Xsuite

Integrated Beam Physics Simulation Framework

G. Iadarola, **R. De Maria, S. Łopaciuk,**
A. Abramov, X. Buffat, D. Demetriadou, L. Deniau, P. Hermes, P. Kicsiny, P. Kruyt, A. Latina,
L. Mether, K. Paraschou, G. Sterbini, F. Van Der Veken, CERN, Geneva, Switzerland

P. Belanger, TRIUMF, Vancouver, Canada

D. Di Croce, T. Pieloni, L. Van Riesen-Haupt, M. Seidel, EPFL, Lausanne, Switzerland

P. Niedermayer, GSI, Darmstadt, Germany

Work supported by: CHART

https://xsuite.web.cern.ch

Outline

- **Introduction**
    - Motivation
    - Design goals and constraints
    - Architecture
    - Agile development
- **Lattice modeling, simulation and optimization**
    - Lattice modeling
    - Single-particle tracking
    - Dynamic parameter control
    - Multi-objective optimizer
- **Simulation of specific processes**
    - Particle-matter interaction
    - Synchrotron radiation
    - Collective effects
- **Final remarks**

Outline

- **Introduction**
    - Motivation
    - Design goals and constraints
    - Architecture
    - Agile development
- **Lattice modeling, simulation and optimization**
    - Lattice modeling
    - Single-particle tracking
    - Dynamic parameter control
    - Multi-objective optimizer
- **Simulation of specific processes**
    - Particle-matter interaction
    - Synchrotron radiation
    - Collective effects
- **Final remarks**

Introduction

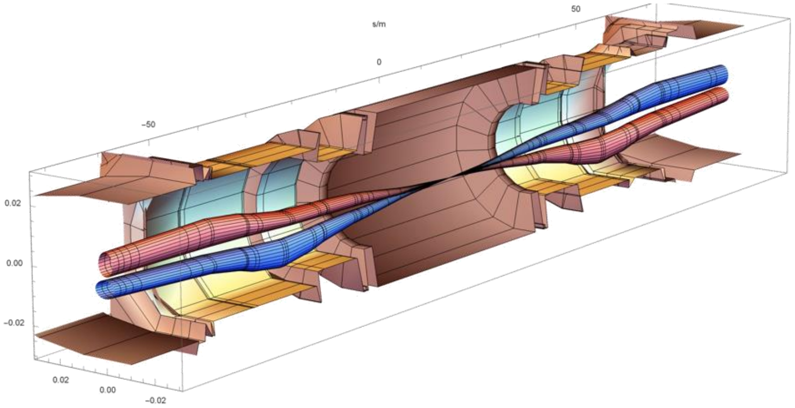
Coming from **long tradition** in the **development of software tools for beam physics**

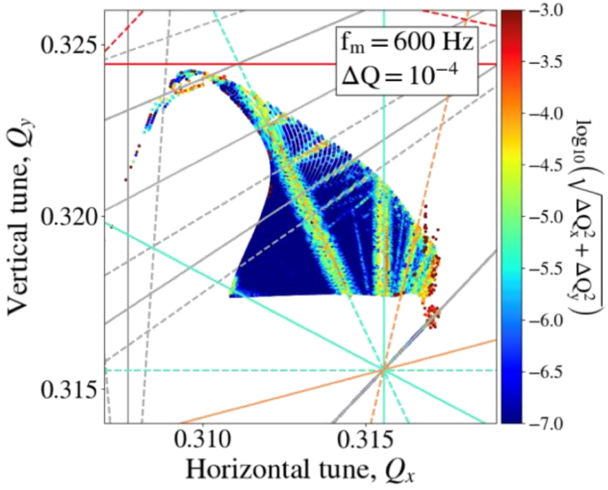
Powerful **tools** provided to the user community:

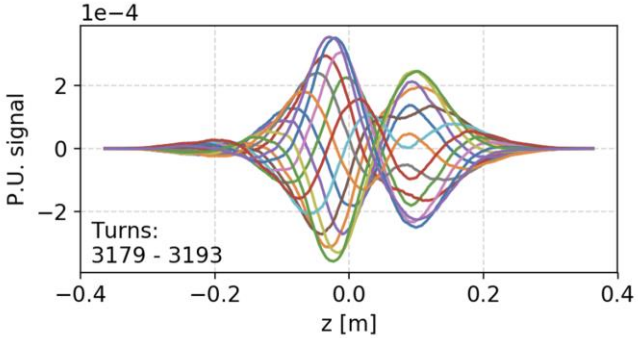
- **MAD-X**, standard for lattice description, optics calculation and design, tracking
- **Sixtrack**, a **fast-tracking program** used mainly for long **single-particle simulations**
- **Sixtracklib**, a **C/C++library for single-particle tracking** compatible with Graphics Processing Units (GPUs)
- **COMBI**, for the simulation of **beam-beam** effects using **strong-strong modelling**
- **PyHEADTAIL**, a **Python** toolkit for **collective effects** (impedance, feedbacks, space charge, and e-cloud).

Developed over decades, providing **powerful features** in their respective domains

- Nevertheless, **limitations also became apparent**…



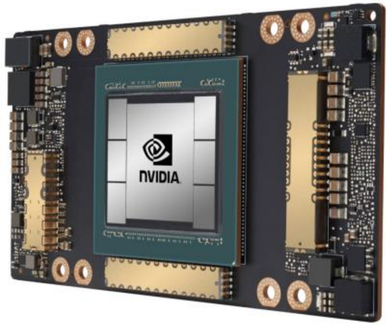




Motivation

Over the years we started to **facing needs that could not be easily fulfilled with legacy tools**:

- **Integration**: very difficult to combine features from different codes to simulate complex heterogeneous effects
- **Extendability:** difficult to extend to present needs (notably those coming from FCC-ee)
- **User interface**: need to move away from custom interfaces (text files / ad-hoc scripting) towards standard **Python packages**
  - Present **de-facto standard in scientific computing**
  - Allows **leveraging an ever-growing arsenal of general-purpose Python libraries** (statistics, linear algebra, frequency analysis, optimization, plot, etc.)
    - Boosted by **large investments from industry** (big data, AI)



- **GPU acceleration**: mandatory for several applications but cumbersome to retrofit in the existing codes

Xsuite

This led to the **launch of the Xsuite project** in **2021**

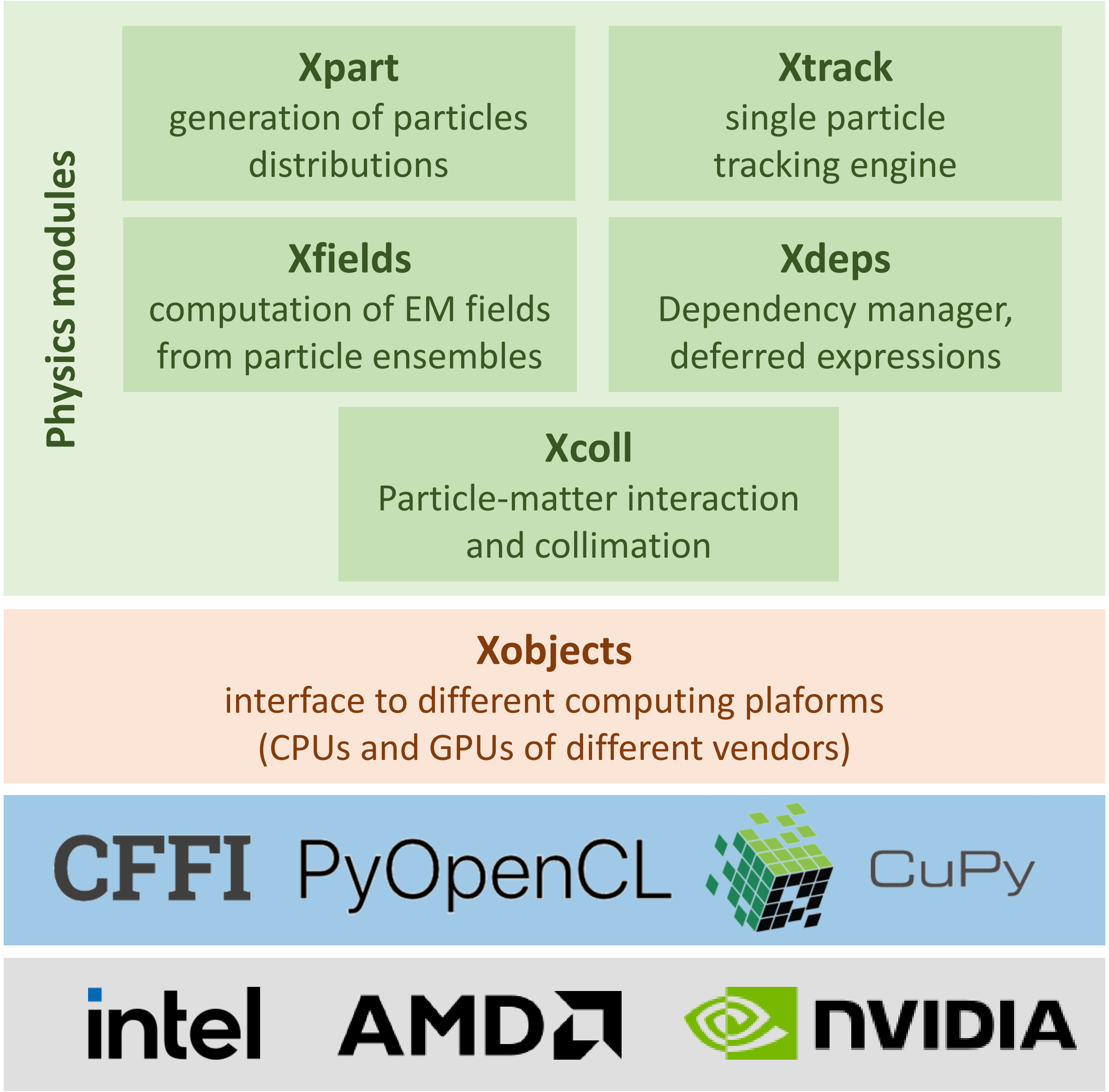
- **Main goal:** bring into a **modern Python toolkit** the know-how built in developing MAD, Sixtrack, COMBI…
    - **Cover with one toolkit** applications ranging from **low-energy hadron rings** to **high-energy lepton colliders**
- Designed for **seamless integration of components** and for **extendibility**
- Support **different computing platforms**, including **multicore CPUs and GPUs** from different vendors

