SASE optimization with OCELOT

Sergey Tomin

other co-workers: I. Agapov, G. Geloni, I. Zagorodnov

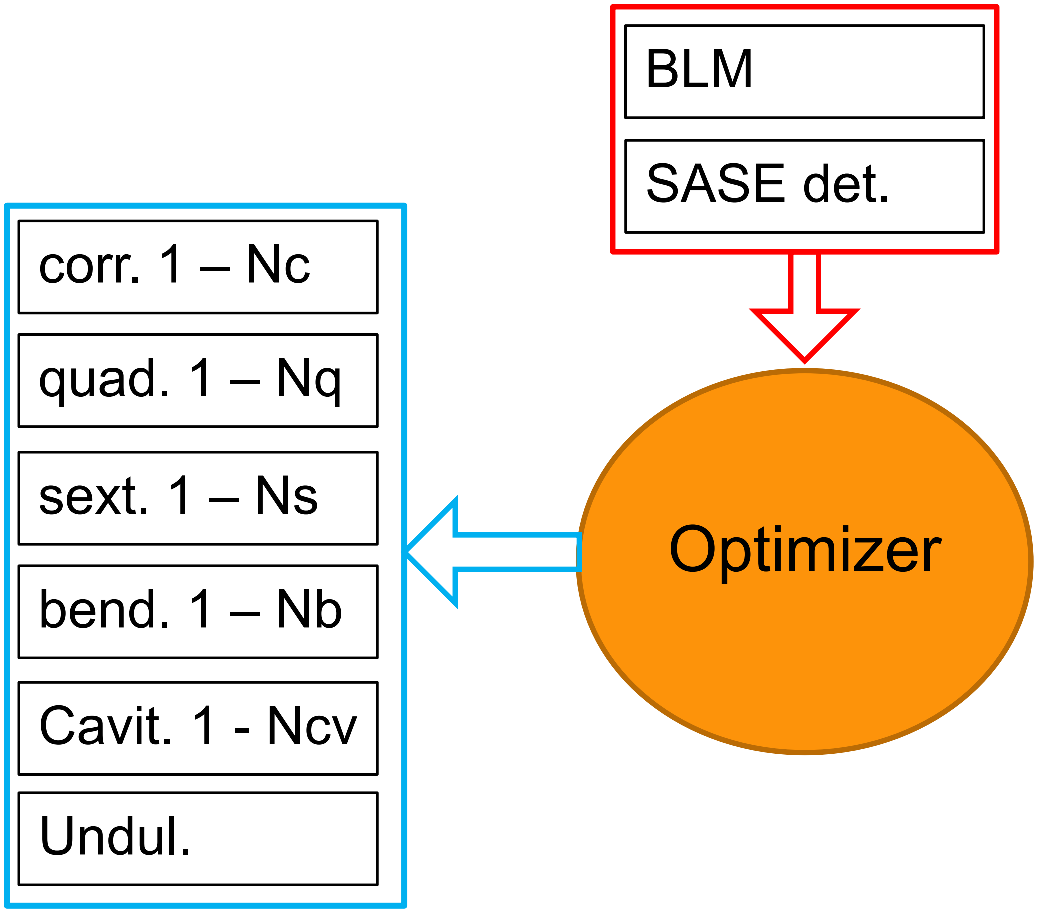
Outline

- Motivation
- How it works
- Recent results of empirical tuning at FLASH (model-free optimization)
- OCELOT features in beam dynamics simulations
- Extending of empirical tuning by model-depending optimization
- Summary

Motivation

The major motivation is the benefit of improving facility availability and performance through more effective and faster tuning

How it works



Typical tuning sequence for FLASH:

- V14, V7, H10, H12, H3, V3
- Q13SMATCH, Q14SMATCH, Q15SMATCH
- FODO QUADS
- intra-undulator orb. correctors
- RF phases and Voltages

Orbit correctors and FODO quads have largest impact

All steps programmed to avoid electron beam losses in the undulators above threshold (solutions above 0.7 alarm level highly penalized and above alarm level forbidden). .

How it works

- Python script

```
dp = FLASH1DeviceProperties()
mi = FLASH1MachineInterface()

#dp = LCLSDeviceProperties()
#mi = LCLSMachineInterface()

opt = Optimizer(TestInterface(), mi, dp)
opt.log_file = 'test.log'
opt.timeout = 1.2

seq1 = [Action(func=opt.max_sase, args=[ ['H10SMATCH','H12SMATCH'], 'simplex'] ) ]

seq2 = [Action(func=opt.max_sase, args=[ ['V14SMATCH','V7SMATCH'], 'simplex' ] )]

seq3 = [Action(func=opt.max_sase, args=[ ['Q13SMATCH','Q15SMATCH'], 'simplex' ] )]

opt.eval(seq1)
#opt.eval(seq1 + seq2 + seq3 + seq4 + seq5)
```

initialization machine interface between Ocelot and Control System

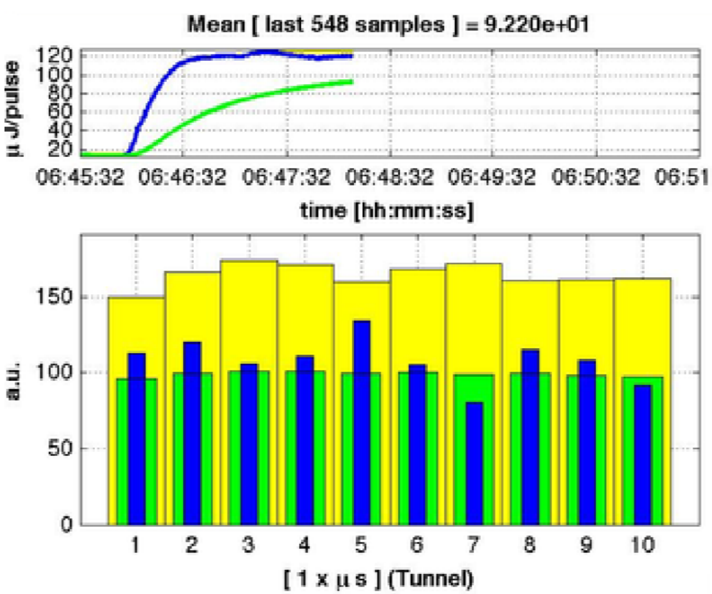
Device ID

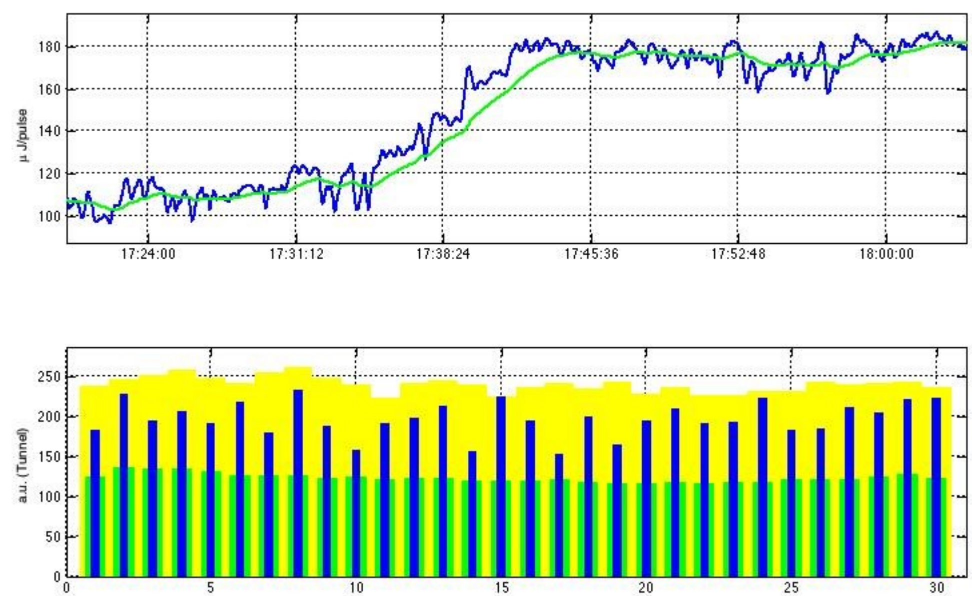
Optimization method

Experience of model-free optimization at FLASH

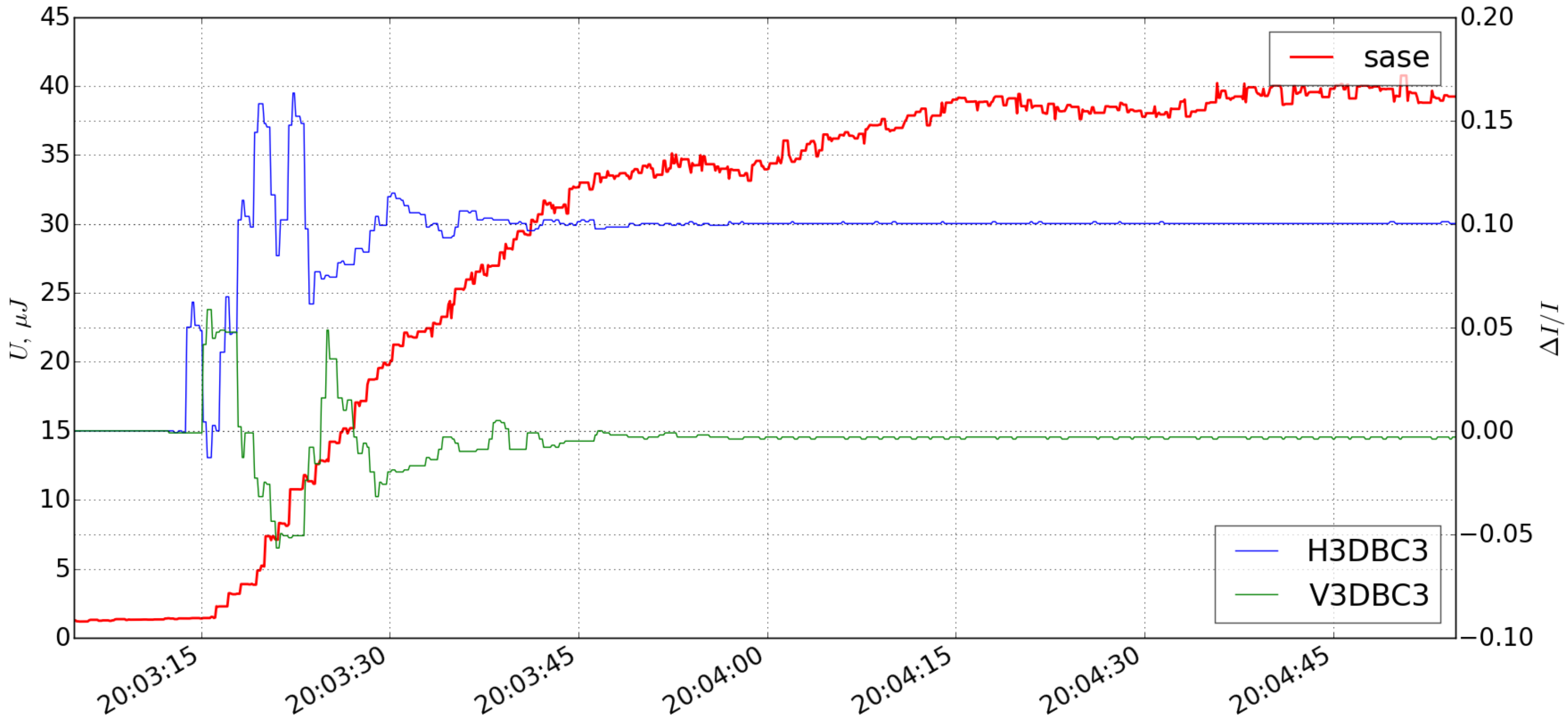
- Automatic SASE tuning works in a few minutes if the machine is in initially stable condition

- Demonstrated at several wavelengths (17nm, 13.5nm, 10.4 nm, 7 nm) with different bunch filling at about 0.3 nC charge



