Manual of the tools for the Analysis of the Crab Cavities

Bruce Yee Rendón, CINVESTAV, México.

Abstract

Here I explain with details the tools that i learned to make the analysis of the CC.

I explain how to use MAD-X in order to calculated the CC voltages and others optics parameters, the Local Loss Map, the executable, the plots of the relatives coordinates for a specify point inside the lattice in Six-Track and create a distribution in Python.

Geneva, Switzerland,
September 2010

Contents

Chapter 1

Crab Cavity Voltage with MAD-X

In order to calculated de CC Voltage using the formulas that we already calculated [1],we use the program MADX [2]. In the program MAD-X we defined the CC for the two differents squemes:

CASE I : Local Crab Cavity (LCC)
CASE II : Global Crab Cavity (GCC).

For the LCC , two CC are put at 130 m in both sides of the IP5 (CMS detector), and for the GCC is at 30 m upstream of IP4 [3].

We introduced Markes in the position of the CC and the IP5, in order to have acces to theirs optics parameters (that we use to calculated the voltage).

For run MAD-X, in your directory, just type the next comand:

madx < name.madx

where **name.madx** is the name of our MAD-X files. It generated the file **twiss.dat** in which are printed the information of the optics parameters, also we can specify which optics parameters (α_x,β_x,μ_x,...) that will be printed for each device of the lattice, can generate plots of the accelerator's variables and also some input files for SixTrack (**fort.2.cc**)[4].

The voltage for the LCC for the left is,

$$V_{CCL} = \frac{cE_s \tan(\frac{\phi}{2})}{q\omega\sqrt{\beta_{CCl}\beta_{IP}}\sin(\Delta\Psi_1)} \tag{1.1}$$

and for the rigth is,

$$V_{CCR} = \frac{cE_s \tan(\frac{\phi}{2})}{q\omega\sqrt{\beta_{CCR}\beta_{IP}}\sin(\Delta\Psi_2)} \tag{1.2}$$

Table 1.1: Optics Parameters for a LCC

Parameters	[Units]	Value
q	e	1
c	$10^8 \frac{m}{s}$	3
E_s	TeV	7
ϕ	$\mu radians$	285
ω	MHz	800
β_{IP}	m	0.55
β_{CCL}	m	1622.285
$\Delta\Psi_1$	$radians$	0.2553
β_{CCR}	m	924.388
$\Delta\Psi_2$	$radians$	0.2561

and the optics parameters are in the Table 1.1,

where, c is the speed of the light, E_s the particle energy, ϕ the crossing angle, ω the CC frequency, β_{IP} is the β funtion in the interacion point (IP), β_{CCL} is the β funtion in the left CC, β_{CCR} is the β funtion in the right CC, $\Delta\Psi_1$ is the difference of the phase advance between the left CC and the IP and $\Delta\Psi_2$ is the difference of the phase advance between the IP and the right CC.

The results is $V_{CCL} = 7.885$ MV and $V_{CCL} = 10.446$ MV

For a GCC, the voltage is

$$V = \frac{cE_s \tan(\frac{\phi}{2})}{q\omega\sqrt{\beta_{CC}\beta_{IP}}} \frac{2\sin(\pi Q)}{\cos(\Delta\Psi - \pi Q)} \tag{1.3}$$

and the optics parameters are in the Table 1.2,

where, c is the speed of the light, E_s the particle energy, ϕ the crossing angle, ω the CC frequency, β_{IP} is the β funtion in the interacion point (IP), β_{CC} is the β funtion in the CC and $\Delta\Psi$ is the difference of the phase advance between the CC and the IP.

The results is $V = 9.07$ MV.

Table 1.2: Optics Parameters for a GCC

Parameters	[Units]	Value
q	e	1
c	$10^8 \frac{m}{s}$	3
E_s	TeV	7
ϕ	$\mu radians$	285
ω	MHz	800
β_{IP}	m	0.55
β_{CC}	m	255.965
Q		64.31
$\Delta\Psi$	$radians$	7.673

With MAD-X we can obtain the **fort.2.cc** which are ones of the input of SixTrack. In the next way

SIXTRACK, CAVALL,
MULT_AUTO_OFF,
APERTURE,
RADIUS = 0.017.

