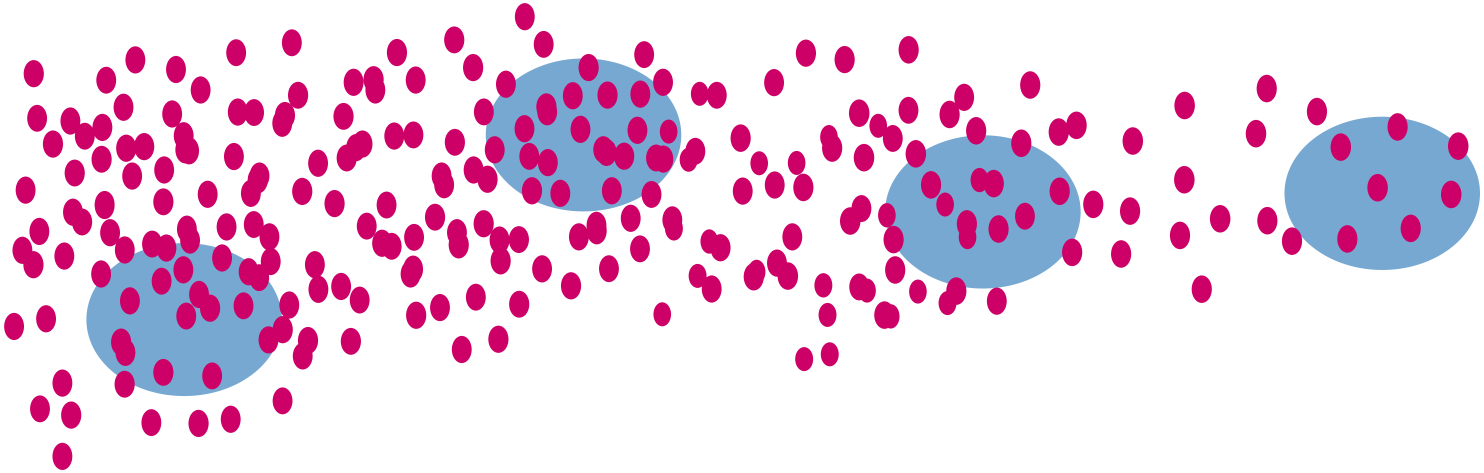
- In synchrotrons without clearing gap, ion density builds up over several turns
    - Conventional ion instability

- For bunch train followed by gap, ions build up only during one train passage
    - Fast beam-ion instability (FBII)

Fast beam-ion instability

- Ion density increases for every bunch passage – effect stronger at tail of train

- Ions transfer information on offsets from bunch to bunch – head and tail of train coupled



- Over time can lead to multi-bunch instability, tune shift, emittance growth

Analytical results

- Treatment based on linear approximation of Bassetti-Erskine formula
  - Velocity kick for ion with mass number A

$$k_{x,y}(x,y) = \frac{2N_b r_p c}{A} \frac{x,y}{(\sigma_x+\sigma_y)\sigma_{x,y}}$$

- N_b = bunch intensity, $\sigma_{x,y}$ = rms transverse beam sizes, r_p = classical proton radius

  - Instability rise time

$$\tau^2_{\text{inst}} \propto \frac{\gamma^2 A\omega_\beta}{n_b^4 N_b^3 P^2 T_b c} (\sigma_x + \sigma_y)^3 \sigma^3_{x,y}$$

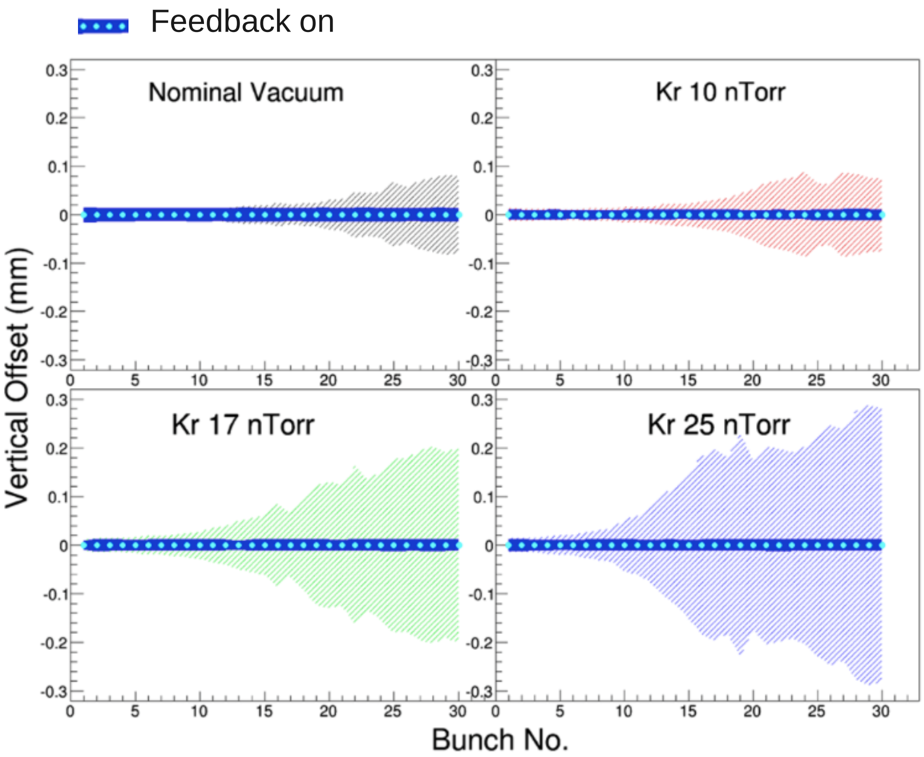
- T_b = bunch spacing, n_b = nr of bunches, P = pressure, ω_β = betatron frequency