Design constraints:

- Need **to grow the code in a “sustainable” way**, being managed and maintained by a **small core team** integrating (in a clean way!) contributions by a wide developer community
- Need **developer learning curve** to be short as possible

→ Field specific features developed **directly by field experts**

Xsuite – architecture



Physics modules

Xpart
generation of particles distributions

Xtrack
single particle tracking engine

Xfields
computation of EM fields from particle ensembles

Xdeps
Dependency manager, deferred expressions

Xcoll
Particle-matter interaction and collimation

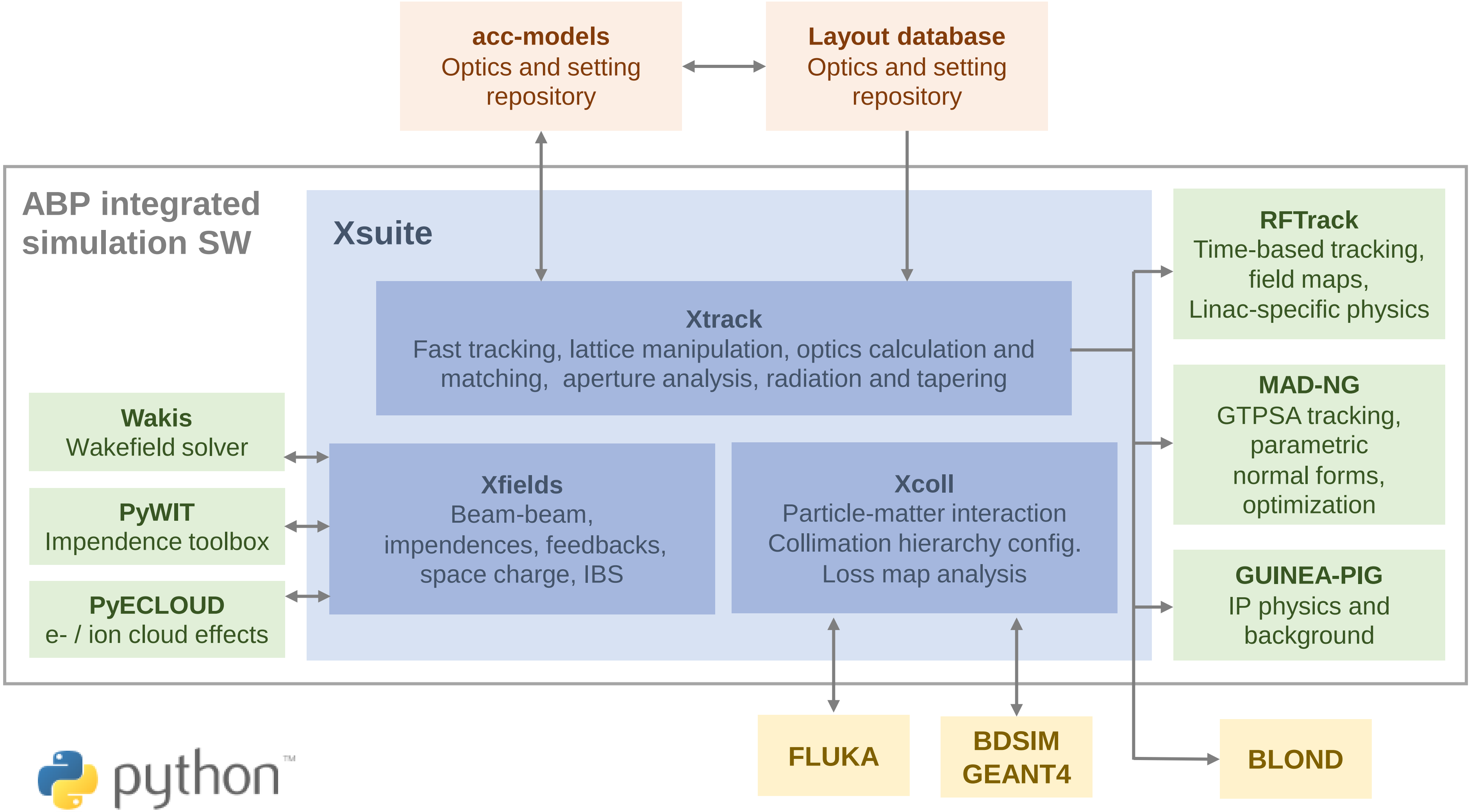
Xobjects
interface to different computing plaforms (CPUs and GPUs of different vendors)

Lower level libraries
(external, open source)

Hardware

Broader software architecture

Xsuite working as aggregation point of a wider Python ecosystem (under development)



acc-models
Optics and setting repository

Layout database
Optics and setting repository

ABP integrated simulation SW

Xsuite

Xtrack
Fast tracking, lattice manipulation, optics calculation and matching, aperture analysis, radiation and tapering

Xfields
Beam-beam, impendences, feedbacks, space charge, IBS

Xcoll
Particle-matter interaction
Collimation hierarchy config.
Loss map analysis

Wakis
Wakefield solver

PyWIT
Impendence toolbox

PyECLOUD
e- / ion cloud effects

RFTrack
Time-based tracking, field maps, Linac-specific physics

MAD-NG
GTPSA tracking, parametric normal forms, optimization

GUINEA-PIG
IP physics and background

FLUKA

BDSIM GEANT4

BLOND

Orthogonal design

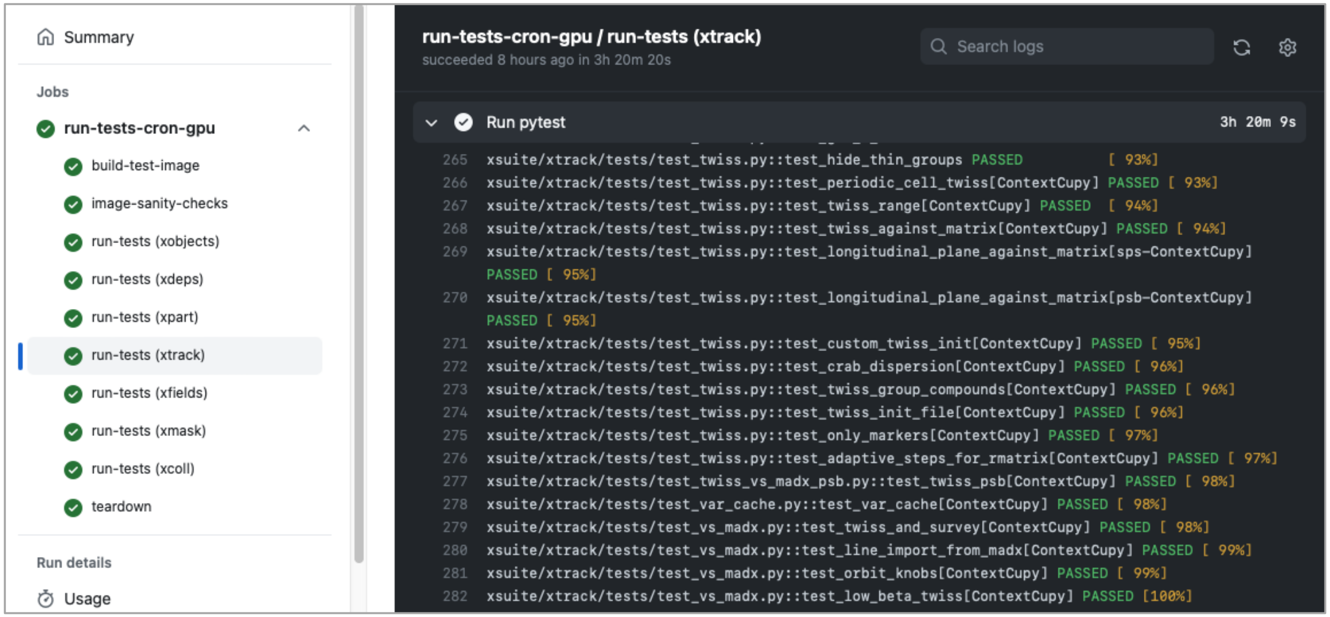
Project employs an **"orthogonal" design strategy**:

- Ensure that **each functional block** remains **well-isolated** and interacts with the others through **clearly defined interfaces**
- This approach has two key **advantages**:
  - Enables **contributors** to **modify or expand** specific components **without full knowledge of other parts** nor of underlying software infrastructure
  - **Minimize codebase complexity**, ensuring that it **increases linearly rather than exponentially** as new features are added

Development strategy

Our development follows the **agile principles**:

- **We engaged with the user community**, encouraging and **supporting users** to **test and exploit** available features in **full-scale studies since early development stages**
- Code **evolves incrementally**, promptly applying needed fixes and improvements
- **Rely on fast release cycle** (new versions released multiple times per month) while ensuring **no disruption due to version changes** on the user's side.
- Made possible by large investment **automatic testing**: each version of Xsuite undergoing over a **thousand automatic checks** (on CPU and GPU)



run-tests-cron-gpu / run-tests (xtrack)
succeeded 8 hours ago in 3h 20m 20s

Run pytest 3h 20m 9s

```
xsuite/xtrack/tests/test_twiss.py::test_hide_thin_groups PASSED          [ 93%]
xsuite/xtrack/tests/test_twiss.py::test_periodic_cell_twiss[ContextCupy] PASSED [ 93%]
xsuite/xtrack/tests/test_twiss.py::test_twiss_range[ContextCupy] PASSED  [ 94%]
xsuite/xtrack/tests/test_twiss.py::test_twiss_against_matrix[ContextCupy] PASSED [ 94%]
xsuite/xtrack/tests/test_twiss.py::test_longitudinal_plane_against_matrix[sps-ContextCupy]
PASSED [ 95%]
xsuite/xtrack/tests/test_twiss.py::test_longitudinal_plane_against_matrix[psb-ContextCupy]
PASSED [ 95%]
xsuite/xtrack/tests/test_twiss.py::test_custom_twiss_init[ContextCupy] PASSED [ 95%]
xsuite/xtrack/tests/test_twiss.py::test_crab_dispersion[ContextCupy] PASSED [ 96%]
xsuite/xtrack/tests/test_twiss.py::test_twiss_group_compounds[ContextCupy] PASSED [ 96%]
xsuite/xtrack/tests/test_twiss.py::test_twiss_init_file[ContextCupy] PASSED [ 96%]
xsuite/xtrack/tests/test_twiss.py::test_only_markers[ContextCupy] PASSED [ 97%]
xsuite/xtrack/tests/test_twiss.py::test_adaptive_steps_for_rmatrix[ContextCupy] PASSED [ 97%]
xsuite/xtrack/tests/test_twiss_vs_madx_psb.py::test_twiss_psb[ContextCupy] PASSED [ 98%]
xsuite/xtrack/tests/test_var_cache.py::test_var_cache[ContextCupy] PASSED [ 98%]
xsuite/xtrack/tests/test_vs_madx.py::test_twiss_and_survey[ContextCupy] PASSED [ 98%]
xsuite/xtrack/tests/test_vs_madx.py::test_line_import_from_madx[ContextCupy] PASSED [ 99%]
xsuite/xtrack/tests/test_vs_madx.py::test_orbit_knobs[ContextCupy] PASSED [ 99%]
xsuite/xtrack/tests/test_vs_madx.py::test_low_beta_twiss[ContextCupy] PASSED [100%]
```