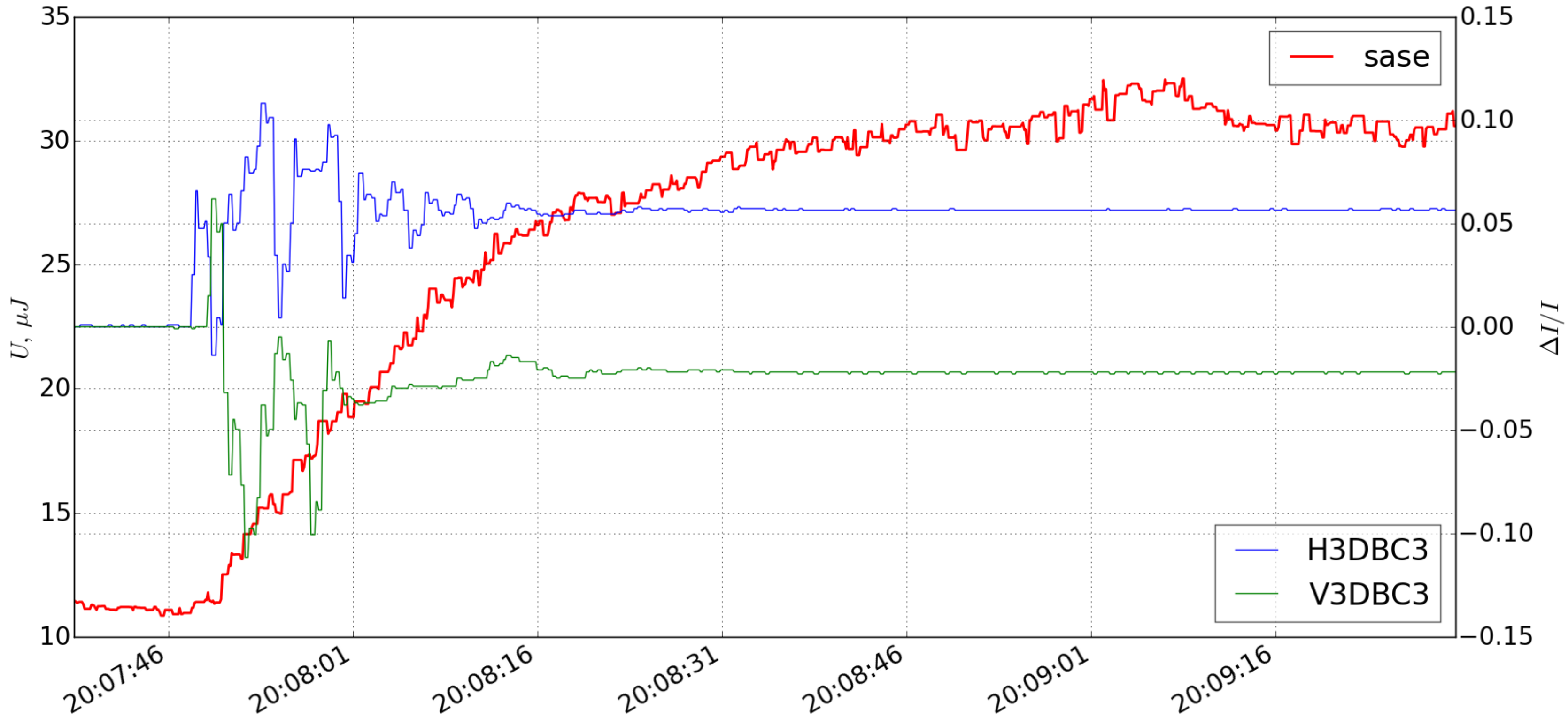


Last shift results: SASE optimization by correctors $\lambda = 10.4\ nm$



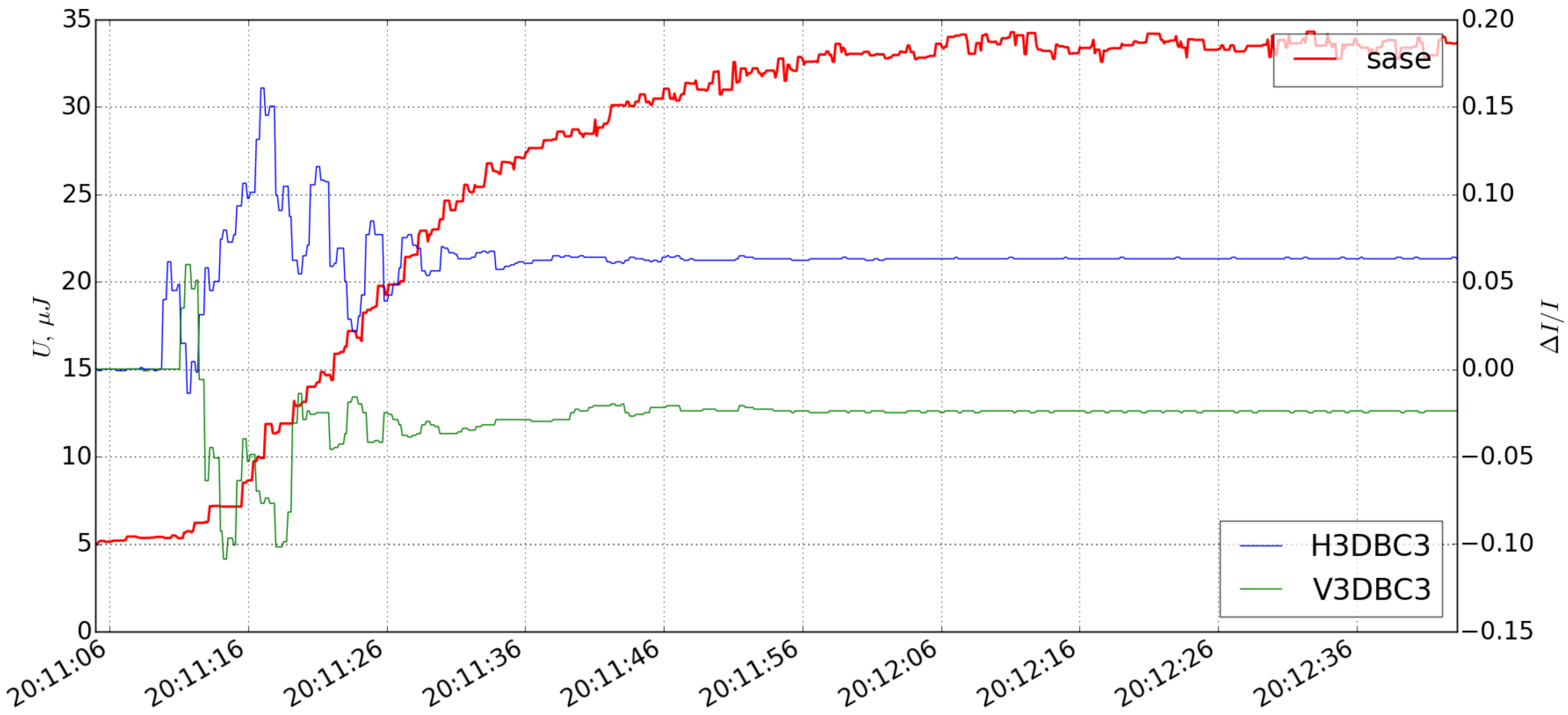
- For optimization is necessary initial SASE signal
- To check repeatability of the optimization techniques we repeated the same experiment after resetting correctors to initial values