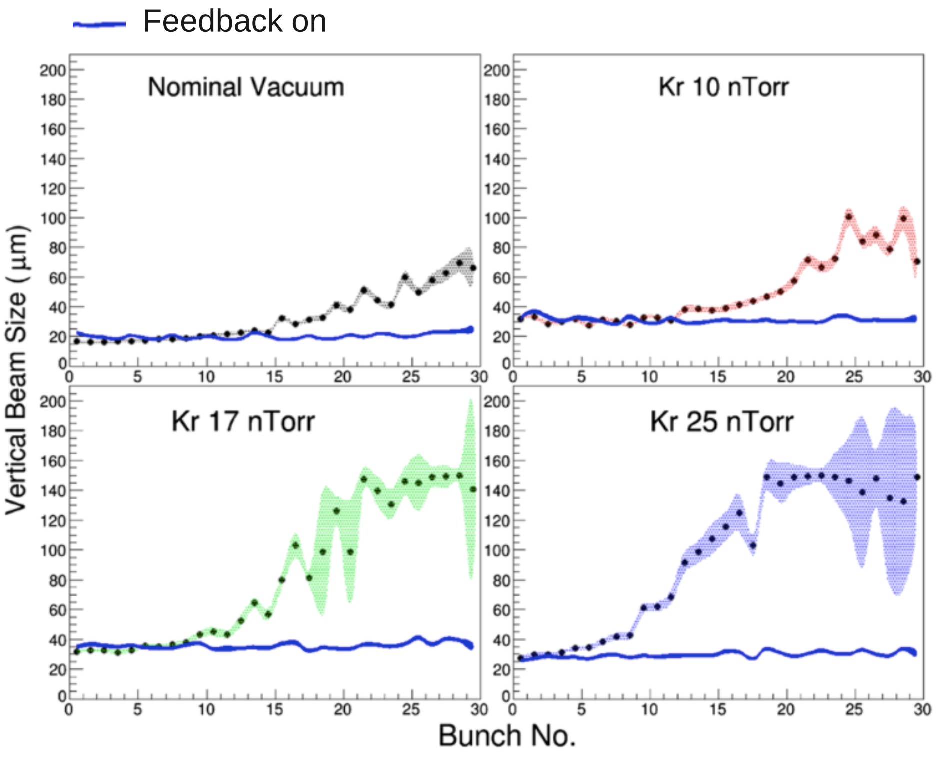
- Long bunch train, high intensity, small beam → short rise time

Fast beam-ion instability. I. Linear theory and simulations
Raubenheimer *et al.* Phys. Rev. E 52, 5, 5487

Observations

- Observed in several machines under vacuum degradation
- Measurements at CESR-TA (Dec. 2013, Apr 2014)
  - Varying ion species, pressure, bunch charge, train structure, feedback etc.





A. Chatterjee *et al*. Phys. Rev. ST Accel. Beams 18 (2015) 6, 064402

Simulation studies

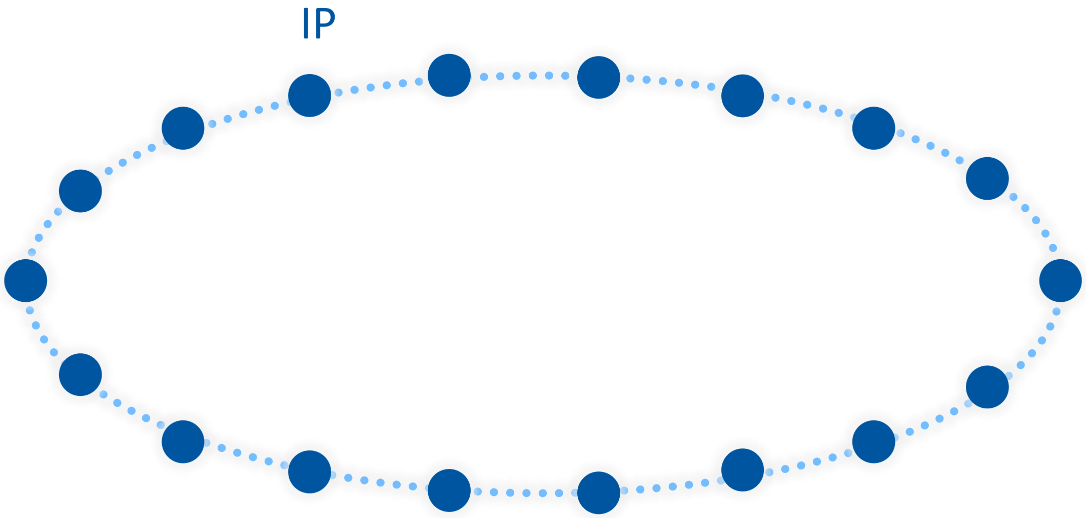
- Simulation with strong-strong 2D macro-particle multi-bunch tracking code

- Aim to estimate the effect of parameters on instability rise-time
  - Beam parameters: intensity, emittance, etc.
  - Constraints for vacuum pressure and composition

- Relevant for several possible future projects at CERN
  - CLIC (Compact Linear Collider)
  - FCC-ee, FCC-he (Future Circular Collider)
  - LHeC (Large Hadron Electron Collider)

