New functionalities in AppLE.py

Prepared by:

Afnan SHATAT

Under the Supervision of:

Alexander Gerbershagen

Dipanwita Banerjee

Gian Luigi D'Alessandro

This Report is submitted for the results of CERN online project.

August, 2020

Orsay, France

Contents

Abstract

A Graphical User Interface tool has been developed at the CERN EN-EA-LE section, called AppLE.py (*App*lication for *L*iaison with *E*xperiments), which is written in python and uses MADX in the backend. AppLE.py is designed for beamline simulation of beam parameters as well as particle tracking, with an addition of an improved matching algorithm.

In this project, new functionalities are added to AppLE.py such as 3 types of beam monitors and 3 types of kickers. Also, the energy loss of particles due to synchrotron radiation was manually calculated.

Chapter 1

Introduction

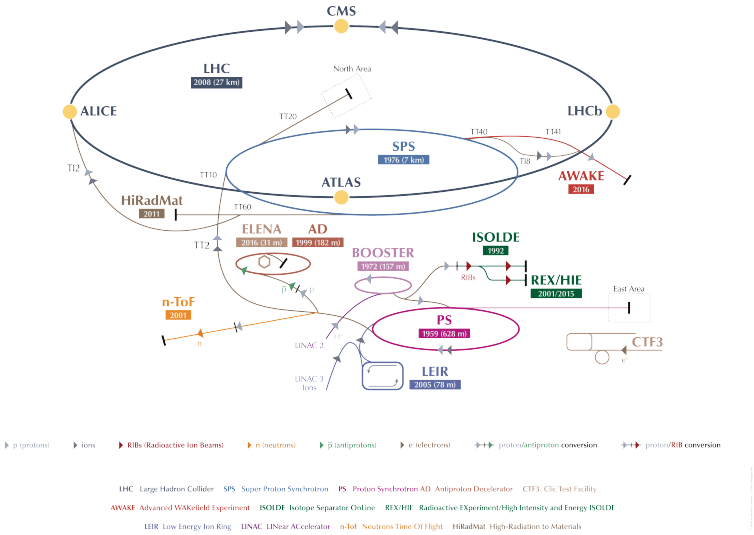

Figure 1.1: Schematic of CERN experimental areas. It includes both primary and secondary beams.

At CERN, the European Council for Nuclear Research, beams of particles are boosted by a series of accelerators including the proton synchrotron and the super proton synchrotron. Many fixed target experiments are setup at the end of the accelerated beam, other than the large hadron collider-LHC. These experiments provide a large variety of particles types and momenta, in addition to a facilitated access and use in beam tests for different purposes. They are mainly located in the North and East area of CERN. In which the section EN-EA-LE (ENgineering-Experimental Areas- Liaison with Experiments) activities are focused on, in addition to test beam activities. This section is responsible for

the design and control of the secondary beams in the experimental areas and other test beam facilities in addition to coordinate with all users.

1.1 Secondary beams

Most of the experimental areas use secondary beams. Secondary beams are the beams produced after a first fixed target interaction. The resultant particles have to be collected. For further use, which is the secondary beam, the beam particle type and its momentum are selected. After that, it will be transported to the target experiment. Both momentum selection and transportation happens with the help of bending magnets and quadrupoles.

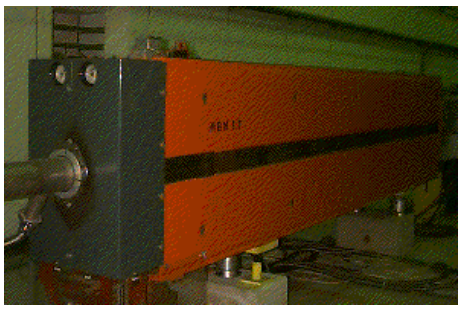

Figure 1.2: Example of a bending magnet element.

