Modeling and optimization of the European XFEL with OCELOT

Sergey Tomin
DESY

CERN BE Seminar, 28 August 2020
Outline

- Introduction and motivation
- OCELOT structure
- Modeling with OCELOT
- Optimization with OCELOT
- Conclusion
The European XFEL

The 3.4 kilometre-long facility runs (in underground tunnels) from the DESY campus in Hamburg to the town of Schenefeld.
**Accelerator Overview**

- **Nominal parameters**
  - Electron beam energy up to 17.5 GeV
  - Pulse rep. rate 10 Hz
  - Bunches per pulse 2700
  - Intratrain rep. rate 4.5 MHz
  - Bunch charge 0.02 – 1 nC

- SC accelerator is able to produce up to 27000 electron bunces per second in burst mode with 10 Hz
- Electron bunches in a single pulse are distributed by a fast kicker system to three SASE undulators

---

*W. Decking et al., “Commissioning of the European XFEL”, https://doi.org/10.18429/JACoW-IPAC2017-MOXAA1*
Motivation
Motivation

The current setup:
- Faster
- More easy to modify
- Can be run by non experts
**OCELOT structure**

- Everything in **Python**. Focus on simplicity. Implement only physics
- Open source (On GitHub [https://github.com/ocelot-collab/ocelot](https://github.com/ocelot-collab/ocelot))

**Photon field simulation**
- FEL simulations (**genesis**)
- Spontaneous radiation
- Wavefront propagation
- FEL estimator

**Charged Particle Beam Dynamics (CPBD) module**
- (linacs, rings)

**Online beam control**
- Orbit correction
- Adaptive FB
- Optimizer

**Data sets**
Modeling of the European XFEL with OCELOT
Modeling of the European XFEL with OCELOT. Beam dynamics simulations

- 6.5 MeV, I ~ A's
- 17.5 GeV, I ~ kA's

OCELOT:
- 3D Space Charge
- 3D wakefields
- 1D CSR
- particle motion with transport matrices of second order
- RF cavity with Rosenzweig-Serafini model
- Chamber with wakes
- ISR and quantum fluctuations

Modeling #1. Beam dynamics for 500 pC, 5 kA

Projected x-emittance growth by 37%
Projected y-emittance growth by 44%
Modeling #1. Beam dynamics for 500 pC, 5 kA

- projected x-emittance growth by 60%
- slice energy spread growth due to quantum fluctuations in SASE2 (K=3.9)
- correlated energy spread growth due to strong wakefields in SASE2
Overview of options for generating high-brightness attosecond x-ray pulses

Evolution of the 2.3 keV radiation emitted by the modulated electron beam.

Current and longitudinal phase space of the 500 pC electron beam at different stages of the XLEAP scheme.

S. Serkez, G Geloni, S Tomin et al, Journal of Optics, Volume 20, Number 2, 024005, 2018
Motivation:
- Measurements show asymmetric beam energy distribution with $\phi_{A1} = 0^\circ$

Simulations:
- Length of the simulated setup (up to the screen) ~ 40 m
- 3D wakefields from A1-, AH1-, TDS-cavities.
- 3D SC along the whole beam line
- Tracking up to 2nd order
- Gun simulations with ASTRA (up to 3.2 m from the cathode)
Modeling and measurement of collective effects in the EuXFEL Injector

$\Delta p_z$ [MeV/c]

- SC & wakes
- measured

$\phi_{\text{gun}} = 2^\circ$

$Q = 250 \text{ pC}$

$\Delta p_z$ [MeV/c]

- SC
- w/o

$Q = 250 \text{ pC}$

Coauthors: I. Zagorodnov, Y. Chen, S. Tomin
What else can you do with OCELOT?
Timing jitter correction for plasma acceleration

Beamline simulations require combination of Ocelot with plasma codes (FBPIC, Wake-T)

Conceptual design for achieving sub-fs timing jitter between laser pulse and RF electron beam for injection into plasma accelerator.


18% RMS energy jitter reduced to 0.7%

Reference: A. Ferran Pousa PhD thesis (to be published)
Accelerator toy for ML studies

GitHub: **Python** is most popular language for Machine Learning

GitHub: Python libraries that used in ML are:
- Numpy
- Scipy
- Scikit-learn
- Theano
- TensorFlow
- Keras
- PyTorch
- Pandas
- Matplotlib

I. Agapov, “Some (possible) ML applications for low-emittance storage rings”. 2nd ICFA Workshop on ML for Particle Accelerators PSI, 27 Feb 2019
Optimization of the European XFEL with OCELOT
Motivation

- Modern Free Electron Lasers are complex facilities with hundreds of free tuning parameters
  - Bunch compression, orbit, beam optics, gun optimization, undulator gaps, phase-shifters, etc

- Even when the main accelerator systems work well, manual fine-tuning is necessary to get the best performance and this is time expensive

