LATTICE TUNING AND ERROR SETTING IN ACCELERATOR TOOLBOX

S. M. Liuzzo, N. Carmignani, L. Farvacque, B. Nash, ESRF, Grenoble, France

Abstract

New lattice designs need to be studied in the presence of magnetic and alignment errors and appropriate lattice tuning procedures. For this reason a set of tools to perform a commissioning-like sequence has been developed for the ESRF-EBS [1],[2] upgrade in Accelerator Toolbox (AT) [3] and is now generalized to be used for other accelerators lattice design. The functions presented here allow to correct first turn trajectory, orbit, tune, chromaticity, optics and coupling, in any order. A set of functions to define errors is introduced to address, among others, the issues of: misalignment of magnets modeled by several slices, multiple errors setting on the same magnet and spatially recursive errors along the lattice.

INTRODUCTION

We present in this paper some tools added in the AT 1.4 [4] pubtools as first steps to a more complete and exhaustive set of tools for error setting and correction of lattices. The tools presented have been used for the definition of error tolerances for the ESRF-EBS upgrade

ERROR SETTING

Only the additional functions introduced in version 1.4 are described below.

Longitudinal alignment errors Longitudinal alignment errors require to change the length of adjacent drift spaces. For Dipoles also a change of trajectory needs to be considered. Figure 1 shows a lattice layout with a displaced dipole (pale blue). The length of the lattice changes when displacing longitudinally dipoles.

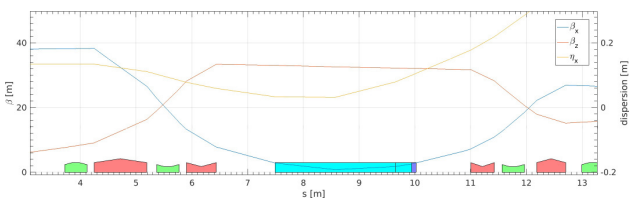

Figure 1: Dipole longitudinally displaced.

Rotation of a dipole about the s-axis Dipoles change the reference frame but their field is usually ignored. To make the effect of dipole rotation and field errors visible, those must be expressed in terms of *PolynomA* and *PolynomB*. The function *atsettiltdipole* implements this feature, and sets the field PolynomB and PolynomA in a dipole to represent the distortion introduced by the rotated bending magnet. In the figures below the effect of the rotation of a dipole are shown in three cases: rotation of a straight multipole using *atsettilt* (Fig. 2), rotation of a dipole using *atsettiltdipole* (Fig. 3), rotation of a dipole using *atsettilt* (Fig. 4).

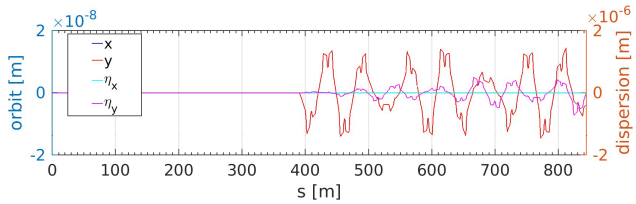

Figure 2: Dipole modeled as kick (PolynomB(1) not zero) rotated using *atsettilt*.

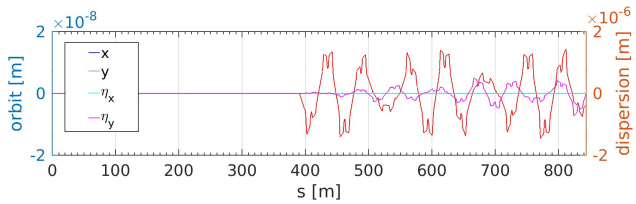

Figure 3: Dipole modeled as bend (BendingAngle not zero) rotated using *atsettiltdipole*.

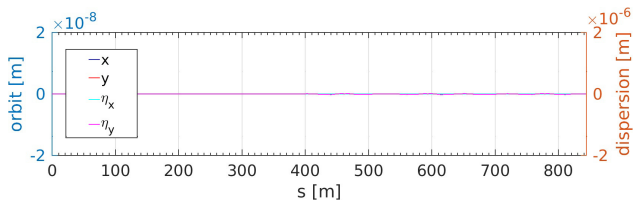

Figure 4: Dipole modeled as bend (BendingAngle not zero) rotated using *atsettilt*. (vertical dispersion is non zero after the rotated bend).

There is no difference between Fig. 2 and Fig. 3, while the tilt implemented rotating the reference system does not show the expected orbit distortion. The same considerations are true for the rotation of a combined function dipole-quadrupole.

BPM errors BPM errors are: offset, rotation, gain and reading precision (random). Those are set in the lattice with the function *atsetbpmerr*. Figure 5 shows an example of closed orbit and BPM readings. To obtain BPM readings including the errors the function *findorbit4Err* and *findorbit6Err* are used. Future AT versions will try to implement this in a dedicated PassMethod in C, to increase the speed of this transformation and avoid the use of extra functions for orbit with errors.