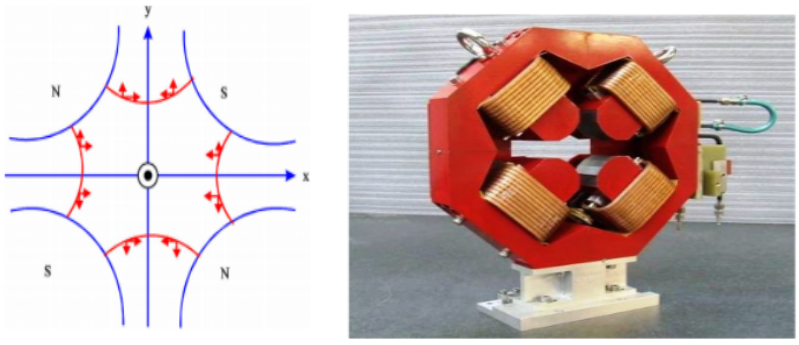

Figure 1.3: At left, a schematic of a quadrupole focusing in a plane and defocusing in the other plane. At right, an example of a quadrupole element.

Many useful beam elements are used for particle selection, momentum selection and beam transportation to an experiment, such as the bending magnets, quadrupoles, collimators and other beam instruments, as follows:

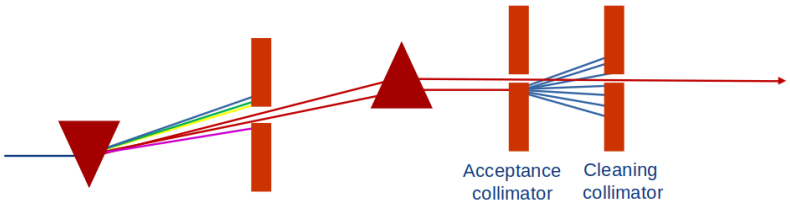

Figure 1.4: An example of collimators used as limiters to the beam size and as a particle selector. The triangles represent bending magnets. The beam starts from left.

1. Bending magnets and kickers are dipole electro-magnets, as in Fig. 1.2. A magnet of length L and a magnetic field of B deflects a particle of momentum p by an angle θ, as follows:

$$\theta[rad] = 0.29979.\frac{BL[T.m]}{p[GeV/c]} \tag{1.1}$$

The higher the particle momentum, the less deviated it is. Following this character, it can be used to select particles with specific momenta. The difference between bending magnets and kickers is its purpose of use. Kickers are used to correct the beam direction, so that it is usually shorter with less strength than the bending magnets.

2. Quadrupoles are magnets with 4 poles, that work as a focusing lens in a plane and defocusing lens in another orthogonal plane at the same time. This property is used to focus the beam and transport it to the target experiment. Usually doublets and triplets of quadrupoles are used for this purpose. An example is shown in Fig. 1.3.

3. Collimators are materials that have a specific aperture size, depending on the purpose of use and its place. It is used mainly in two ways. At first, it limits the beam size and defines the beam acceptance[1]allowing to lower the beam intensity. At second, it helps in cleaning the beam from specific type of particles. In other words, it is a particle selector also. As a result, collimators are called "target attenuators". Fig. 1.4 shows an example of useful collimators.

4. Other elements, as follows:

a. Beam monitor: is a component that measures the beam intensity in the plane transverse to the beam propogation direction.

b. Beam dumps: are blocks of material that absorb the beam after it has been delivered

[1]Beam acceptance is the maximum beam emittance that the beam transport system is able to transmit. And the beam emittance is the average beam size in the phase space.

to the beam users. Usually, iron and concrete of few meters length are used.
c. Other beam instruments: intensity counters, scintillators, detectors and profile monitors are examples of many beam instruments that is used in the experimental areas.

Particle selection is achieved by exploiting particles properties, such as:
1. Synchrotron Radiation: could be used to separate light particles from heavier ones in a bending or quadrupole magnets, in which the lighter particles lose more energy than the heavier ones and thus they are bent more. This is detailed in section 2.3.
2. Absorber: in which a specific type of particles loses most of its energy while others are nearly not affected. Then, it passes by a bending magnet, where the light particles will be bent more, separating the heavier beam particles from the lighter ones.
3. Radiator: to get a photon beam, secondary electrons lose energy via bremsstrahlung in an absorber, called radiator. The products pass by a bending magnet, separating the photon beam from the low energy electrons that is bent by the magnet.