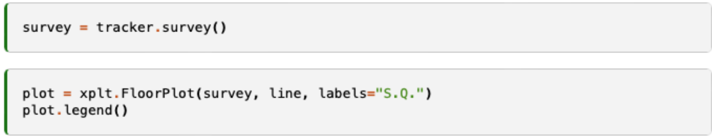
Outline

- **Introduction**
  - Motivation
  - Design goals and constraints
  - Architecture
  - Agile development
- **Lattice modeling, simulation and optimization**
  - Lattice modeling
  - Single-particle tracking
  - Dynamic parameter control
  - Multi-objective optimizer
- **Simulation of specific processes**
  - Particle-matter interaction
  - Synchrotron radiation
  - Collective effects
- **Final remarks**

Lattice modelling

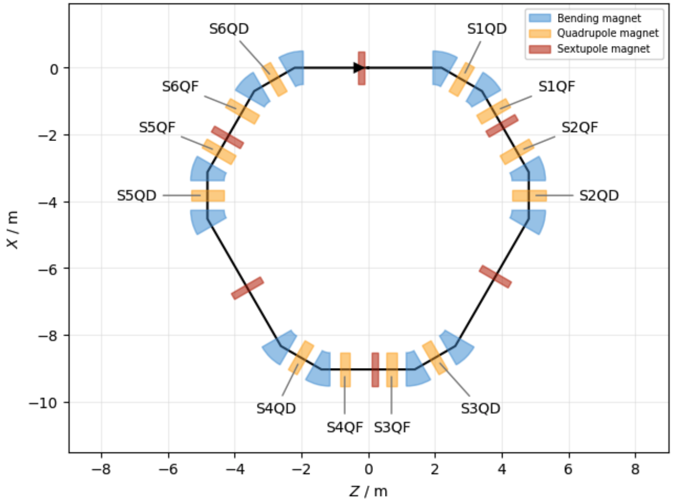
- The **beam line** is represented as a **sequence of Python objects**, each corresponding to an accelerator element or to other physical processes (e.g. magnets, cavities, aperture restrictions, etc.).
    - Can be **defined manually** or **imported from MAD-X**
        - Including **tilts**, **misalignments** and **multipolar errors**

Xsuite model of a ring
(represented with the [Xplt package](#))



```
survey = tracker.survey()
```

```
plot = xplt.FloorPlot(survey, line, labels="S.Q.")
plot.legend()
```



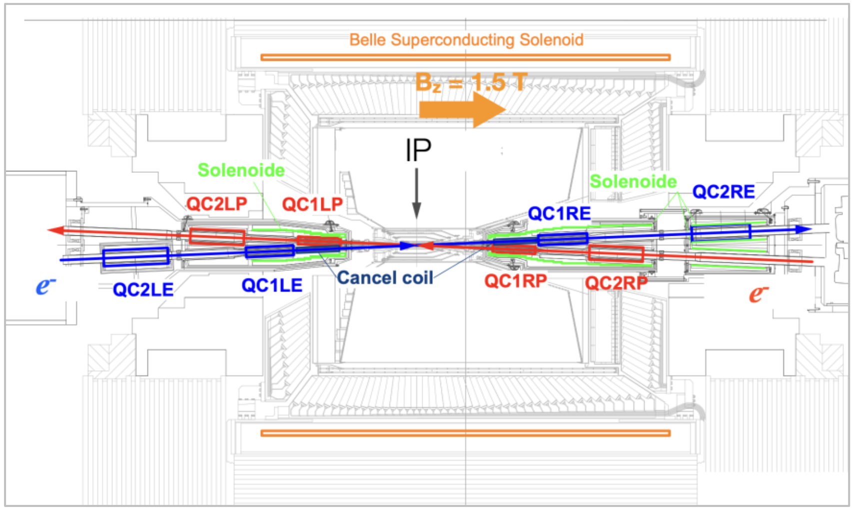
We provide:

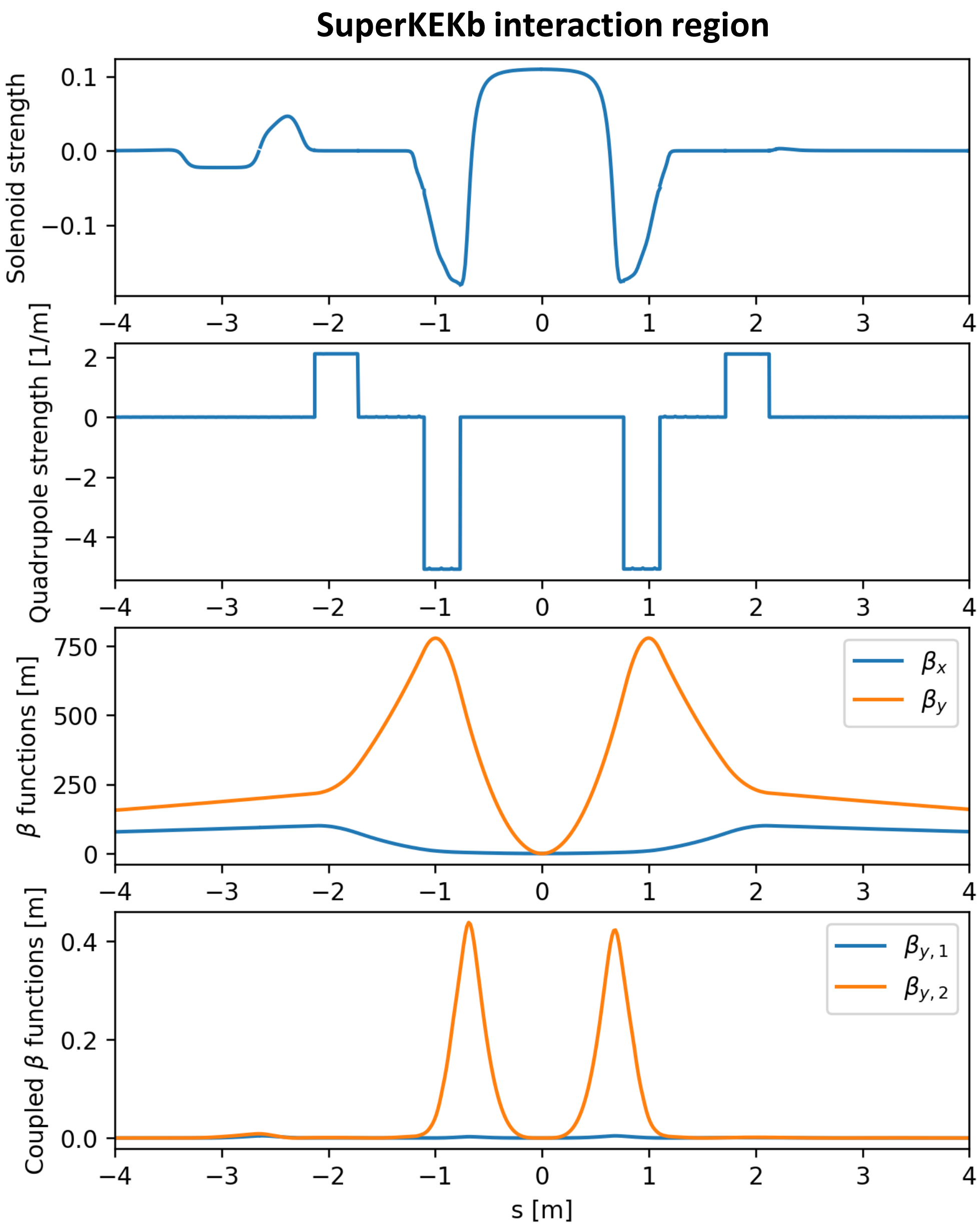
- **"Thin" lattice integration**, largely based on the Sixtrack and Sixtracklib experience
- **"Thick" maps** for bending and quadrupole magnets
- **Dipole edge effects** including **fringe fields** can be modeled either in their **linearized form** or as **full non-linear maps** (same fringe model as in MAD-NG and PTC).