The MAD-X programs are in the the folder “/afs/cern.ch/user/b/byee/public/MADX”.

Chapter 2

SixTrack and Collimation

In the folder "/afs/cern.ch/user/b/byee/public/cc" are all the necessary files to run Sixtrack simulations for our proposals.

We want to reproduce the previous results computes to qualify the performance of the collimation system, namely cleaning efficiency (loss distribution per proton incident on the primary collimator, and also residual halo vs. amplitude) and the impact parameter on the primary collimator with the help of Local Loss Map (LLM) [3].

In order to achieve this task we requeried to have the next files:

sixtrackcc : is the executlable that we obtain from Sixtrack code.

The way that you can create your own executable is in the next way.

1) You need to login in like:

user@lx64lc4.cern.ch or **user@lx32slc4.cern.ch**

we use the last one.

2) You need to type in the console:

```
csh
```

3) If we just make a small detail for the CC we will work on the file **Track.f** and then we compile with the intruction:

lf95 *.f -Wa, –32 –o <sixtrack.executable>

but we use the next way, go the folder Six-code, remove the file **sixtrack.s** and replace for the version that we changed , remove the executable and compile with the next order:

./make_six collimat -crlibm (with out crlibm and include collimat)

We need to include collimat if we want to get the LLM.

CollDB_V6.500_lowb_st.b1.data, CollPositions.b1.dat : It has the LHC collimators database.

allapert.b1 : It contains all the information about the beam 1 aperture.

BeamLossPattern_2005-04-30_gcc2.9 : It analyze the tracking data, and get the local loss map.

fort.2.cc : Its the input file of Sixtrack which give us all the geometry of the lattice.

In the first part it defined all the elements of the lattice its called the **Single Elements**. The first column is the name of the element. Each kind of element has a certain number, which appears in the second column. For example, BPMs and pointers have number 0, quadrupoles are number 2, crab-cavities (globalcc) are 23, etc. The crab-cavities are introduced by us, they are not in the file the collimation team uses. These cavities are called with certain parameters in the files **fort.3.cc** and **fort.3.v.cc**.

We defined like:

Name of the CC	*Type*	*Voltage* $[MV]$	*frequency* $[MHz]$	*Phase lag* $[radians]$
Gcc	23	9.07	800	0

Here we put an example for the a GCC , with a horizontal crab beam (23 if is a vertical crab beam if must be -23), the voltage is 9.07 MV . Then start the **Block Definitions** (in four-dimensional transverse tracking, the linear elements between nonlinear elements can be combined to a single linear block to save computing time).

And the last part is the **Structure Input** (in which appear the sequence of the elements in the way that they are install in the lattice.)

We can select the starting point with " **GO** ", writing it just before the **IP** that we want to be the first. That is very important since the Beta-functions are very different depending of the starting point. By default, the beginning is set to the **IP1** (look into the file and search **GO** to see how to insert it).

fort.3.cc and **fort.3.v.cc** : These are the other input for SixTrack that we use in this work and give all parameters of the lattice that sixtrack uses as input. **fort.3.cc** has to be used for the horizontal plane and **fort.3.v.cc** has to be used for the vertical one (but, we you use the **fort.3.cc**).

In **fort.3.cc** exits structured by blocks. Between two NEXT's there is a block. The most useful for us are:

- **TRACKING** : the first number is the total number of turns (202 by default for collimation we can no use more than those). The third number is the number of particles we track per bunch divided by 2 (by default 32, so we have 64 particles per bunch, and normally we have 50 bunches, so we'll have 3200 particles in total).

we just change the first line the first and third column the rest of the parameters are still the same like the previous work (Collimation's team and Yi-Peng)[3].

```
TRACKING
202 0 32 0 17 0 1
1 1 0 0 0
0 0 1 1 1 20000 2
```

- **INITIAL COORDINATES**: It's only for single particle tracking, (we changed when we are interested in obtain the plot of relative horizontal position versus the relative longitudinal position with respect to synchronous particle).

- **CRAB** : It's the new block (create for Javier Barranco) which lead us to change the voltage and the phase of CC in an independent way.

```
CRAB
1 10 191 0
1 10 190 1 1.0472
11
```

The first line is for the change of the voltage, the first column for the number of free turns, the second for the ramping up, the third one for plato and the fourt one for the ramping down.