Last shift results: SASE optimization by correctors $\lambda = 10.4\ nm$



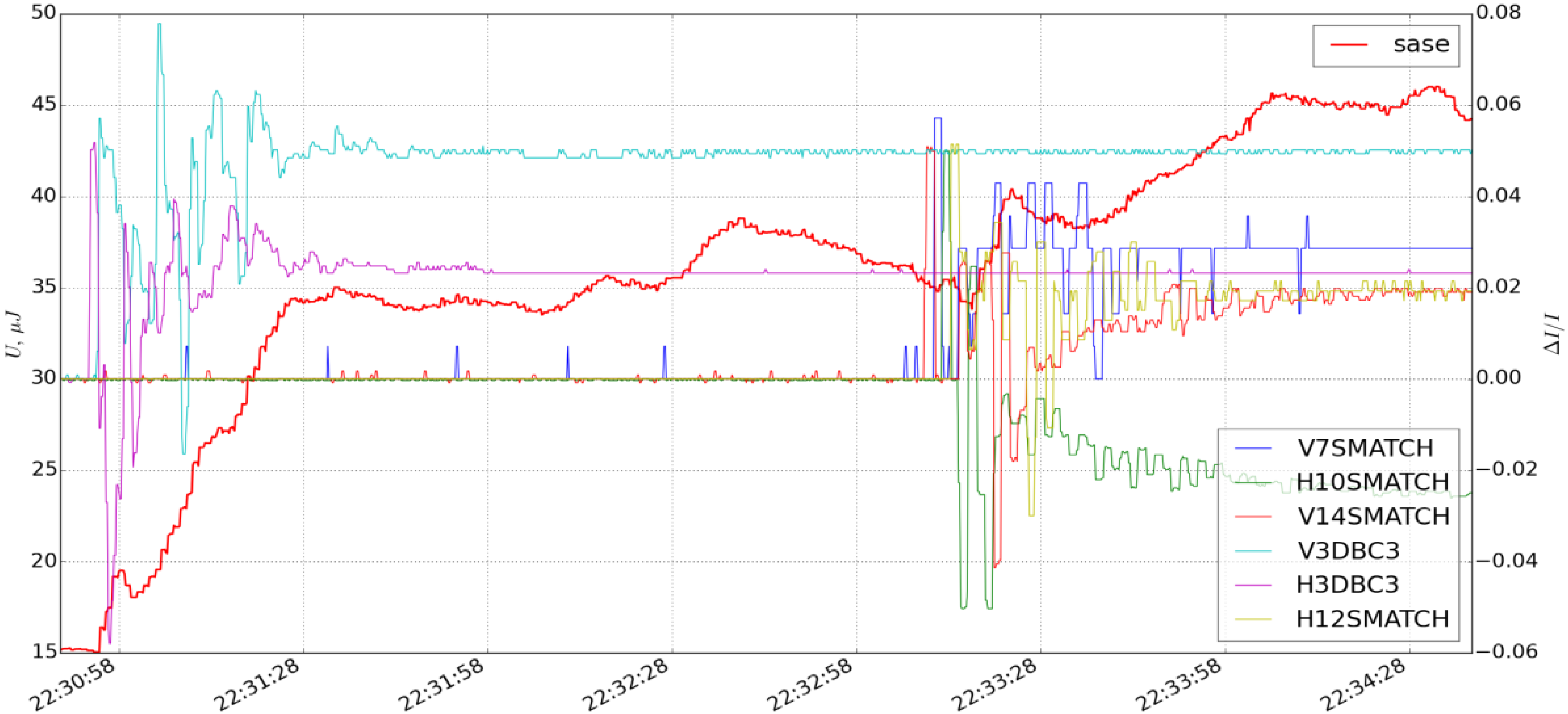
- after resetting correctors

Last shift results: SASE optimization by correctors $\lambda = 10.4\ nm$



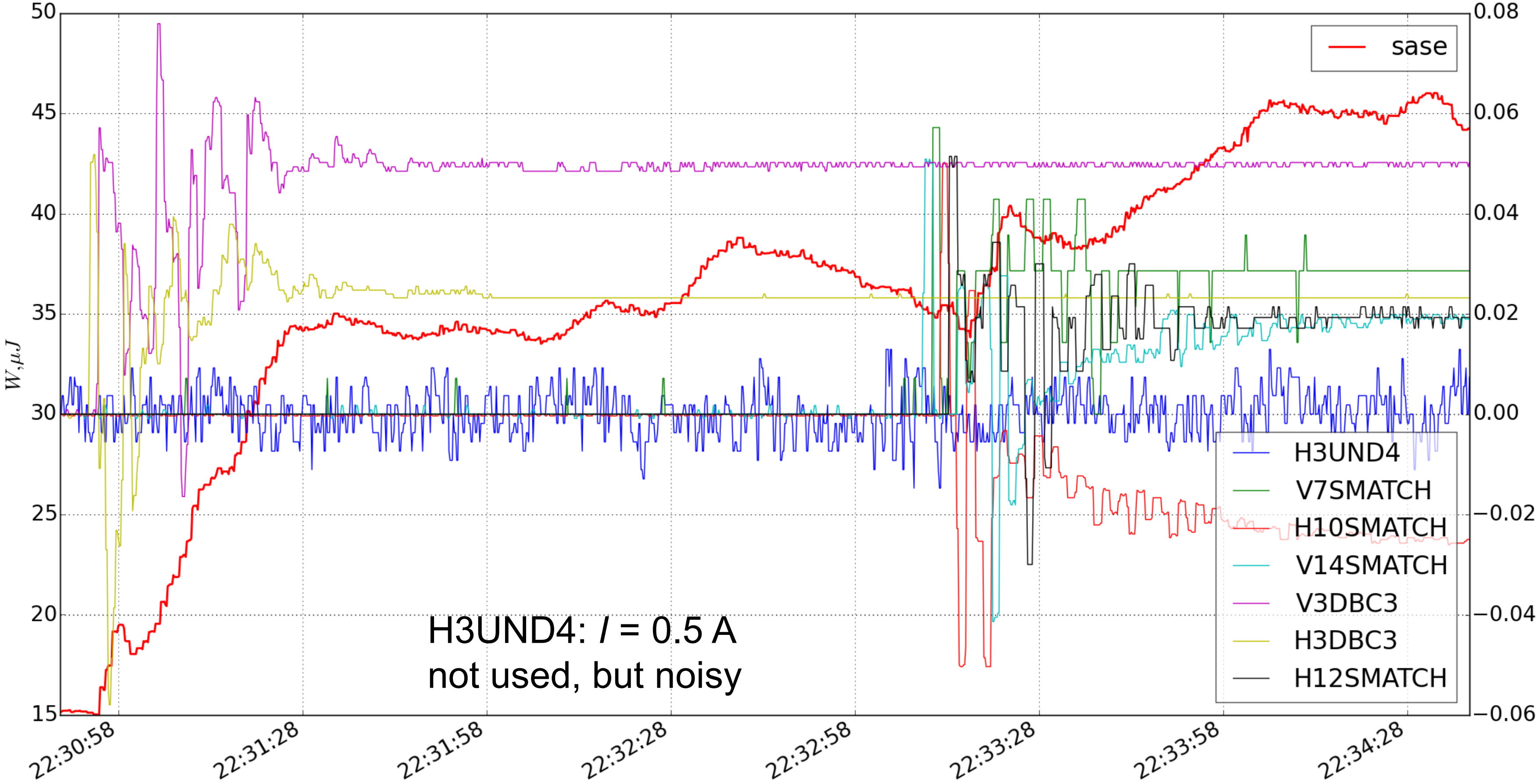
- … and after correctors cycling

Last shift results: SASE optimization by correctors $\lambda = 10.4\ nm$



- Extended set of correctors used
- All changes made by the OCELOT optimizer logged (without online processing)

Last shift results: SASE optimization by correctors $\lambda = 13.4\,nm$



- From previous experiments we found most effective correctors and repeated optimization process using them for new machine settings $\lambda = 13.4\,nm$

Resume

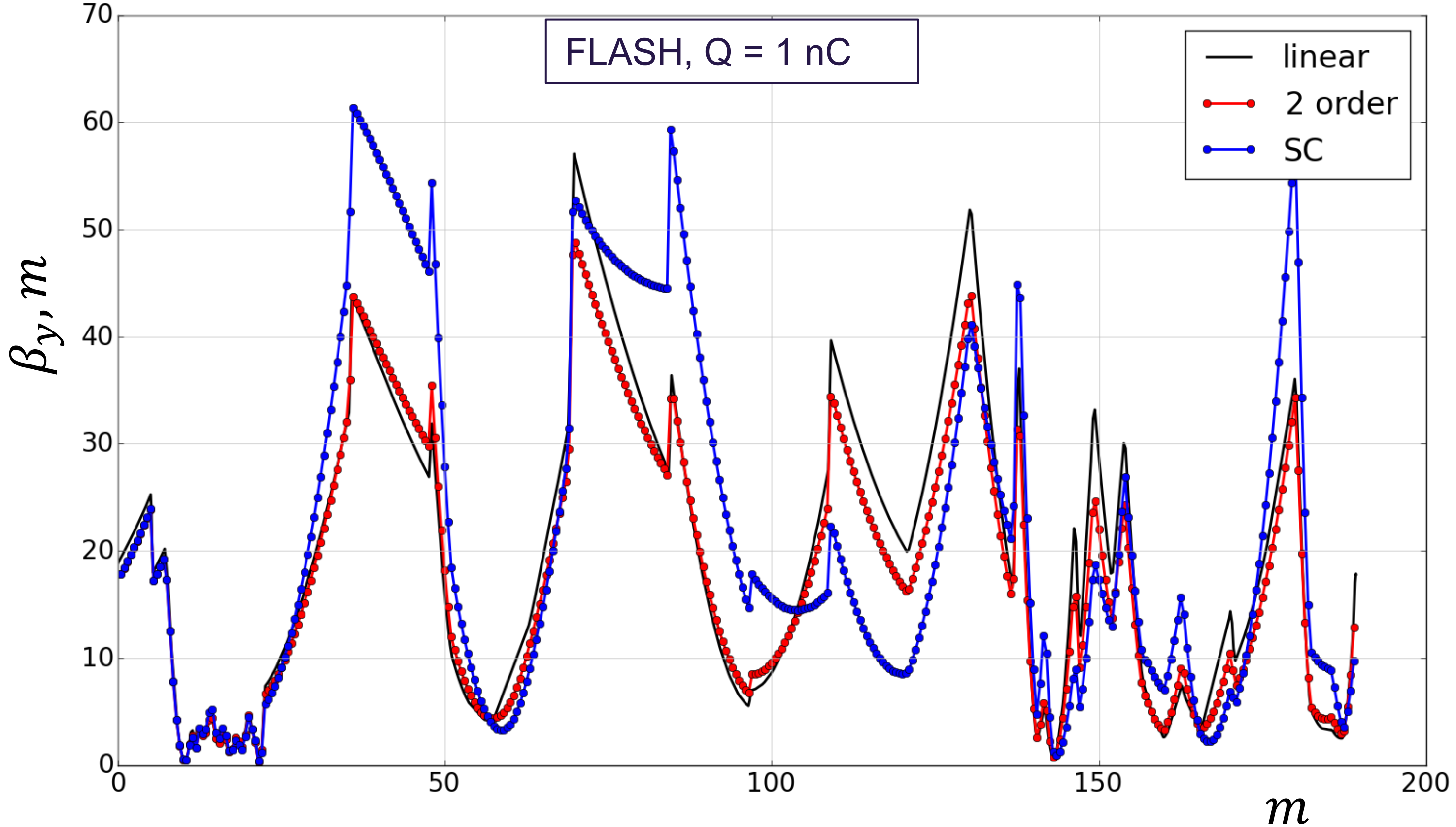
- The empirical tuning works and was demonstrated at FLASH at several wavelengths (17nm, 13.5nm, 10.4 nm, 7 nm) with different bunch filling at about 0.3 nC charge
- The method was demonstrated at SLAC (mainly uses quadrupoles in the linac. They call this tuning procedure “Ocelot optimization“)
- Stability of the machine operation is necessary. From our experience in ~10-15% of cases SASE fluctuations were too large and the method does not work.
- A universally effective sequence of operations for fast SASE tuning does not exist (it is necessary to control the optimization from operator’s side).
- The initial SASE signal is necessary
- The optimization is model-free.

Ocelot overview

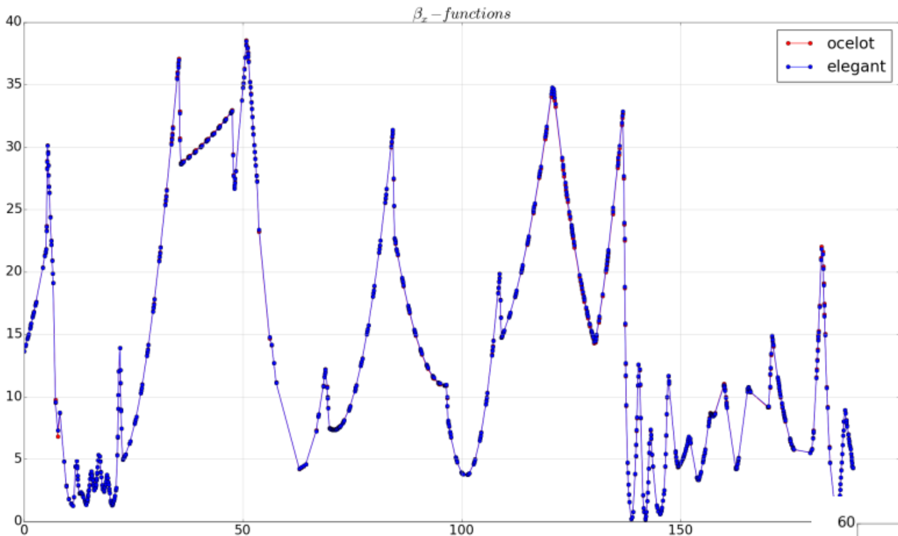
- OCELOT is a multiphysics simulation toolkit
- Twiss parameters calculation (CPBD module)
- Particle tracking module (CPBD module)
- Matching module (CPBD module)
- Orbit correction module (was implemented for Siberia-2 Light Source at 2013 see Tomin et al., proc. IPAC 2013) (CPBD module)
- Native module for spontaneous radiation calculation
- Native module for photon ray tracing
- SASE calculations using GENESIS
- Python based and open source https://github.com/iagapov/ocelot
- OCELOT was designed with on-line capability in mind (see Agapov et al., NIM A. 768 2014)
- Developed infrastructure for switching between flight simulator/controls mode with binding to DOOCS and EPICS. Already used for on-line beam control at Siberia-2 and for SASE tuning at FLASH and LCLS.

Developing Charge Particle Beam Dynamics (CPBD) module in OCELOT.

- ✓ added second order matrices.
- ✓ added space charge solver (@ M.Dohlus and I.Zagorodnov)
- ✓ CSR solver in progress (@ M.Dohlus)