Solenoids and overlapping elements

Recent developments allow modelling of **experimental solenoids** of lepton colliders also in the presence of **overlapping multipole fields**

- Tested on **FCC-ee** and **SuperKEKb** models




SuperKEKb interaction region

Many thanks to G. Broggi, J. P. Salvesen, KEK experts

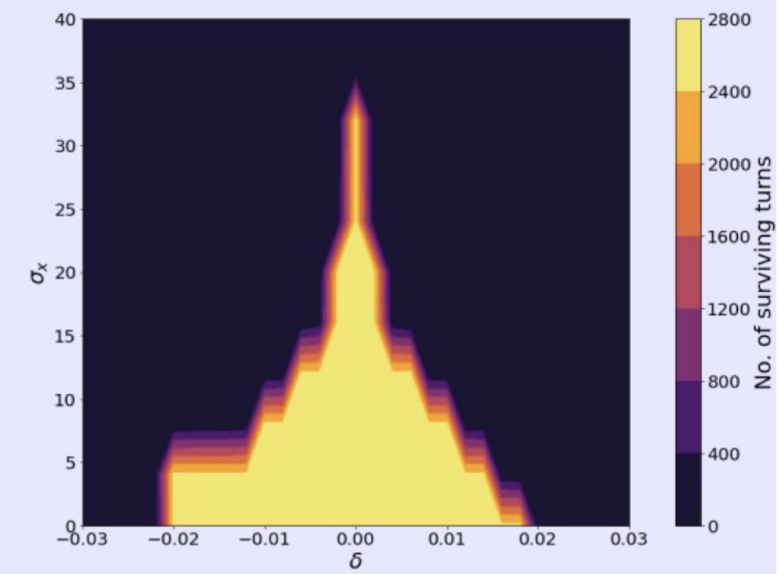
Single particle tracking

- Multiturn **element-by-element tracking speed** is critical for several application
- To **speed up tracking simulations**, Xsuite assembles and compiles a **C kernel** (callable from Python) **optimized for the given beamline** and **specialized for the chosen platform** (CPU or GPU)
  - The **tracking speed** is found to be **similar to Sixtrack** for single-core CPU and **about two orders of magnitudes faster than that on high-end GPUs**
  - Developments are well advanced to deploy Xsuite on the **LHC@Home** volunteer computing platform (collaboration with EPFL)

Tracking time for a typical LHC simulation

Platform	Computing time
CPU (single core)	190 (μs/part./turn)
GPU (NVIDIA V100, cupy)	0.80 (μs/part./turn)
GPU (NVIDIA V100 pyopencl)	0.85 (μs/part./turn)

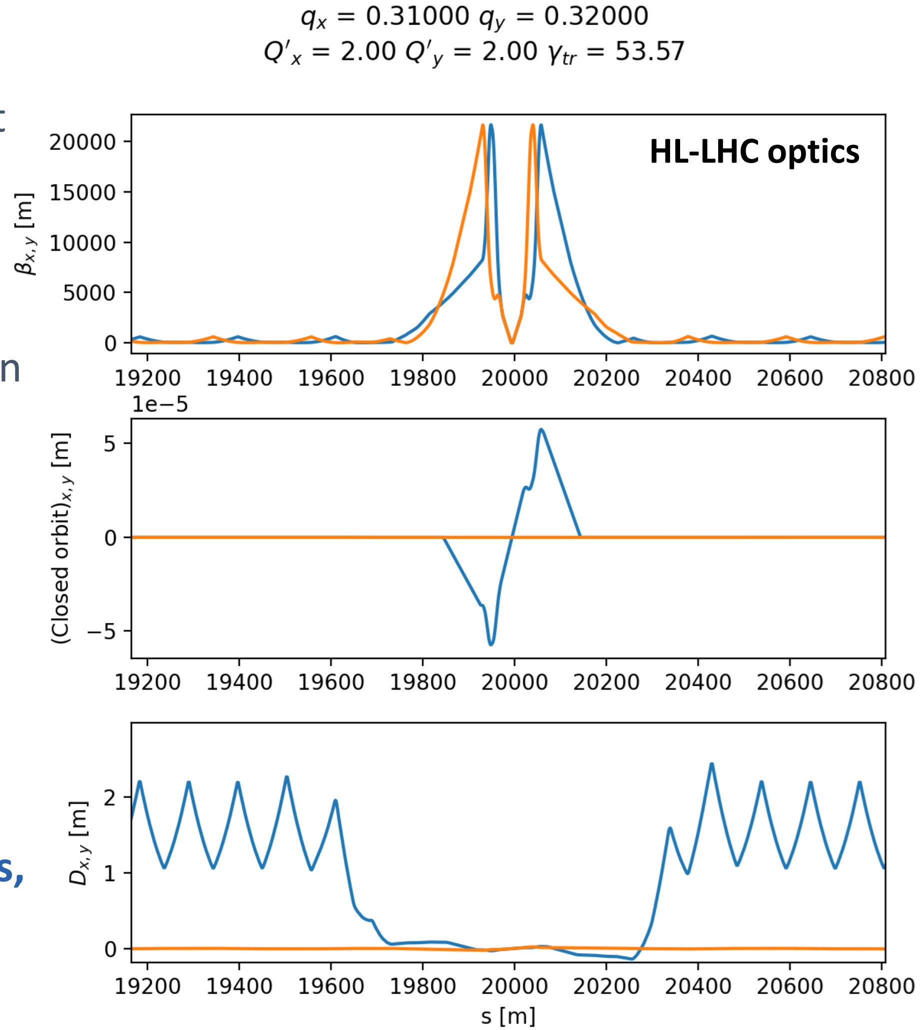
Example: Xsuite simulations used to study FCC-ee DA and momentum acceptance



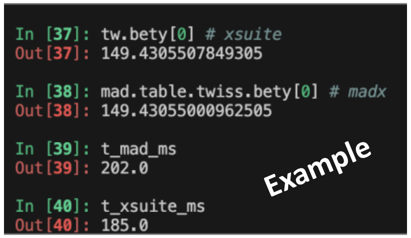
Courtesy M. Hofer

Twiss module

- The **Xsuite Twiss** module can be used to extract **lattice functions** of a ring or a beamline
- The calculation **probes the lattice <u>simply by tracking particles</u>**:
    - **Closed orbit** obtained by applying a Python root finder on the tracking
    - The **Jacobian** matrix obtained by **tracking (central differences)**
    - Compute **"Linear Normal Form"** of the Jacobian matrix (diagonalization)
    - **Propagate eigenvectors** by **tracking**
    - Obtain from the eigenvectors **Twiss parameters (α, β, γ) , dispersion functions, phase advances, coupling coefficients**
- Computation can be done with **assigned beam momentum** to get **off-momentum beta-beating**, **non-linear chromaticity, non-linear dispersion,** etc.



Accuracy compared to MAD-X: $\Delta\beta / \beta << 10^{-4}$
Computation time is very similar



```
In [37]: tw.bety[0] # xsuite
Out[37]: 149.4305507849305

In [38]: mad.table.twiss.bety[0] # madx
Out[38]: 149.43055000962505

In [39]: t_mad_ms
Out[39]: 202.0

In [40]: t_xsuite_ms
Out[40]: 185.0
```

Dynamic control of beam line parameters

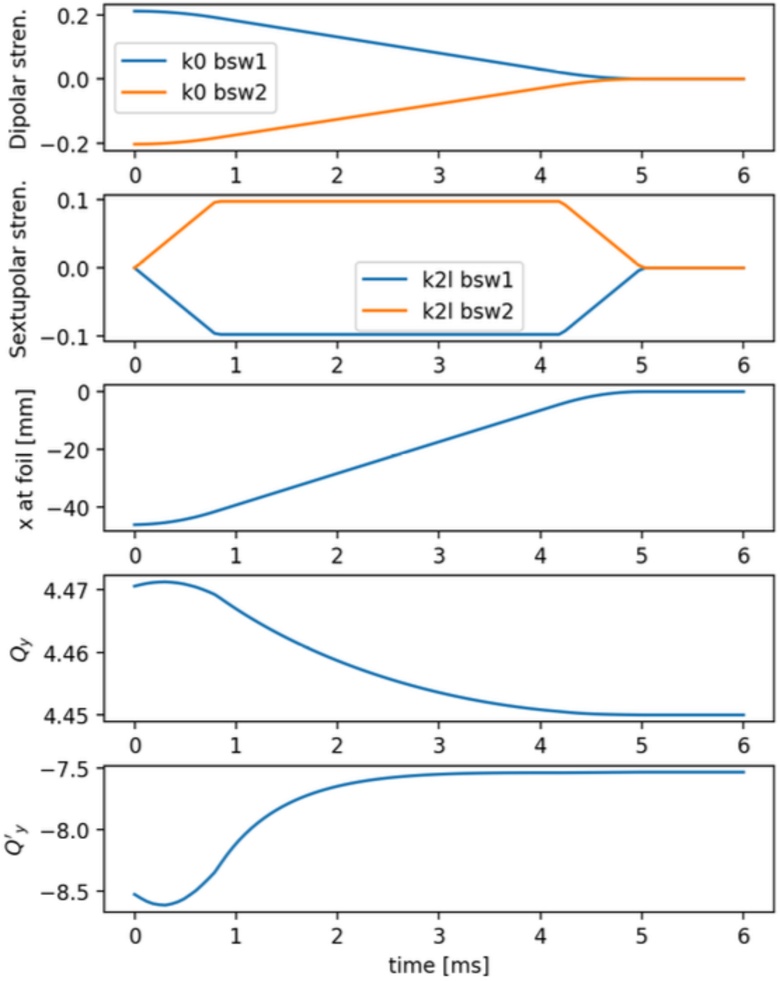
- In accelerators often a single **high-level parameter** can be used to control **groups of components** with complex dependency relations. (e.g. circuits with multiple magnets, groups of RF cavities, etc.)
- The **Xdeps module** provides the capability **to include such dependencies in the simulation model** (as done by MAD-X deferred expressions)
- **Example**, LHC crossing angle knob:

  At any time, the user can set:

  `lhc.vars['on_x1'] = 160 # murad`

  which **automatically changes the strength of 40 dipole correctors** to get the required crossing angle
- User can also use **"Time functions"**, i.e. **time dependent knobs** that are updated automatically during the simulation

Simulation of a fast orbit bump used for the H⁻ injection into the CERN PS Booster



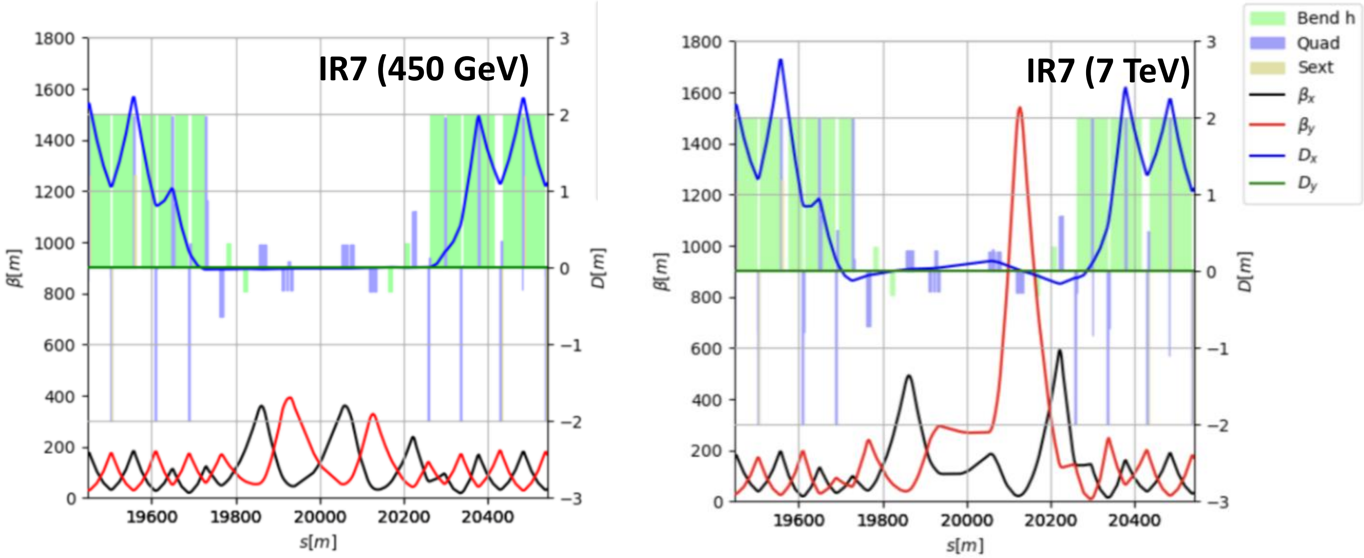
Optimizer

Xsuite provides a **multi-objective optimizer** to "match" model parameters to assigned constraints (e.g. control tunes, chromaticity, build orbit bumps, design the optics)

