Xcoll
Xboinc
Xdyna

Frederik Van der Veken

ABP Technical Meeting on Large Simulation Codes, 19/10/2023

Xcoll

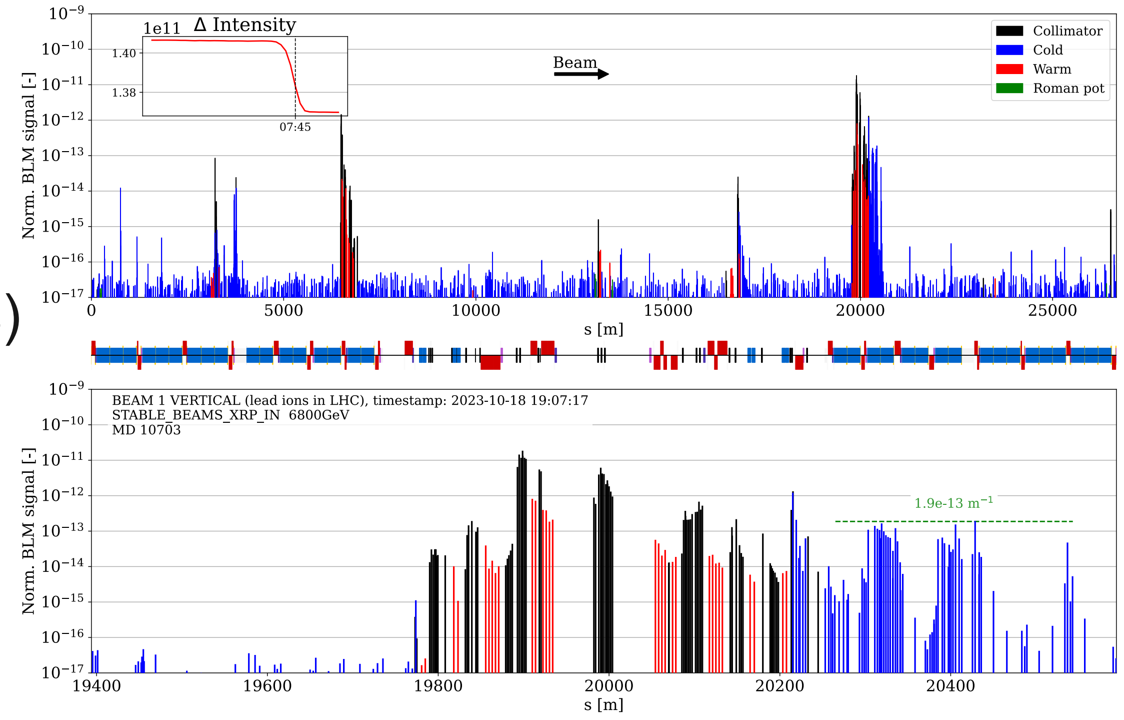
Everything you need for your collimation simulations in Xsuite

Design Philosophy

- **Standardisation:**
  - common approach for easy comparison between different simulation setups and measurement
  - e.g. direct integration with recent lossmaps tool

- **Flexibility:**
  - user-friendly modularity stimulates autonomy (not dependent on developers for small changes)
  - while guaranteeing robustness and reliability

- **Maintability:**
  - code readability and documentation is a must to ensure future-proofing code development
  - robust and encompassing test suite



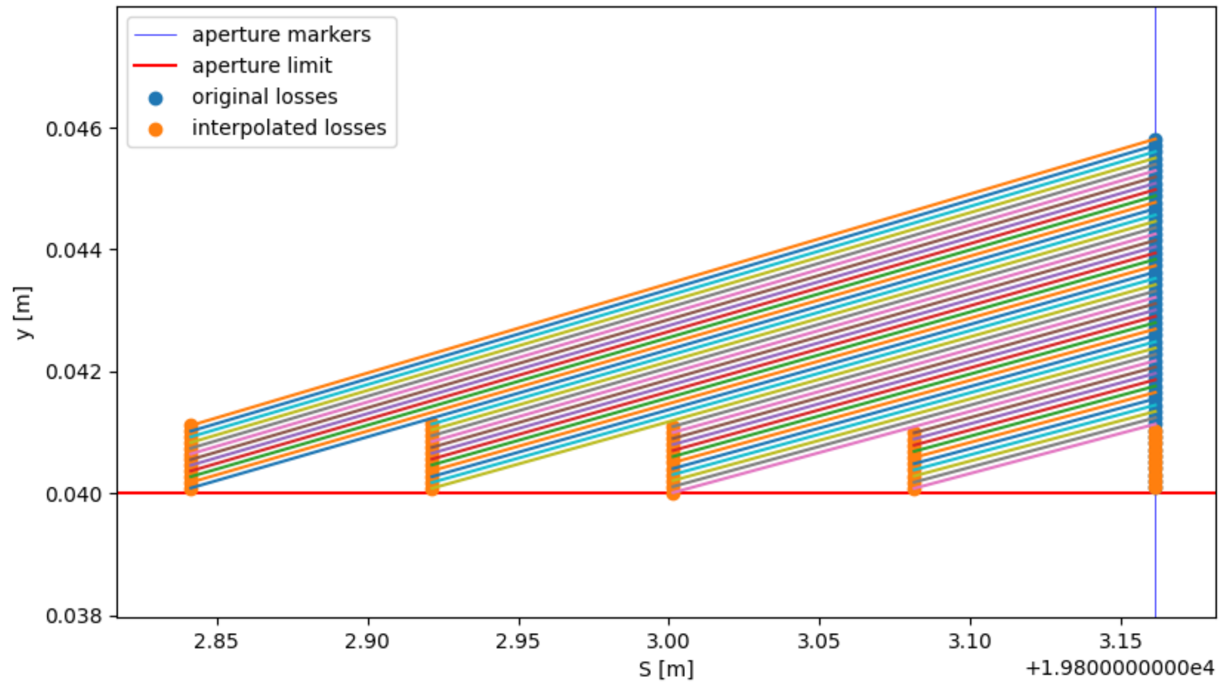
Features

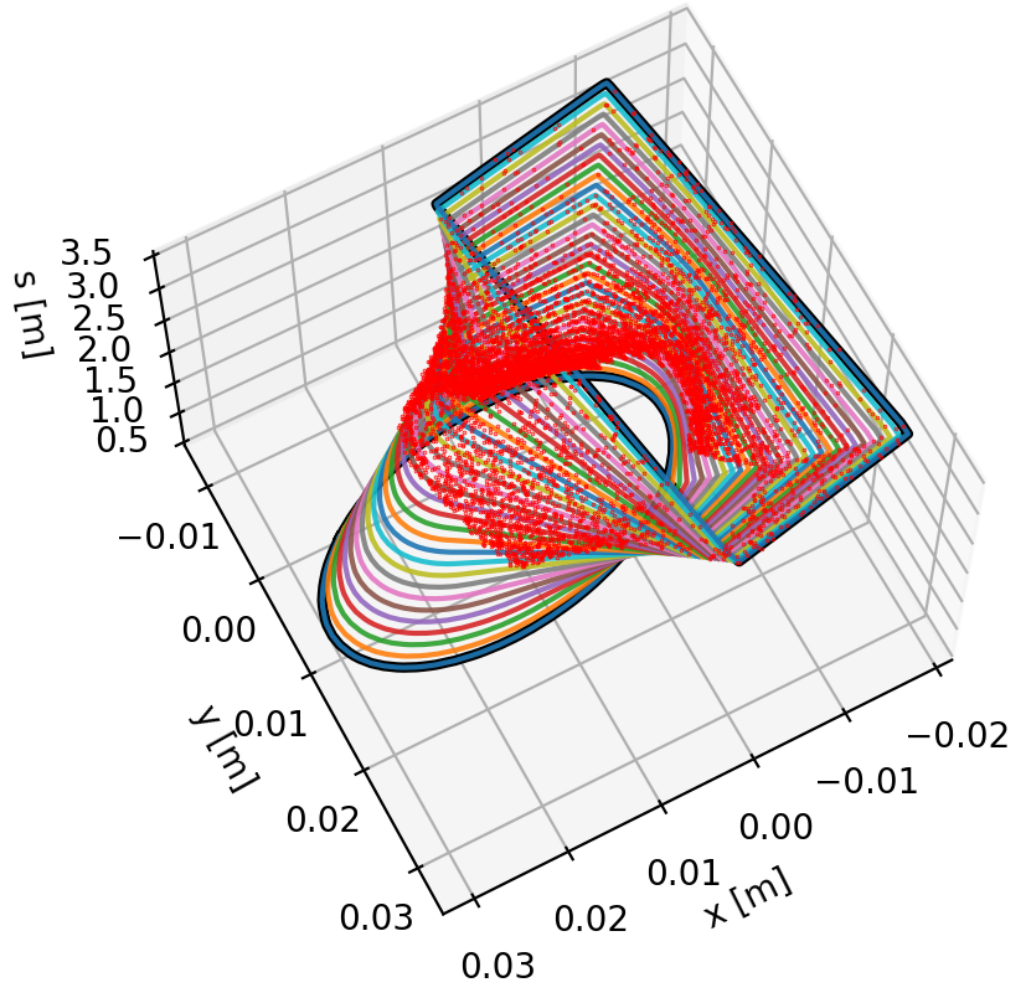
- **Collimator installation:** special care for consistency (cryo tank, entrance markers, aperture, ...)
- **Collimator settings management:** CollimatorDatabase (YAML) <=> BeamElement
- **Initial distributions** (Xpart)**:** annular halo, pencil beam (betatron and/or off-momentum), ...
- **Scattering Engines:** Everest (native), FLUKA (via FlukaIO), Geant4 (via collimasim+BDSIM)
- **Aperture interpolation** (Xtrack): consistency of aperture model along the lattice
- **ImpactTable:** precise logging of different scattering interactions
- **Loss Map:** longitudinal histogram of losses along the lattice
- **RF Sweep:** realistic modelling of off-momentum loss-maps

