TOWARDS MORE ACCURATE MODELING OF THE FEL RADIATION FOR THE EUROPEAN XFEL

I. Agapov*, G. A. Geloni, European XFEL GmbH, 22761 Hamburg, Germany

Abstract

For the operation phase of the Eurpean XFEL the possibility to characterize the FEL radiation taking realistic machince model into account is important. To achieve this, a software framework is being developed. It allows for interoperability of various simulation codes by means of a common graphical user interfaces, common input and output files, and common programming model for scripting; includes the possibility of modeling beam jitters and machine imperfections to set errorbars on the simulation results, and has a connection to the control system for data acquisition. We report on the progress in the developing of this framework and give examples of FEL property calculations performed with it.

SOFTWARE OVERVIEW

The requirements for simulation software for commissioning and operation of an accelerator facility are rather different from those posed on software used in the design phase. Whereas the focus in the latter is probably more on rendering the physics processes correctly and showing the feasibility of various designs, the former are required to be suited to the needs of operation automation and performance optimization. A software project refered to as *xframework* (or *xcode* when shipped with 3rd party codes) has been under development for slightly over a year now and first introduced in [1]. It is partially based on the developments presented in [2]. It reflects one of the authors' previous experience in developing the commissioning software [3], [4], [5] and is aimed at commissioning and operation phase of the European XFEL due to start in 2015. Here we report the developments done so far and the development plans for the next years which have emerged after a series of discussions with machine and instrument scientists. Basic requirements are summed up below:

Physics. The diversity of physics phenomena to be simulated – space charge effects, wakefields, single-particle electron optics, spontaneous and FEL radiation, and x-ray optics, – make their coverage by a single code impossible. On the other hand, the major bulk of every simulation code consists of auxiliary routines, and converting between formats takes a considerable effort. A modular software design where physics processes can be added into a common framework whereas the geometry description, statistical analysis and similar functionality is common, is well suited for such situations. Such approaches have been very successful in particle physics [6] and in fluid dynamics [7]. In accelerator and light source community this has not been done, perhaps for reasons of lower simulation needs, and perhaps for other reasons. The authors believe that the next genaration of codes should follos such a modular approach.

Language and flexibility. Capabilities of input languages of programs like *MAD* [8] severely limit the functionality, even if the basic 'building blocks' are there. Such examples include. e.g. simulating beam jitters, ground motion, making parameter scans and calculating response matrices. Interfacing to a high level programming language which can be embedded in the control system is also necessary. (as was done in [3], [4], [5]). *Xframework* adopts python as the main language, so no interfacing is required, and capabilities of an advanced high level programming language are available.

Model. The need to create more and more complicated accelerator models reflecting, e.g. both magnetic field properties and geometric information, has lead to creating rather involved description languages [9], [10]. Languages derived from MAD lack extensibility, whereas XML-based languages lack reasonable control flow such as loops, if-statements, function calls, etc. We have converged on a loose API-level standardization instead. The geometry description is thus restricted to python, but it is a natural language to write converters in, and the amount of parsing is minimal should conversion to other languages be required.

Control system. With the tuning of radiation parameters becoming complex, high-level controls should possibly be based on a similar model as simulations. The tradeoff is of course, that the level of hardware detail needed for controls is typically not needed in simulation tools. Regardsless of if *xframework* will or will not be included in high level online tools for E.XFEL, the possibility to access the control system and data acquisition for. e.g. machine studies, is important. To this end, appropriate interfaces are being implemented. Two major controls interfaces will be available for the XFEL undulators, Doocs and Karabo [11]. In both cases, python binding would be provided. This would allow to buid graphical or command-line steering tools in the same software framework as simulations.

The architectural solution was to provide a rather loose set of classes and functions with basic capabilities.

Model. Set of base classes for standard components like quadrupoles, bending magnets, sextupoles, undulators etc. The level of validation is low (e.g. one cannot supply negative length) and one can introduce custom components or attach additional information to the basic classes in an arbitrary way. Such additional information would be ignored in modules where it is irrelevant, i.e. attaching geometry in

* ilya.agapov@xfel.eu

ISBN 978-3-95450-122-9

any way would not affect the electron beam optics calculations.

Electron beam optics. Uncoupled linear optics function calculations are in place. Nonlinear effects (sextupoles) are included in tracking. Generally, an element transfer map can be altered at any time, even during execution. In such a way elements with arbitrary transfer maps, even not present among the base classes, can be introduced by users. Misalignments are included for tracking calculations – transverse offsets, and tilts. Space charge is not yet included but is foreseen. Using the capabilities of python optimization libraries, matching and orbit correction can be relatively easily introduced as external routines.