- Based on the **extensive experience of MAD-X** → Uses the **same optimization algorithm** (Jacobian, proven robustness)
- Interface **designed for usage flexibility**. User can **intervene in the optimization** by:
    - Enabling/disabling targets or knobs, rolling back optimization steps, changing knob limits, target values, convergence tolerances
- Used for **optics matching of the LHC and of FCC-ee colliders**

→ Proved capability of handling **large problems** with several constraints and degrees of freedom

First full cycle designed with Xsuite tested at the LHC in 2024
(including combined ramp&squeeze for all insertions)



Courtesy R. De Maria and B. Lindstrom

Outline

- **Introduction**
  - Motivation
  - Design goals and constraints
  - Architecture
  - Agile development
- **Lattice modeling, simulation and optimization**
  - Lattice modeling
  - Single-particle tracking
  - Dynamic parameter control
  - Multi-objective optimizer
- **Simulation of specific processes**
  - Particle-matter interaction
  - Synchrotron radiation
  - Collective effects
- **Final remarks**

Particle-matter interaction

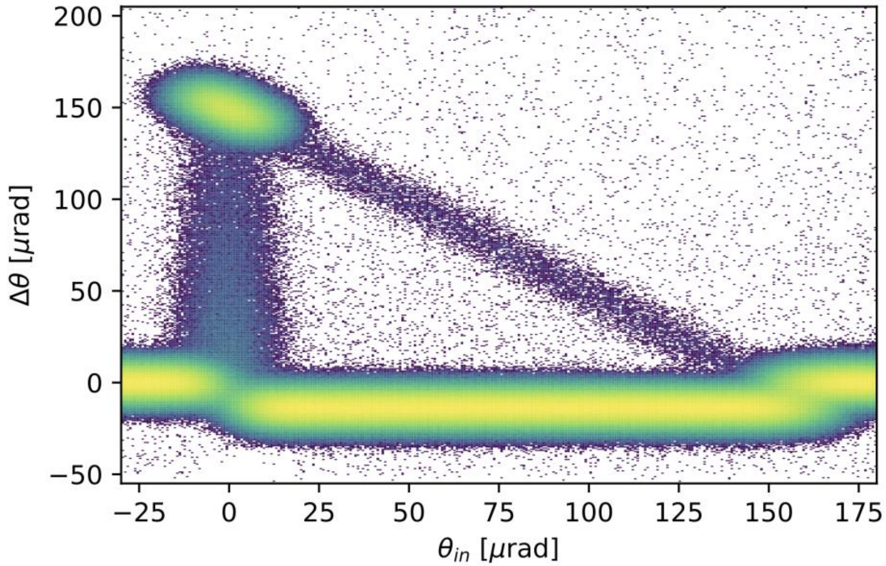
For collimation studies, the **Xcoll module** provides **three particle-matter sim. engines:**

- The **“Everest” engine** embedded in Xcoll (evolution of K2 module from Sixtrack)
- The **“Geant 4” engine**, based on an **interface with BDSIM-Geant4**
    - **Used for FCC-ee collimation studies** (see presentation by G. Broggi)
- The **“FLUKA” engine**, based on an interface with the **FLUKA** Monte Carlo code

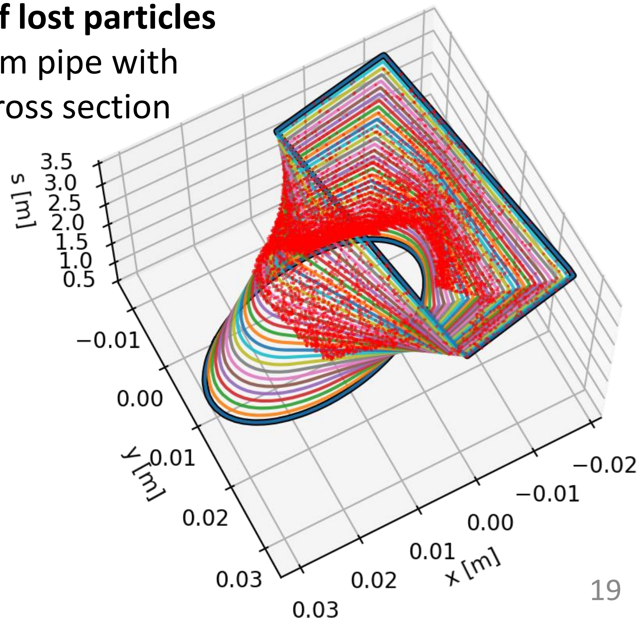
To support collimation studies, Xsuite provides:

- Tools to **automatically install and set collimators** in the simulation model
- Support for **complex aperture modelling** and **accurate localization of the lost particles along** the beam line (typically within 1-10 cm)

Particle deflection from a bent crystal (Everest engine)



Localization of lost particles
along a beam pipe with
changing cross section

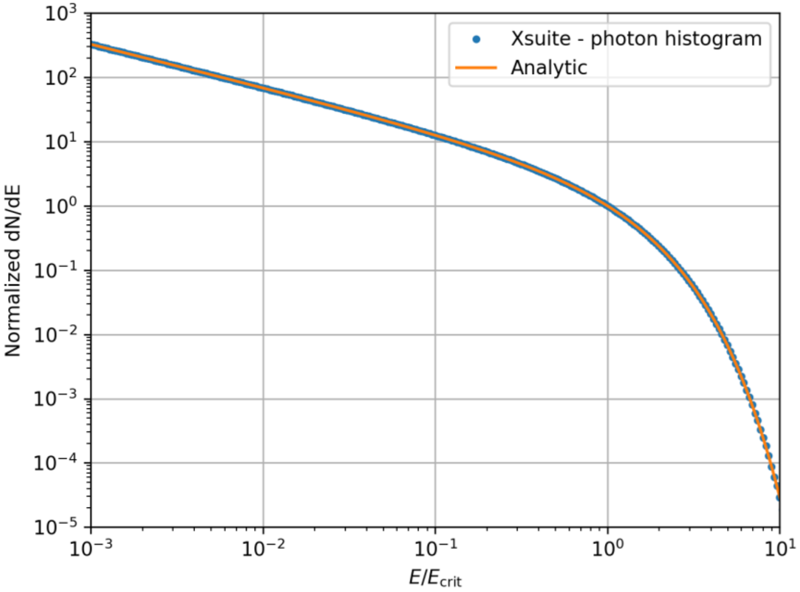


Synchrotron radiation

The effect of **synchrotron radiation** can be included in Xsuite tracking simulations. Two models available:

- The **"mean" model**, for which the energy loss from the radiation is applied particle by particle without accounting for quantum fluctuations;
- The **"quantum" model** for which the actual photon emission is simulated(1).

Validation against analytical photon spectrum



---

(1) *Based on H. Burkhardt, "Monte Carlo generator for synchrotron radiation", 1990. Implementation ported from PLACET (A. Latina)*

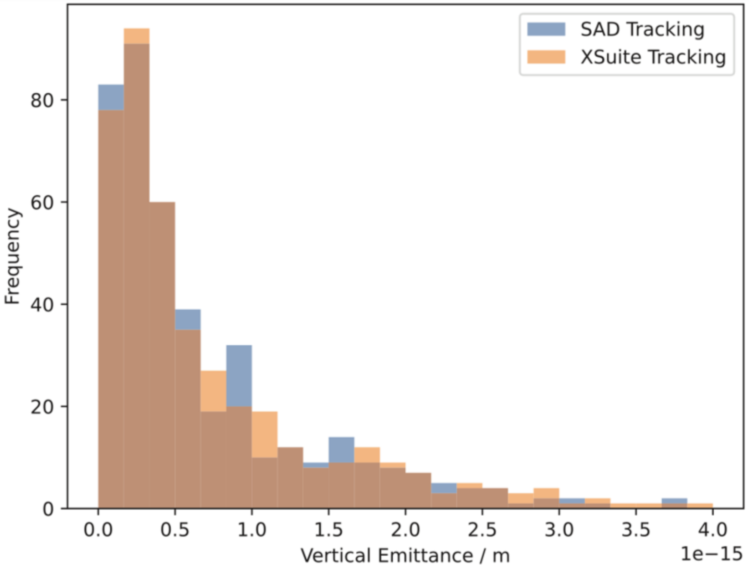
(2) *E. Forest, From tracking code to analysis: generalised Courant-Snyder theory for any accelerator model. Springer, 2016*

Synchrotron radiation

The effect of **synchrotron radiation** can be included in Xsuite tracking simulations. Two models available:

- The **"mean" model**, for which the energy loss from the radiation is applied particle by particle without accounting for quantum fluctuations;
- The **"quantum" model** for which the actual photon emission is simulated(1).