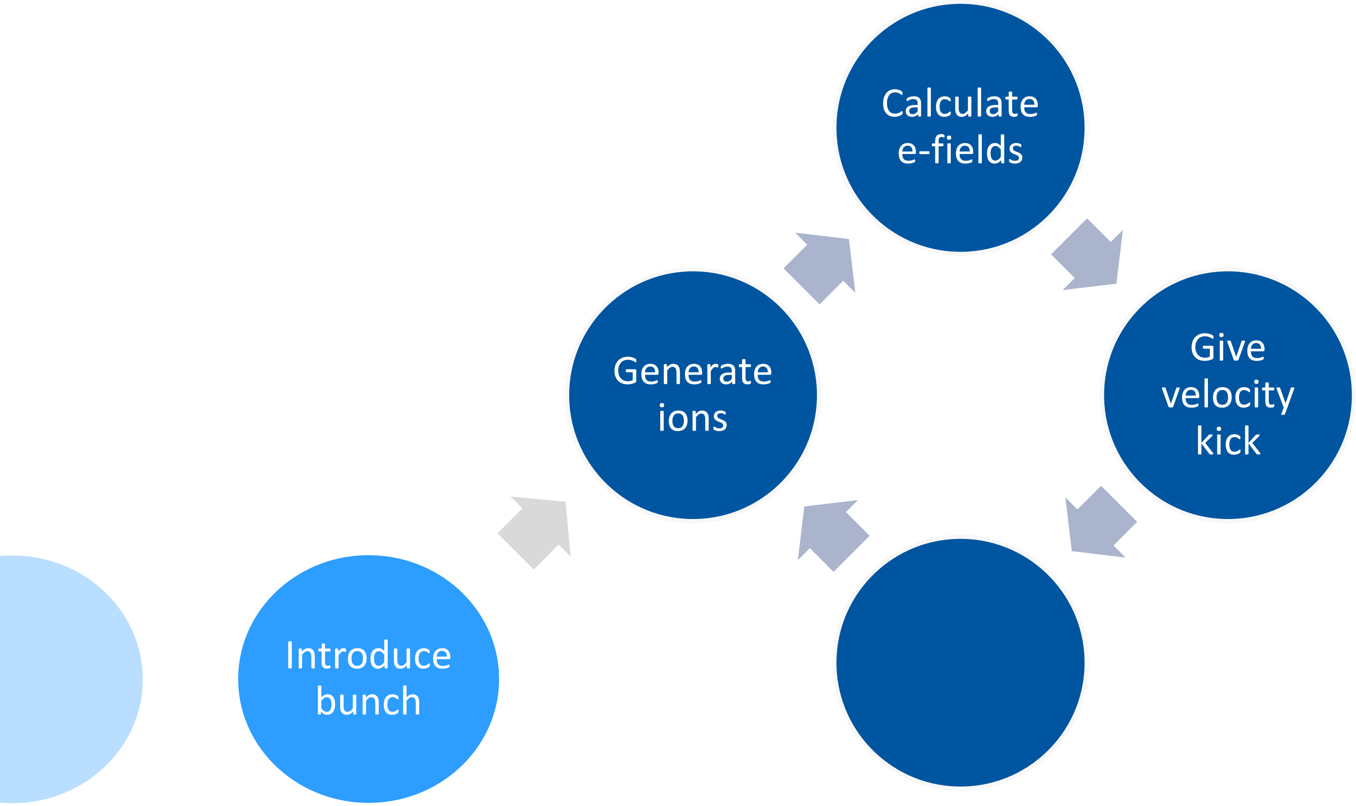
Simulation outline

- The machine lattice is divided into a number of interaction points (IP)
    - An electron bunch train is tracked through the lattice
    - In every IP, the beam-ion interaction is simulated