1.2 Introduction to MADX and AppLE.py

In high energy accelerator physics, a common software is used, called MAD-X. MAD-X is the methodological accelerator design code, MAD, and X is the latin letter of version 10. MADX is used extensively in beam optics design and related studies of charged particles beams. Since the graphical output of MAD-X is limited and does not include most of the functions needed for effective operation of the secondary beam lines, another software framework was developed in the EN-EA-LE section. This graphical user interface of MADX is called AppLE.py and it stands for Application for Liaison with Experiments written in python. AppLE.py provides a simulation for the beam line together with a convenient graphical display of the beam parameters and beam line elements. It also includes the functionality of optimization and particle tracking.

1.3 Beam parameters

The beam particles' motion can be described mathematically in the transverse plane of trajectory. Assuming right hand coordinates (x, y, s), s is the distance along the reference particle trajectory, x is the horizontal coordinate in direction of the reference orbit

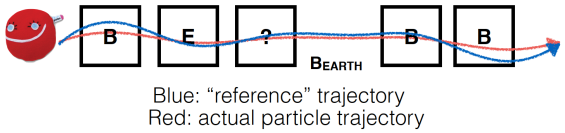

Figure 1.5: A particle reference trajectory in blue and the real trajectory in red.

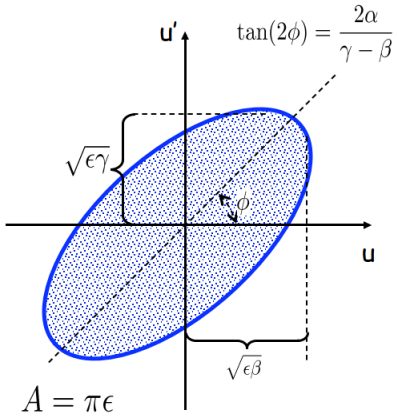

Figure 1.6: Beam ellipse in phase space (u, u'), where u is the position, u' is the transverse momentum. β is related to the beam shape and size, α is related to the beam tilt and ϵ is related to the beam spread. other parameters are dependent on the previous 3 parameters.

curvature and y is the coordinate perpendicular to both. The beam is designed so that the particles follow a reference orbit, ideal trajectory. Indeed, most of particles deviates from the reference trajectory. So that, the idea in studying the transverse motion of the particles is to subtract the reference trajectory from the actual one and later use some approximations.

The transverse motion of particles with respect to the reference trajectory is governed by Hill's equations, that is solved in order to get the particle trajectory. To solve them, a set of variables is used. Each particle has the reference momentum p_0 with its components (p_x, p_y, p_z). To normalise the momentum, an average momentum p_s is defined as $p_s = p_0(1 + \delta_s)$ where δ_s is an arbitrary value imposed by the user. For particles of variable momentum, the differential momentum error is called $p_t = \Delta E/p_s c \approx \beta_s(\Delta p/p_s)$. It has the phase space coordinates (x, x') and (y, y'), where $x' = dx/ds$ and $y' = dy/ds$ are the transverse momentum in x and y direction in radians.

Each beam element effect on the beam line can be described by 6×6 matrix, called R-matrix. The R-matrix allows to relate the initial particle phase to that after the element.

It is possible in MADX to use the phase space column vector Z. Both Z and an example of the R-matrix of the drift[2] space are defined below, where β and γ are the usual relativistic parameters and L is the drift space length:

$$\mathbf{Z} = \begin{pmatrix} Z_1 \\ Z_2 \\ Z_3 \\ Z_4 \\ Z_5 \\ Z_6 \end{pmatrix} = \begin{pmatrix} x \\ p_x \\ y \\ p_y \\ t \\ p_t \end{pmatrix}, \quad \mathbf{R}_{drift} = \begin{pmatrix} 1 & L & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & L & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \frac{L}{\beta_s^2 \gamma_s^2} \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

If the initial particle phase is Z_0, and it passes by an element of R_1, then R_2, etc, then its phase after these elements is Z'

$$Z' = R_{...} R_2 R_1 Z_0 = R_{final} Z_0$$

In the R-matrix, the optical parameters (R11, R12, R16) are for the horizontal plane and (R33, R34, R36) for the vertical plane. These optical parameters relates the final position to the initial position taking into account the passed elements in the beam. The units of these parameters will be (length/length, length/transverse momentum, length/$\Delta p/p$) for each plane, respectively.