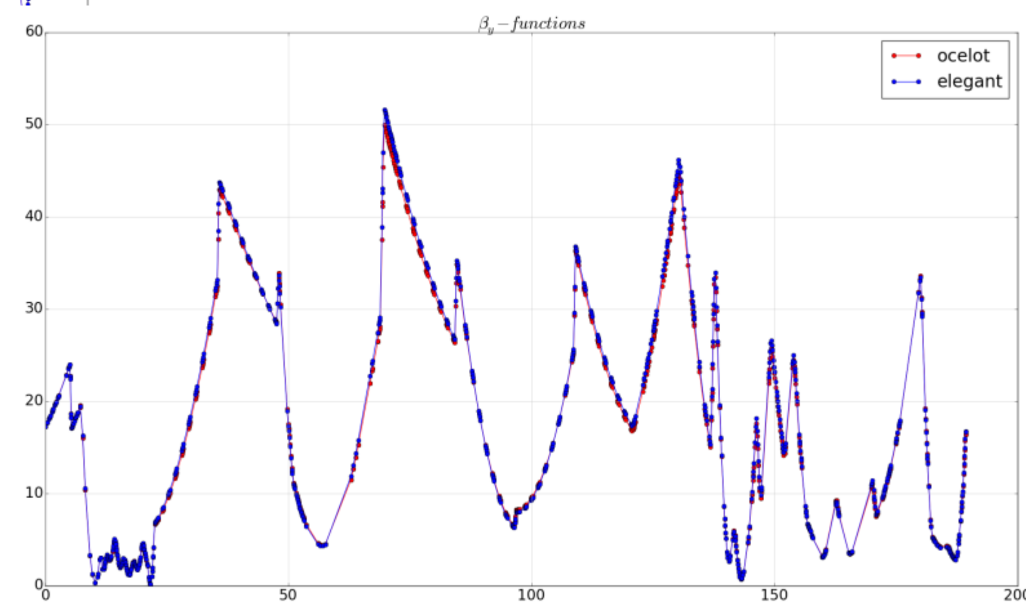


Cross-checking with Elegant: Tracking including second order matrices

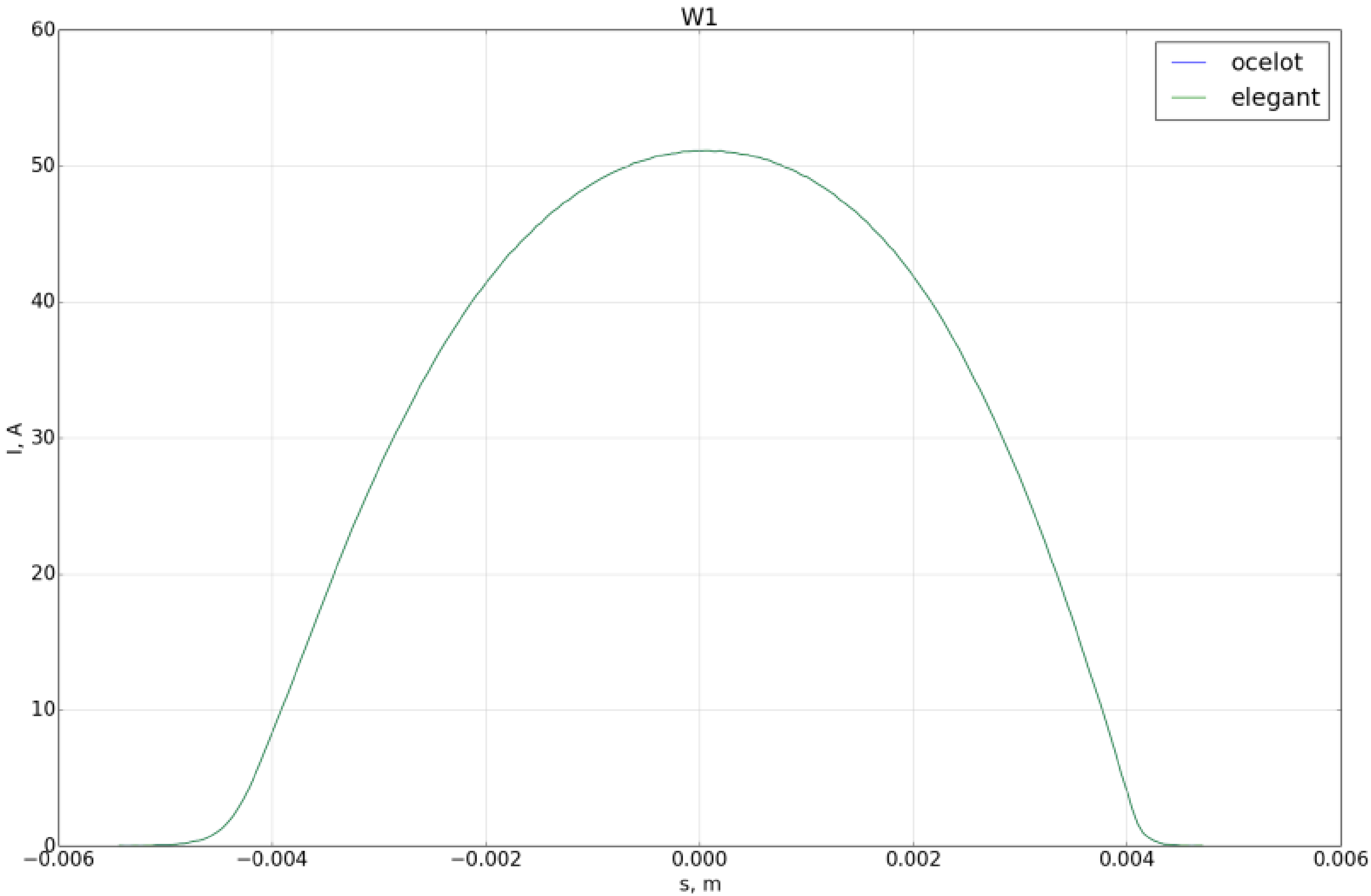


β_x cross-checking

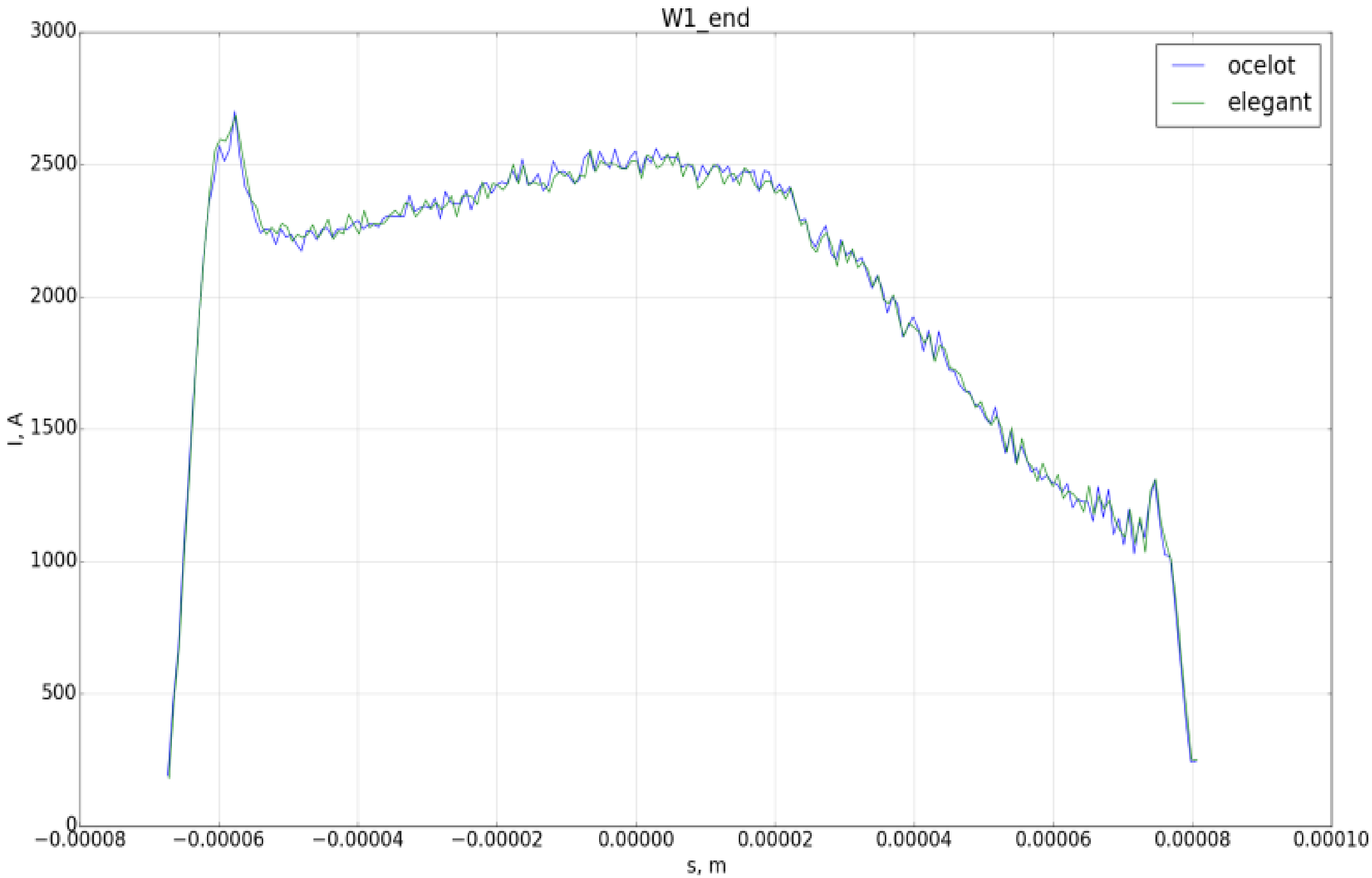
β_y cross-checking



Cross-checking with Elegant: Current profile at the Start point (FLASH)



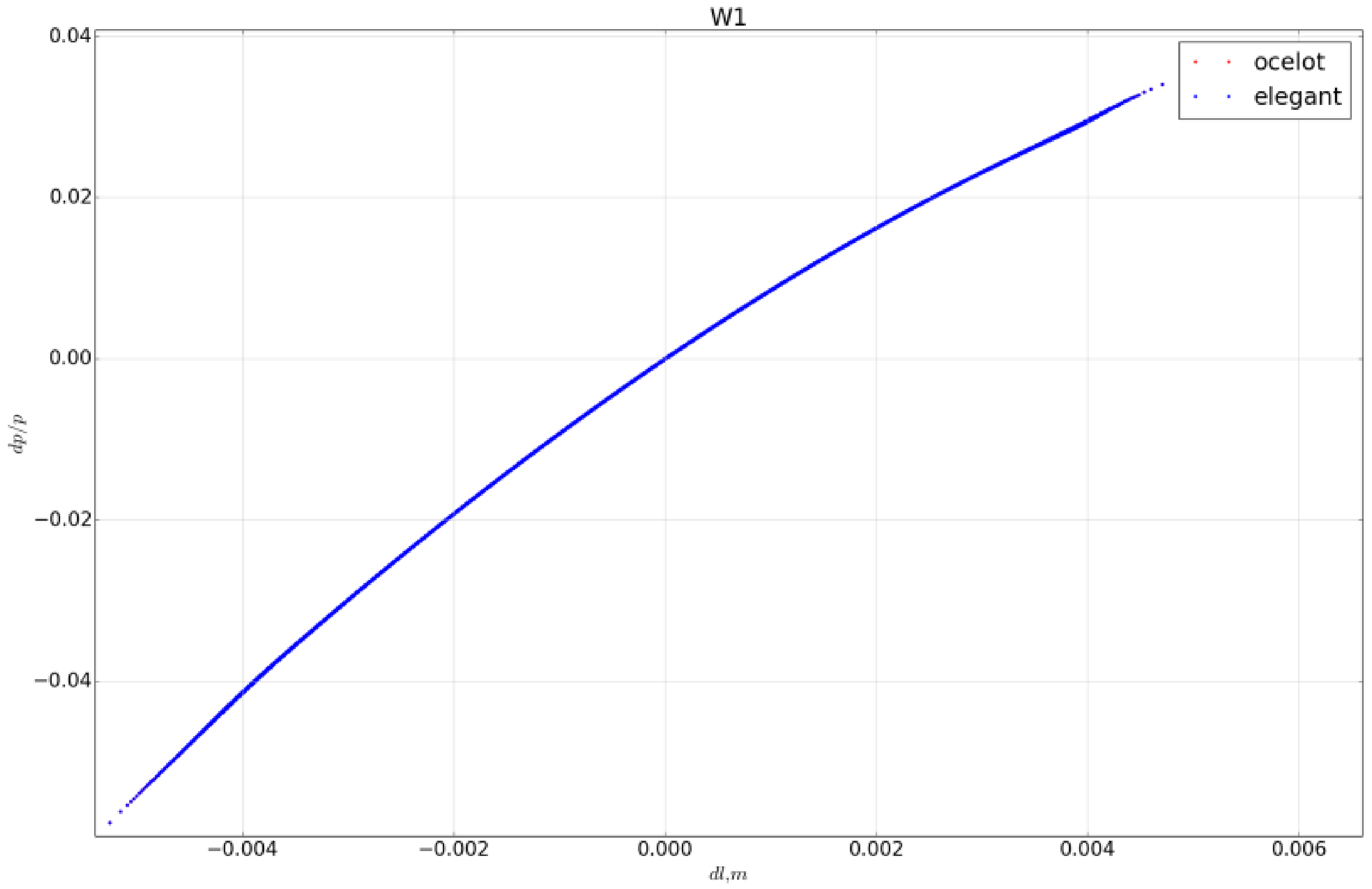
Cross-checking with Elegant: Current profile at the End point