Benchmark of equilibrium emittaces from tracking (with lattice errors)



L. Van van Riesen-Haupt , T. Pieloni, et al., EPFL

---

(1) *Based on H. Burkhardt, "Monte Carlo generator for synchrotron radiation", 1990. Implementation ported from PLACET (A. Latina)*

(2) *E. Forest, From tracking code to analysis: generalised Courant-Snyder theory for any accelerator model. Springer, 2016*

Synchrotron radiation

The effect of **synchrotron radiation** can be included in Xsuite tracking simulations. Two models available:

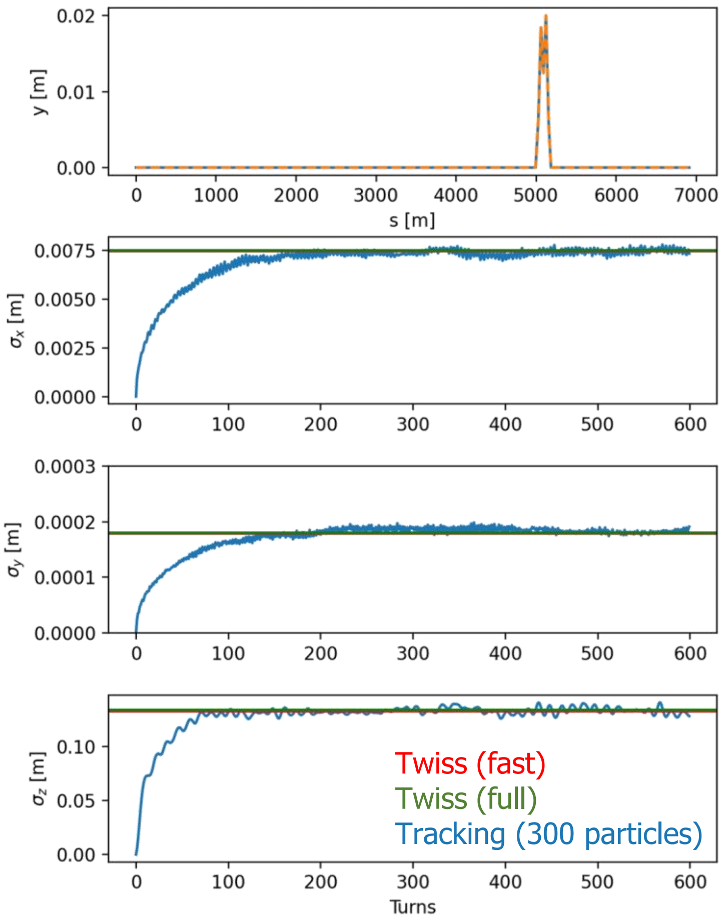
- The **“mean” model**, for which the energy loss from the radiation is applied particle by particle without accounting for quantum fluctuations;
- The **“quantum” model** for which the actual photon emission is simulated(1).

The **Xsuite Twiss** also includes:

- Dedicated algorithm for **non-symplectic one-turn map**(2)
- Computation of **radiation energy loss, damping times and equilibrium emittances**

An **automatic tool** is provided for **phasing the RF cavities** and **adjusting magnet strengths** to **compensate the radiation energy loss (“tapering”)**

Equilibrium emittance (twiss vs track)

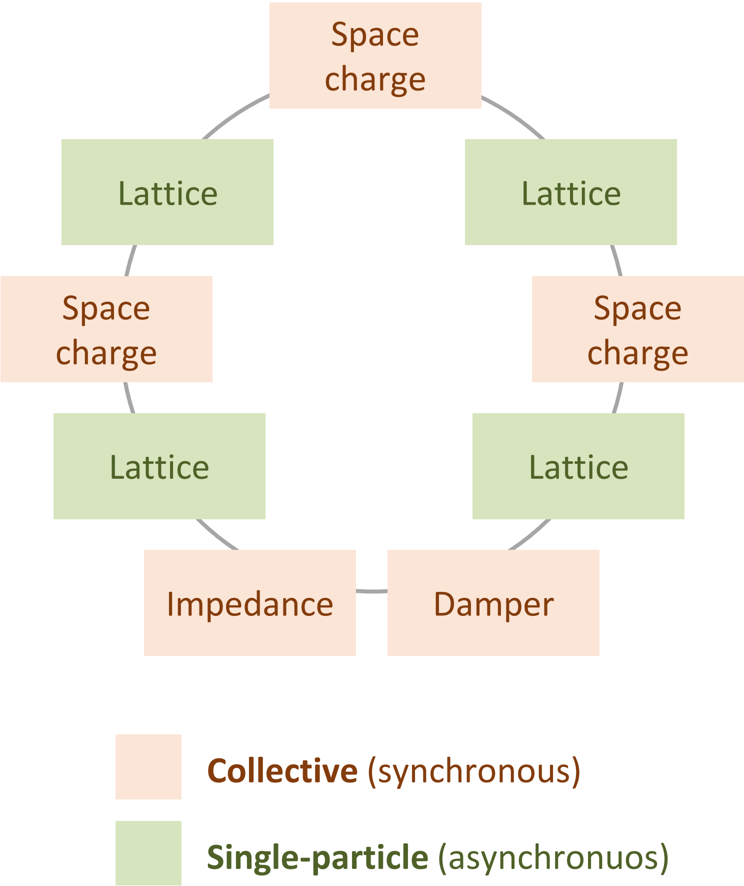


---

(1) *Based on H. Burkhardt, “Monte Carlo generator for synchrotron radiation”, 1990. Implementation ported from PLACET (A. Latina)*

(2) *E. Forest, From tracking code to analysis: generalised Courant-Snyder theory for any accelerator model. Springer, 2016*

Collective effects



Space charge

Lattice

Lattice

Space charge

Space charge

Lattice

Lattice

Impedance

Damper

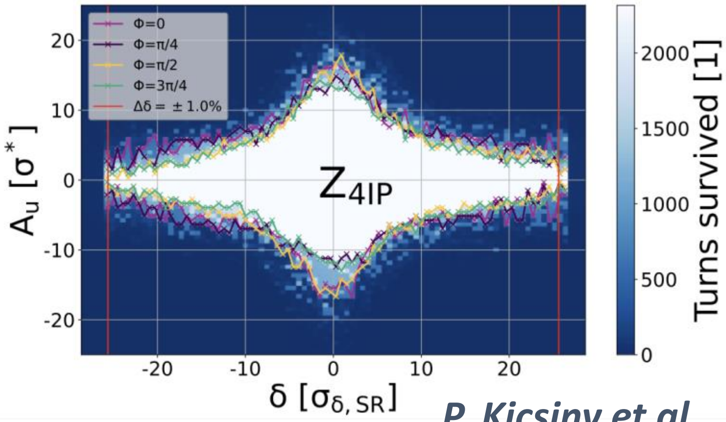
🟧 **Collective** (synchronous)

🟩 **Single-particle** (asynchronuos)

- Xsuite is designed to include **collective effects** in the simulations
- Handling of collective elements is **fully automatic** → The Xtrack module identifies the collective elements and **splits the sequence**:
  - The **non-collective** parts are handled **asynchronously** to **gain speed**
  - The simulation of the **collective effects** is **performed synchronously**
- **Space-charge**, **beam-beam**, **e-cloud** (weak-strong) are handled **natively**
- **Impedances** and **feedback systems** are handled through an interface with **PyHEADTAIL**
- An automatic tool for the **computation of stability diagrams** from amplitude detuning is also provided

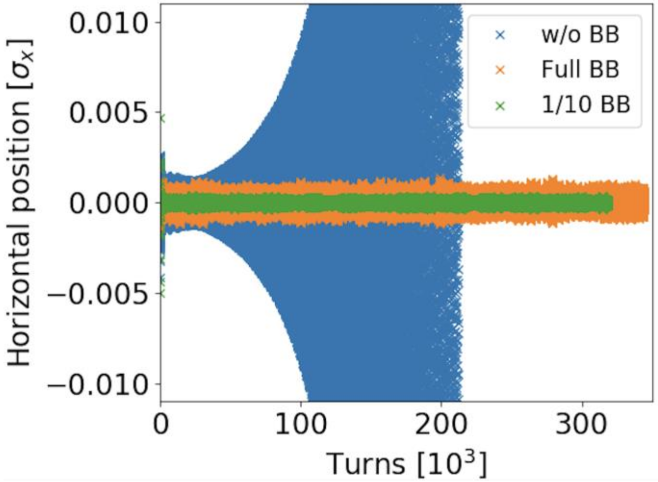
Beam beam effects

FCC-ee DA studies (with bb)



P. Kicsiny et al.

Wakefield + beam-beam simulations for HL-LHC (strong-strong modelling)

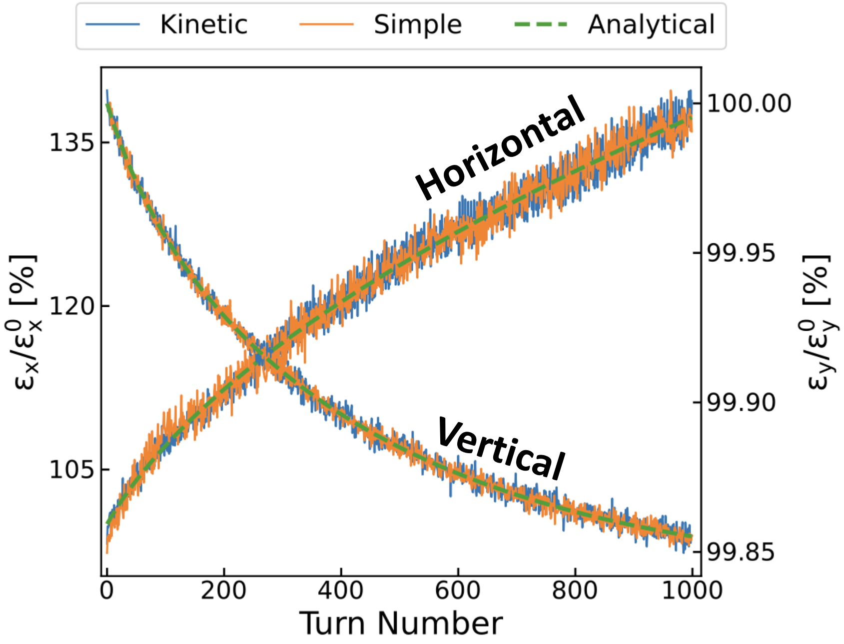


- Xsuite implementation based on experience from **Sixtrack** and **COMBI**
- Two models are provided:
  - The **“4D” model**, which applies **only transverse** forces **independent on the longitudinal motion**
  - The **“6D” model**, which applies **longitudinal and transverse** forces accounting for the synchrotron motion (method by Hirata et al.)
- Both models can be used either in **“weak-strong“ mode** (fixed assigned distribution for the other beam) or in **“strong-strong” mode** (self-consistent two-beam simulation, “soft Gaussian”)
- For the simulation of lepton colliders, the code can also simulate for **beamstrahlung** and **Bhabha scattering** (developed in collaboration with EPFL)
- **Strong-strong simulations** are accelerated by **parallel computing on HPC clusters** (based on MPI)
  - **“Pipeline” algorithm(1)** used to optimize workload distribution across the nodes