Spontaneous synchrotron radiation. Spontaneous synchrotron radiation (SR) is not the primary research tool at an FEL facility, however it's understanding is useful for diagnostics and tuning purposes. E.g., to adjust the model to particular lasing wavelength, SR calculations can be used. For a desired wavelength, SR can also be used to adjust phase shifter settings, undulator gap settings and if necessary, undulator tapering. SR is also used for undulator tuning [12]. Synchrotron radiation capabilities are included in *xframework*, primarily based on but not limited to the SRW library [13].

FEL. Genesis [14] is probably the most widespread FEL code at present and is included, together with python functions to control the run parameters, generate input lattice from the common *xframework* model, and perform parameter scans. Postprocessing functions to deal with the resulting radiation and beam output are also present. A set of semi-analytical FEL models, and a 1D FEL code are also included, and the possibility to include capabilities beyond Genesis is under investigation.

OPTIMIZATION OF RADIATION PARAMETERS

For a SASE FEL there exist estimates of key performance parameters such as output power, bandwidth, etc. in therms of the Pierce parameter ρ [15].

$$\rho = \frac{1}{\gamma}\left(\left(\frac{KA_{JJ}l_w}{8\pi\sigma_b}\right)^2 \frac{I}{I_A}\right)^{\frac{1}{3}} \tag{1}$$

where $\lambda_r = \frac{l_w}{2\gamma^2}\left(1+\frac{K^2}{2}\right)$ is the radiation wavelength, γ the relativistic factor, K the undulator parameter, A_{JJ} the coupling factor, l_w the undulator period, σ_b the beam size, I the peak current, and $I_A = 17KA$. For better estimates, as well as for extracting more involved parameters such as coherence properties, 3D numerical simulations are required. There also exist fitting formulae for various parameters taking more effects into account [16], but they do not necessary cover the whole set of parameters of interest. Optimization based on time-consuming simulations is performed, but its applicability is of course also restricted. But even if the accuracy of analytical estimates is sufficient, or the numerical simulation is arbitrarily fast, more serious issues are that a) the optimization problem cannot always be easily formalized and b) the parameters which enter the optimization are not directly controllable and depend in a non-trivial way on other parameters. For example, consider a question of minimizing the radiation bandwidth which is interesting from the xfel user perspective [17]. The bandwidth at saturation can be to a sufficient accuracy taken as

$$\frac{\Delta\omega}{\omega} \approx 2\rho \tag{2}$$

On the other hand the average power at saturation is

$$P_{sat} \sim I\rho \tag{3}$$

Minimizing the bandwidth is achieved by minimizing ρ which in turn reduces the output power, but what is the optimal value here is not clear per se, but only from the experimental considerations, which might in turn rely on complex simulations. Thus, the opmimization criterion here is difficult to formulate. Now, even if the working point in terms of I and ρ is chosen, one is still free to choose the corresponding γ, K and σ_b. All this parameters can be in theory controlled directly by changing the RF voltage, undulator gaps, undulator optics or the compression level. The question of choosing the working point becomes related to machine operation.

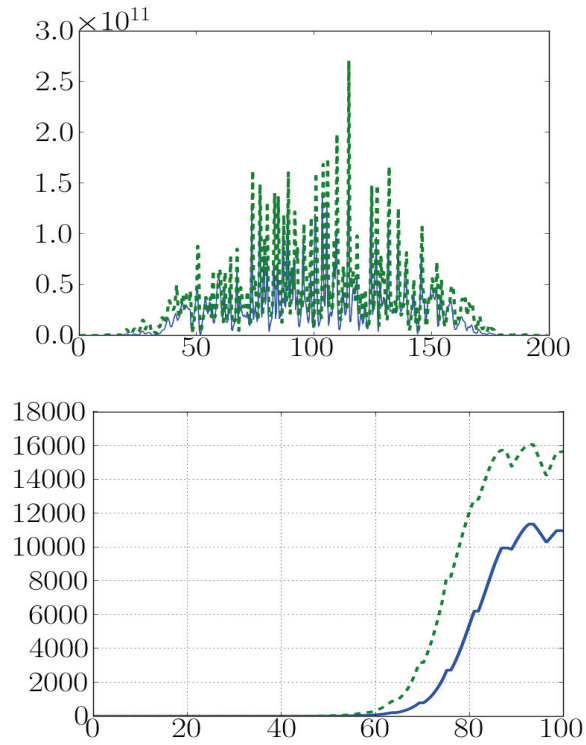

Figure 1: Output signals correspondiong to 4KA (solid line) and 5KA (dashed line) input current, for the same seed, SASE3 undulator.