- Using second order matrices

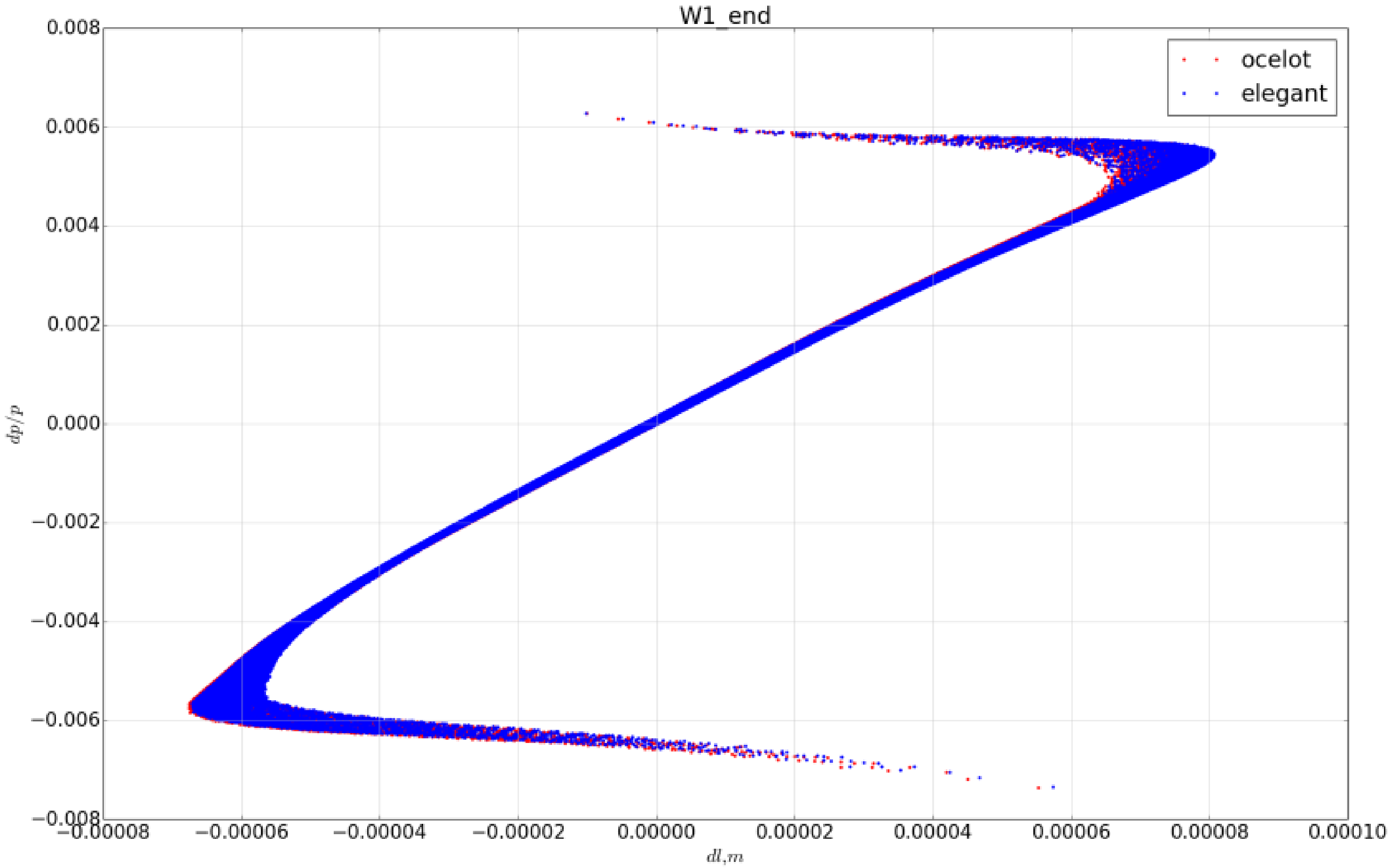
Cross-checking with Elegant: Beam distribution in space $(\Delta l - {}^{\Delta p}/_{p})$

at the Start point of FLASH



Cross-checking with Elegant: Beam distribution in space $\left(\Delta l - {}^{\Delta p}/_{p}\right)$

at the End point of FLASH



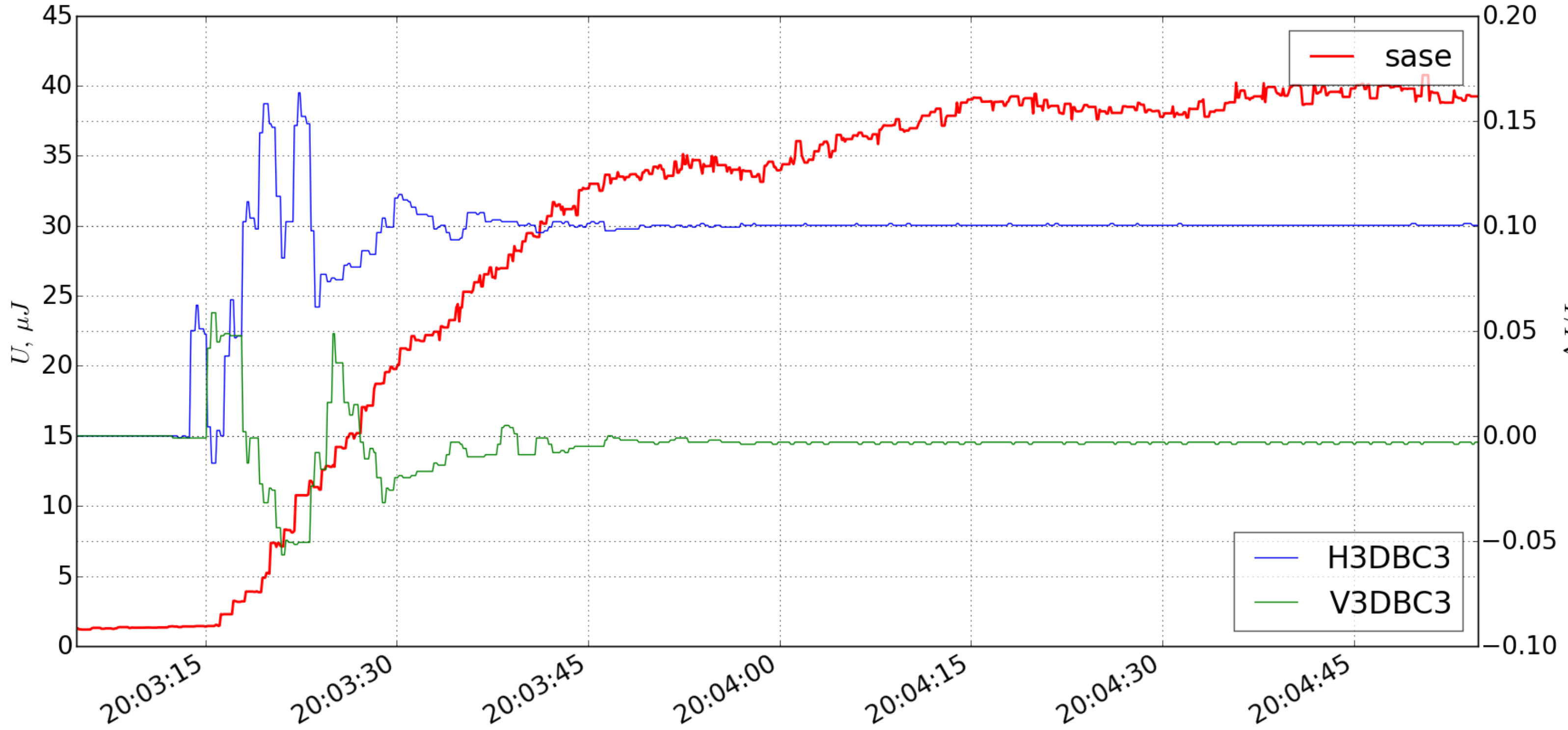
Future plans of development CPBD module

- Completing CSR solver in cooperation with DESY
- Add wakefield effects in cooperation with DESY
- Further development and testing of symplectic tracking
- Further development of matching module

Extending of empirical tuning by model-depending optimization

- Creation of realistic model of accelerator
  - Visualization of changes during Ocelot optimization.
  - Orbit steering
  - Algorithm of strategy selection

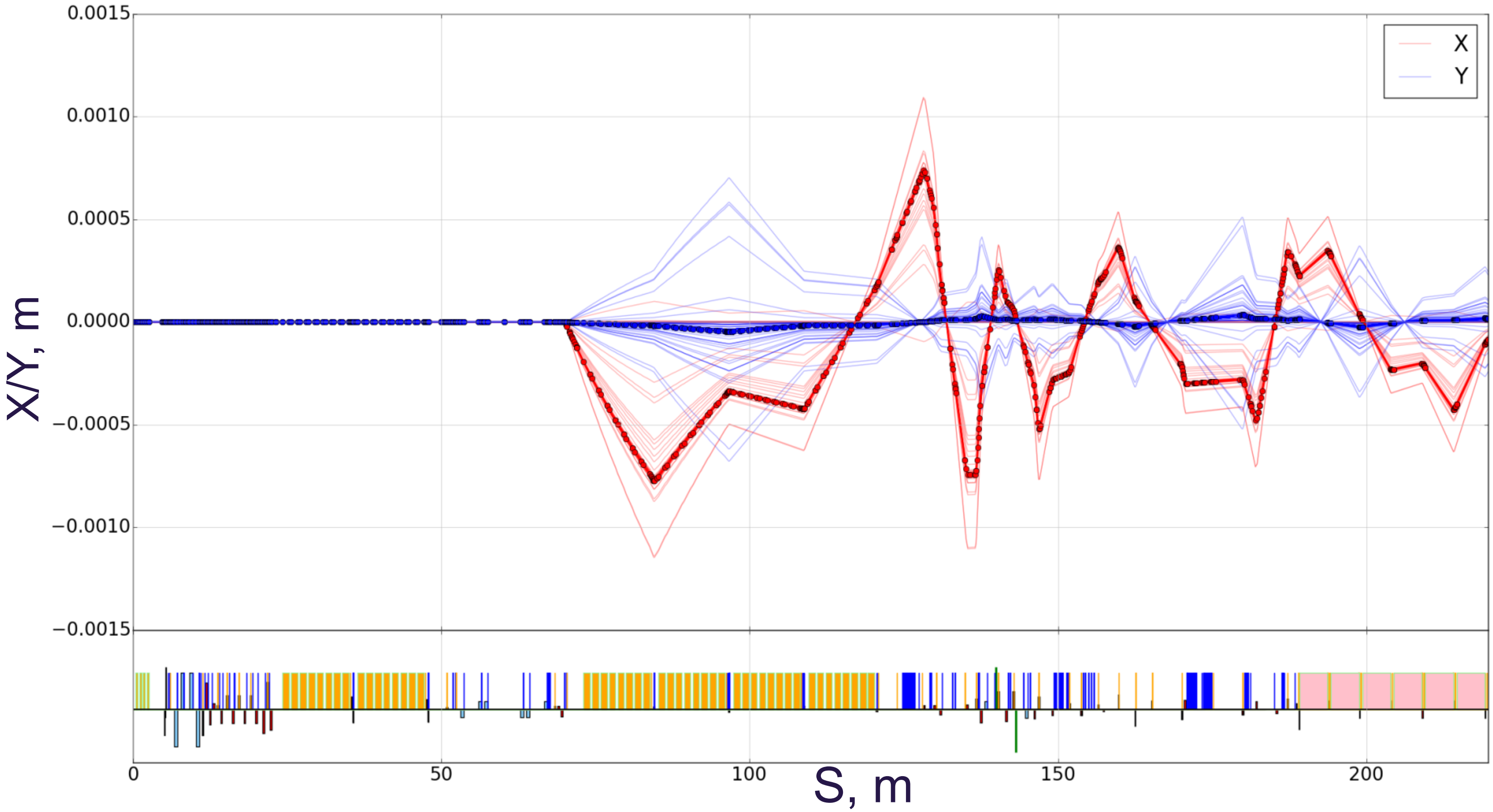
Visualization of changes during Ocelot optimization