(1) *S. Furuseth and X. Buffat, Comput. Phys. Commun. 244 (2019)*
(2) For more details, see presentation by P. Kicsiny

Intra Beam Scattering

Benchmark case for the SPS ring (Pb ions)



Intra Beam Scattering (IBS) simulation capabilities have been recently introduced:

- **IBS growth rates computation** from beam parameters and optics. Two methods available:
  - Nagaitsev (very fast, vertical dispersion neglected)
  - Bjorken-Mtingwa (slower, D_y correctly accounted)
- Effect of **IBS can be included in multiparticle simulations** in combination with all other effects available in Xsuite. Two methods available:
  - Effective kick
  - Kinetic formalism

For more info: F. Soubelet et al., "Development of numerical tools for intra-beam scattering modelling", IPAC24

Outline

- **Introduction**
  - Motivation
  - Design goals and constraints
  - Architecture
  - Agile development
- **Lattice modeling, simulation and optimization**
  - Lattice modeling
  - Single-particle tracking
  - Dynamic parameter control
  - Multi-objective optimizer
- **Simulation of specific processes**
  - Particle-matter interaction
  - Synchrotron radiation
  - Collective effects
- **Final remarks**

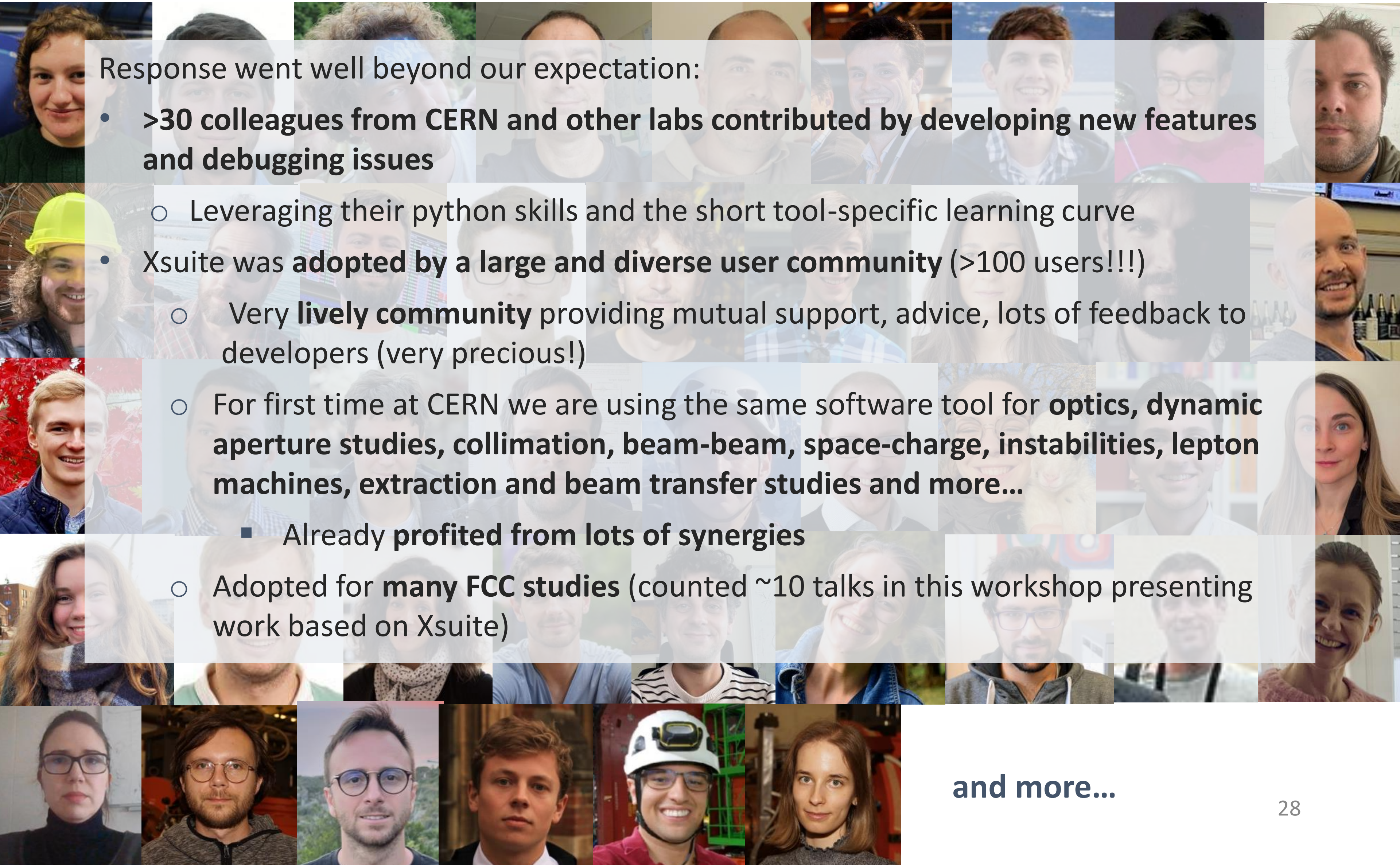
Final remarks

After three years the software has grown very rapidly, thanks **to many people contributing code and expertise…**


and more…

Final remarks

After three years the software has grown very rapidly, thanks **to many people contributing code and expertise…**



Response went well beyond our expectation:

- **>30 colleagues from CERN and other labs contributed by developing new features and debugging issues**
  - Leveraging their python skills and the short tool-specific learning curve
- Xsuite was **adopted by a large and diverse user community** (>100 users!!!)
  - Very **lively community** providing mutual support, advice, lots of feedback to developers (very precious!)
  - For first time at CERN we are using the same software tool for **optics, dynamic aperture studies, collimation, beam-beam, space-charge, instabilities, lepton machines, extraction and beam transfer studies and more…**
    - Already **profited from lots of synergies**
  - Adopted for **many FCC studies** (counted ~10 talks in this workshop presenting work based on Xsuite)

and more…

Final remarks

Xsuite simulations have been already used for **studies covering a variety of rings**:

CERN

- ELENA
- LEIR
- PSB
- PS
- SPS, TI2, TI8
- LHC
- FCC-ee, FCC-hh
- Muon collider
- LEP

GSI

- SIS-18
- SIS-100

Medical facilities

- HIT (Heidelberg)
- MEDAUSTRON
- PIMMS
- NIMMS

BNL

- RHIC
- Booster
- EIC

Fermilab

- Main injector
- Recycler
- Booster
- IOTA

Light sources and damping rings:

- PETRA
- DESY injector ring
- ELETTRA
- BESSY III
- CLIC-DR

… and more

Each of these tought us something and contributed to extend and improve the software!

Final remarks

- CERN Accelerator and Beam Physics group is **committed to maintain Xsuite and support its users for the years to come**
  - For other legacy tools (e.g. MAD-X, PyHEADTAIL) we will keep present level of functionality and support for as long as needed, while focusing resources on the development of this stack.

- **Xsuite development continues:**
  - Next items on our plate include **3D PIC for beam-beam, spin tracking, insertion devices, quadrupole fringes, and more…**

- The code is **publicly available** on GitHub and can be installed through pip
  - You are **vey welcome to give it a try** for your use case
  - **Installation instructions and many examples** available in the documentation pages
  - We are very interested in getting your **feedback** (don't hesitate to contact us for issues, questions, suggestions)

- Code is **open-source** and is open to **developments from the community**
  - Get in touch if you are interested in contributing to the development

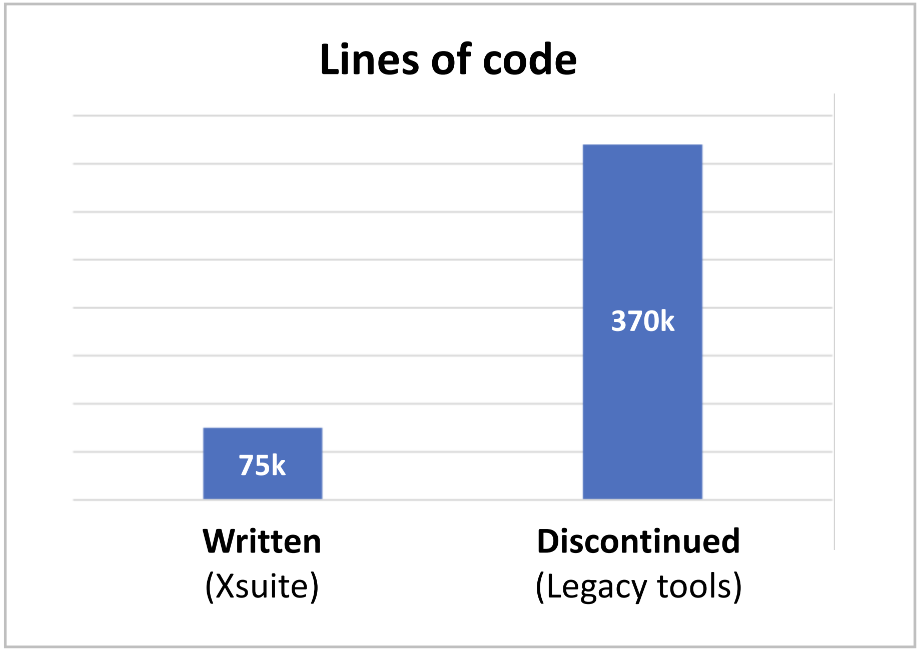
Thanks for your attention!

Discontinuation of legacy tools

In 2022-23 we have **essentially discontinued** the development and, to a very large extent, the usage of the following tools:

- Sixtrack
- Sixdesk
- Sixtracklib
- COMBI
- PySSD
- DistLib

This led to a **massive simplification** of our code base.



