FEED-FORWARD IN THE LHC

M. Pereira, X. Buffat, K. Fuchsberger, M. Lamont, G.J. Müller, S. Redaelli
R.J. Steinhagen, J. Wenninger, CERN, Geneva, Switzerland

Abstract

The LHC operational cycle is comprised of several phases such as the ramp, the squeeze and stable beams. During the ramp and squeeze in particular, it has been observed that the behaviour of key LHC beam parameters such as tune, orbit and chromaticity is highly reproducible from fill to fill. To reduce the reliance on the crucial feedback systems, it was decided to perform fill-to-fill feedforward corrections. The LHC feed-forward application was developed to ease the introduction of corrections to the operational settings. The LHC Feed-Forward software has been used during LHC commissioning and tune and orbit corrections during ramp and squeeze have been successfully applied. As a result, the required real-time corrections for the above parameters have been reduced to a minimum. In parallel, successful trials have been made to apply feedforward corrections before commissioning with beam which are based on MAD-X simulation scans over the unused setting functions. In this paper we present the evolution of feedforward for the LHC and discuss further improvements of this software.

INTRODUCTION

The performance of the LHC is highly dependent on the capability to keep under control some key beam parameters (tune, chromaticity, orbit, ...). These parameters must be kept within acceptable limits to allow safe and efficient operation of the accelerator. Early conclusions pointed out that this challenging task would require the implementation of a real-time control system for satisfactory operation of the LHC. Such system, also known as feedback, would correct perturbing effects on the beam reducing beam loss inside the accelerator. However, the prevailing idea was that the mitigation of external effects on the beam should be done by both feedback and feedforward control [1]. The feedforward control would use the past experience from previous fills and apply corrections to the subsequent ones, eliminating some of the observed effects. The feedback systems would then correct the remaining effects through real-time corrections.

Feedforward is seen as a succession of corrections being applied which become progressively smaller, provided a certain degree of reproducibility exists. Consequently, the measurements obtained will gradually tend to the expected result. In the LHC, feedforward corrections are very similar. Corrections are retrieved from measurements of previous fills and then applied to settings to be used in subsequent fills. Currently, both feedback and feedforward control have been successfuly implemented ensuring safe and

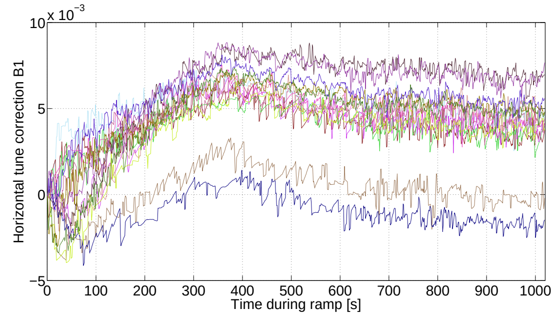

Figure 1: Feedback correction on beam 1 tune on the horizontal plane over 15 fills.

performant operation of the LHC. This paper presents implementation details of the LHC Feedforward application, results obtained and discusses future improvements.

FEEDFORWARD IMPLEMENTATION

Feedback System

Real-time corrections are typically based on a feedback loop. In the case of the LHC one single feedback controller is responsible for correcting tune, chromaticity, orbit, coupling and energy [2]. The hardware receives measurements from the beam position monitors and tune diagnostics systems and calculates the corrections. These are then sent to corrector circuits and RF systems which should stabilize the beam around the established reference value for the beam parameter being corrected. The input data and corrections of the feedback controller are then re-published to be logged in the Logging Database or to be reused in other applications.

Reproducibility

The success of the feedforward correction approach is highly dependent on the reproducibility of the machine on a fill to fill basis. In the case of the LHC, the high level of reproducibility was evident early in beam commissioning. The reproducible behaviour was first observed in tune measurements during ramp. Through the analysis of successive tune corrections made by the feedback systems it was evident that the discrepancy between measurements was quite small. Figure 1 compares feedback corrections of beam 1 tune in the horizontal plane over 15 fills.