G.Iadarola

Aperture Interpolation

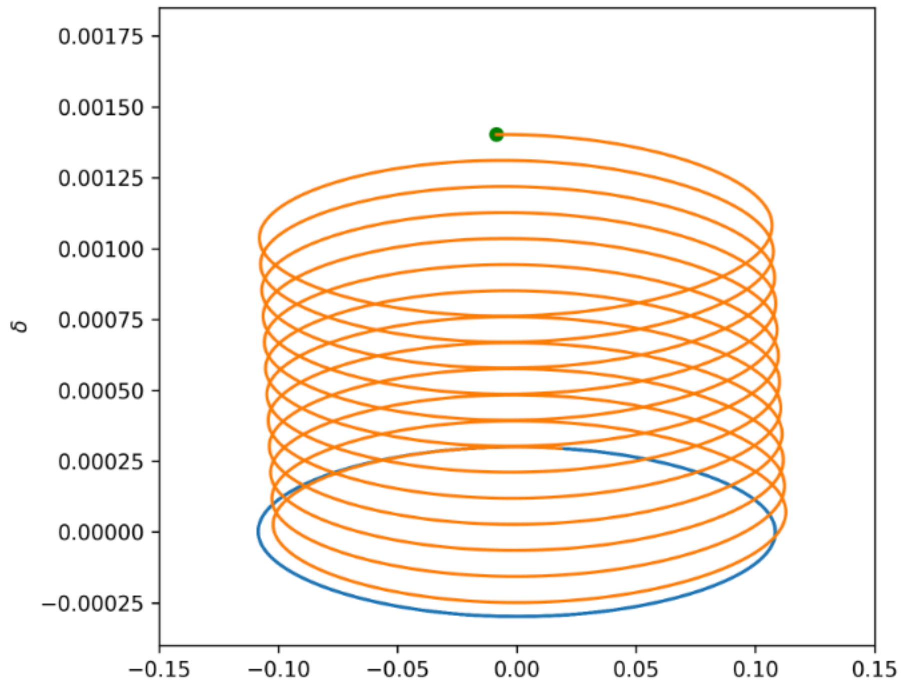
- Aperture markers can be limited to locations where aperture changes (for CPU efficiency)
- But need precise arbitrary resolution => **backtrack**
- **Interpolation** between different aperture types

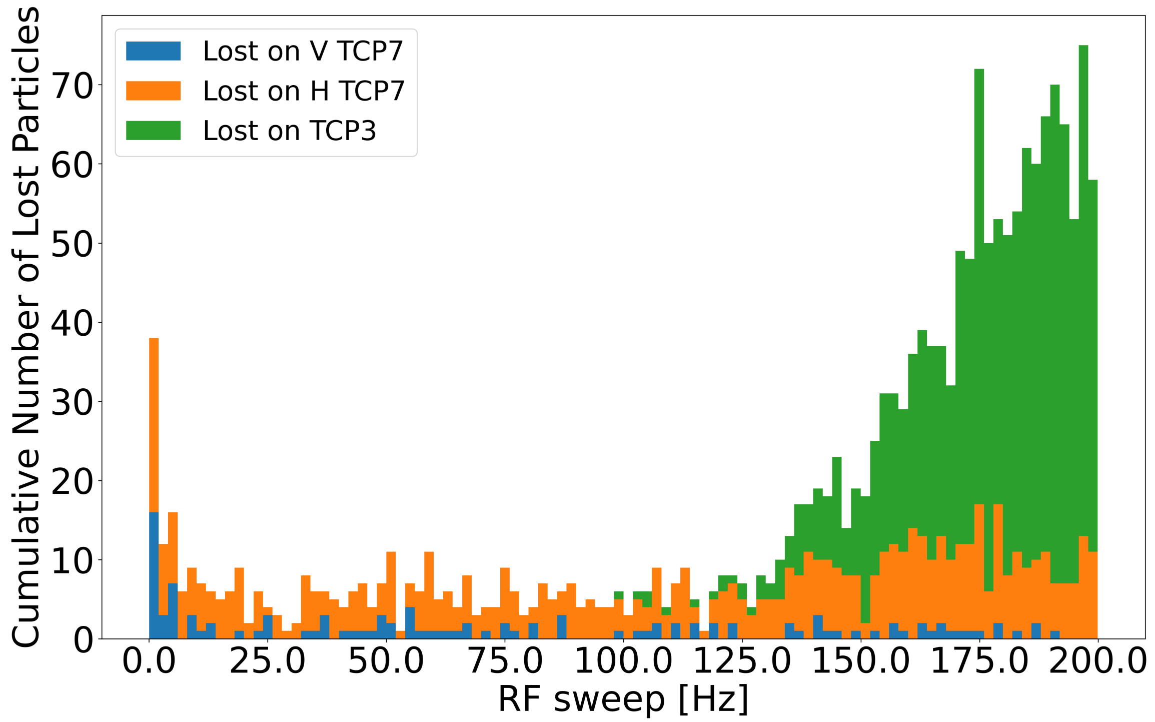




RF Sweep

- Sweeping frequency of RF cavities moves separatrix up/down along delta
- In simulations, particles are assumed synchronous to longitudinal reference trajectory
  => need to shorten/lengthen trajectory to simulate sweep
- See https://www.overleaf.com/read/kcgwmgwrfwhw for details

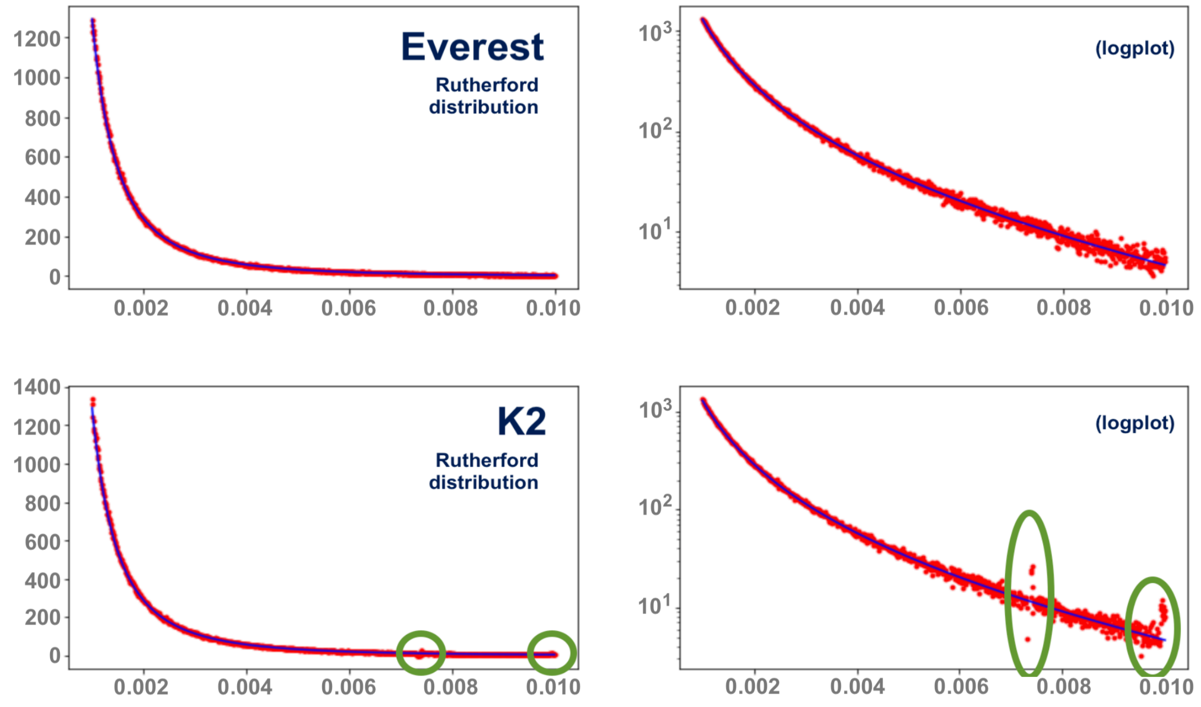




D. Demetriadou, FFVdV

Everest

- **Native C** implementation (translated from FORTRAN)
- **Speed gain** of factor ~6 compared to K2 (single CPU)
- Low-level control exposed to user (e.g. tilts) **(NEW)**
- Rutherford random generator improved
- Code **readability** and logic flow improved
- **OpenMP**-compatible **(NEW)**
- Impact table improved
- Small updates/improvements to physics, like MCS (WIP)



Example application:

N. Triantafyllou, HB'23

Betatron cleaning during Ramp

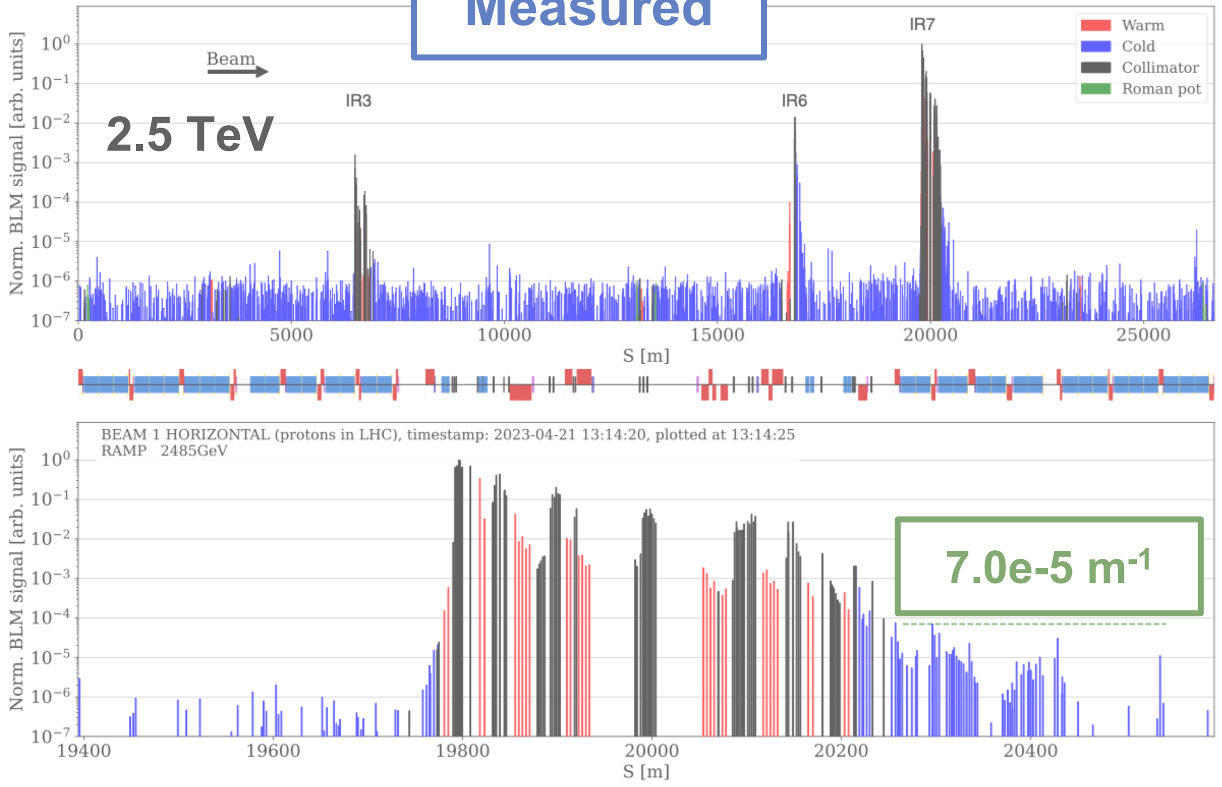
- **Good qualitative agreement** between measurements and simulations
  - Highest losses in **IR7**: similar **loss pattern**

Cleaning inefficiency simulations

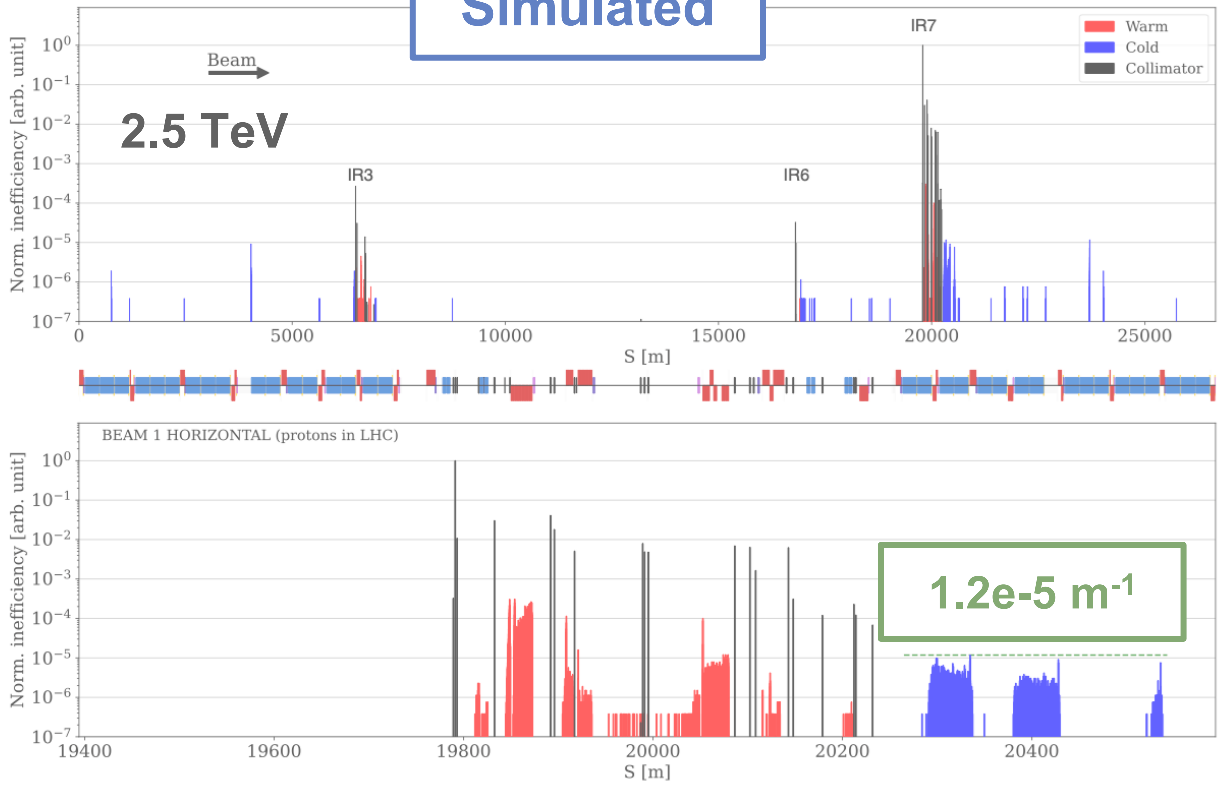
$$\eta = \frac{N_{\text{loc}}}{N_{\text{tot}}\Delta s}$$

where N_{loc} the local losses over distance Δs and N_{tot} is the total number of losses in the collimation system

Measured



Simulated



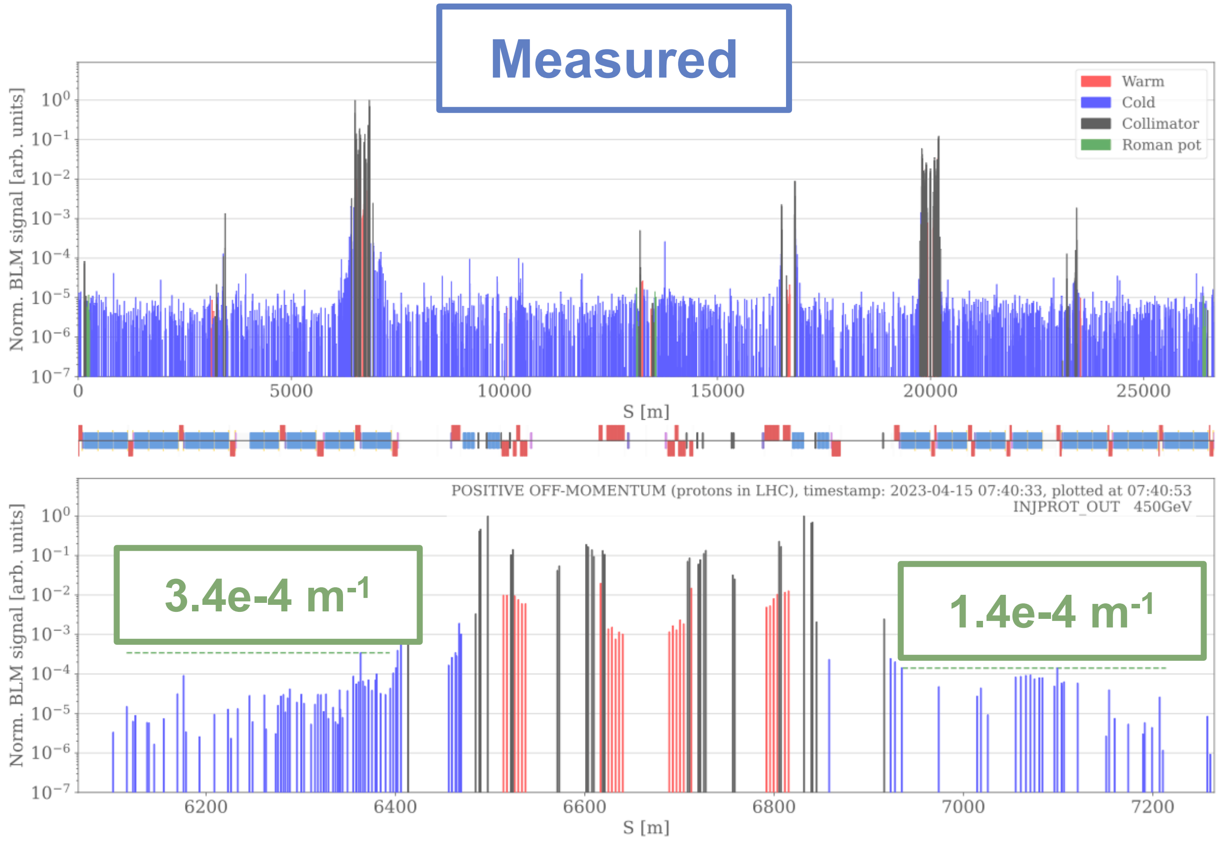
Example application:

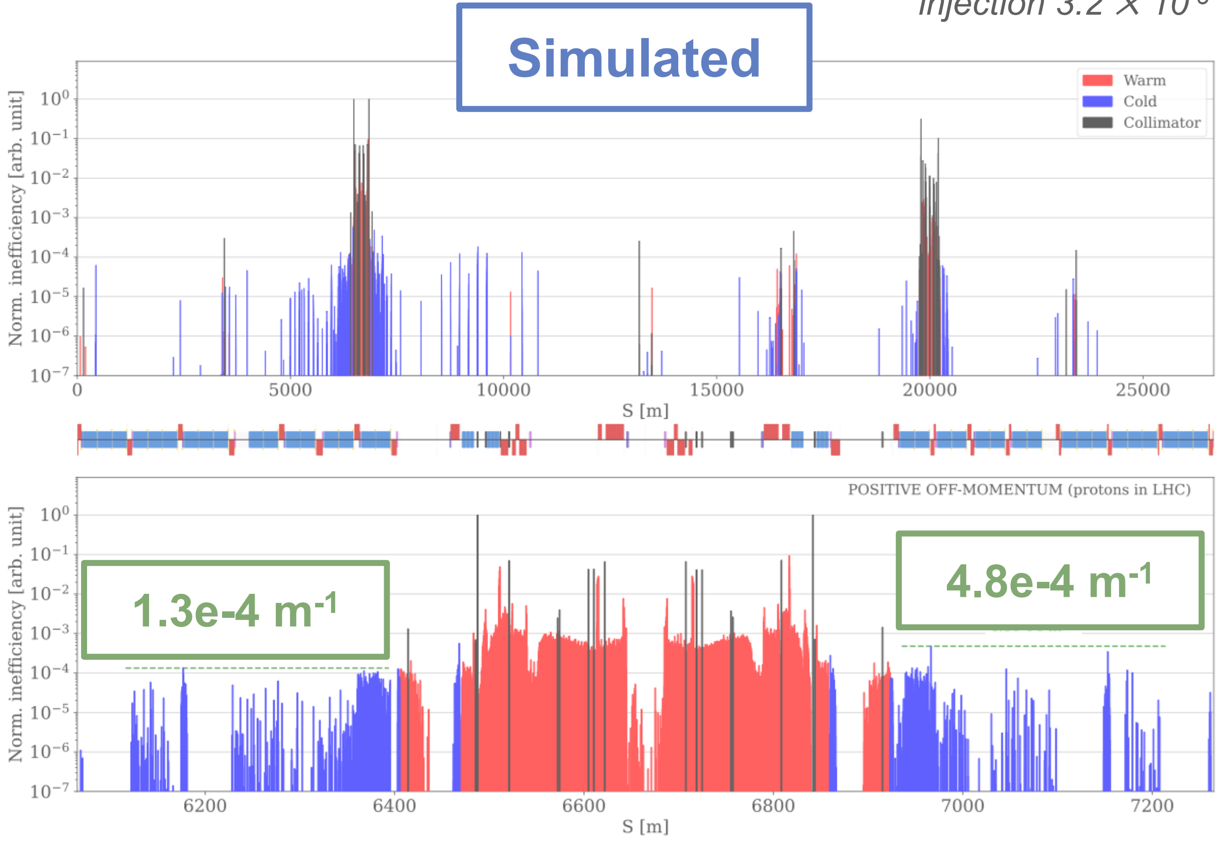
N. Triantafyllou, HB'23

Off-momentum cleaning

- Example **positive off-momentum** loss maps at **injection** energy, **RF sweep -200 Hz**

Off-momentum cut at injection 3.2 × 10^{-3}

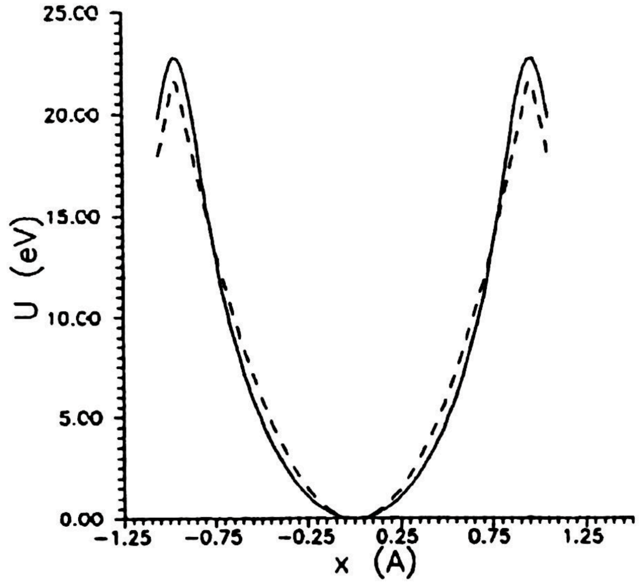




M. D'Andrea, FFVdV

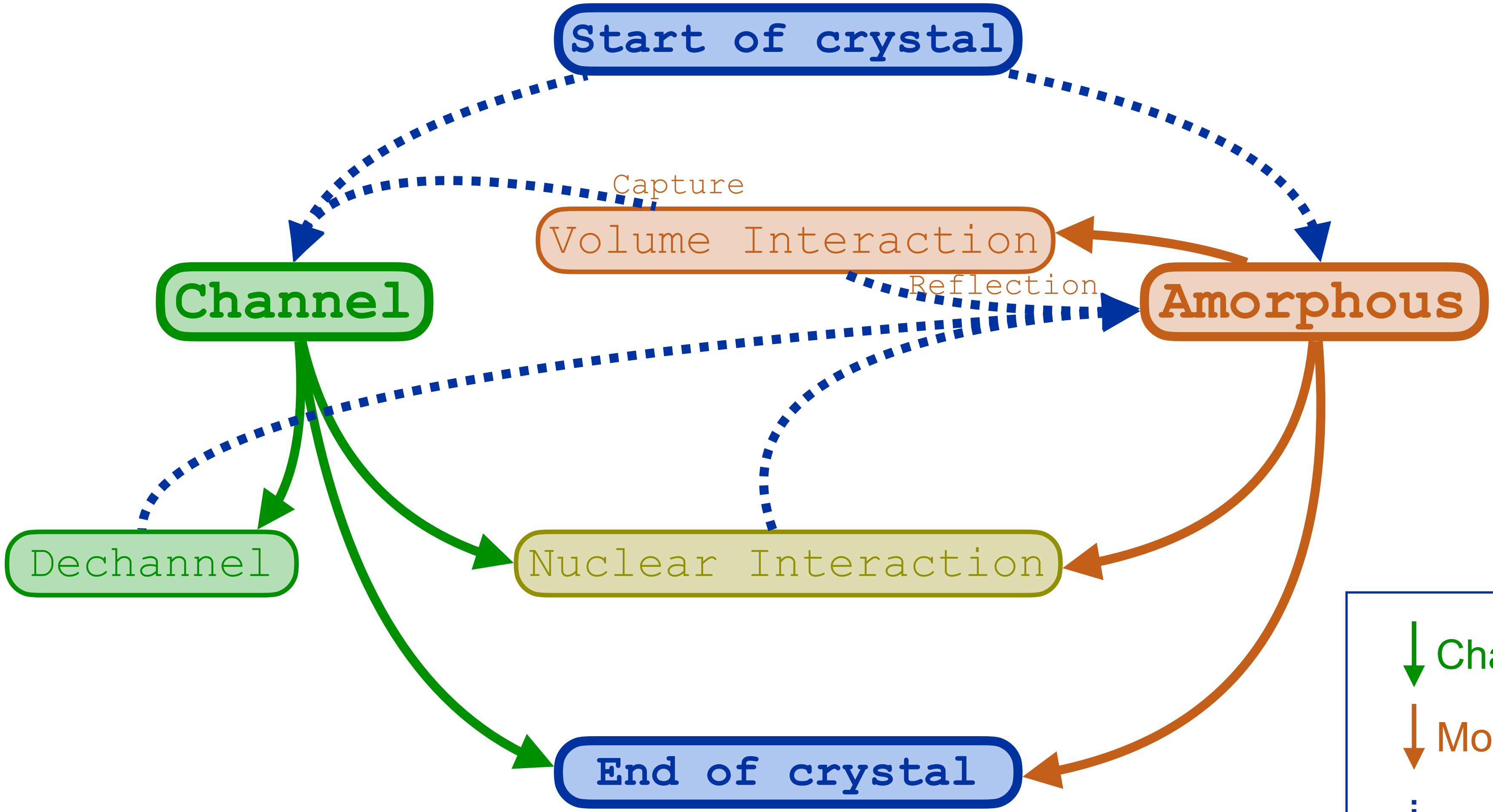
Everest Crystals

- **Complete revision** of crystal routine (both programming logic and physics)
- Needed to simulate **long crystals** (previously only single interaction possible)
- Replaced copy-pasted code with new logic
- **Low-level fixes** on physics**:**
  - Computation of Volume Interaction (two errors that kind of cancelled each other)
  - Bending angle was over-approximated (was still within acuraccy)
  - Saturation factor of channeling probability was compensating typo
  - Hard-coded material parameters are removed
- **Work in progress:**
  - Questionable need of transition regions
  - Crititial angle calculation