Visualization of changes during Ocelot optimization

Using converter tpi2k() (@Mathias Vogt) → corrector kicks

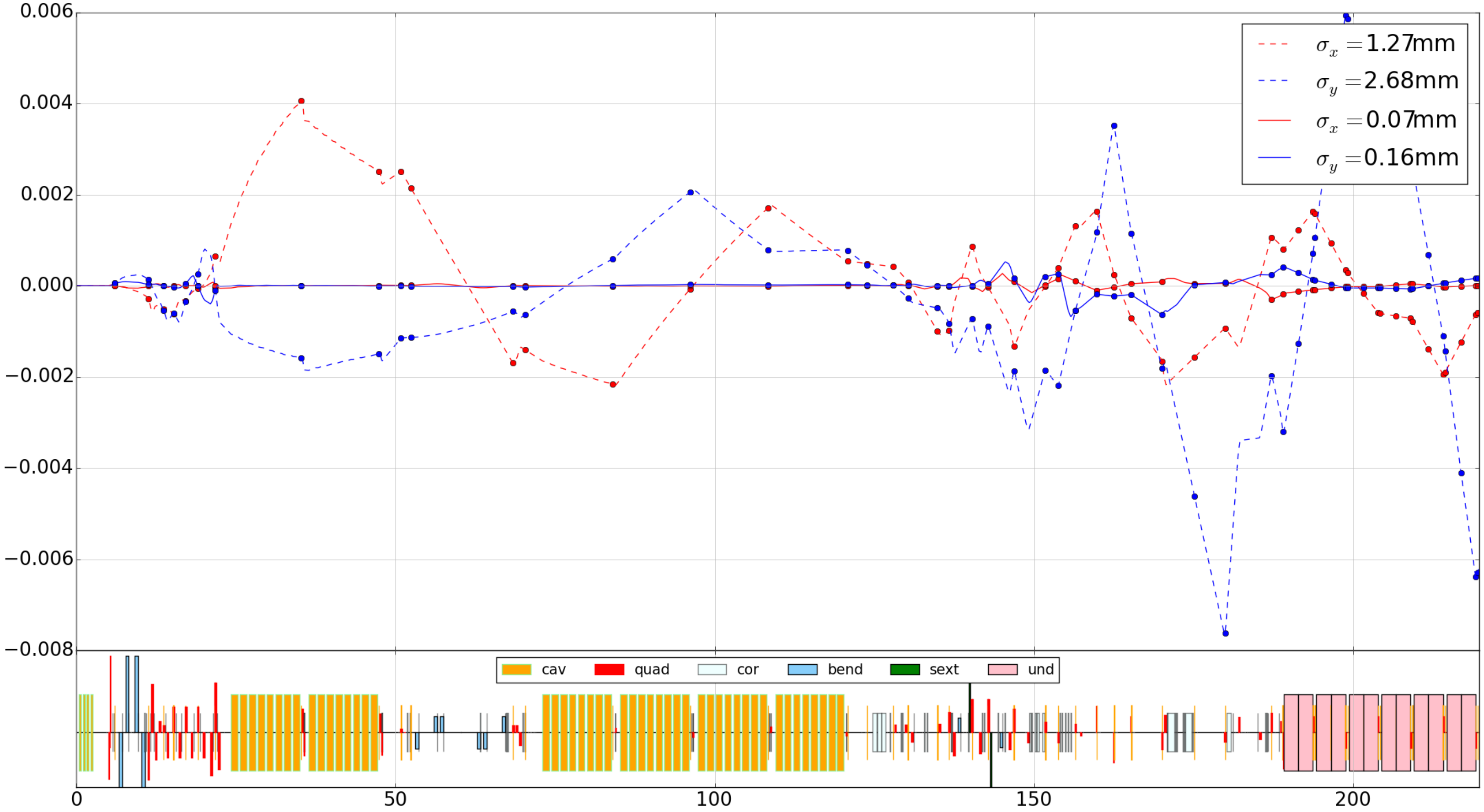
Kicks fed into ocelot → particle tracking → relative changes in beam trajectory



- It can give idea that problem was in horizontal direction

Orbit steering

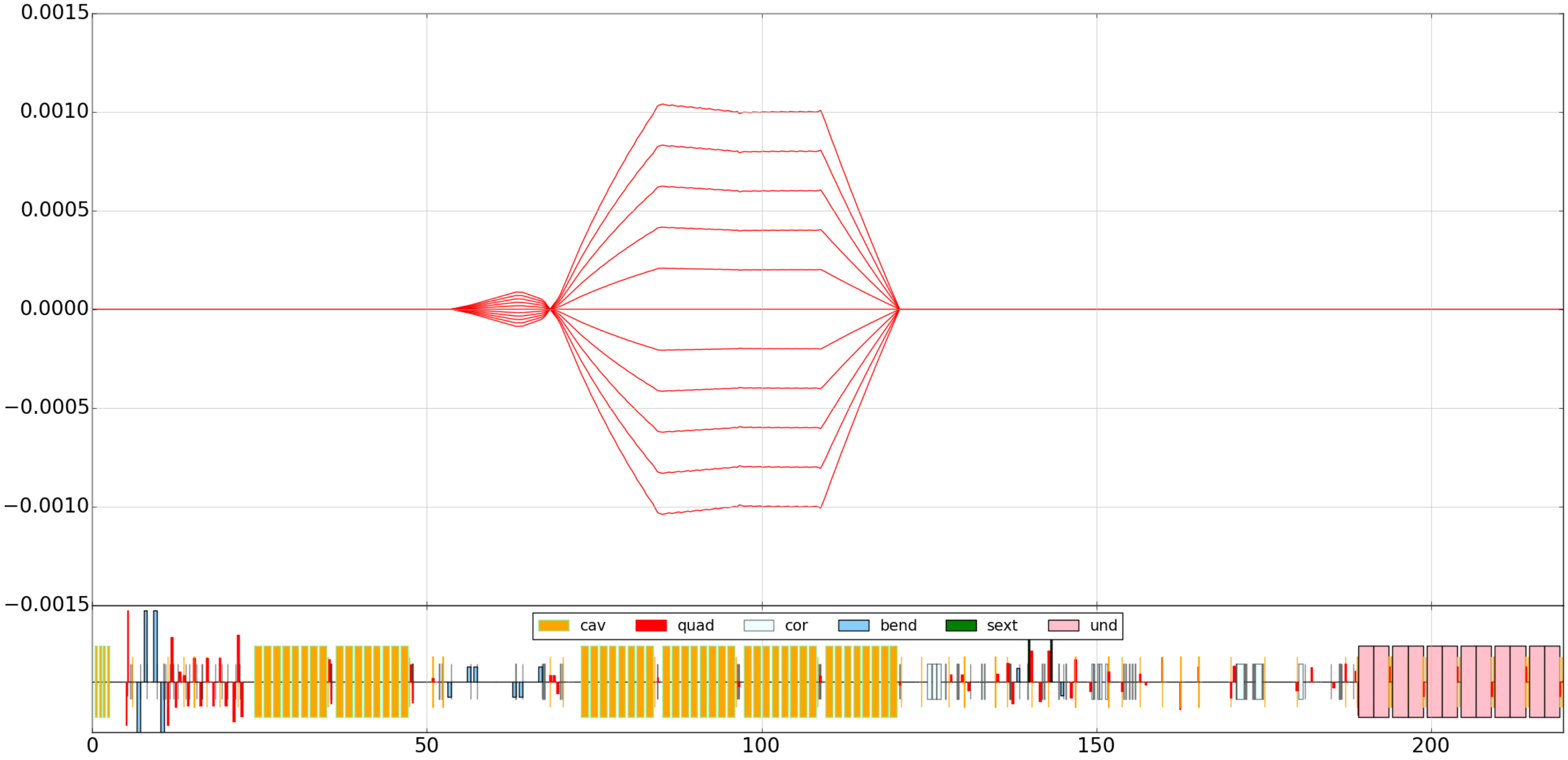
- Restoring orbit from previous successful optimization to get initial SASE signal → SASE optimization.



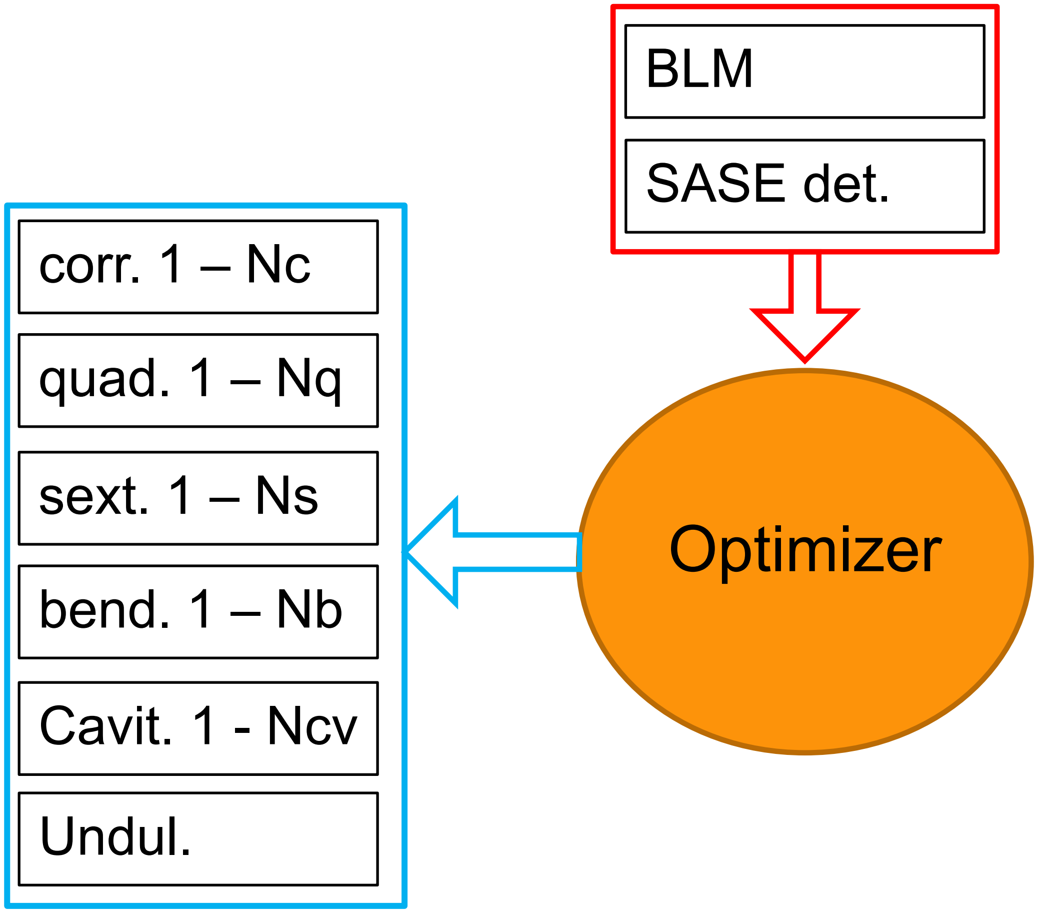
- Ideal lattice

Orbit steering

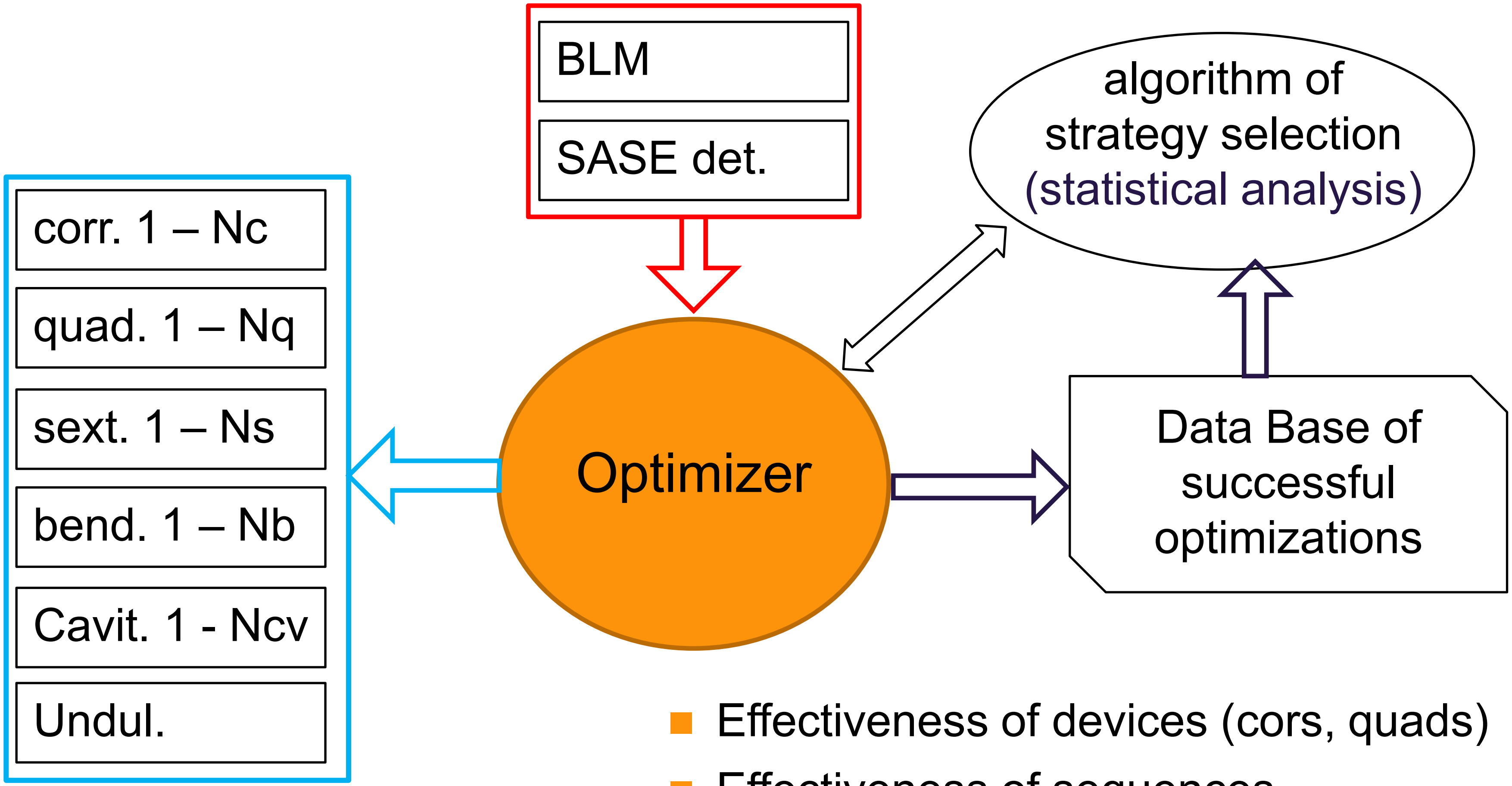
- Changing orbit in some point of the lattice to maximize SASE level.