Simulation outline

- In every interaction point, the beam is passed bunch by bunch



Introduce bunch

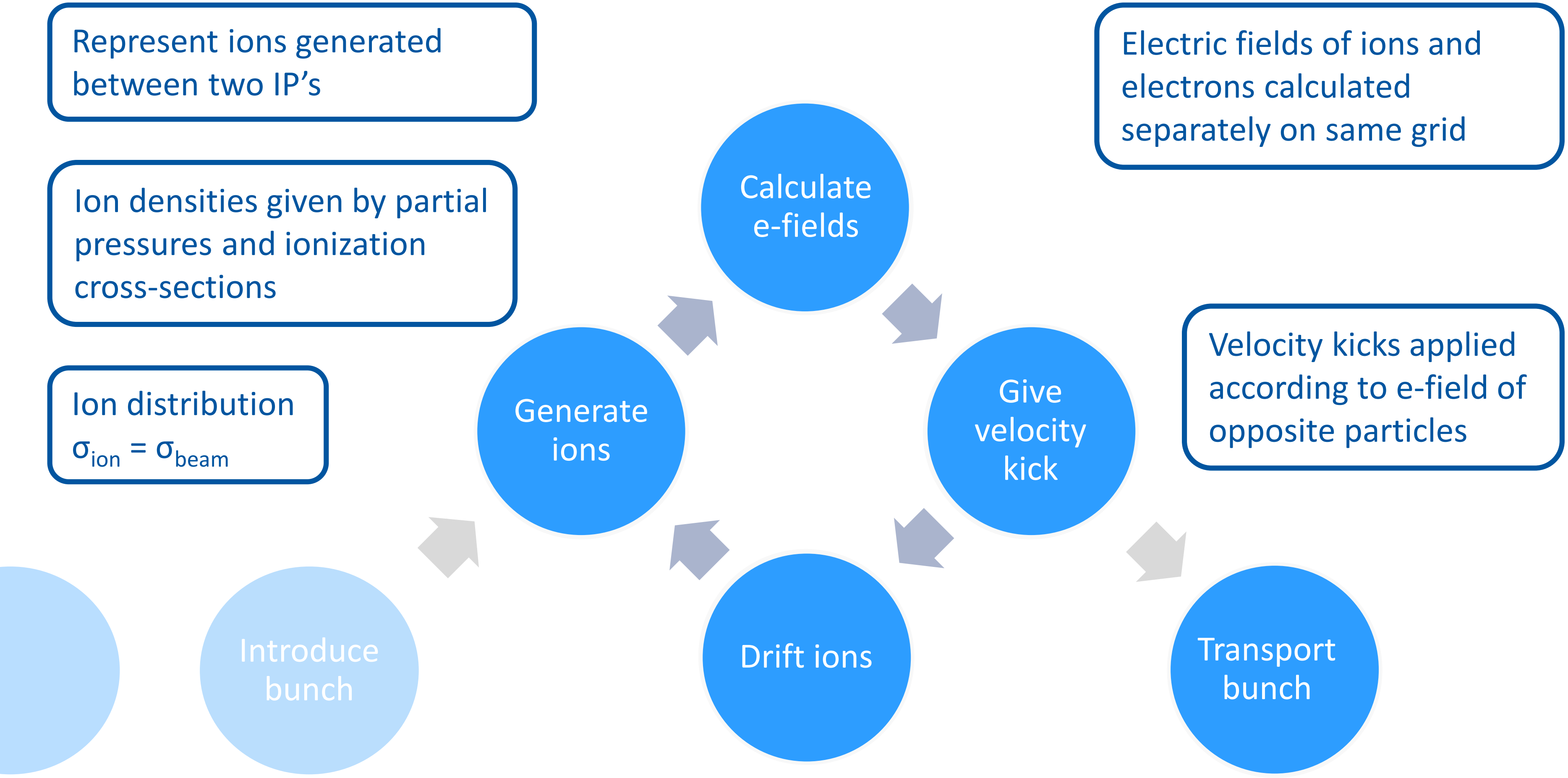
Generate ions

Calculate e-fields

Give velocity kick

Simulation outline

- In every interaction point, the beam is passed bunch by bunch



Represent ions generated between two IP's

Ion densities given by partial pressures and ionization cross-sections

Ion distribution $\sigma_{ion} = \sigma_{beam}$

Electric fields of ions and electrons calculated separately on same grid

Velocity kicks applied according to e-field of opposite particles

Simulation outline

- In every interaction point, the beam is passed bunch by bunch

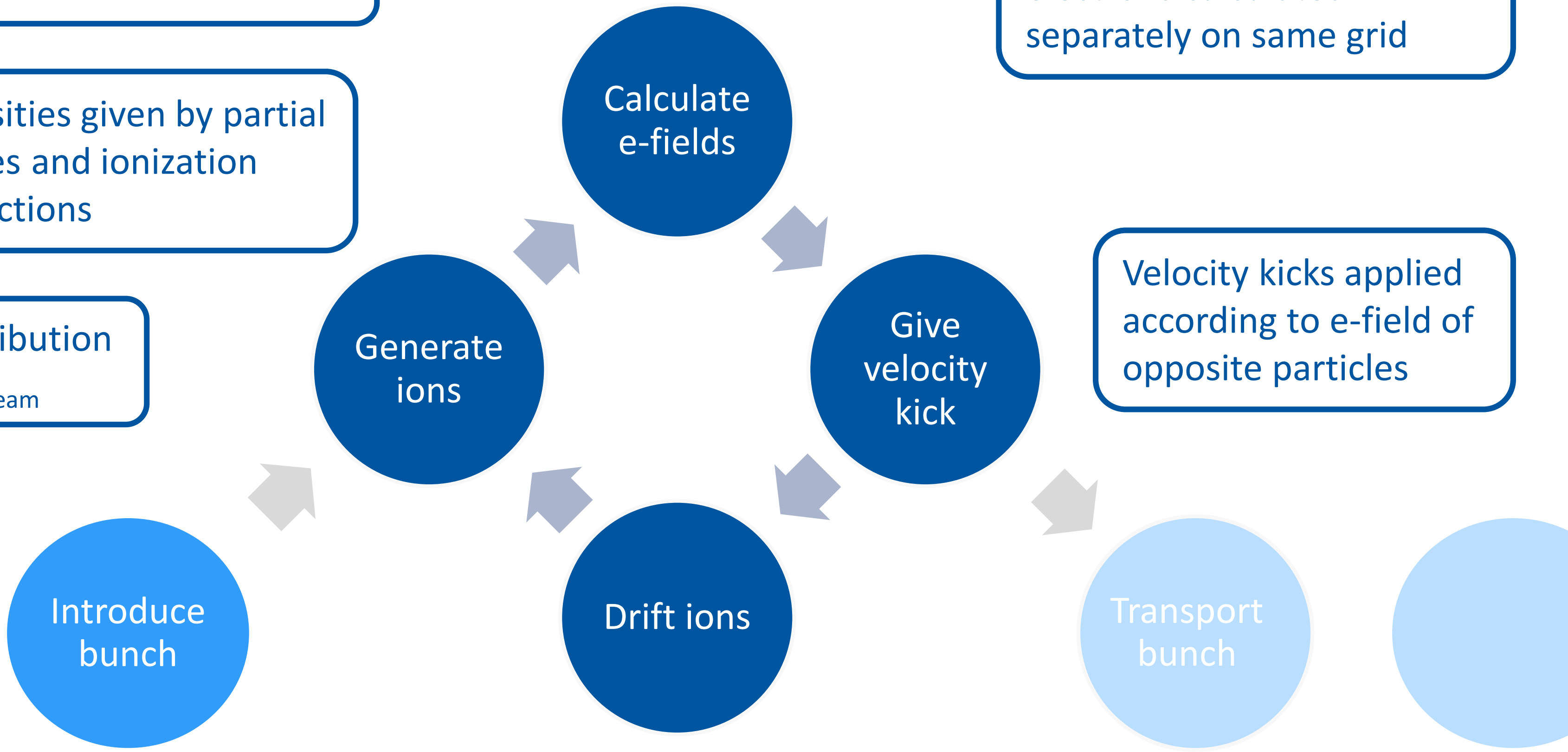
Represent ions encountered between two IP’s