M. D'Andrea, FFVdV

Everest Crystals: New Logic Flow



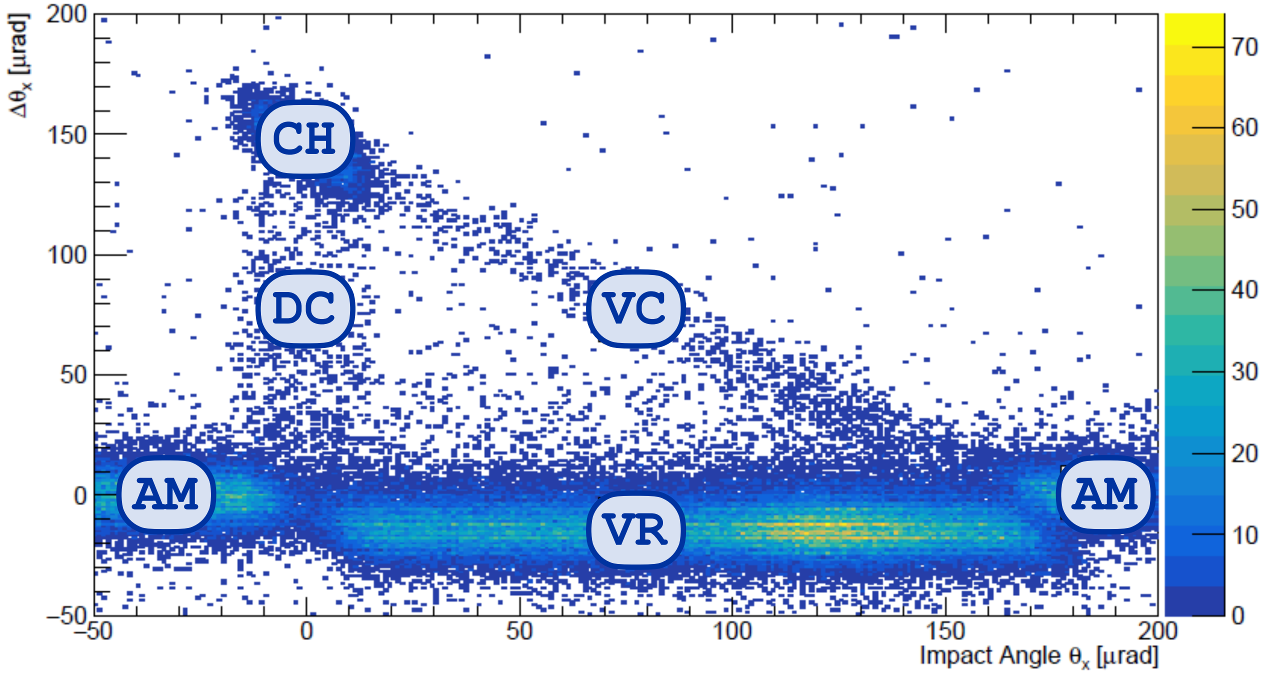
Channel over distance until ...

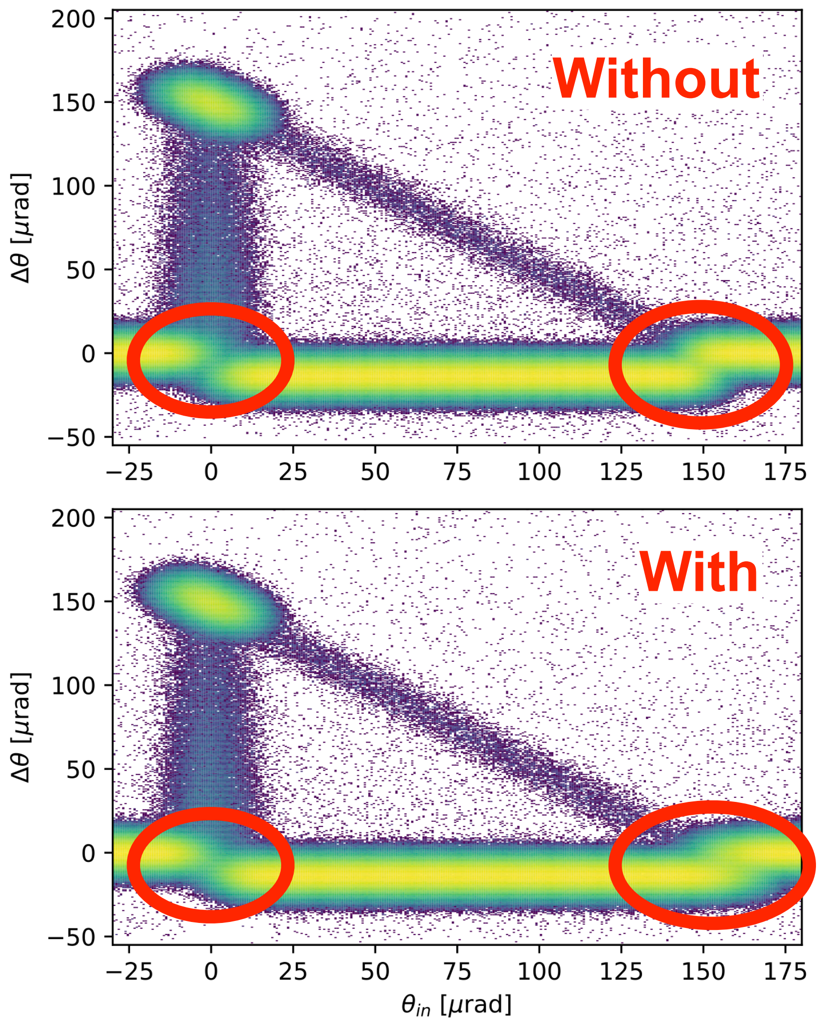
Move amorphous over distance until ...

Next step in program flow ...

M. D'Andrea, FFVdV

Everest Crystals: Transition Regions

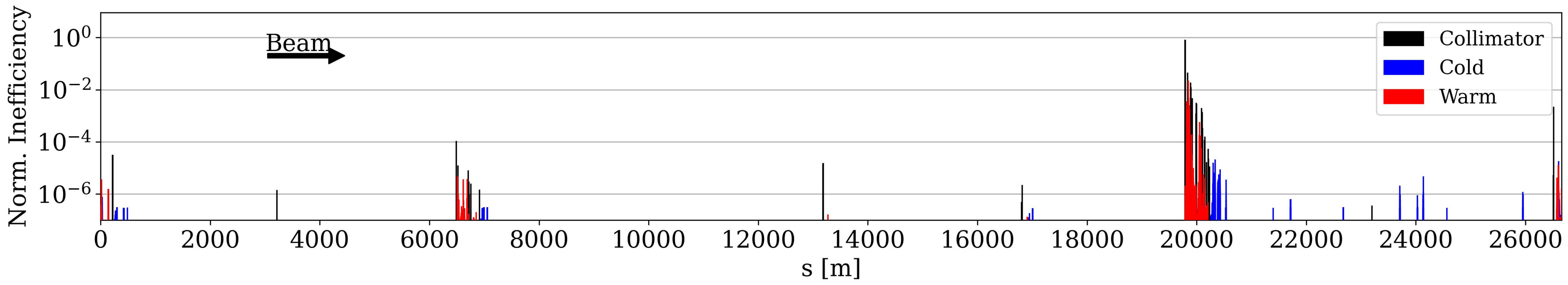




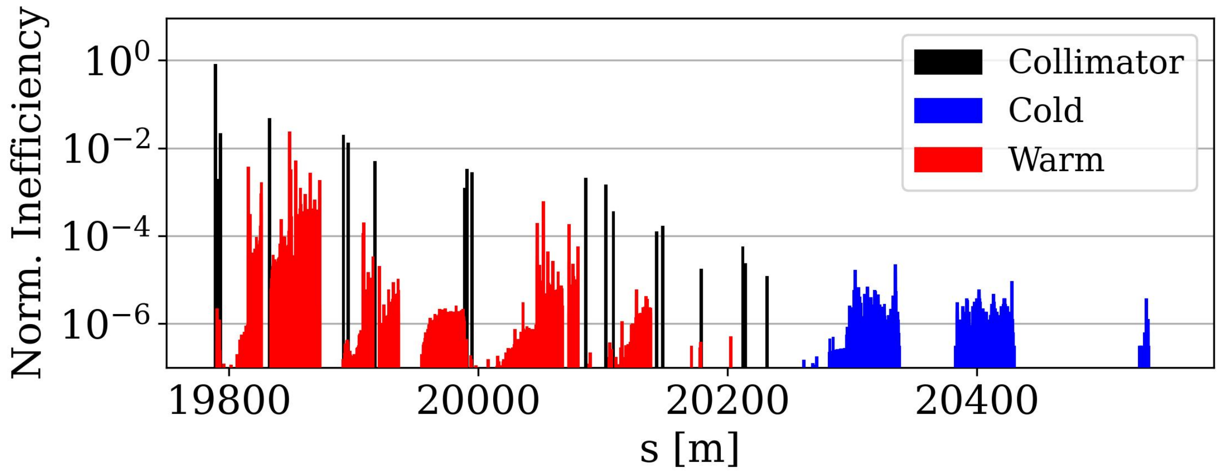
- **Are transition regions realistic?**
- **Physical motivation seems weak** (geometrical constraints)
- **In-depth benchmark coming up** (all 3 codes + new measurements)