Even this simplified example shows that optimization is a mostly empirical process which requires human knowl-

edge of device specifics. Furthermore, assuming the accelerator can be perfectly controlled, the optimization target is well understood, and all calculations can be done sufficienly fast, the FEL process is still intrinsically nonlinear as demonstrated in Fig 2 which still could lead to difficulties.

Putting it all together, it seems reasonable to compile a table of possible working points which could help users plan their experiment. The possibilities for choosing a working point for a particular experiment depend on parameters such as intensity, bandwidth, pulse duration and so forth. XFEL parameters calculated with the code FAST are summarized in [18]. It is however desirable to have a more systematic presentation of the baseline parameters, which can be traced back to certain machine settings. Data mining routines could be run on the simulated results to search parameter space, and the parameters used for certain simulations can be recovered in case of questions. To achieve this *xframework* includes a radiation and beam parameter database, with an API to submit simulation results to it and search results in it. The database will consist of a bulk of simulation outputs on a mass storage, and an SQL index database. The simulation results can be processed independently from the input database, and the index database can be later complemented with additional data. The database is linked to a repository of input decks and beam parameters, so that the input used in a particular simulation can be recovered. During the operation phase, the actual machine performance might noticably differ from what is predicted by simulations. The database can be further complemented by parameters inferred from operation. The question if non-trivial information can be produced in such a way is open. Statistical learning techniques [19] often prove of limited value when applied to complex real life systems. A systematic presentation of the best knowledge of radiation parameters to the users can, in contrast, always be aimed at.

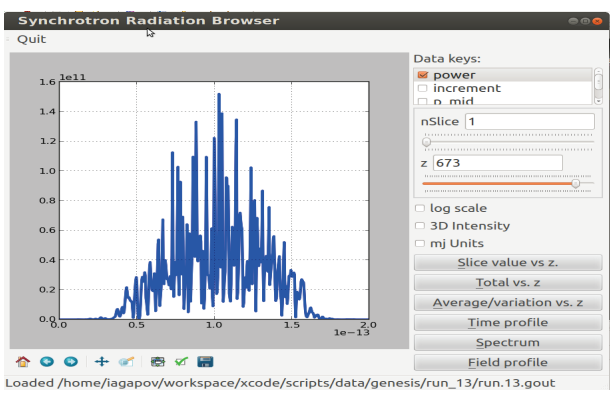

Figure 2: Xframework contains a graphical (Qt4-based) postprocessing tool supporting plotting basic simulation output. More comlicated operations are accessible from python scripts.

ACKNOWLEDGEMENTS

The authors are acknowledging input of S. Tomin and A. Bozhevolnov as part of the xcode development team, I. Zagorodnov for discussing start-to-end accelerator simulations, S. Esenov, V. Rybnikov and C. Youngman for useful discussions on control systems, M. Myer and N. Kabachnik for discussing the experimental issues and S. Molodtsov for support of this work.

REFERENCES

[1] I. Agapov et al., "Physics Simulations ob the DESY HPC platform", proc. ISC 2012.

[2] I. Agapov, "Ocelot", presented at the Seminar on Advances in Electromagnetic Research, KWT 2011

[3] I. Agapov et al., "LHC on-line model", proc. PAC'07.

[4] I. Agapov, "LHC on-line model, design and implementation", CERN-AB-Note-2008-053.

[5] C. Alabau Pons et al., "The Online Model for the Large Hadron Collider", proc IPAC'10.

[6] Geant4, http://geant4.cern.ch/

[7] OpenFOAM, http://www.openfoam.com

[8] MAD-X, http://mad.home.cern.ch/

[9] I. Agapov, "The GMAD accelerator description language". EUROTeV-Memo-2006-002-1, AML, http://www.lepp.cornell.edu/ dcs/aml/

[10] I. Agapov et al., "BDSIM: A particle tracking code for accelerator beam-line simulations including particle-matter interactions", NIM A 10.1016

[11] DOOCS, http://doocs.desy.de, Karabo, an in-house XFEL software framework

[12] M. Tischer et al. " Photon diagnostics for the X-ray FELs at TESLA", NIM A 483 (2002) 418-424

[13] O. Chubar and P. Elleaume, "Accurate and Efficient Computation of Synchrotron Radiation in the Near Field Region", proc. EPAC-98.

[14] Genesis, http://genesis.web.psi.ch/

[15] E. Saldin et al., "Physics of free electron lasers". Springer 2000

[16] E. Saldin et al., DESY-04-012, M. Xie, NIM A 445(2000) 59

[17] M. Myer, Private communications

[18] E. Schneidmiller and M. Yurkov, "Photon beam properties at the European XFEL", DESY-11-152

[19] T. Mitchell, "Machine learning", McGraw Hill 1997

ISBN 978-3-95450-122-9