Practically, huge number of particles are used in beams so that the previous Hill's equations no more governs the beam dynamics. To overcome this problem, the beam is parameterized and the transport equations are written for the entire beam in terms of these parameters instead of one particle equation. A good approximation for the beam in the phase space is an ellipse. Its area and tilt are described by many parameters, called twiss parameters. They are 3 independent parameters and one dependent parameter. At first, β[3] that is related to the beam shape and size. Secondly, α is related to the beam tilt. Thirdly, ϵ is the beam emittance, that is the average beam spread in the phase space, so it is related to the beam size. Finally, γ is the dependent variable on α and β, $\gamma = (1+\alpha^2)/\beta$.

[2]The drift space in the beam line is the space in which no element is located.

[3]This β has nothing to do with the relativistic parameters β = v/c, where v is the particle velocity and c is the speed of light.

Chapter 2

New functionalities in AppLE.py

In AppLE.py, beamline elements are represented in several classes. It handles MADX input and output files. Also, it implements the calculations of converting between power supplies current and magnetic fields. These links allow to provide both calculations and visualization of optics parameters and beamline tracking. These abilities are very useful for many users who need to tune many beam parameters, the beam spread as an example.

2.1 AppLE.py requirements and outputs

AppLE.py requires some initial files to be called, and used to process for the online calculations and visualization. It mainly needs the following files:
1. MADX twiss ouput file named **_Ref_Magnets.twiss*[1] : generated by a madx file describing the beamline optics. The twiss file contains the sequence of the beam magnets and its types in addition to other properties.
2. MADX magnets file named **_Magnets.madx* : for each magnet type, its length and aperture are defined.
3. Magnets currents database : I2BL files (from current to magnetic field and vice versa). Also, power supply file *PowerSupplies.txt*: a text file associates each magnet name, defined in Magnets.madx file, to its power supply.

In the graphical user interface (GUI) of AppLE.py, some initial beam information are to be determined as the particle type and its energy. After these information are de-

[1] the star * indicates any name

termined, AppLE.py code calls the previous listed files(twiss file, magnets information, magnets currents). AppLE.py produces the following files:

1. MADX input file **_input.madx*: contains the read beamline sequence of elements, with its location and defined properties.

2. Output magnets twiss file *Out_Magnets.twiss* file: contains the sequence of the read beam magnets and its types.

3. Optics twiss file **_Optics.twiss*: contains the calculated R-matrix parameters, taking into account all beam elements.

4. Tracking file **_Tracking.txtone*: contains information about the beam particles positions, transverse momentum and its energy detailed after each beam element.