Ion densities given by partial pressures and ionization cross-sections

Ion distribution
$\sigma_{ion} = \sigma_{beam}$

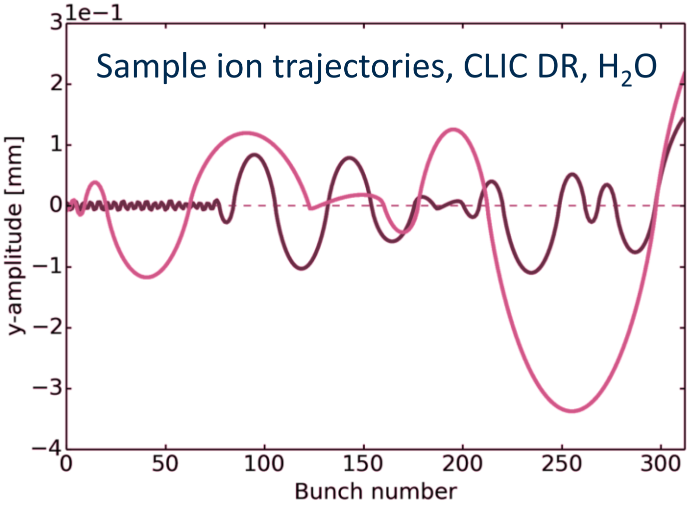
Electric fields of ions and electrons calculated separately on same grid

Velocity kicks applied according to e-field of opposite particles



Simulation codes

- FASTION
  - Developed at CERN, by G. Rumolo
  - Written in C
  - Based on HEADTAIL for electron cloud
  - Used e.g. for CLIC linear structures, CESR-TA
  - Recent work to adapt to future synchrotrons



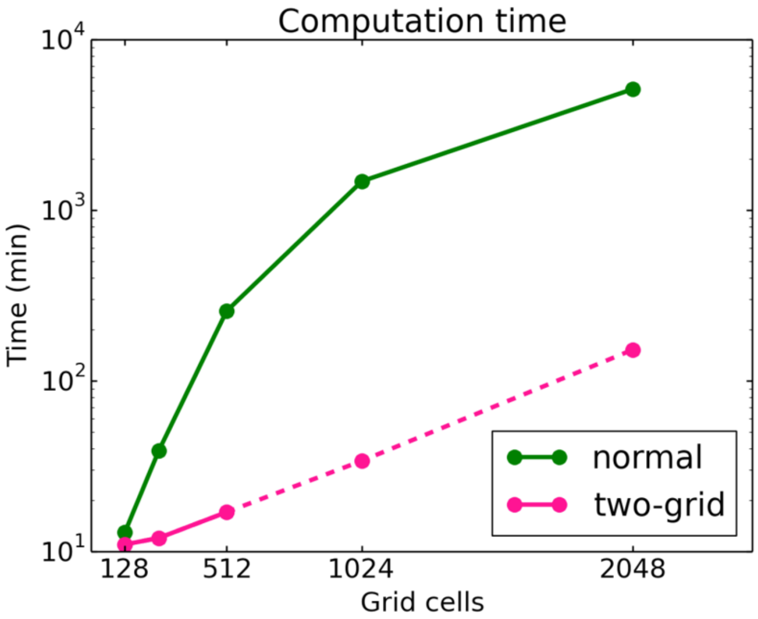
- Several CERN beam dynamics codes re-designed in the past few years
  - Aim to make codes more user friendly, maintainable and flexible
  - Electron cloud build-up: ECLOUD → PyECLOUD
  - Collective effects: HEADTAIL → PyHEADTAIL

coupled

- Decision to incorporate fast beam-ion instability simulations into framework of PyECLOUD and PyHEADTAIL codes