ISBN 978-3-95450-182-3

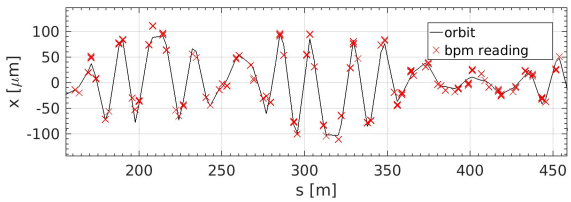

Figure 5: Closed orbit and closed orbit reading with BPM errors.

Several errors in the lattice In most cases several errors need to be set in the lattice simultaneously. The function *atsetrandomerrors* accepts a more structured input to simplify the specification of long lists of errors. A common seed is specified in order to have deterministic error sets. In Fig. 6 a sample set of errors: girder errors in the horizontal, vertical plane and rotation about s, are specified and set simultaneously. The functions also considers splitted elements (often defined to obtain optics at magnets centers, or for longitudinal gradient elements) grouped using a flag named *MagGroup* in each element of the AT structure fields.

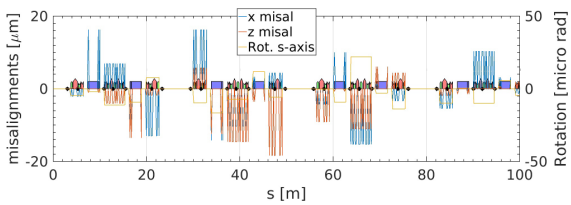

Figure 6: List of girder errors set using *atsetrandomerrors*.

Survey errors Measurements of the magnets positions along the lattice (survey) are routinely performed at ESRF. Knowing these measurement and their errors, it is also possible to simulate survey curves. This curves are a better estimation of the global alignment errors, and can eventually replace the girder-to-girder errors. If no measurements are available, studies of errors can be performed using spatially recursive errors (waves), defined as sum of spatial waves of given amplitude and frequency. The function *atsetwaveerrors* allows this studies and grants that the generated error pattern is continuous. These errors can be very large, but as they are smoothly varying along the lattice, their effect is limited. BPM alignment errors (T1,R1,...) are used in *findorbit4Err* and *findorbit6Err* to offset the BPM reading. This defines a reference closed orbit that follows the global misalignment curves (Fig. 7).

Multipole errors Multipole errors may be systematic or random. To set on a given magnet a series of multipole errors, in AT it is sufficient to define them in PolynomB and PolynomA, and set accordingly the MaxOrder field of the element. However care has to be taken since different conventions might be used for the magnetic field decomposition.

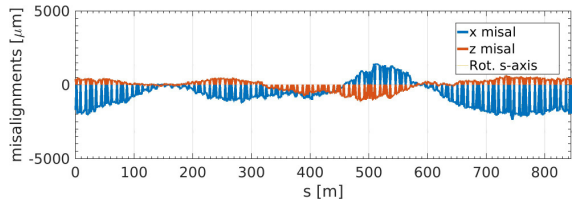

Figure 7: Survey Errors.

The field decomposition in AT is:

$$B = B\rho * \Sigma_{n=1}^{\infty}((b_n + i * a_n)(z)^{(n-1)})$$

where b_n and a_n are PolynomB and PolynomA while for the magnet design group of ESRF is:

$$B = B_N(\rho_0) * \Sigma_{n=1}^{\infty}((B_{N,n} + i * A_{N,n})\left(\frac{z}{\rho_0}\right)^{n-1})$$

where ρ_0 is the reference radius and $B_{N,n}$, $A_{N,n}$ the multipole components. The function *AssignFieldErr* does the conversion and sets the multipole errors, scaling with the main field.

CORRECTIONS

This section describes several possible correction tools. All the functions are based on the pseudo-inversion using SVD (available in matlab) of an adequate Response Matrix (RM). All functions rely on orbit, trajectory and dispersion computations in 6D. Also BPM errors are always included in the simulations. The functions used for this purpose are:

- *findtrajectory6Err* : linepass starting at given initial coordinates + BPM errors.
- *findorbit6Err* : Closed Orbit Distortion in 6D (COD)
- *finddispersion6Err* : moves RF frequency to observe orbit variation off energy (relies on the *RFCavityPass* Integrator to change RF frequency)

The functions implemented for correction (detailed below) are:

- *atfirstturntrajectory* : correct first turn trajectory
- *atcorrectorbit* : correct closed orbit distortions using steerers
- *atcorrectdispersion* : correct dispersion using quadrupoles
- *atdispersionfreesteering* : correct COD and dispersion using steerers
- *atRDTdispersioncorrection* : correct Resonance Driving Terms (RDT) and dispersion using quadrupoles

Also fitting functions can be implemented, but are often very dependent on the fit choices and on the available computing clusters. So they are not included here.

Response matrices

All response matrices are computed by a common function that outputs the structure collecting the responses that

is used by all correction functions. The function *getresponsematrices* centralizes the computation of all the matrices. A vector of integers is used to specify which matrices to compute. All functions have implemented a default RM computation calling *getresponsematrices*. The output structure *ModelRM* contains all the computed RM in different fields. All RM are normalized by the kick used for the computation. The responses are the difference between a perturbation applied in the negative and in the positive plane ("two sided").