5. MADX plotting file **_Optics.pdf*: contains the illustration of the beam optics (elements and beam parameters).

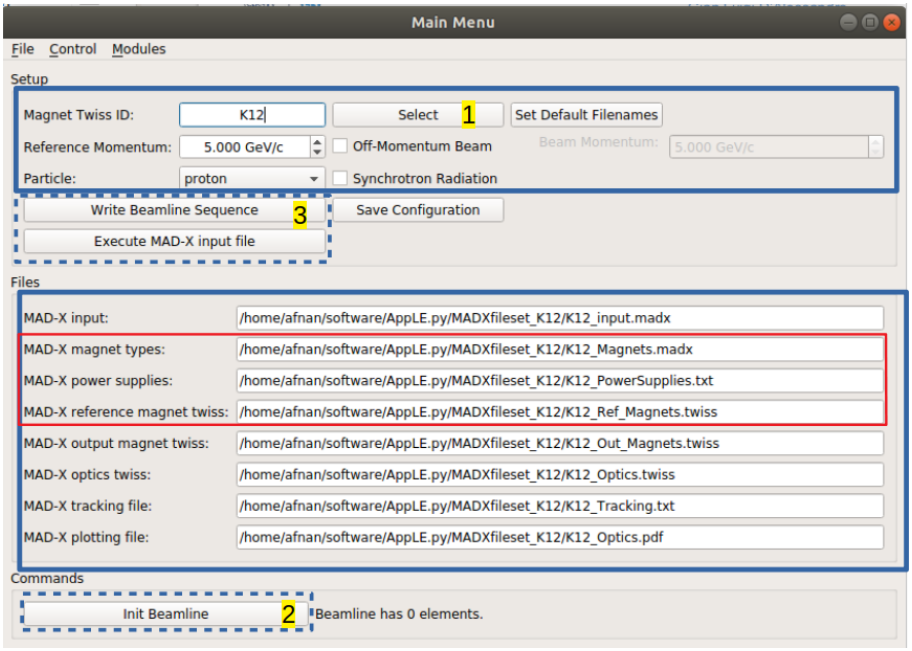

Figure 2.1: The main graphical interface of AppLE.py. The first numbered box includes the initial beam information. The second numbered box in the bottom is a command to initiate the beamline sequence. After which AppLE.py calls the files listed in the red box and produces all other listed files. The third numbered box is a command to write the beamline sequence. From the upper menu bar, from "control", one can display the "MADX control" module that will be shown in Fig. 2.2.

Fig. 2.1 shows the main GUI when AppLE.py is lunched, with the list of input and output files and commands to do the beam calculations and simulation.

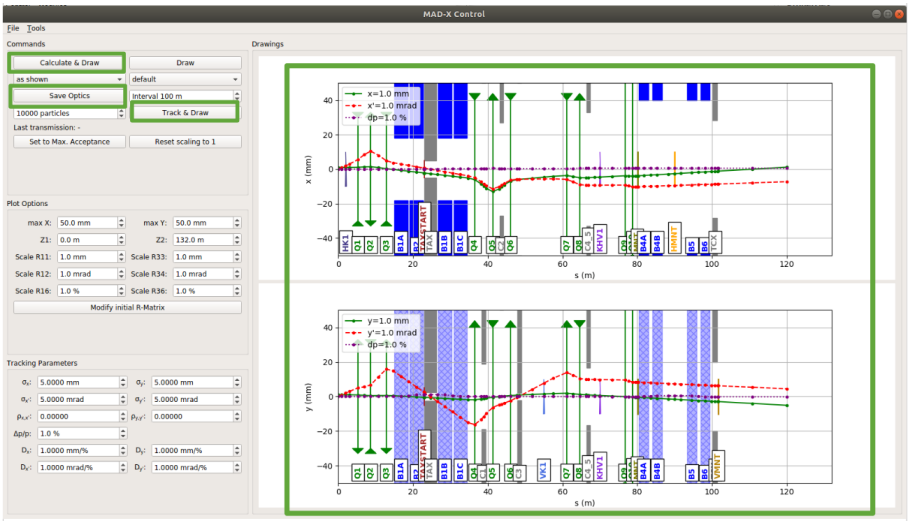

Figure 2.2: MADX control module of AppLE.py.

Fig. 2.2 shows "MADX control GUI" that is launched from "control" in the main GUI menu bar. The beamline elements and parameters are visualized in the left green box. The upper and bottom part are the horizontal and vertical plane in the direction of the particle trajectory. x and y are the position coordinates. x' and y' are the transverse momentum. dp is the relative energy error. In the left part in the figure, one can choose to calculate and draw the beam parameters (R11, R12, R16, R33, R34, R36). Also, one can save the beam optics graphics (the green box in the right). Moreover, one can draw the track of the beam particles in addition to the related tracking parameters using a randomized initial beam distribution. The number of beam particles are also editable.

2.2 AppLE.py functionalities

AppLE.py in its latest version was including several beam elements, like the bending magnets, quadrupoles and collimators. New functionalities were added to AppLE.py, including beam monitors and kickers. The graphical interface AppLE.py has to be lunched by the command in the terminal : *$python3 GUI.py*. The *GUI.py* calls two base files, called *BLToolBase.py* and *BLToolGUI.py* that is thousands of lines written in python. In

BLToolBase.py, each element is defined in classes and the *input.madx* file is constructed here. However, *BLToolGUI.py* file that is built using *pyQt* python library, all beam parameters calculations, drawings and tracking is implemented here. And in this file, all buttons that appear in the GUI are defined with their functions.