The high reproducibility of the LHC came as a pleasant surprise to those who before had seen quite the opposite in LEP. The lack of reproducibility in the machine was evident and only after 8 years of operation the first feedforward corrections were made.

Logging Database

The Logging Service [3] is responsible for logging all major activity related to the accelerator. This includes a wide range of heterogeneous data ranging from cryogenic temperatures and magnetic field strengths to beam positions and intensities. Data queries are performed either through a dedicated Java API or a desktop tool called *TIMBER*. While the former allows other software applications to access data programatically, the latter is aimed at users who want to explore and visualize the data or eventually export the data to a file.

Smoothing & Sampling

Real-time corrections stored in the Logging Database cannot be directly introduced into LSA. The feedback system logs data at a frequency of 1Hz which translates into a large number of data points that LSA simply does not support. Moreover, the data contains a reasonable amount of noise, which if not removed, would most certainly produce erroneous trim functions. Smoothing is a well known method used in signal processing for mitigation of noise and capture of patterns in data.

There is a wide variety of algorithms but the most commonly used is the ”moving average”. Two variants have been tested during the development of feedforward software: simple or unweighted moving average (*SMA*) and the weighted moving average (*WMA*). The *SMA* is in essence an average value based on a window calculated with the arithmetic mean of subsequences of *n* terms. This average is calculated over an equal number of data points on either side of a central value. Consider the following example of a horizontal tune correction with 10 data points ($P_0, P_1 ... P_9$) and a 5 point averaging window. The points in the extremities are kept intact in the resulting function. The points P_1 and P_8 become the average of their value and of their adjacent points (P_0, P_2 and P_8, P_9 repectively). For all other points the formula is:

$$SMA = \frac{P_{m-2} + P_{m-1} + P_m + P_{m+1} + P_{m+2}}{5}$$

After smoothing, the correction function goes through a sampling process. As the power converters in the LHC cannot ”follow” setting functions for current with data point spacing inferior to 0.1s, caution is needed. Bad data point positioning typically causes a power converter to trip. In the present configuration the LHC Feedforward application will smooth the trim correction using an unweighted moving average of 5-point sliding window and sampling every 10 seconds. This has been proved to be more than enough to efficiently remove signal noise and provide enough data points for setting corrections.

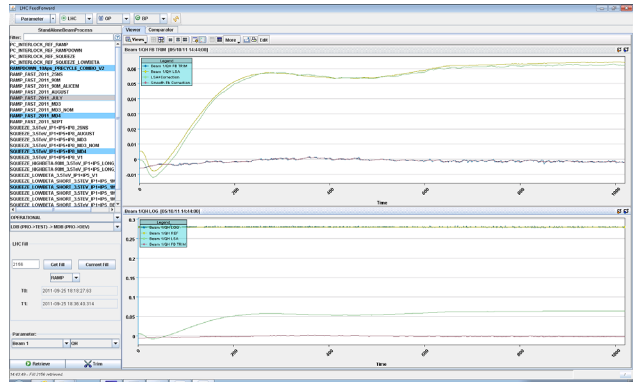

Figure 2: LHC Feedforward application.

LHC Software Architecture

The LHC Software Architecture (LSA) is a framework implemented in Java that covers all essential aspects of control of the LHC such as generation, modification and management of settings, measurements and hardware exploitation. This and other functionalities are made available through a set of clean and well defined APIs divided by domains (optics, parameters, contexts, etc). LSA revolves around three key concepts: *parameter*, *context* and setting. Parameters are organized in a hierarchical structure with each level being dependant on the level above. Thus, changes made on top level parameters are propagated down the hierarchy until the lowest level. The parameters on the top level typically represent physics-related concepts namely tune, chromaticity or orbit, while the lower are related to hardware i.e. power converters. Operators and experts commonly only manipulate top level parameters known as *knobs*. LSA automatically reflects those changes to the lower-level parameters using algorithms called *make rules*. Parameters can have different values depending on the time period considered. This relationship between a parameter value and the time period is called a *context*. There are three types of contexts, *supercycle*, *cycle* and *beam process*. For the LHC, a *context* is called *beam process* and several have been created for the different machine phases: injection, ramp, squeeze, etc.