Orbit

Orbit correction is performed using *atcorrectorbit*. The function implements several features for correction:

- the average steerers strengths is imposed to be 0 (additional column in RM),
- iteration of the correction varying the number of eigenvectors
- correction of the frequency (as in *atRFcorrection*).
- possibility to limit the steerers strengths
- closed orbit bumps

Dispersion

Dispersion correction is performed using *atcorrectdispersion*. The functions implements all features of orbit correction but uses quadrupole correctors

Dispersion-free Steering

The function *atdispersiofreesteering* uses the steerers to simultaneously correct orbit and dispersion (dispersion free steering [5]. The function implementation is very similar to the one of orbit and dispersion correction, but adds a weight parameter to tune the relative correction of orbit and dispersion.

Correction of RDT

Following [6] we implement in AT the correction of normal and skew quadrupole resonance driving terms (RDT) to tune optics and coupling. This kind of correction is suited for the correction of a fitted lattice model that includes normal and skew quadrupoles errors. The functions *atskewRDTdispersioncorrection* and *atnormalRDTdispersioncorrection* retrieve the normal and skew quadrupole components in the lattice and compute the RDT values. Then the real and imaginary part of those terms are corrected in a system with dispersion and tunes (only normal quadrupoles). An example of the application of the correction is shown in Fig. 8 for skew quadrupoles. This technique is routinely used at ESRF for optics and coupling correction.

Open Trajectory Correction

If the errors set in the lattice are too large, it is possible that no COD is found by AT. This problem is often found in real commisioning, when the first turn is not granted. The function *atfirstturntrajectory* finds an initial closed orbit by using the available trajectory at BPMs. The algorithm follows these steps:

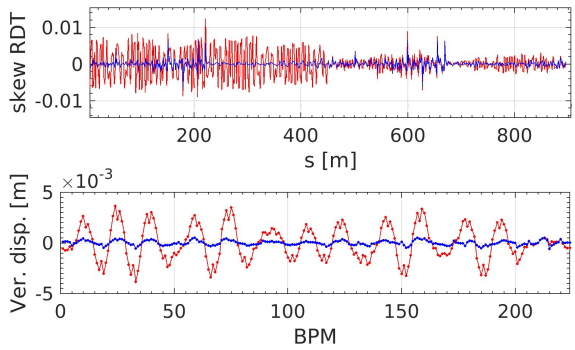

Figure 8: RDT and Dispersion correction using *atRDTdispersiocorrection*. Dispersion weigth = 0.8.

1. look for all BPMs with signal below a given threshold (ex: 3mm) and correct the trajectory using a response computed from the model without errors.
2. if all BPM see a signal, close the trajectory using the last 2 correctors in the lattice to match the reading of the first 2 BPMs.
3. if the correction fails to have signal at all BPMs, try to increase the threshold up to +1mm
4. if still no signal at all BPMs, look for an optimized injection point.
5. if still failing to get to the end of the lattice, compute trajectory response on lattice with errors.

In most cases the first three steps are sufficient to find a closed trajectory. The number of correctors used can be limited to speed up the search. Figure 9 shows an example of trajectory in the horizontal plane before and after the search.

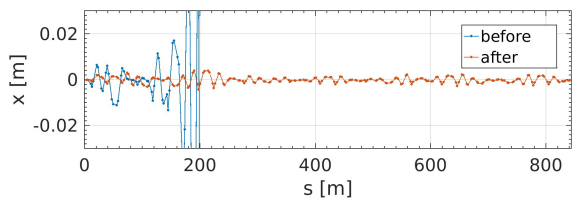

Figure 9: Trajectory correction using *atfirstturntrajectory*.

Commissioning Like Correction Sequence

The above correction are often requested in a sequence. The function *CorrectionChain* allows to perform a full correction sequence calling the above functions from a single entry point.

CONCLUSION

Several additional features for error setting and correction have been listed and can be found in AT 1.4 pubtools. These functions are being developed to be later introduced in AT 2.0, expected before the end of 2017.

ISBN 978-3-95450-182-3

REFERENCES

[1] J.C. Biasci et al. , "A low emittance lattice for the ESRF", Synchrotron Radiation News, vol. 27, Iss.6, 2014.

[2] "ESRF upgrade programme phase II", ESRF, December 2014.

[3] B. Nash, et al., "New Functionality for Beam Dynamics in Accelerator Toolbox (AT)", MOPWA014, IPAC'15, Richmond, Virginia,USA (2015).

[4] `https://sourceforge.net/p/atcollab/code-0/HEAD/tree/`

[5] R.Assmann, et al., "Emittance optimization with dispersion free steering at LEP", Phys. Rev. ST Accel. Beams 3 , 121001 (2000).

[6] A. Franchi, et al., "Vertical emittance reduction and preservation in electron storage rings via resonance driving terms correction", Phys. Rev. ST Accel. Beams 14, 034002 (2011).