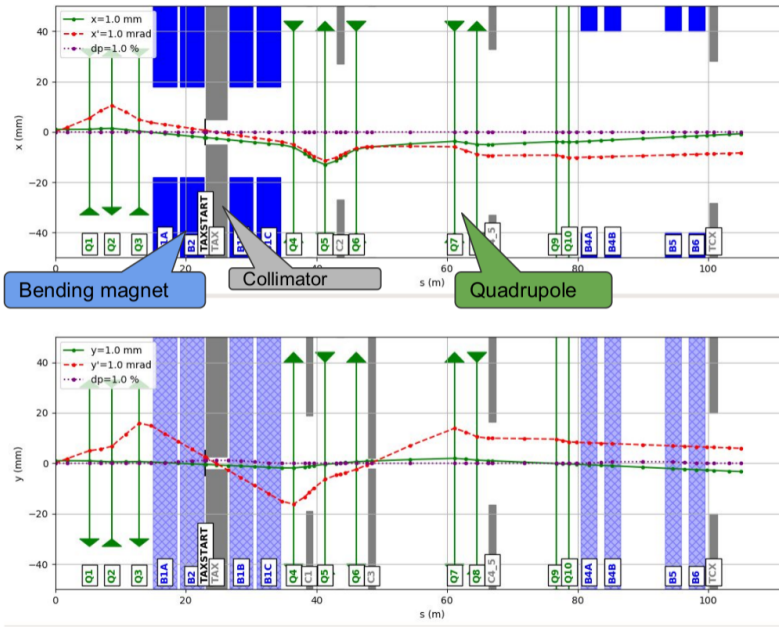

Figure 2.3: The graphical interface optics output of AppLE.py before adding new functionalities (during this project).

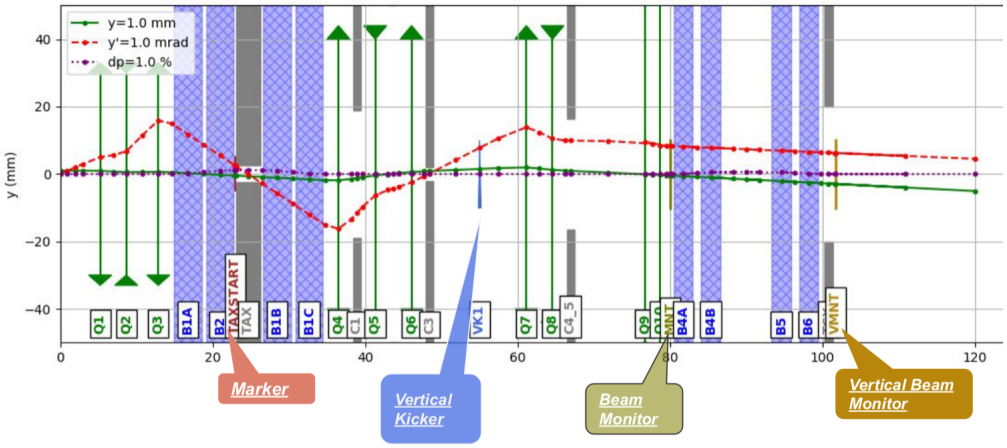

Figure 2.4: AppLE.py optics output in the vertical plane after adding the vertical kicker, vertical monitor and a marker.