Algorithm of strategy selection

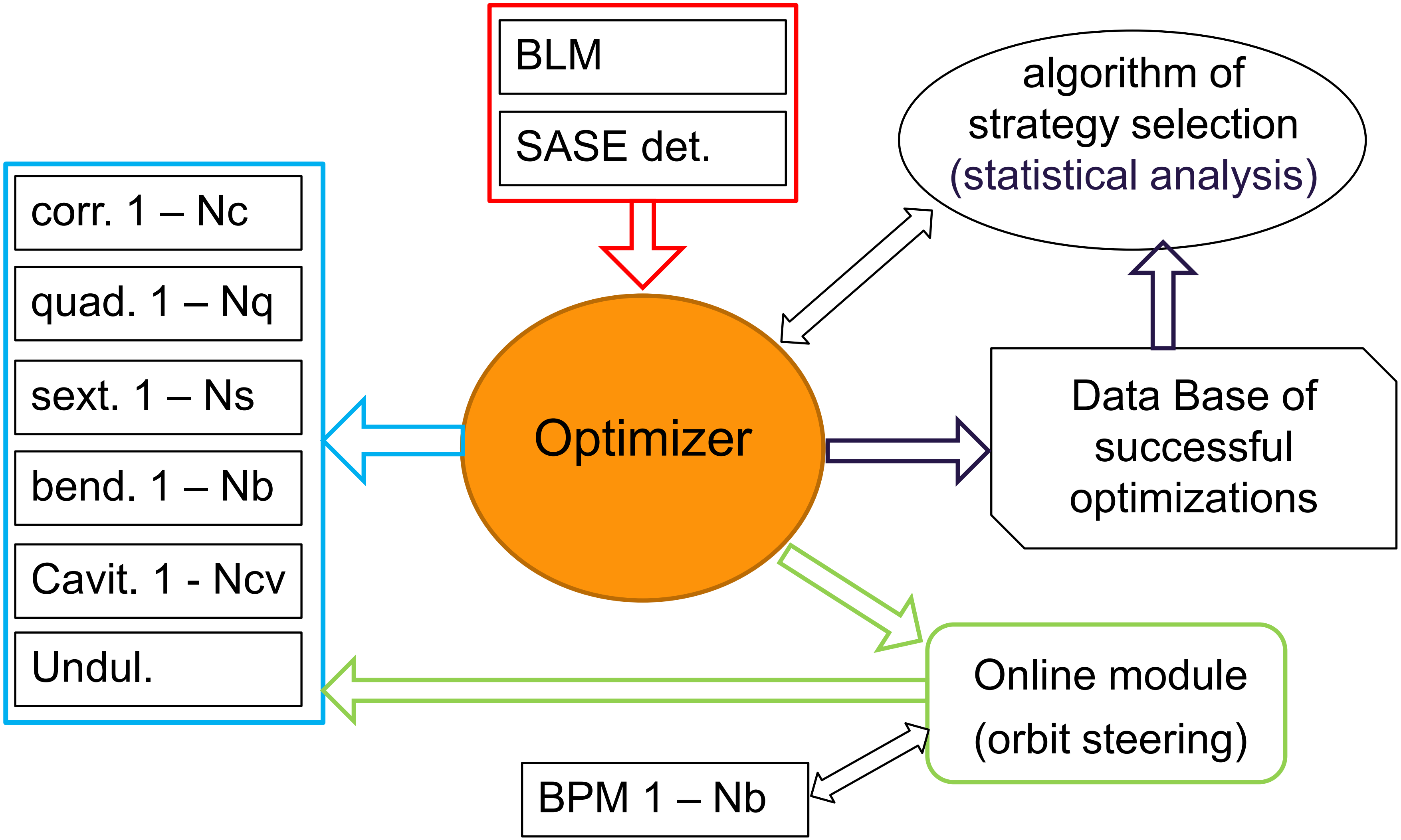


Algorithm of strategy selection



- Effectiveness of devices (cors, quads)
- Effectiveness of sequences
- Orbits with maximum SASE signal

Algorithm of strategy selection



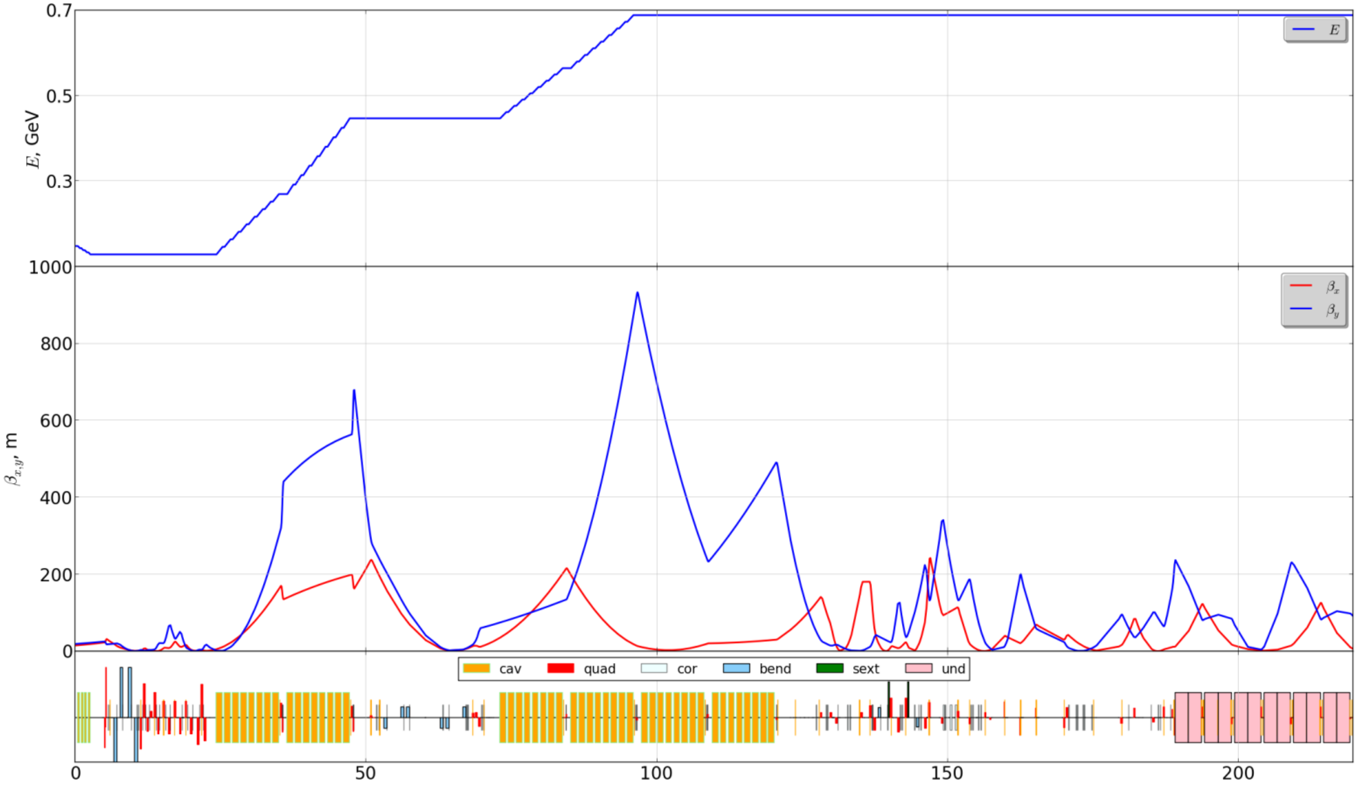
Online model of accelerator ?

Used last optics file (MAD8 and Elegant)

Taken the amplitudes and phases of cavities from control system by PyDoocs

Taken quadrupoles currents from the control system and using tpi2k we got the strengths

And we got these beta functions…



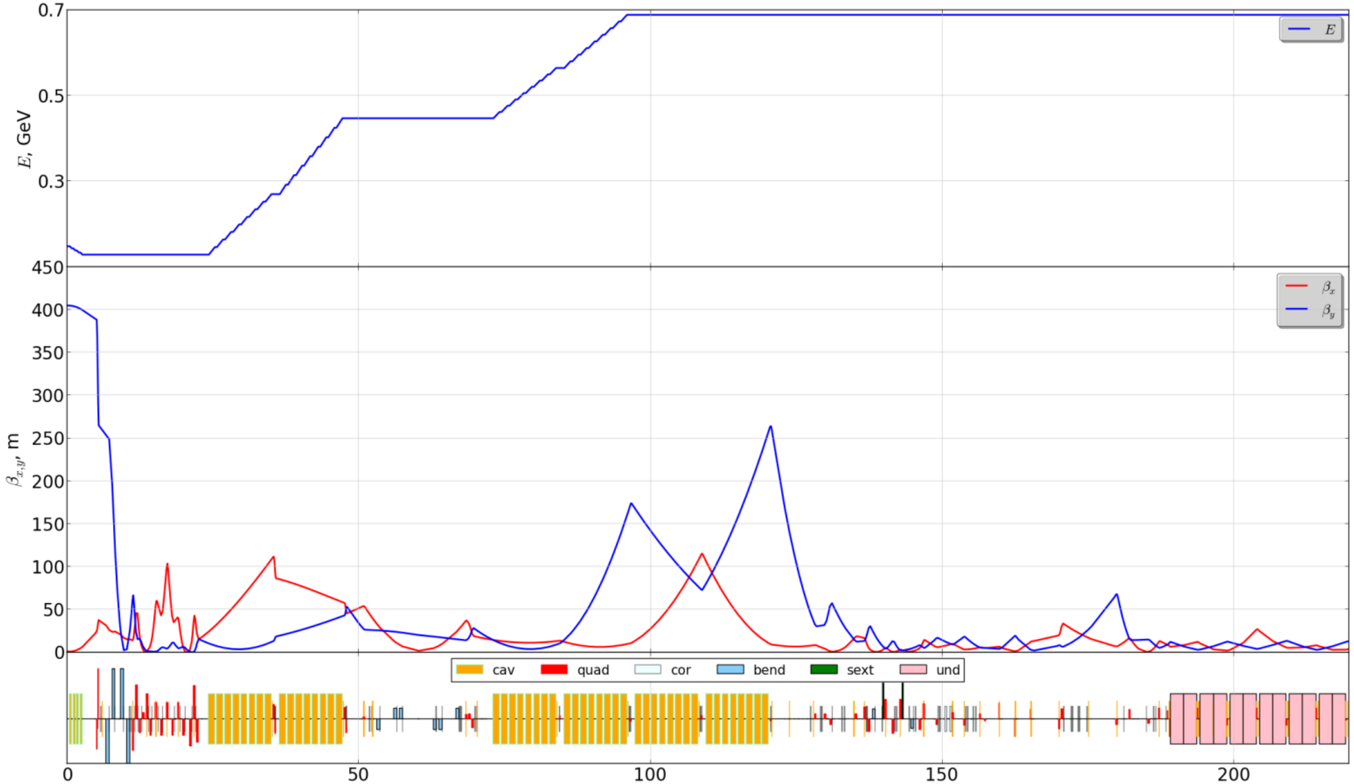
Online model of accelerator ?