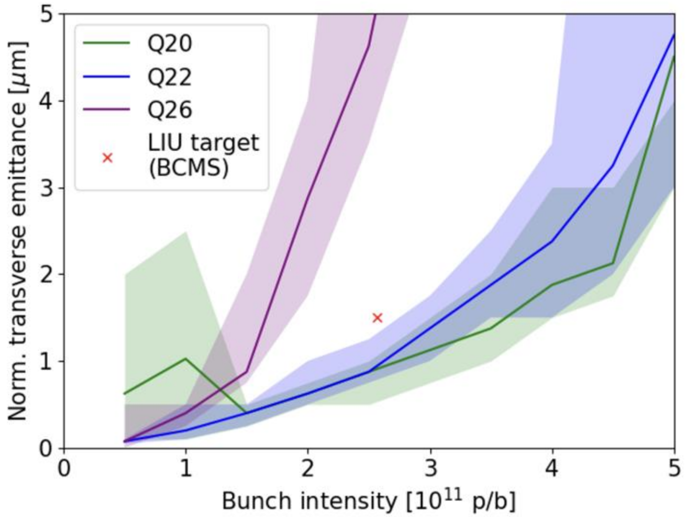
Space charge

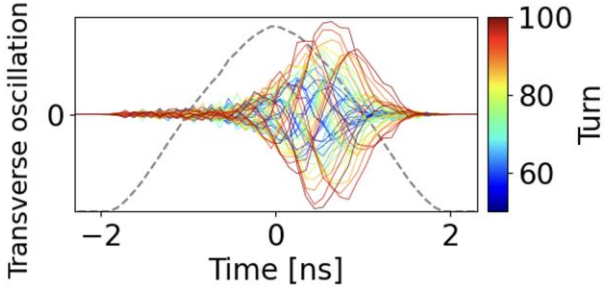
The implementation is largely based on **PyHEADTAIL-PyPIC**

Different **space-charge models** are implemented:

- The **“frozen” model**, in which particles interact with fixed charge distributions
- The **“quasi frozen”** model, in which the **beam intensity and beam sizes are recomputed at each interaction**
- The **“Particle In Cell (PIC)”** model:
    - Charge of tracked particles distributed on a **rectangular grid**
    - **Fast Poisson solver** based on **FFT** method with Integrated Green Functions
- Space charge simulations **strongly profiting from GPU acceleration**

Simulation campaign for the CERN SPS including full non-linear lattice, space charge and wakefields





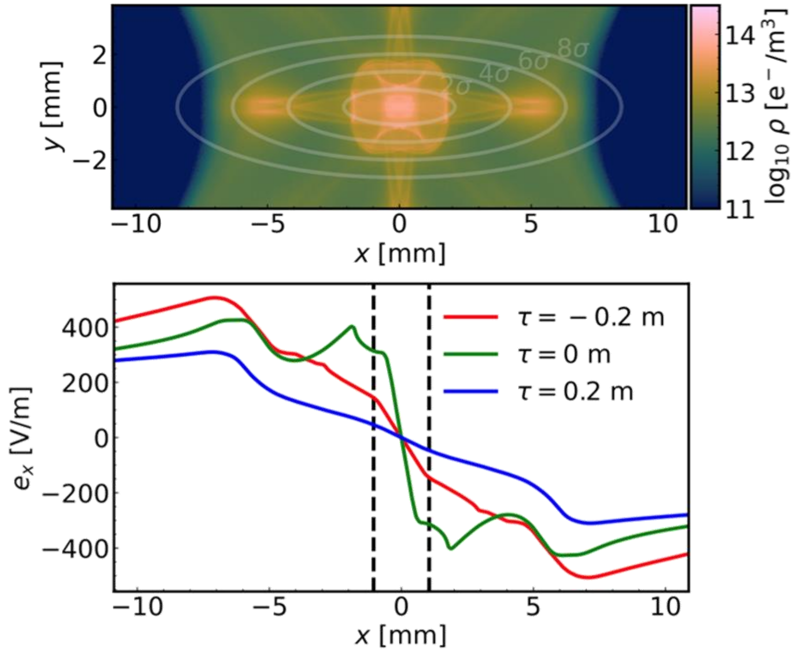
N. simulations	400
Number of PIC calculations per turn	540
Number of turns per simulation	40’000
Computing time per sim. (GPU)	~3 days
Computing time per sim. (CPU serial)	> 12 months

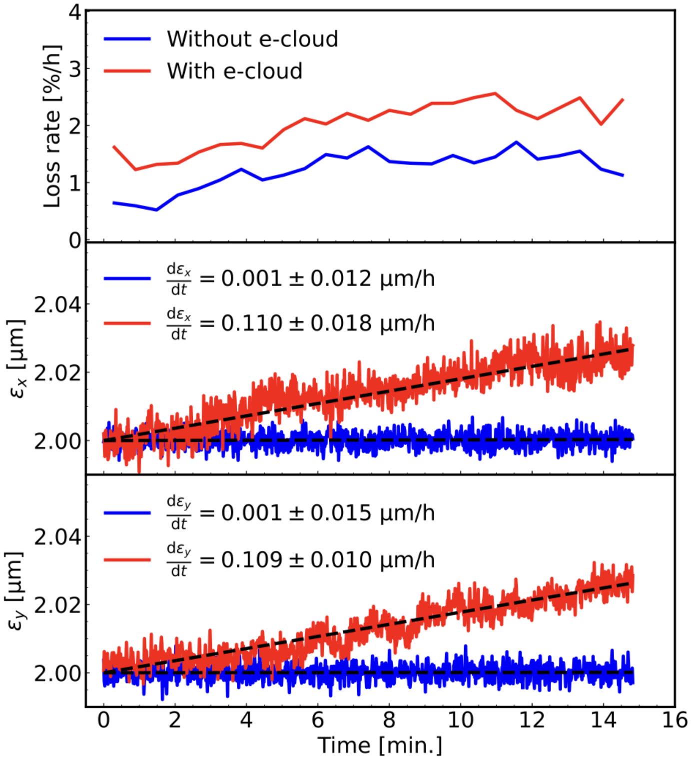
Courtesy X. Buffat

Electron cloud modeling

Xsuite has been exploited to study the **effect of electron cloud** on **slow beam degradation** (emittance growth, lifetime degradation).

- Done by applying a **high-order interpolation scheme** to the **e-cloud potential imported** from a dedicated multipacting simulator.
    - Scheme designed to **preserve the symplecticity of the resulting map** by ensuring the global continuity of the potential and required derivatives.
- **Use of GPUs is mandatory** to simulate the required long time scales (>10^6 turns).



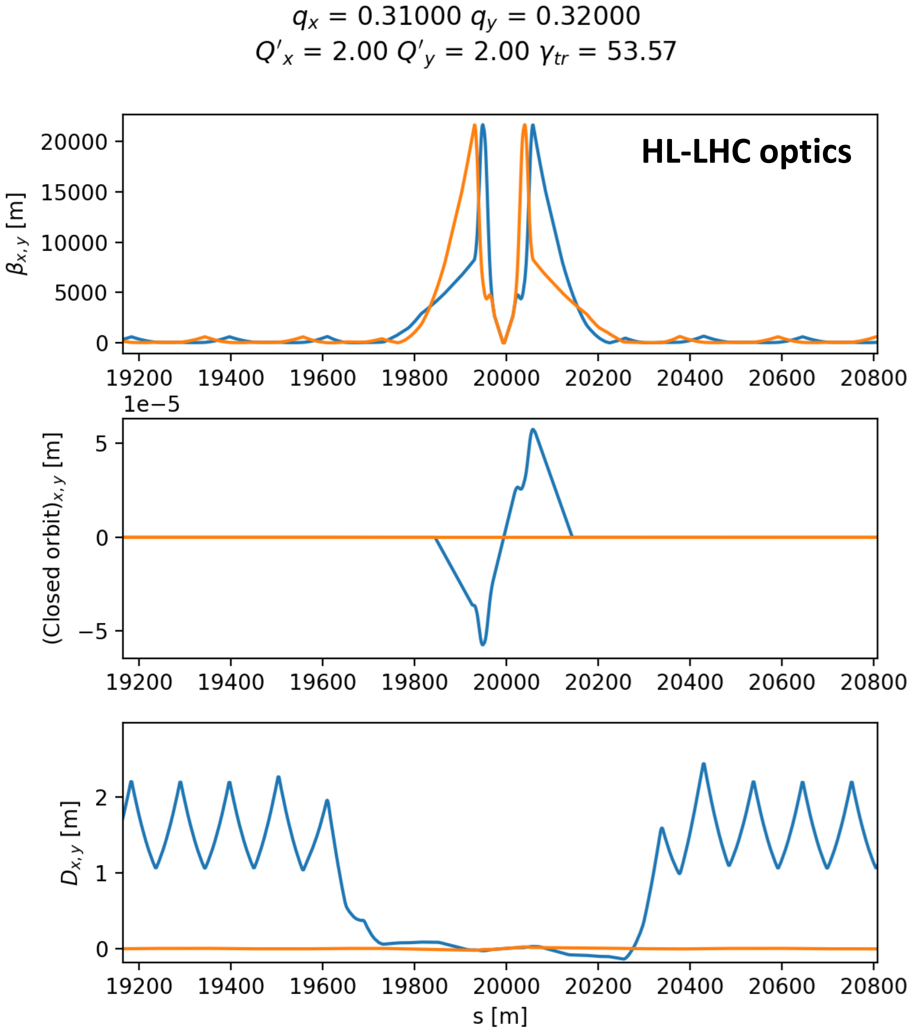


See also: K. Paraschou, THBP16

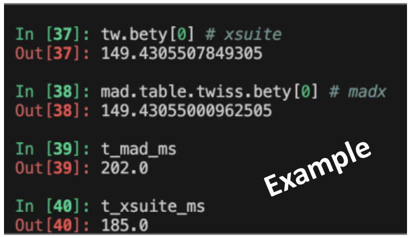
Twiss module

Computation of **Twiss parameters based on the tracking** has **two main advantages**:

- Any **physical model included in the tracking** is **automatically usable in Twiss**
  - Without additional development effort
- Twiss becomes a **powerful diagnostics tool** on the built **tracking model**
  - Allows **measuring directly on the tracking model** tunes, chromaticities, closed orbit, beta functions, etc.
  - Can be done **effortlessly** and **without exporting or manipulating the model**.
  - **Used daily** to for validating simulation models, catching mistakes, investigating issues



Accuracy compared to MAD-X: Δβ / β << 10^{-4}
Computation time is very similar



```
In [37]: tw.bety[0] # xsuite
Out[37]: 149.4305507849305

In [38]: mad.table.twiss.bety[0] # madx
Out[38]: 149.43055000962505

In [39]: t_mad_ms
Out[39]: 202.0

In [40]: t_xsuite_ms
Out[40]: 185.0
```