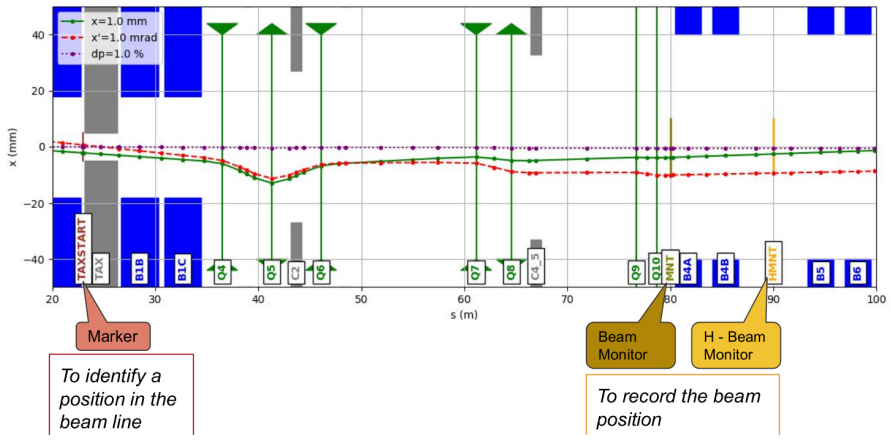

Figure 2.5: AppLE.py optics output in the horizontal plane after adding the horizontal kicker, horizontal monitor and a marker.

Fig. 2.3 shows AppLE.py elements, while Fig. 2.4 and Fig 2.5 show new added elements to AppLE.py in the vertical and horizontal planes. The new added elements are the following:

- Kicker: It is first defined in the beam sequence in the madx file, that generates the *Ref_Magnets.twiss* file. It has many possible attributes, its length, its tilt from the longitudinal axis and its kick. The kick is defined as the relative change in momentum with respect to the particle reference momentum, indicating the bending strength of the kicker. The kick, denoted by *kick*, can be in one plane of the transverse motion. It is either a horizontal kick, $= \delta p_x/p_0$ or a vertical kick, $= \delta p_y/p_0$. It is also added to the *Magnets.madx* file and the *PowerSupplies.txt* file as it needs current for the dipole magnet. These 3 files are needed to launch AppLE.py .

  After making sure that the magnet type is in the I2BL database, the kicker is defined in the base file *BLToolBase.py*. Three classes are defined corresponding to 3 elements based on the kicker correction direction. One class for the horizontal plane only. Another one for the vertical plane only. And the last one for the two planes at once. In the same base file, all are added to the list of elements that need a power supply with the same physics as the bending magnets, as in equ. 1.1. Each class includes a description of the drawn element in the GUI. Kicker symbol is a filled triangle. Different color is assigned to each kicker class. Kicker is also added in *BLToolGUI.py* file to the list of elements that is included in the beam

parameters calculations and displaying of the beam parameters and beam tracking. Finally, when AppLE.py is launched from *GUI.py* file, it calls both of the files *BLToolBase.py* & *BLToolGUI.py*.

- Beam position monitor is defined in a similar fashion as the kicker element, except that the beam monitor is not a magnet and does not need a power supply. It has only one attribute, that is the length. It is also defined in 3 different classes. One provides a monitor only in the horizontal plane. Another one for the vertical plane. And the last provides both directions. Each one is represented in the GUI as a thick vertical line in its corresponding plane. Different colors are assigned for each class.

- Marker is a virtual element that helps in finding for example the beam parameters at specific position. It is visualized in AppLE.py as a thick vertical line since it is already defined in the *Ref_Magnets.twiss* file. It has no attributes.

2.3 Synchrotron radiation losses

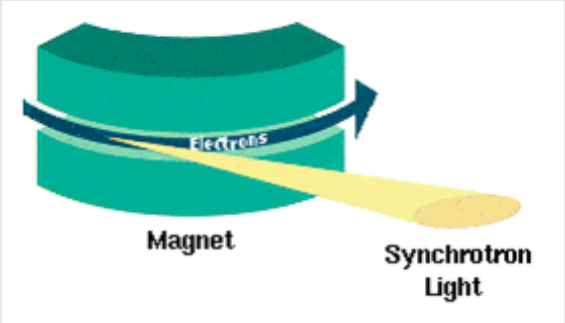

Figure 2.6: Synchrotron radiation loss of a charged particle in a bending magnet.