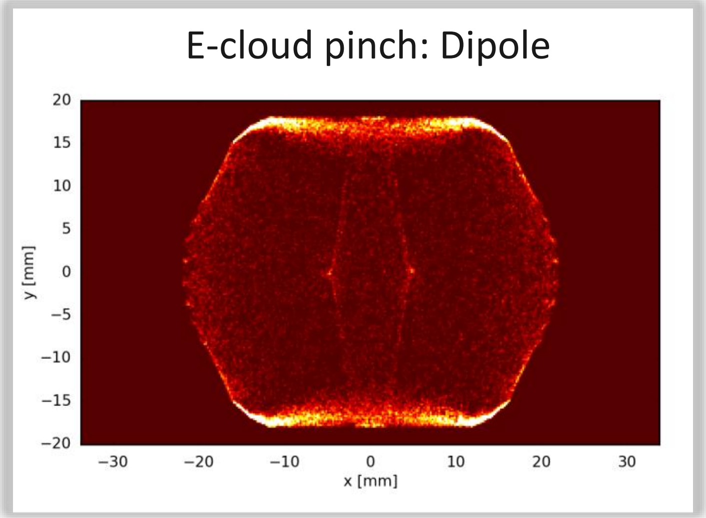
Recent work on FASTION

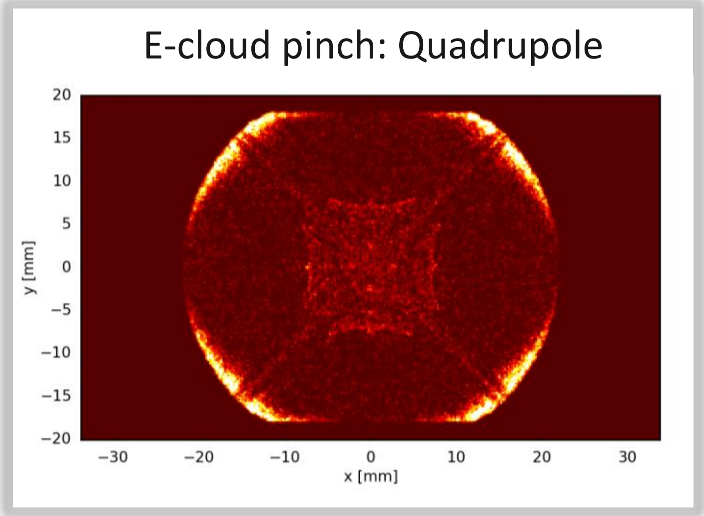
- Run-time optimization
  - For CLIC damping ring, simulate at least one damping time, $\tau_d \approx 1400$ turns
    - Initially: 1 turn, 27 min → 26 days for 1400 turns
    - After optimization: 1 turn, 16 min → 15 days for 1400 turns
- Resolution problems
  - Due to rescaling of grid to adapt to expanding ion distribution
    - Good resolution → impossibly long run-time
- Solution: two-grid method
  - Keep grid around beam fixed
  - Use second grid to track outer ions
    - 1 turn, 15 min → 14 days for 1400 turns
    - With improved resolution
- Next step: parallelization



PyECLOUD

- Simulation code for electron cloud formation
  - Developed and maintained at CERN since 2011 by G. Iadarola
  - Following the legacy of the ECLOUD code (F. Zimmermann et al., 1997…)
  - 2D macro-particle code
  - Written in Python with Fortran (f2py) and C (CYTHON) extensions
- Features
  - Gas ionization
  - Boris algorithm for moving particles in magnetic fields
  - PIC module with multiple solver methods for arbitrarily shaped chambers