FLUKA

G. Hugo, FFVdV



- **Large amount of progress made:** first working prototype, **fully integrated into Xcoll API!**
- Still need to manually create FLUKA input files (*.inp, using SixTrack-tuned code)
- **Everything else managed in Python:**
  FLUKA, flukaserver, FlukaIO, network, ...
- Many layers:
  Python => SixTrack (FORTRAN)
  => FlukaIO (FORTRAN)
  => FlukaIO (C)
  => FLUKA



FLUKA

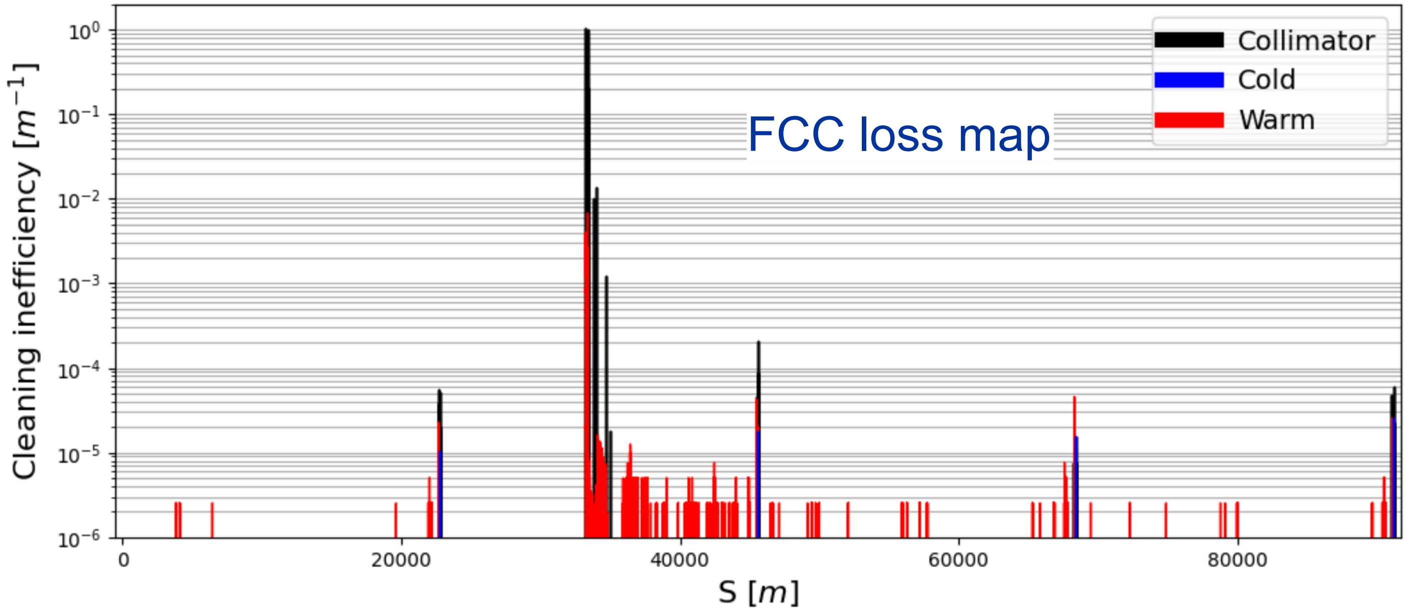
G. Hugo, FFVdV

- **Work in Progress:**
  - File generation based on **xt.Line** and **xc.CollimatorDatabase** in order to remove SixTrack dependency
  - Reduce programming layers: Python => xobjects C API => FlukaIO (C) => FLUKA
  - Automatic FlukaCollimator generation for standard studies
  - Retrieve more impact information (interaction type etc) from FLUKA
  - Test and benchmark time efficiency for optimal implementation

Geant4

A. Abramov, G. Ricci, FFVdV

- **Working implementation of Geant4 coupling:**
  BeamInteraction => collimasim => BDSIM => Geant4
- First steps made towards integration in Xcoll API (to leverage full functionality)
- Used in full production for FCC studies



- **Work in Progress:**
  - No need to use BeamInteraction
  - Reduce programming layers: Python => xobjects C API => BDSIM => Geant4

Xcoll Summary

- Very active development, several new features, more are planned
- **Quickly becoming new standard for collimation team:**
    - **Everest:** default for proton - matter interactions (K2 discontinued)
    - **Geant4:** only used with xsuite (SixTrack+Geant4 discontinued)
    - **FLUKA:** first prototype, SixTrack+FLUKA still in active use

Xcoll Summary

- **Future outlook:**
  - Detailed **manual**, including underlying physics (in progress)
  - Further remove dependency on colldb: allow easy collimator installations
  - **Everest:** detailed crystal comparison, jaw flatness models, and GPU-compatibility
  - **Geant4:** complete integration into xcoll
  - **FLUKA:** translate to C, and add user-friendly API without files

Xboinc

Facilitating large-scale volunteer computing

Development supported by CHART

Xtrack on BOINC

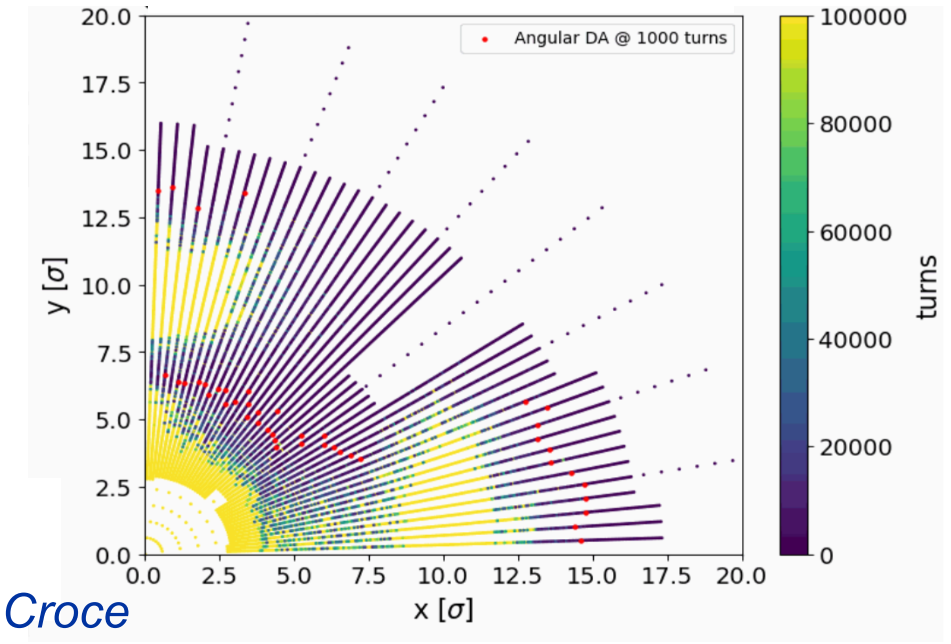
- **Similar approach as SixTrack:**
  - Single light-weight application (pre-compiled stand-alone executable)
  - Custom environment (submission, assimilator, validator)

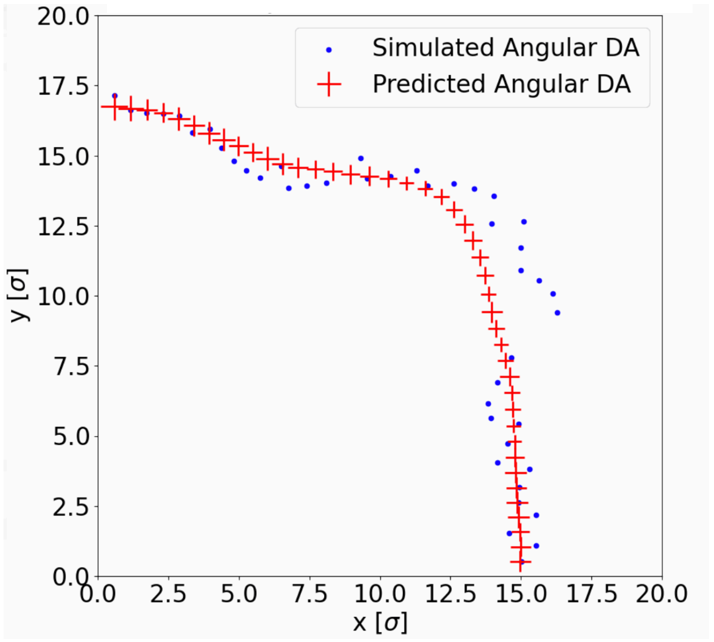
- **Current status and outlook:**
  - Executables for Windows (64 bit), Mac, and Linux on test server (lhcathome-dev.cern.ch)
  - Custom server environment nearing completion
  - Not limited to one type of study (though only compileable parts of toolkit)
  - **CPU at a first stage** (investigate potential for GPUs later)