The second line is for the change of the phase, and in the fourt first columns are the same like for the voltage, but it has a fifth column in order to specify the final phase of the CC.

And the third line is the turn that you turn onn the collimators, before that turn the collimators acts like a drift space.

-**SYNC**: Synchrotron oscillation parameters.

- **COLLIMATION** : It's made by the collimation team. That's the most important. It begins like this in the file **fort.3.cc**.

```
COLLIMATION
.TRUE.
50 7000000
3 5.958 .0015 0. 0. "nothing" 1.129E-4 75.5
```

In the second line, the first column are the number of packs of 64 particles to be tracked (**breaks internal limitation**) and the second column is the energy of the beam to be tracked. In the third line the first column give to us the choice for the distribution, sixtrack has three defaults distributions and have the option four when you can introduce you own distribution, for understand how its work we use the option three in the distribution (distribution in the plane for which the parameters are specified + 1 sigma distribution in the other plane + energy distribution and bunch length according to the values given with **ENERROR** and **BUNCHLENGTH**) but now we create a distribution with Python [5]. The second and third column are referred to the horizontal plane, and the fourth and fifth to the vertical one. This line is the only one which changes between the two files (**fort.3.cc** and **fort.3.v.cc**), and that's why in this one the fourth and fifth values are set to 0. 5.958 is the x (or y) position in units of σ and 0.0015 is the error in position.The sixth we introduced the name of our distribution (if we put in the first column the option four if we dont do, just put "dummy"). The seventh is for the energy spread of tracked beam and the eigth is the bunch length of the tracked beam in millimeters (This is just a fragment of the collimation's block, here we just put the part that we used).

my job: Script to copy files to lxplus nodes, and copy results back to your home folder. It's necessary to change the path ("/afs/cern.ch/user/b/byee/scratch0/sixtrackcc/crabcavity"). Here also if we use **fort.3.v.cc** (in we want the vertical crab beam) u other files (**fort.2.cc**, **sixtrack.cc**)

joblauncher.sh: Script to launch jobs (to send them to the computer farm). It will call "**myjob**".

2.1 How to run

Now for submit a job just write the command:

```
./joblauncher.sh lowb .coll hor b1 1nd 1 20
```

where each position has the next meaning:

1)Call the **joblauncher.sh**.
2)Choice of **lowb** (for low beta) or **inj** (for injection).
3)Choice a name for the result folder (it will be the name of your folder inside results, but preceded with a dot (.), remember that the result it will keep in the files lowb/inj.name).
4)Put **hor** (for horizontal) or **ver** (for vertical). Depending of this parameter we have to change **fort.3.cc** or **fort.3.v.cc** in **myjob** (line 18 if nothing is changed).
5)Referred to the beam, **b1** or **b2**.
6)The queue (**1nm** means "1 minute", **8nh** means "8 hours" (normally it will be what we need), **1nd** means "1 day" (can be useful if the number of jobs is big)).
7)The number in which we want to start (normally 1).
8)The number of jobs we submit to the farm.

In order to see the evolution of the jobs, we can use the commands **bjobs** and **bqueues -u user**.

We can delete all the reports from the farm as well as the mails you receive telling if the job was successful. Once we have run it, the results will be copied back at "../crab-cavity/result/lowb.name".

2.2 Obtain the Local Loss Map

For the each job we obtain five files as a result (the names are given for **lowb**, **hor** and **b1**):

FirstImpacts.lowb.hor.b1.(jobnumber).dat: This files shows (only) the lost particles in the collimators.

eff.lowb.hor.b1.(jobnumber).txt: This file contains the number of lost particles for diferent number of sigma.

LP_lowb_hor_b1_(jobnumber).s: This file has the positions in which the particles are lost.

LPI_lowb_hor_b1_(jobnumber).s: This file has the positions in which the particles are lost with a resolution of 10 cm.

summ.lowb.hor.b1.(jobnumber).txt: Contains a resume of all the particles which are absorbed by the collimators.