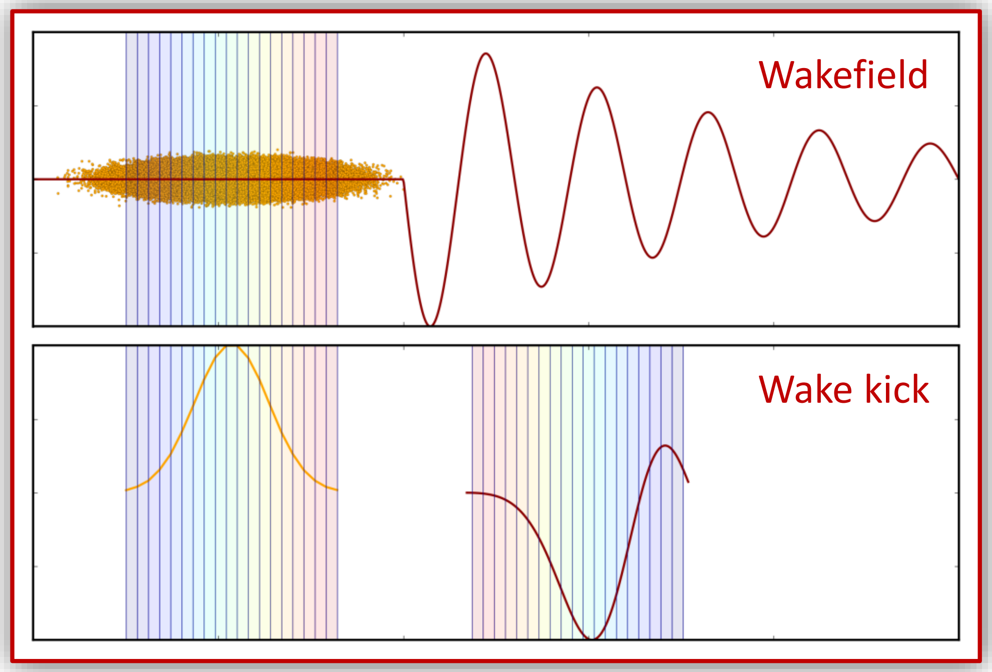

E-cloud pinch: Dipole


E-cloud pinch: Quadrupole

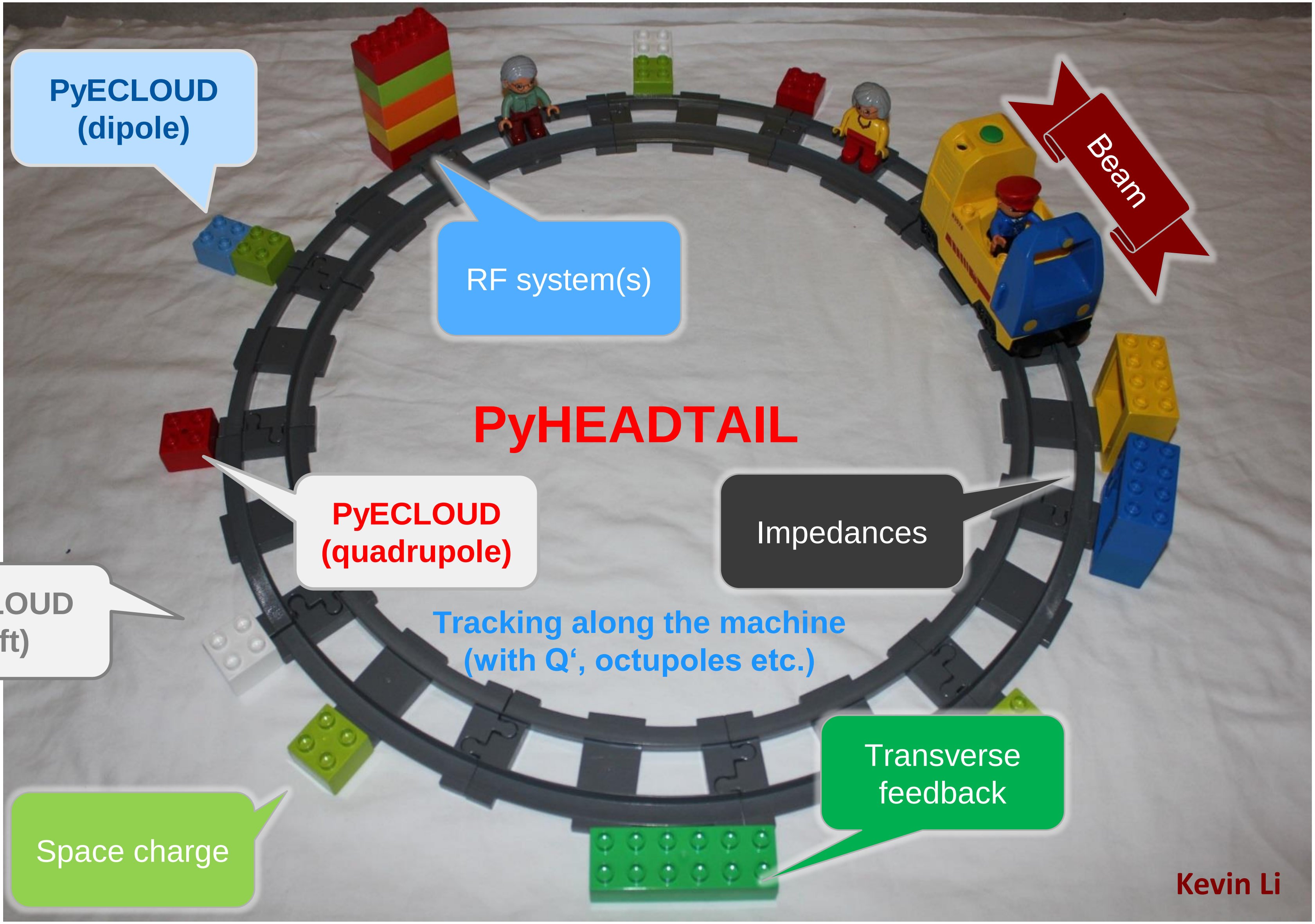
G. Iadarola

PyHEADTAIL

- Simulation code for collective effects: impedance, e-cloud instability, etc.
  - Developed and maintained at CERN since 2014 by H. Bartosik, K. Li et al.
  - Based on the HEADTAIL code (G. Rumolo et al., 2000)
  - 2-3D macro-particle code
  - Written in Python with C (CYTHON) extensions
- Features
  - RF systems and synchrotron motion
  - Chromaticity
  - Transverse feedback
- github.com/PyCOMPLETE
  - PyHEADTAIL
  - PyECLOUD
  - PyPIC



PyHEADTAIL coupled to PyECLOUD



PyECLOUD (dipole)

RF system(s)

Beam

PyHEADTAIL

PyECLOUD (quadrupole)

Impedances

PyECLOUD (drift)

Tracking along the machine (with Q‘, octupoles etc.)