A particle of momentum p and charge q bends in an orthogonal magnetic field of strength B with a radius ρ following a central magnetic force as follows:

$$B\rho = \frac{p}{q} \tag{2.1}$$

A unit charged ultra relativistic particle, of mass m and energy E, emits radiation of ΔE and bends with a radius ρ in a bending magnet, of length L and bending angle θ. The emitted radiation ΔE, from a charged particle of mass m and kinetic energy E bends

with a bending raduis ρ per one revolution in a circular orbit, reads as follows:

$$\Delta E = \frac{e^2}{3\epsilon_0 (mc^2)^4} . \frac{E^4}{\rho} \tag{2.2}$$

where ϵ_0 is the electric permittivity.

With the help of equ. 2.1 and equ. 1.1, the loss of energy for a part of the revolution covering the bending angle θ out of 2π, ΔE becames:

$$\Delta E = \frac{e^2}{6\pi\epsilon_0 (mc^2)^4} . \frac{E^4 . \theta^2}{L} \tag{2.3}$$

For electrons, $mc^2 = 0.511 MeV$, ΔE becomes:

$$\Delta E[GeV] = 1.550610^{-5} . \frac{E^4[GeV^4] . \theta^2[rad^2]}{L[m]} \tag{2.4}$$

And for protons, $mc^2 = 0.93827 GeV$:

$$\Delta E[GeV] = 1.250410^{-18} . \frac{E^4[GeV^4] . \theta^2[rad^2]}{L[m]} \tag{2.5}$$

However, for electrons, a phenomenlogical formula has been benchmarked against the beam data collected in the North Area secondary beam lines as follows:

$$dE_e[GeV] = 1.3935.10^{-5} . \frac{\theta^2[rad^2] . p^4[GeV^4]}{L[m]} \tag{2.6}$$

L = 3.6 m, θ = 0.02 rad				
p (GeV)	100	200	300	400
$dE_e(GeV)$	0.154833	2.47733	12.5415	39.6373
$\Delta E_e(GeV)$	0.172289	2.75662	13.9554	44.106
$\Delta E_{proton}(10^{-13} GeV)$	0.138933	2.22293	11.2536	35.5669

Table 2.1: Manual calculations of synchrotron radiation loss for electrons and protons of different momentum p. dE_e is the energy loss by electrons using the benchmarked formula, equ. 2.6. ΔE_e and ΔE_{proton} are the energy losses using the analytical formula, equ. 2.3.

Table 2.1 shows some expected energy loses of protons and electrons emitting synchrotron radiation in a bending magnet of length L and bending angle θ.

Chapter 3

Conclusions

Beam particles does not follow the reference trajectory most of the time, so the real trajectory is simulated with respect to the reference one with the help of beam parameterizing and both of software codes, MAD-X and the graphical user interface AppLE.py. The beam optical parameters are also calculated and displayed in AppLE.py. AppLE.py also provides particle tracking and beam parameters optimization functionality.

Several elements of new functionalities were added to AppLE.py as part of the internship project. The beam position monitors are added in 3 different ways. Either only in the horizontal plane or only in the vertical plane of the transverse motion or in both planes at once. The kickers are also added in the same fashion, the 3 different ways are available.

In the near future, the synchrotron radiation loses calculated by MADX and visualized by AppLE.py will be checked and compared to the expected calculations. Manually calculated synchrotron radiation loss of protons is approximately smaller than for electrons by a factor of $^{-13}$ since the proton is about 2000 times heavier than the electron. And the higher the particle's momentum, the higher the loss of energy as synchrotron radiation is.

References

[1] L.Deniau(editor)et al., The MAD-X Program, Version 5.05.02, Users Reference Manual, July(2019), Geneva, Switzerland.

[2] M. Rosenthal, Note on AppLe.py, EN-EA-LE-INTERNALNOTE-2019, Geneva, Switzerland.

[3] F. Christoph Iselin, The MAD Program, Version 8.13, Physical Methods Manual, September(1994), Geneva, Switzerland.

[4] G.L. DAlessandro, Internal slides: "Particles transport and matrix formalism"

[5] J. Holmes et al., Transverse Beam Optics, Part I , USPAS, January(2009), Vanderbilt. https://uspas.fnal.gov/materials/09VU/Lecture6.pdf