To obtain the LLM in the Results folder you must to have the files : **imp_summary.m, list_collimators_V6.500.txt** and the **loss _coll.m**. This 3 Matlab [6] scripts to compute the results graphically, they have to be in the same folder than the data files (Figure 2.1)..

imp_summary.m: Its the main program whichs calls the other ones (**loss_coll.m** and the **list_collimators_V6.500.txt**) also calls the **summ.lowb.hor.b1.(jobnumber).txt**.

list_collimators_V6.500.txt: It has the list of Collimators and its position in the lattice.

loss_coll.m: It will be called from **imp_summary.m**, here you can put some labes to the plot, or also you can do it in the matlab editor. This scripts calls the **LPI_lowb_hor_b1 ...s** files in order to generate the LLM.

In order to run, first you need to have installed Matlab in your directory, if you dont have you need to have an account on **parc** and the when you get it, you need to type in the Result Folder the next command

```
/afs/cern.ch/project/parc/bin/matlab
```

Then open it with Matlab. Select the "current directory"' where the script and the data are. Write **imp_summary.m** in the Matlab prompt. VERY IMPORTANT: Change the number of jobs. By default is set to 1800. It's written as "n=1800".

After running it we obtain a plot of the local inefficiency vs. the space coordinate (s [km]). All parameters are set for top energy (7 TeV) and full beam (2808 bunches). For those parameters is calculated the Quench limit. In grey we observe the losses at the collimators (of course the bigger ones will be in the IP7, corresponding the highest line to the primary collimator), in blue the losses at the superconducting magnets and in red the ones at the warm magnets.

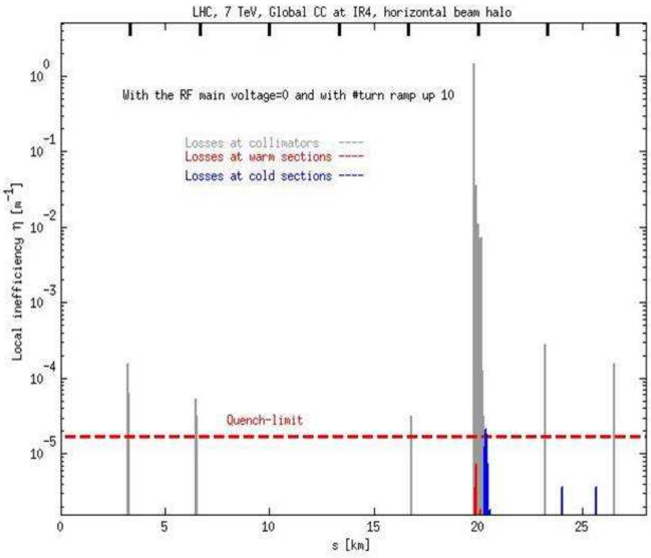

Figure 2.1: The Local loss map for the primary TCP.C6L7.B1 in the nominal LHC (top energy, β_{IP5}=0.55m, without a transverse, longitudinal position of $1{\times}10^{-4}$ and with a offset energy of $2.5\sigma_p$). Using a GCC of 800MH (The number of the tracking turns for the ramping up is 10).

2.3 Obtain the plots of the relatives coordinates

If you want to generate the plots of the relative horizontal position again the relative longitudinal position with respect to the synchronous particle. First you need to introduce a **BPM** in the place that you are interested to see, the way that you can do is defined in the **Single Elements** of **fort.2.cc** in the next way

$$BPMname \qquad 0 \quad 0.000000000E+00 \quad 0.000000000E+00 \quad 0.000000000E+00.$$

After that in the **Structure Input** you need to put BPM in the place that you are interested (In the las part of **fort.2.cc**). When you create your executable on SixTrack you can run like this:

./make_six -collimat -crlibm bpm (with out collimat,crlibm but with bpm)

Finally take the new **sixtrack.executable** and put it in the same folder that you have **fort.2.cc** and **fort.3.cc**, you need to change it both for **fort.2** and **fort.3** ,then you run the **sixtrack.executable** in the next way:

./sitrack.executable

For this you dont need to submit a job you can run locally in your pc, the result will give to you a outputfile with the name BPMname. This ouput file has seveth column that are X, X', Y, Y', CT, $\frac{\Delta p}{p}$ y E.

And with your favorite program you plot X vs CT (Figure 2.2).