LHC Feedforward Software

As aforementioned, the LHC Feedforward application has been developed to ease the computation of trim corrections from measurements and their merge with operational settings. It is a standalone application and it is presently available in the CERN control room. The application has been developed using the standard set of technologies for LHC related software: Java, Spring [4] and Swing. Figure 2 shows the interface of the LHC Feedforward application.

At the moment the application is able to perform feedforward corrections on tune, chromaticity and orbit for ramp and squeeze beam processes.

Feedforward corrections on a particular parameter consist of a simple three step process enumerated:

- Logged feedback corrections for the said parameter are retrieved from the Logging Database through its Java API.
- Measurements undergo a smoothing process to eliminate possible noise. Only a subset of these data points are used for the feedforward correction.
- Corrections are applied to the operational settings of the selected parameter using the LSA API.

From the implementation perspective the application is divided into three logical components. The DAO layer is responsible for the communication with both Logging and LSA databases. Direct database access is not possible for security reasons and therefore queries must be done through the already mentioned Java APIs. The business layer contains the correctors and the smoothing algorithms. There are three *correctors*, each one responsible for the calculation of a correction function of a particular parameter, tune, chromaticity or orbit. From the Java implementation perspective these correctors are in separate classes implementing a common Java interface. Hence, *correctors* are homogeneous in functionality and the implementation of a new corrector for a new beam parameter simply requires the implementation of the corrector Java interface. The user interface was created in Swing reusing some of the available standard components from LSA. The interface is divided in three subpanels: a panel for the selection of the beam process to which corrections will be applied; a panel for the selection of the parameter to be corrected and the fill from which the feedback corrections will be taken; a visualization panel where current parameter settings, the proposed correction and the corrected settings are displayed. Additionally, a rudimentary mechanism for comparison of settings and measurements has been implemented. Although quite limited in functionality it allows some brief analysis of fill and settings data.

SIMULATION BASED FEEDFORWARD

In addition to the feedforward of corrections calculated from logged feedback correction data, a second path was used to reduce the stress on the feedback systems. For the optimization of the duration of the nominal squeeze [5] tools have been developed in the toolchain for online modeling [6] that allow to transfer the power converter (PC) K parameter settings to MAD-X and recalculate the optics functions. After the settings are generated in a beamprocess for the LHC the online model framework application, `Beamprocess Scanner` is used to calculate the optics functions from the settings extracted at a given time in the PC setting functions. Among other features, the evolution of the optics key parameters (tune, chromaticity,...) can be plotted and stored. The value functions are inverted and applied as correction trims to the available highlevel parameter knobs.

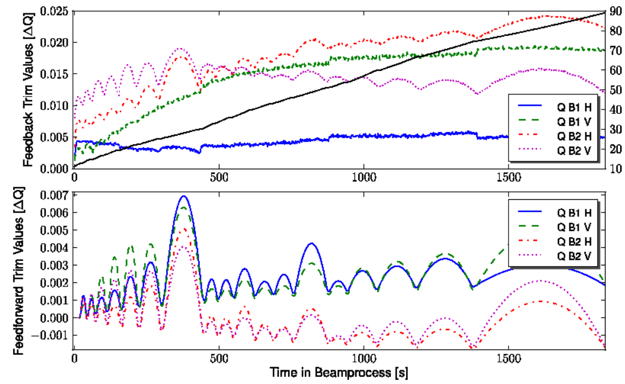

Figure 3: Evolution of the tune knob trims along the 90m un-squeeze. The upper plot shows the real time trims of the tune feedback applied during the un-squeeze, the lower the calculated feedforward corrections from simulation (only the beam 1 corrections were applied). The β* in IP5 from 11 to 90m is shown in the upper plot as well.

This procedure was successfully applied for the tune of beam 1 during the commisioning of the 90m Un-Squeeze [7]. Figure 3 shows the comparison of the tune correction trims required by the feedback system and the proposed corrections from the simulation run. The profit of the feedforward can be clearly observed by the nearly flat real-time trims for beam 1 in the horizontal plane. Real-time trims very similar to the proposed feedforward correction were required for beam 2 where no feedforward had been applied.