Zoo of High Level Control tools

- Beam Based Alignment
- Beam matching
- Longitudinal profile measurement
- Orbit correction
- Scan tool
Optimization algorithm

- Optimization algorithms are faster than scanning
- Optimization methods can be model-independent or model-dependent
OCELOT Optimizer

- OCELOT optimizer is a flexible platform for optimization:
  - Interchangeable optimization methods
  - GUI
    - Add/select device or group of devices
    - Craft/modify target function
  - Infrastructure for testing new methods
  - Save/load configs
  - Logging

- Collaboration

- M.W. McIntire et al, DOI:10.18429/JACoW-IPAC2016-WEPOW055
OCELOT Optimizer: Use cases

- European XFEL
  - FEL pulse energy maximization:
    - Orbit inside an undulator
    - Phase-shifters
    - Orbit in injector
    - Matching quads
    - RF settings

Phase-shifters

- ~400 uJ
- ~800 uJ
- 4 mins
OCELOT Optimizer: Use cases

- European XFEL
  - FEL pulse energy maximization:
    - Orbit inside an undulator
    - Phase-shifters
    - Orbit in injector
    - Matching quads
    - RF settings
  - Local dispersion correction in injector

Before correction

After correction

Laser Heater chicane
In most optimizations 4 devices are used and the average time duration of a single optimization is 3.5 minutes.
Hyperparameters. Beam matching

SASE optimization with 4 quads

- Initial quad step
- Group of quads

Quads tuning in front of SASE1 is one of the steps of standard optimization procedure with Optimizer.

Hyperparameters, such as initial steps, number of iterations, were corrected as experience increased.
Beta mismatch parameter* optimization

- OCELOT beam dynamics module is already used for orbit correction
- Bounds and hyperparameters are defined for all matching sections
- Only 4 actuators are used while number of quads can be more - reduction of dimensionality in some cases

* M. Sands, SLAC-AP-85 1991
Orbit correction tool with adaptive feedback.

- OCELOT orbit correction is the standard tool for orbit correction (using SVD/MICADO algorithms).

- The Adaptive Feedback is a statistical optimizer\(^1\) exploiting the orbit jitter and its correlation with a fast FEL intensity signal (shot-to-shot resolution) to optimize the undulator launch orbit.

\(^1\)G. Gaio, M. Lonza, Automatic FEL Optimization at FERMI, Proc. of ICALEPCS2015
Adaptive Feedback: how it works

Horizontal orbit in undulator

- Golden orbit
- Optimal orbit

FEL pulse energy

Iteration #1
Adaptive Feedback: how it works

Horizontal orbit in undulator

FEL pulse energy

Iteration #1
Adaptive Feedback: how it works

Horizontal orbit in undulator

Golden orbit

Optimal orbit

FEL pulse energy

Iteration #1
Adaptive Feedback: how it works

Horizontal orbit in undulator

FEL pulse energy

*golden orbit*

Optimal orbit

Iteration #1
Adaptive Feedback: how it works

Horizontal orbit in undulator

FEL pulse energy

Iteration #2
Adaptive Feedback: how it works

Horizontal orbit in undulator

FEL pulse energy

Iteration #2
Adaptive Feedback: how it works

Horizontal orbit in undulator

FEL pulse energy

Iteration #2
Adaptive Feedback: how it works

Horizontal orbit in undulator

- Golden orbit
- Optimal orbit

FEL pulse energy

Iteration #4
Adaptive Feedback: how it works

Horizontal orbit in undulator

FEL pulse energy

Iteration #7
Adaptive Feedback: how it works

Horizontal orbit in undulator

FEL pulse energy

golden orbit

Optimal orbit

Iteration #14
Adaptive Feedback

Adaptive FB was started
Conclusion

OCELOT modules and functions for accelerator modeling and optimizing
Conclusion

OCELOT modules and functions for accelerator modeling and optimizing

Available Python modules:
Math., data analysis, ML, plotting, optimization algorithms etc
## Tutorials

https://github.com/ocelot-collab/ocelot

## Description of functions and classes in source files

OR use `help()` in python/ipython/Jupyter

## Read code
Unit-tests

Main functions and modules are covered with unit-tests
But bugs are unavoidable, if found any please report it:
https://github.com/ocelot-collab/ocelot/issues
Thank you for your attention

And thanks to all contributors (the order is random 😊):

G. Geloni, S. Serkez, T. Tanikawa – *the European XFEL*

I. Agapov, M. Dohlus, W. Kuropka, A. Ivanov, Á. Ferran Pousa, I. Zagorodnov – *DESY*

J. Duris, H. Slepicka - *SLAC*

E. Fomin – *Kurchatov Institute*

A. Trebushinin – *BINP (Novosibirsk)*

M. Veremchuk – *Kyiv National University*

....