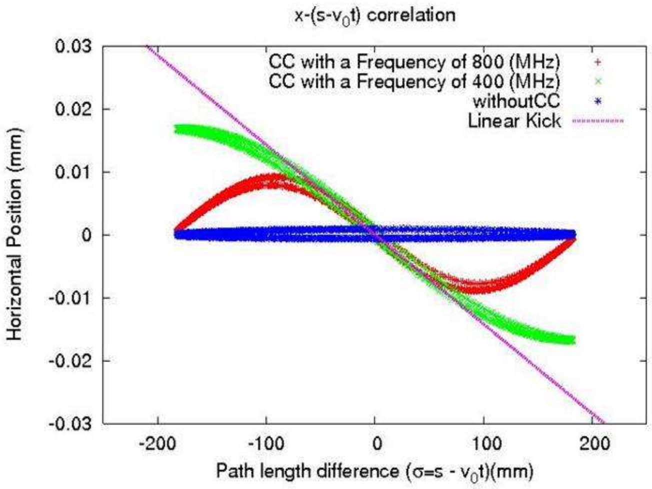

Figure 2.2: This is the correlation of the relative horizontal and longitudinal position with respect to the synchronous particle, without a transverse and longitudinal position offset and with a offset energy of $2.5\sigma_p$. Using a CC of 800MHz, 400MHz and without CC. The number of the tracking turns is 1000.

Chapter 3

Python Distribution

For the distribution we use a script in Python, so we generated a random gaussian distribution (For now, then we can generate a distribution which describe better our situation).

We just need to specify the starting point (in our case is IP1), in order to get the α_x, α_y,... the geometry emittance for x and y (which can be obtained for the **twiss.dat** file from MAD-X, in the case of the emittance we can obtain from the normalized emittance 3.75 mrad with the formula), the sigma E (the energy spread we use 1×10^{-4}), the sigmal (the bunchlength 75.5 mm).

We run the program from our directory we the next command:

python file.py

where **file.py** is the program on python in our case is **CreatingDistribuction.py**, and the generate two outputfile:

- **DistributionforTracking.eps** which are the plots for the distribution (Figure 3.1).

- **Initial_Distribution.txt**. In this files contains the values for X [mm], X'[mrad], Y[mm], Y'[mrad], CT[mm] and $\frac{\Delta p}{p}$[MeV], which are the input format for a distribution in SixTrack.

The Python programs are in the the folder "/afs/cern.ch/user/b/byee/public/Python".

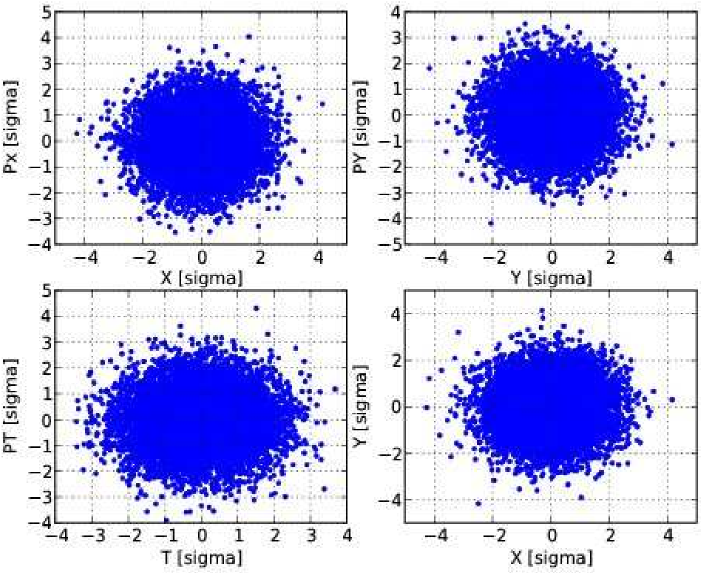

Figure 3.1: Example of initial particle distribution used as input for Sixtrack.

Bibliography

[1] B. Yee Rendon, Crab Cavity Voltage Calculation, http://byee.web.cern.ch/byee/.

[2] MADX manual, http://mad.web.cern.ch/mad.

[3] Yi-Peng Sun, Phys. Rev. ST Accel. Beams 12, 101002 (2009).

[4] Python manual, http://www.python.org/ .

[5] SixTrack manual, http://frs.web.cern.ch/frs/six/six.html .

[6] Matlab manual, http://www.mathworks.com/products/matlab/.