Used last optics file (MAD8 and Elegant)

Taken the amplitudes and phases of cavities from control system by PyDoocs

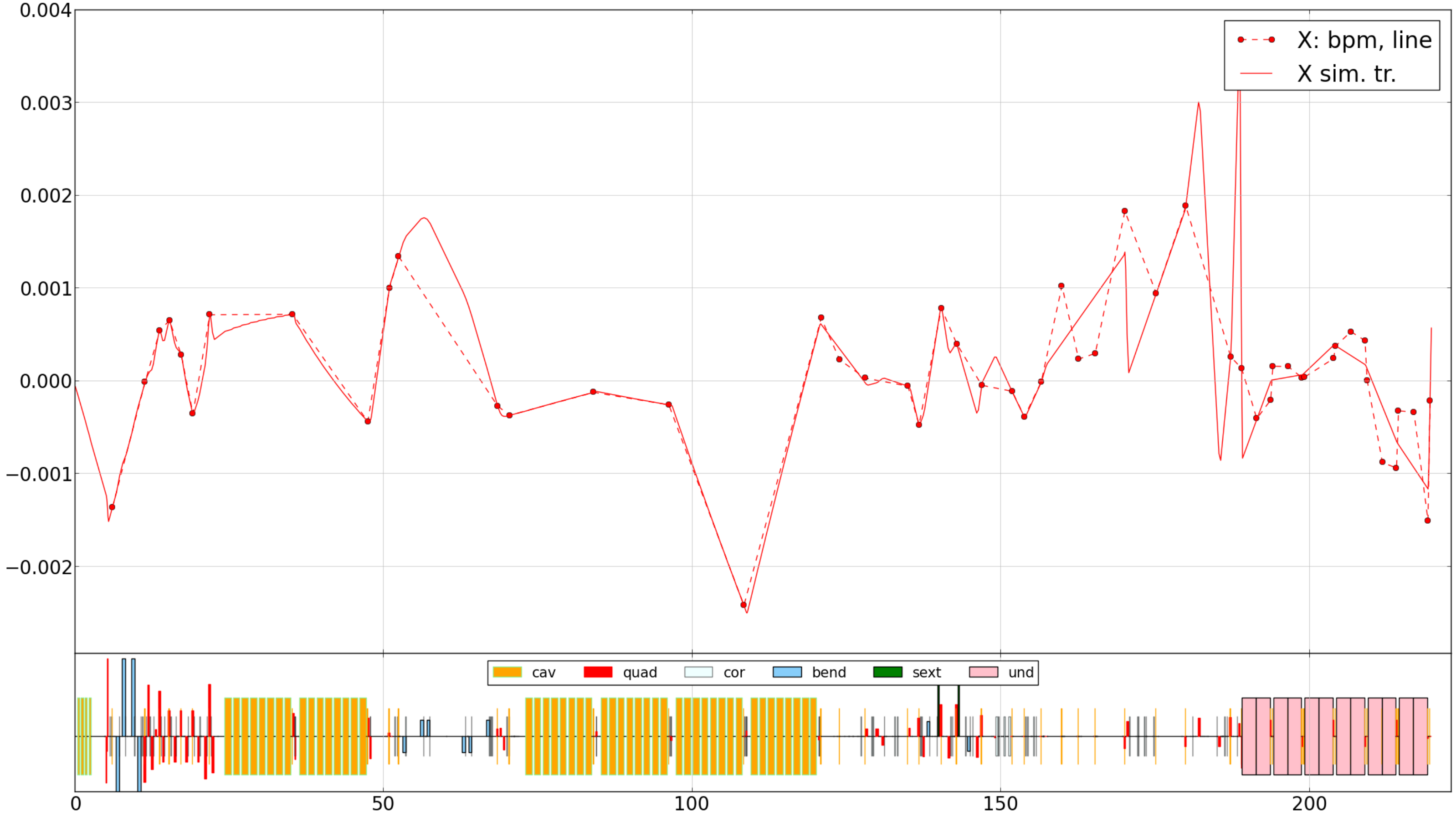
Taken quadrupoles currents from the control system and using tpi2k we got the strengths

And we got these beta functions… **after matching**



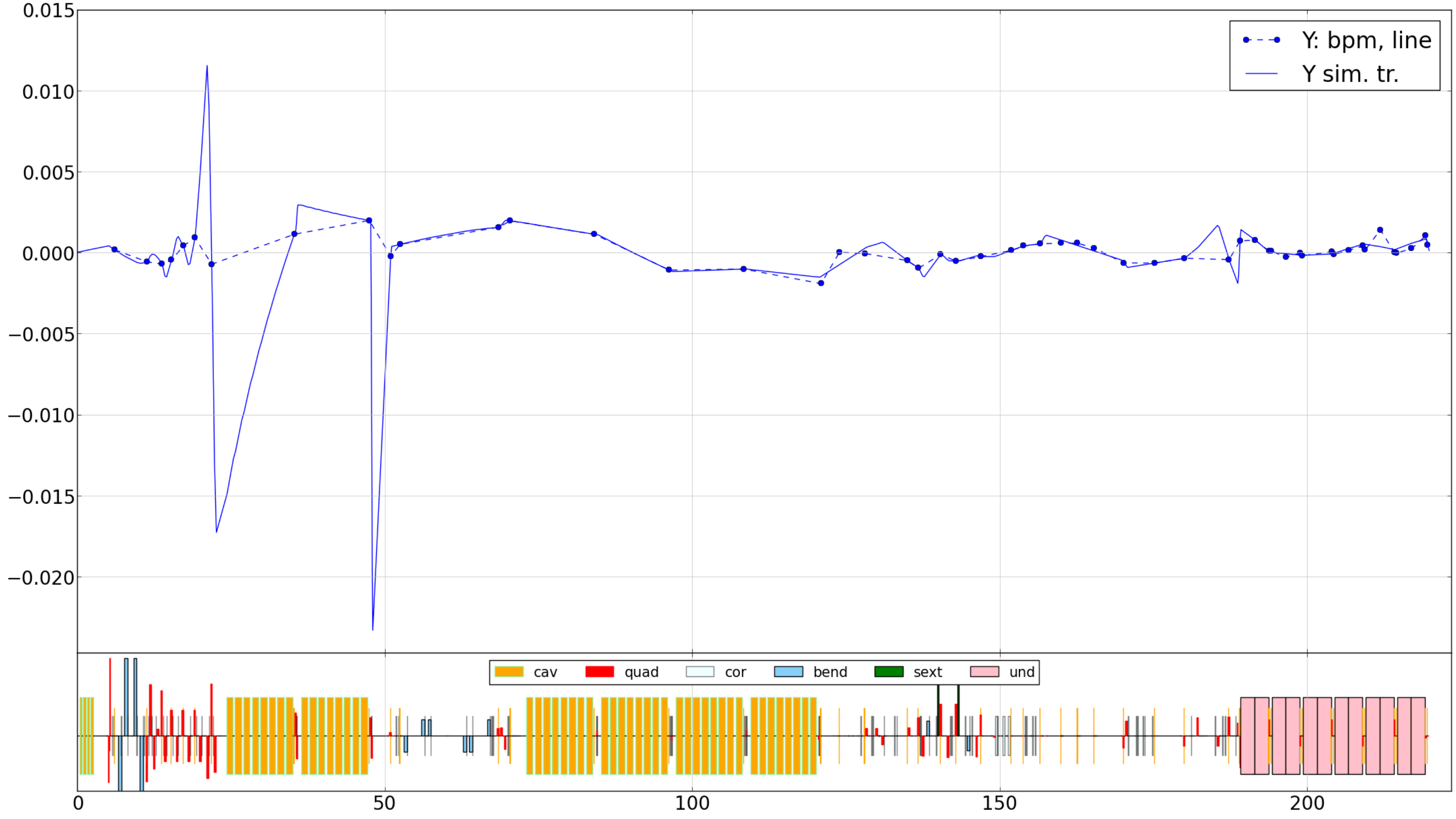
Reading real orbit and fitting beam trajectory

Horizontal plane



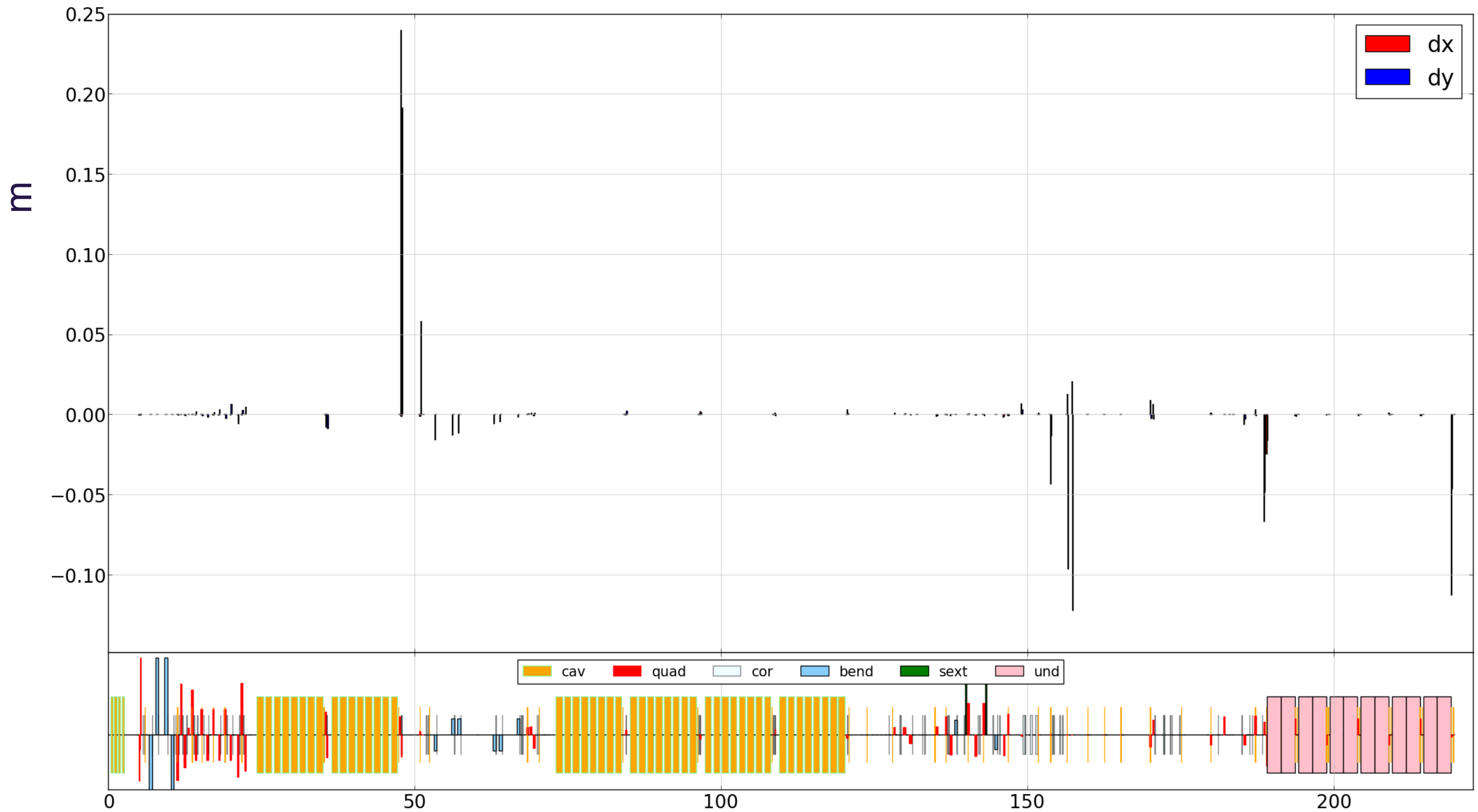
Reading real orbit and fitting beam trajectory

Vertical plane



Reading real orbit and fitting beam trajectory

Quadrupole misalignments



- Creation of realistic model is not simple task.

Summary

- On-line control python tools have been developed
- Demonstrated automatic SASE tuning at FLASH and LCLS
- Proposal for further FLASH beamtime to study more advanced tuning methods has been approved and further developments are ongoing
- Implementation of GUI for FLASH/XFEL in progress (~02.2016)
- Planning of further steps wrt. XFEL.EU is underway. The first step of optimization tool implementation can be beam transmission tuning during commissioning.
- R&D into more complex tuning methods are needed.
- Even the simplest empirical tuning method promises significant advantage for facility availability.

Thank you for your attention