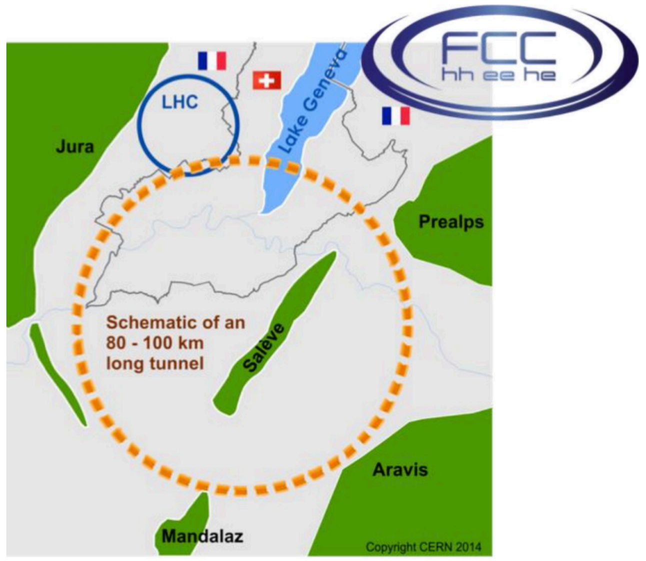
Development supported by CHART

Xtrack on BOINC: Example study

- **Currently, most people moved to Xtrack**
  (explaining the drop in SixTrack jobs submitted to BOINC)
- The tailoring of SixTrack BOINC to one type of study **limited the number of users**
- Very large-volume study in the pipeline: surrogate ML model for LHC and FCC in collaboration with EPFL, SDSC (under the CHART project)







Study supported by CHART

D. Di Croce

Xboinc

D. Di Croce, FFVdV

- To avoid compatibility issues, Xboinc version **freezes** all Xsuite packages' versions
- Generating new **executables** for CPU is now straightforward (with collimation included)
- Logic flow:
  - User registration on server ✓
  - User submits jobs using xboinc.SubmitJobs ✓ (on user EOS or AFS workspace)
  - Xboinc-server fetches jobs from user workspace ✓
  - Xboinc-server submits to BOINC ✓
  - Xboinc-server retrieves results ✓ and organises them by user and type **WIP**
  - User reads results using xboinc.retrieve **WIP**
- Careful development (thorough testing and documenting) of Xboinc-server because of **scalability**

Xboinc Outlook

D. Di Croce, FFVdV

- **Current users (SixTrack): ~1** (will move to xboinc soon)
- **In development:**
  - Finish automatic retrieval and return of jobs
  - Address credit issue by smart sampling of initial conditions
  - Add enconding to protect against cheaters (currently only in conceptual phase)

- **We need users!** (not enough jobs - volunteers are eager to crunch)
  - Not only DA studies, but any non-collective (compilable to C) study

Xdyna

Dynamic aperture is fun

Features

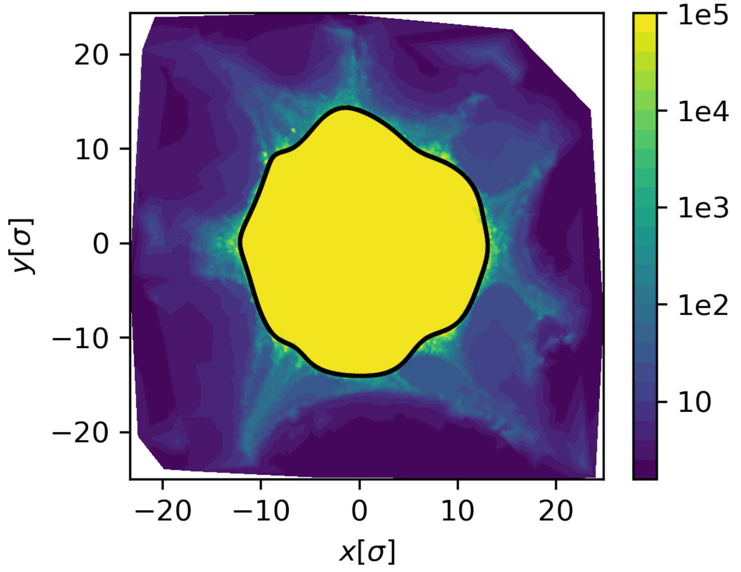
M. Hofer, T. Pugnat, FFVdV

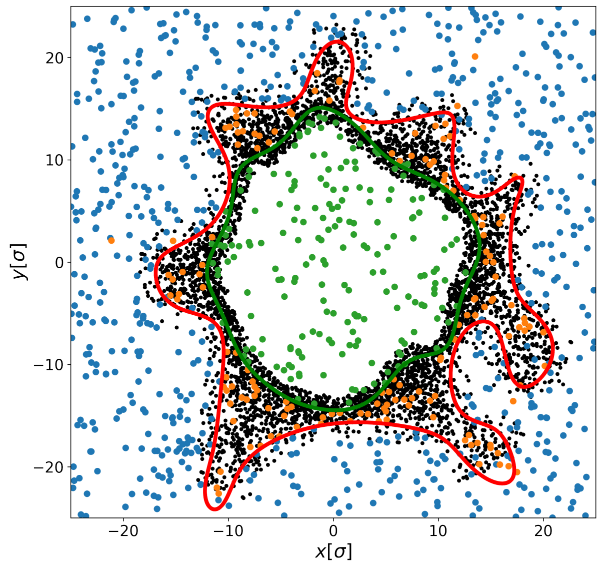
- Xdyna is the **Xsuite successor of SixDesk**, expanding its functionality
- Initial conditions can be radial, on a grid, or random (with resampling)
- In active development, with several new DA border detection methods ("classical" and ML)
- An improved **fitting algorithm** to describe the **evolution of DA** is work in progress
- A lot of progress, on using **dynamic indicators** to describe chaos, is being merged

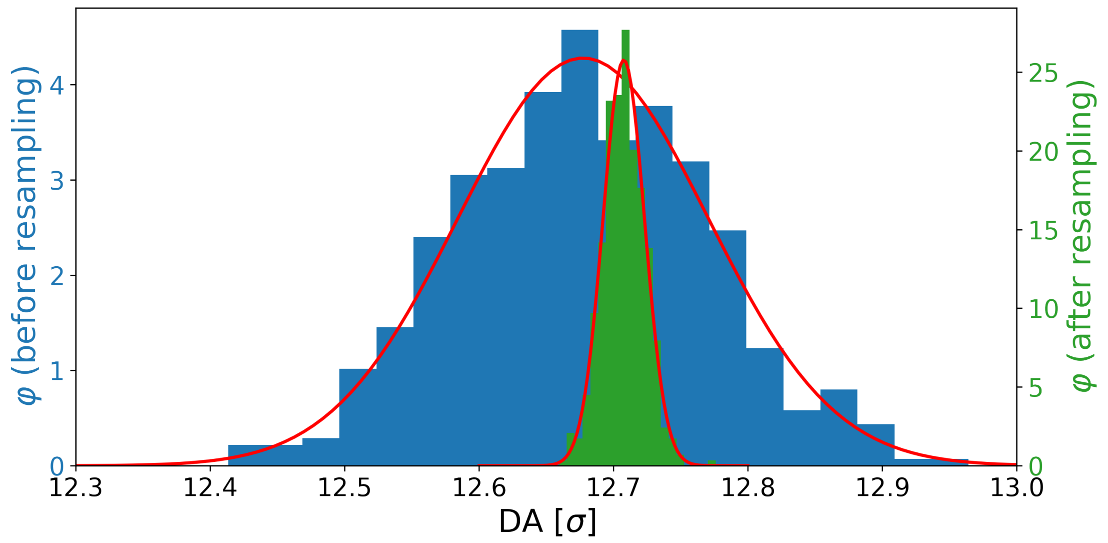
- **Small team of users, feel free to join**

ML Border Detection

- **The border of stability in phase space exhibits chaotic behaviour**
- High resolution is desirable for more accurate evolution (and hence extrapolation) but not known in advance where to sample more initial conditions
- Xdyna provides ML Border models to iteratively recognise the region of stable particles



