LHC OPTICS MEASUREMENT AND CORRECTION SOFTWARE PROGRESS AND PLANS

J. Cardona, Y. Rodriguez, Universidad Nacional de Colombia, Bogota, Colombia

Abstract

LHC Optics Measurements and Corrections (OMC) require efficient on-line software applications to acquire and analyze data and to compute the necessary corrections. During Run 2 various measurement and correction techniques have been merged to yield unprecedented optics quality, increasing the required number of steps to finalize the optics commissioning and the size of the software project. In turn, this calls for a higher level of automation, with possible implementation of machine learning techniques. During the Long Shutdown 2 the codes are being largely re-factored to improve performance, maintainability and extensibility. A description of the current status of the software and future plans is given.

DESCRIPTION OF THE SOFTWARE

The OMC software aims at enabling accurate and efficient beam-based optics measurements and corrections on-line. LHC machine safety and performance have been the main drivers for the software development over more than a decade [1–27]. The main structural philosophy is to place the data analysis algorithms in independent Python codes that can be invoked from a command or from GUIs written in Java to be compatible with the LHC controls software, LSA [28]. This structure has allowed to easily adapt analysis codes to other accelerators [19,29–38] and to package them using Docker software for use in other applications [39]. Profiting from the Long Shutdown 2 (LS2) between end of 2018 and end of 2020, a thorough review, extension, and consolidation of software is taking place. Python software is being migrated to Python 3.6 as Python 2 development will stop by the end of 2019 and, more importantly, the Python 2 backwards compatibility is already abandoned for new features in scientific packages such as numpy, scipy, pandas and matplotlib.

Figure 1 shows a schematic view of the ensemble of the OMC applications used in the control room. Multiturn is a Java GUI that controls the transverse beam exciters (AC dipoles, tune kickers or aperture kickers) and acquires the turn-by-turn (TbT) Beam Position Monitor (BPM) data. It automatically executes the Python codes for the first analysis of TbT data, especially to allow for fast coupling correction [40]. Online Model is a set of Java libraries to extract information on the machine settings [13]. TIMBER is the software developed by the CERN Beams Department Controls group to log and access time-series data [41]. Beta-beat and K-mod represent the two core applications to measure and correct optics from TbT data and from quadrupole strength modulations. The Beta-beat GUI written in Java executes appropriate Python codes for analysis and provides interactive view of the results. Optionally, it performs additional data cleaning and submits corrections to the hardware. The K-mod GUI [25], written in Java, modulates quadrupole strengths and records tune data, which are later automatically analysed by Python suite. The Beta-beat GUI imports results from the k-modulation analysis to be included in the optics corrections calculations.

The flow chart within the Python 3 suite corresponding to data analysis and optics calculations is shown in Fig. 2. The TbT data in binary SDDS format [42] is first cleaned. This

* rogelio.tomas@cern.ch

MC4: Hadron Accelerators
A04 Circular Accelerators

Figure 1: Flow chart of software applications involved in the optics measurement and corrections.

Figure 2: Flow chart of the Python 3 suite for harmonic analysis of TbT data and calculation of the optical parameters.
is achieved by removing malfunctioning BPMs and applying noise reduction to the data via singular value decomposition [43]. Harmonic analysis is performed either with the Python code harpy [20] or with the Fortran code Sussix [44]. The harmonic analysis results are output into files with suffixes “linx” and “liny” for horizontal and vertical planes, respectively. Further BPM cleaning can be applied at this stage based only on tune values or on a collection of features applying Isolation Forest [21–24].

The cleaned linx/y files are passed to the Python suite for optics calculations, which is called GetLLM in the Python 2 suite. A sketch of the different modules in the Python 3 suite to compute optics parameters and their dependencies is shown in Fig. 3. Measured optics parameters are displayed in grey boxes including tunes, phase advances, coupling [8], β functions (from phase and from amplitude) [10, 17] at BPMs and at Interaction Points (IPs) [9, 12, 15], dispersion [27], amplitude of the excitation (kick) and resonance driving terms (RDTs) [45–47].