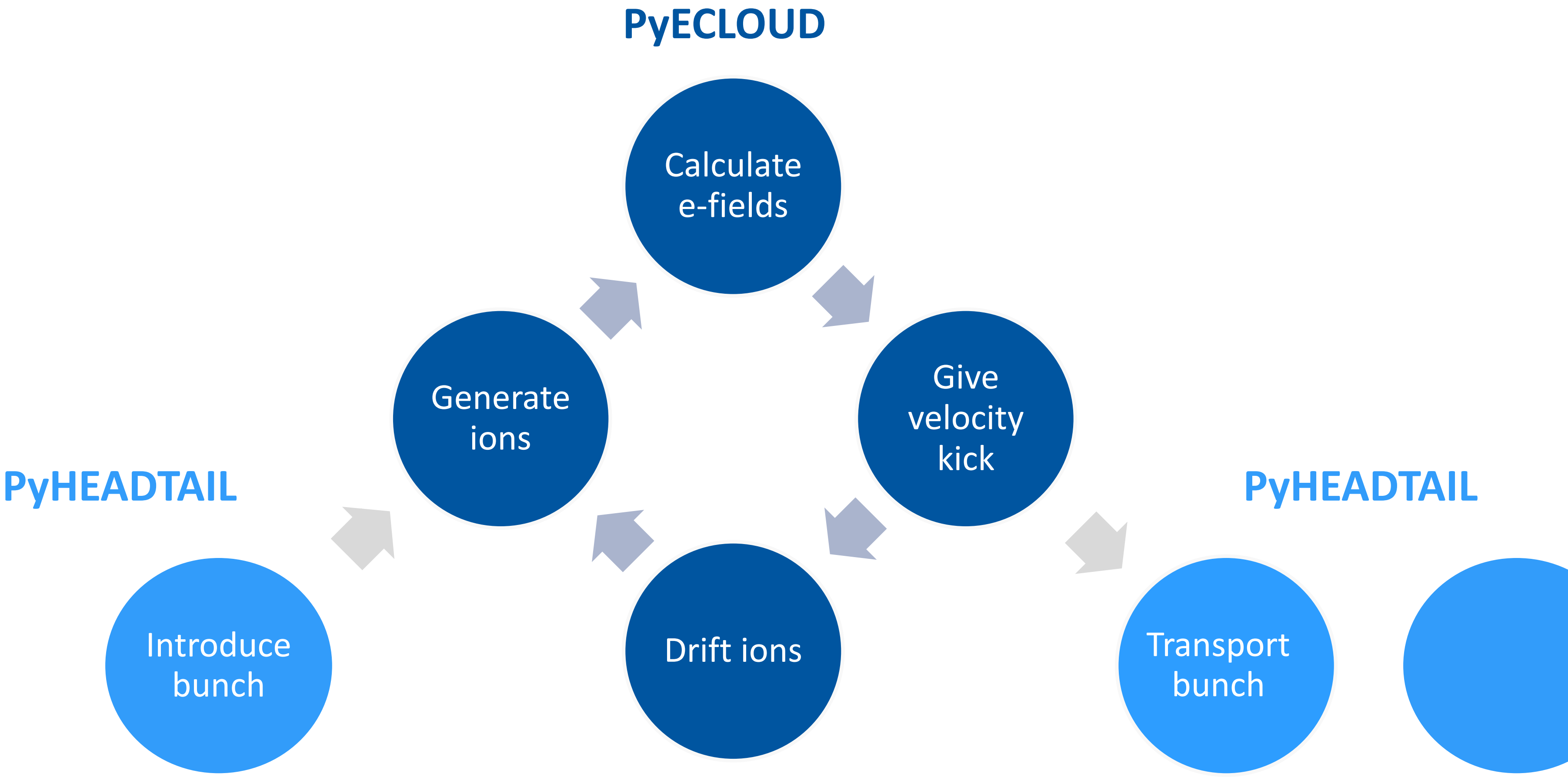
Space charge

Transverse feedback

Kevin Li

Ion simulations with PyECLOUD and PyHEADTAIL

- Interface for PyECLOUD as PyHEADTAIL module for beam dynamics simulations



PyECLOUD

Calculate e-fields

Give velocity kick

Drift ions

Generate ions

PyHEADTAIL

Introduce bunch

PyHEADTAIL

Transport bunch

Progress

- ✓ Generalization to arbitrary charge and mass in PyECLOUD/PyHEADTAIL

- ✓ Implementation of multi-bunch in PyHEADTAIL

- ➢ Extension of PyECLOUD gas ionization routines — In progress
  - ○ Multiple species, field ionization

- ➢ Implementation of grid methods — In progress
  - ○ Rectangular (non-square) grid cells
    - Necessary for very flat beams (e.g. CLIC)
  - ○ Two-grid method

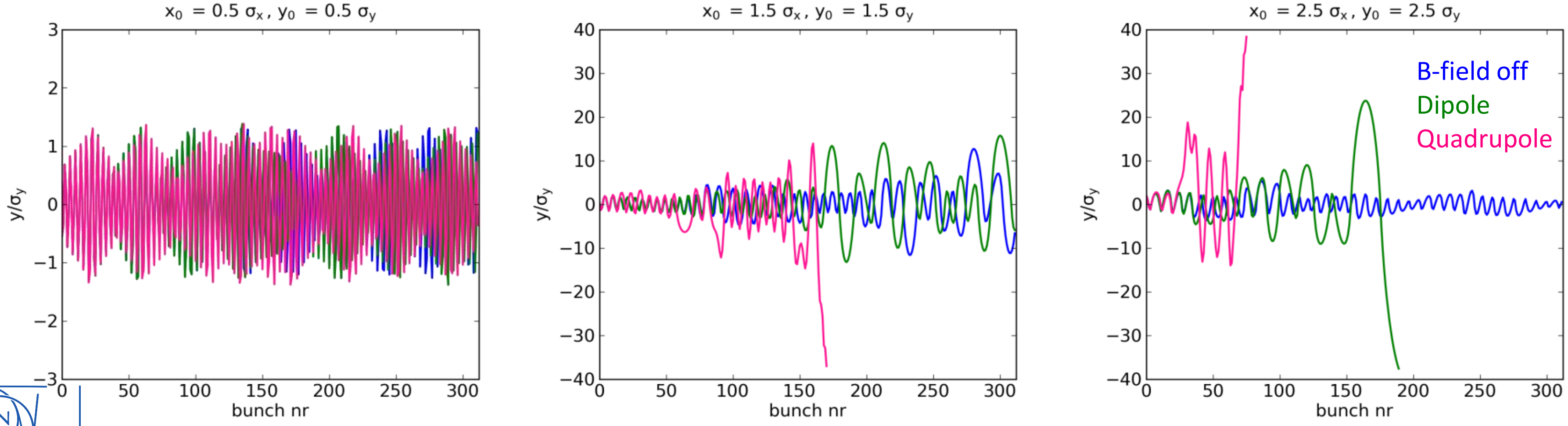
- ➢ Testing

- ➢ Parallelization

Benefits

- Facilitate continued maintenance
- Share workload of common developments, e.g. parallelization
- Benefit from features already implemented in PyECLOUD/PyHEADTAIL
    - Ion self space charge
    - E-field boundary
    - Magnetic fields
    - Bunch slices
    - Synchrotron motion
    - Transverse feedback
- Separate ion build-up and instability simulations for multi-turn ion trapping

Trajectories for H_2O ion during passage of CLIC-like bunch train, with B-fields



Summary

- The FBII can affect electron machines with clearing gaps
  - Usually not a problem in current machines with nominal vacuum
  - Expected to be relevant for future CERN projects: CLIC, FCC-ee, LHeC
  - Also new and upgraded low emittance light sources

- Simulation studies
  - 2D macro-particle, multi-bunch tracking simulations
  - Estimate the effect of beam and vacuum parameters on instability

- FASTION → PyECLOUD + PyHEADTAIL
  - Work in progress
  - Add maintainability and flexibility
  - Share workload of common development
  - Introduce many new features and possibilities

Thank you!

www.cern.ch