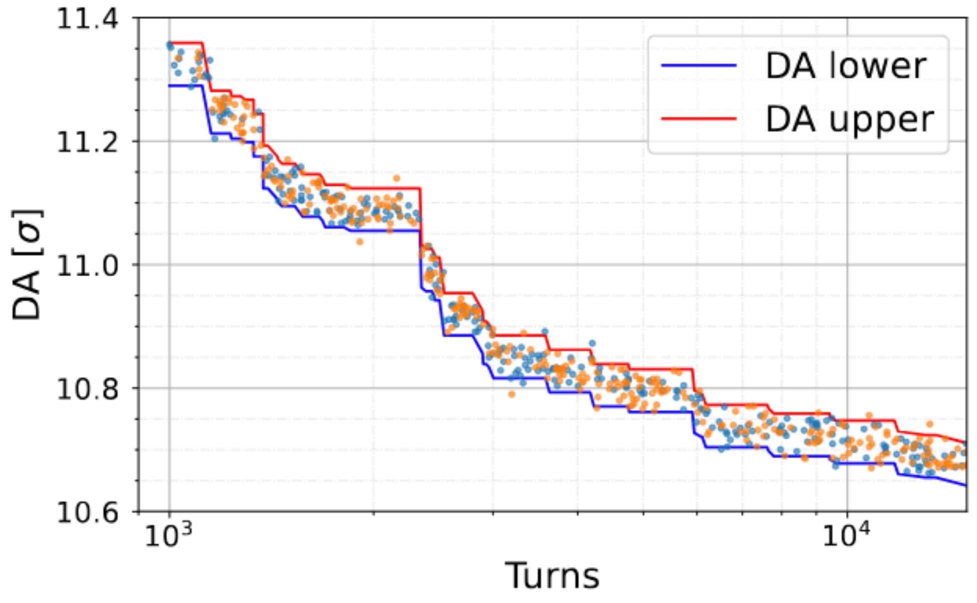


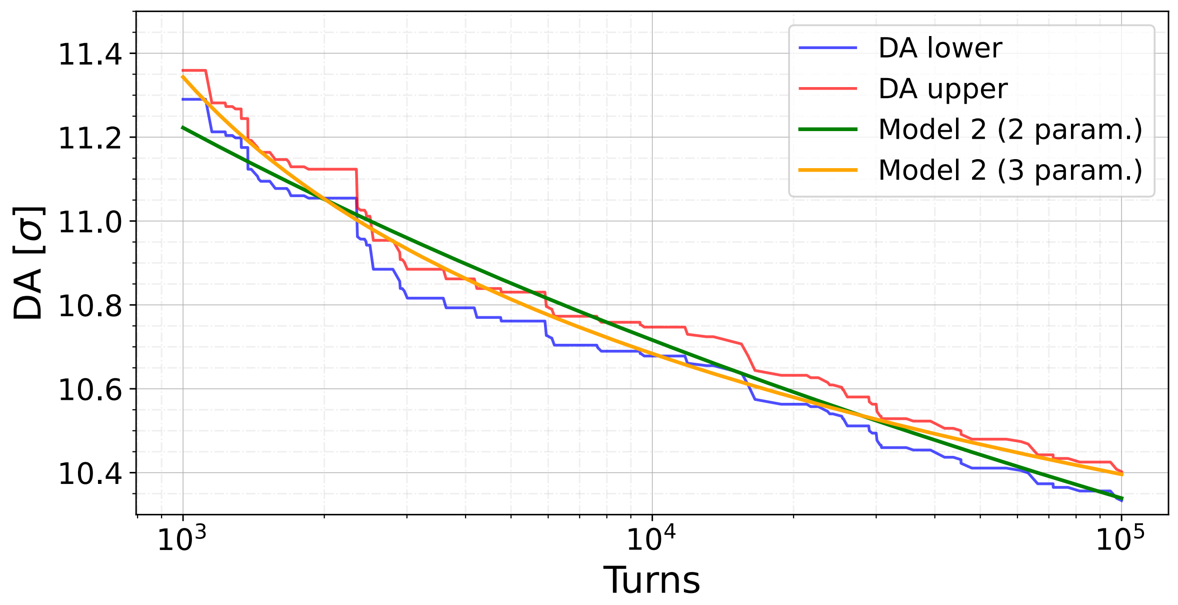


Describing the evolution of DA

T. Pugnat, FFVdV

- 2 DA estimations: **lower** (last surviving) **and upper** (first loss) (SixDesk used lower)
- Implementation of DA evolution with recent models, use sampling to improve fitting
- Exploring different fitting methods, tuning of border detection, order of integrator, sampling distribution, ...

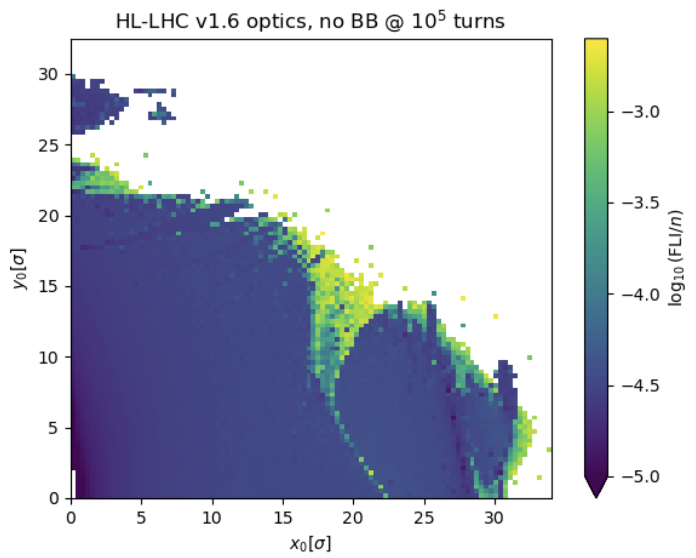


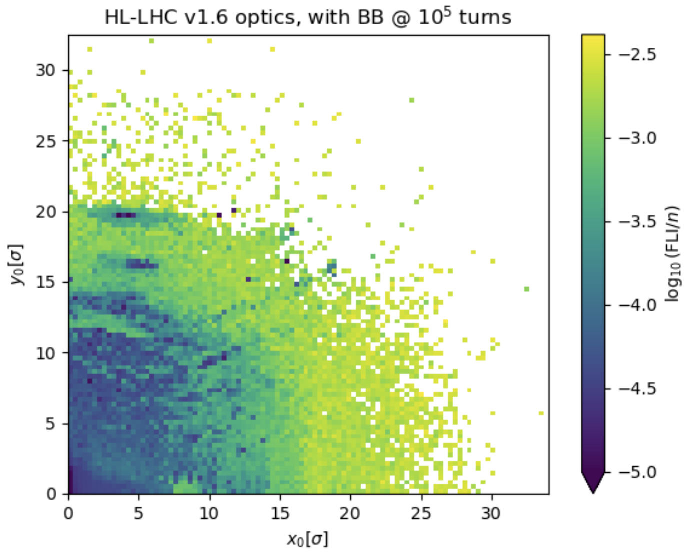


Chaos indicators (pre-alpha)

C.E. Montanari

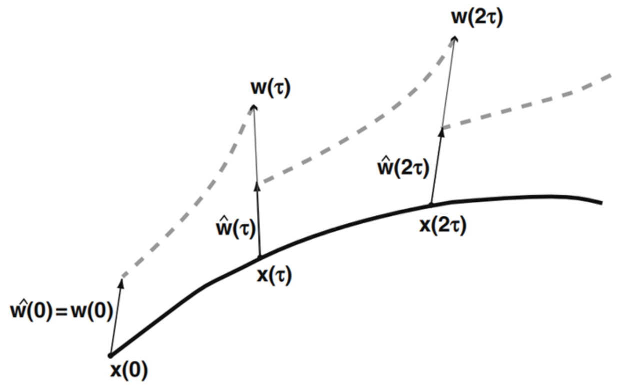
- Mathematical tools to inspect **chaoticity** of the phase-space
- Well established tools such as **Fast Lyapunov Indicators** (FLI) and **Frequency Map Analysis** (FMA), and more novel tools
  - Recently investigated here
- More details **today** at NLBD-WG
- Need proper testing, and finish implementation into Xdyna
- Pre-alpha stage on GitHub fork

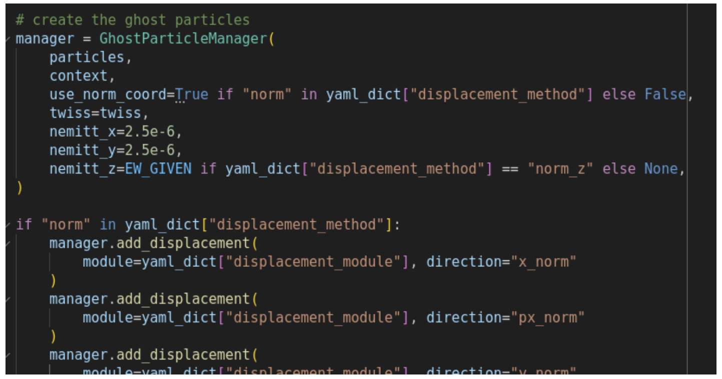




Tools for evaluating FLI

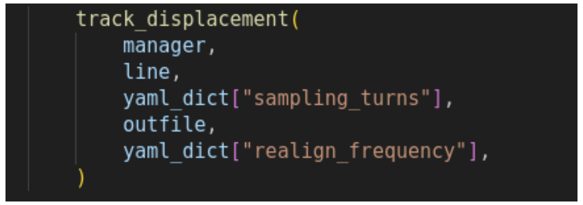
- Use **ghost particle** method to evaluate FLI:
  - for each particle, track a displaced particle and evaluate the evolving distance turn by turn
  - necessary to renormalize the distance while also keeping track of the accumulated distance
- The **GhostParticleManager** object offers the possibility to track multiple ghost particles with ease:
  - can be executed on CupyContext without breaking parallelism
  - supports displacements and renormalisations in normalised coordinates (good for avoiding geometrical biases!)





```python
# create the ghost particles
manager = GhostParticleManager(
    particles,
    context,
    use_norm_coord=True if "norm" in yaml_dict["displacement_method"] else False,
    twiss=twiss,
    nemitt_x=2.5e-6,
    nemitt_y=2.5e-6,
    nemitt_z=EW_GIVEN if yaml_dict["displacement_method"] == "norm_z" else None,
)

if "norm" in yaml_dict["displacement_method"]:
    manager.add_displacement(
        module=yaml_dict["displacement_module"], direction="x_norm"
    )
    manager.add_displacement(
        module=yaml_dict["displacement_module"], direction="px_norm"
    )
    manager.add_displacement(
        module=yaml_dict["displacement_module"], direction="y_norm"
```



```python
    track_displacement(
        manager,
        line,
        yaml_dict["sampling_turns"],
        outfile,
        yaml_dict["realign_frequency"],
    )
```

Conclusions

Cool stuff, right?

Thanks for your attention!

Feel free to use at will, comment/correct/contribute!

home.cern