RESULTS

The first real test of feedforward corrections was during the commissioning of the first 1.18 TeV ramp. The first ramp trial had already failed due to major tune shifts and the tune feedback was not operational. These were the perfect testing conditions to prove the worth of feedforward corrections. Despite the software being immature and the algorithms not optimized, the accelerator successfully completed the ramp. Since then corrections for chromaticity and orbit have been commissioned for ramp and squeeze. The development focused first on the ramp process which was the more problematic. The next beam parameter to be corrected was chromaticity. Ever since the initial commissioning only one feedforward correction was needed until the present day. Finally, orbit trims have been performed both in ramp and squeeze. Support for orbit corrections was more complex due to the number of parameters involved. Corrections are introduced in LSA at the K level since no *knobs* have been defined. Corrections have been successful although very infrequent.

Figure 4 shows the reduction in tune feedback correction after feedforward corrections had been applied to ramp settings.

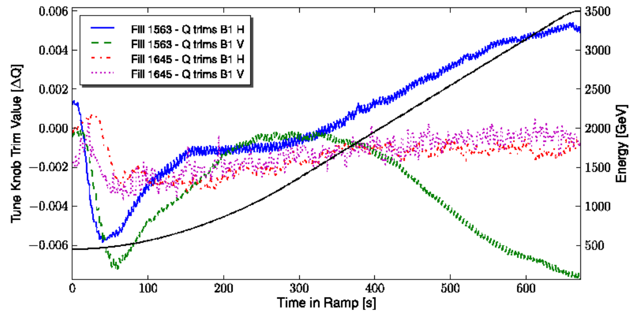

Figure 4: Comparison of the evolution of the tune feedback trims for two fills: 1563 before feedforward and 1645 after feedforward.

CONCLUSIONS & PROPOSED ENHANCEMENTS

The LHC Feedforward application can currently perform corrections during the ramp and squeeze for tune, chromaticity and orbit. The software has been successfully used during 2010 and 2011 achieving the expected results. In parallel, the real-time feedback system has been performing well and was used during every ramp and squeeze in the past year. Hence, feedforward corrections haven't been pursued rigorously and are only performed when new beam processes are created. In addition, due to the high reproducibility of the machine very few corrections were needed throughout 2010 and 2011. However, one should not underestimate the importance of feedforward corrections since they ensure that the beam is not lost in case of feedback malfunctions has occasionally occurred. The simplicity of the LHC Feedforward from an implementation standpoint should be credited not only to the remarkable reproducibility of the LHC but also on the exceptional capabilities of the LSA system. One of the major shortcomings of the application is its limited capability for comparison of parameter corrections and settings over the course of time. This feature would facilitate analysis the effect of feedforward corrections over several fills.

Morever, the application requires a manual selection of the beam process to be corrected, an action prone to mistakes. At the moment of this writing a new feature has been added to the LSA API which allows the retrieval of the name of the beam process used during a determined time period. The integration of this functionality in the LHC Feedforward application would automatize the selection of the beam process and lessen the possibility of mistakes.

REFERENCES

[1] T. Wijnands et al., *Requirements for Real Time Correction of Decay and Snapback in the LHC Superconducting Magnets*, Proceedings of EPAC'00, 2000.

[2] R.J. Steinhagen et al., *Commissioning and Initial Performance of the LHC Beam-based Feedback Systems*, Proceedings of IPAC'10, 2010.

[3] C. Roderick, *The LHC Logging Service - Capturing, storing and using time-series data for the world's largest scientific instrument*, Proceedings of UKOUG'06, 2006.

[4] Spring Source, http://www.springsource.org/

[5] X. Buffat, G. Müller, et al., *Simulation of linear beam parameters to minimize the duration of the squeeze at the LHC*, Proceedings of IPAC'11, 2011.

[6] G. Müller, S. Redaelli, et al., *Toolchain for online modeling of the LHC*, these proceedings.

[7] S. Cavalier, G. Müller, et al., *90m Optics Commissioning*, Proceedings of IPAC'11, 2011.