Non-linear measurements and corrections, in particular amplitude detuning and crossing angle scans, require a large human effort. Amplitude detuning is measured by acquiring many AC dipole excitations with increasing strength in the horizontal and vertical planes. The natural tunes represent a small signal in the spectra of the driven TBT data. It is fundamental to use noise-cleaning techniques to uncover the natural tunes. Often, human intervention is required to avoid confusing the natural tunes with spectral lines arising from the non-linear motion. Crossing angle scans are performed to sample the non-linear components of the triplet by measuring their feed-down to tune and linear coupling. The bump non-closure when changing the crossing angle is corrected to avoid contributions from other non-linear components in the machine. AC dipole excitations are also applied at every step of the scan. A variety of extra codes are used for these measurements, which are not fully incorporated into the above main environment. Figure 4 shows the work flow corresponding to these non-linear measurements and corrections with the connections to modules already described.

Table 1 shows the lines of code, the commits in 2018, and the static issues of the different software packages. The incomplete Python 3 suite, currently being migrated from Python 2, is included in the table. A sizable improvement is observed in the static analysis issues.

**DEVELOPMENT PLANS**

A new frequency analysis framework has been developed [48] within the Python 3 showing superior performance with respect to the current software. During Run 2, optics measurements based on 3D beam excitation were experimentally demonstrated [49, 50]. In the LHC, such measurements are an order of magnitude faster compared to standard methods. Dedicated analyses are being implemented in the Python 3 suite.

The measurement of second order dispersion is currently being tested with experimental data from Run 2 [27] and should become fully operational on-line for 2021.

The possibility of improving the linear optics corrections by using machine learning techniques is currently under
Table 1: Lines of code, commits, and static code analysis issues in 2018. Corresponding data during 2017 for a fraction of these codes can be found in [16]. In Python 3 repository, the git squash command is used. Therefore, the actual number of commits would be considerably higher.

<table>
<thead>
<tr>
<th>Language</th>
<th>Lines of Code</th>
<th>Commits</th>
<th>Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Python 2</td>
<td>48007</td>
<td>1214</td>
<td>16051</td>
</tr>
<tr>
<td>Python 3</td>
<td>4196</td>
<td>31</td>
<td>69</td>
</tr>
<tr>
<td>Java (GUIs)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>β-beat</td>
<td>32139</td>
<td>108</td>
<td>82</td>
</tr>
<tr>
<td>Multiturn</td>
<td>20425</td>
<td>20</td>
<td>788</td>
</tr>
<tr>
<td>K-mod</td>
<td>16272</td>
<td>32</td>
<td>43</td>
</tr>
</tbody>
</table>

investigation [51]. First promising results suggest first applications in Run 3. Strength limitations in the quadrupolar circuits might appear when operating at 7 TeV, requiring new algorithms for the calculation of corrections. True local observables of linear imperfections are being developed in [52] in view of the 2021 commissioning.

During Run 2 various limitations in the current algorithms for the measurement of IP optics parameters were observed and new techniques were successfully tested [53]. One improvement consists in using the existing high-resolution DOROS BPMs [54] in the IRs to measure the minimum $\beta$ function near the IP via phase advance. Currently, these BPMs are not included in the regular acquisition requiring new developments. Luminosity scans will be fundamental for accurate measurement of the betatron waist displacement from the IP [53] and for verifying the linear coupling correction at the IP. Studies are on-going to explore how these tasks could benefit from existing operational software for luminosity optimization and scanning.

An experimental demonstration of RDT correction was accomplished in [45–47]. The actual software to perform this correction on-line needs to be developed for 2021.

Automation of crossing angle scans and amplitude detuning is being investigated. This includes monitoring and correcting tune and orbit jitters of the machine [55, 56]. A challenging target for Run 3 is to fully validate IR dodecapole corrections as a first step towards the even more complex HL–LHC non-linear commissioning in Run 4 [56].

With the increasing use of Python in LSA, it should be possible to establish new communications between the Python suite and the machine, that is currently done in the Java GUIs. More efficient solutions for data management concerning input data and output results should be explored to allow staged analyses and easy access to stored measurements.

REFERENCES
