From PTC-ORBIT to PTC-pyORBIT for PS Booster space charge simulations

JB. Lagrange, CERN

March 29, 2018

1 Introduction

Since PTC-ORBIT is no longer maintained, the goal is to continue the studies done initially with this code to PTC-pyORBIT. A proper benchmark study between the 2 codes is then needed, especially in the case of the PS Booster study at CERN. The versions of the codes used here are for PTC-ORBIT:
/afs/cern.ch/project/LIUsc/space_charge/Codes/
PTC22.10.2014-ORBIT10_SLC6_mpich2,
and for pyORBIT:
/afs/cern.ch/project/LIUsc/space_charge/Codes/
py-orbit_revison1291_dev_FrozenPIC.
Additions to pyORBIT to match the needs of the simulations are also developed.

2 Simulations without space charge (PTC only)

Simulations of the 2 codes have been done without space charge. The parameters of the simulations are 10^5 particles, 3000 turns in the PSB. The results of emittances and rms momentum spread are presented in Fig. 1 and Fig. 2, respectively. The results are identical within the computing precision.

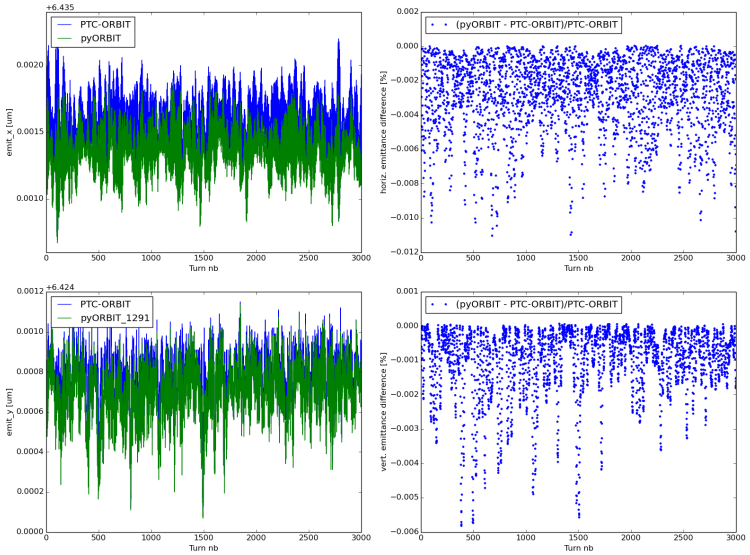

Figure 1: Top: horizontal emittance evolution without space charge as a function of turn number in PTC-ORBIT and pyORBIT (left) and their relative difference (right). Bottom: vertical emittance evolution without space charge as a function of turn number in PTC-ORBIT and pyORBIT (left) and their relative difference (right).

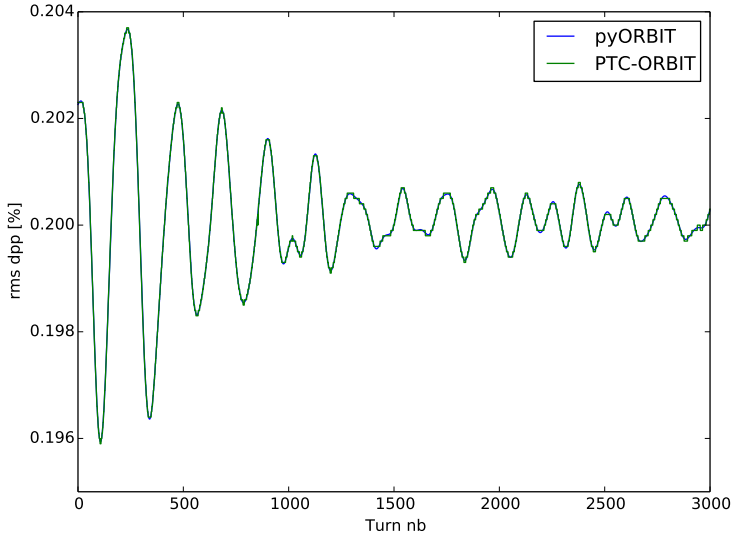

Figure 2: rms momentum spread in PTC-ORBIT and pyORBIT without space charge.

3 Simulations with transverse space charge only

Simulations of the 2 codes have been done with transverse space charge, with the longitudinal space charge turned off. The parameters of the simulations are 10^5 particles, 10^4 turns in the PSB nominal case at 62 GeV without acceleration, with a space charge grid $128 \times 128 \times 64$. The results of emittances and rms momentum spread are presented in Fig. 3 and Fig. 4, respectively. There is a good agreement between the two codes, with 1.5% emittance difference in both planes after 10^4 turns.

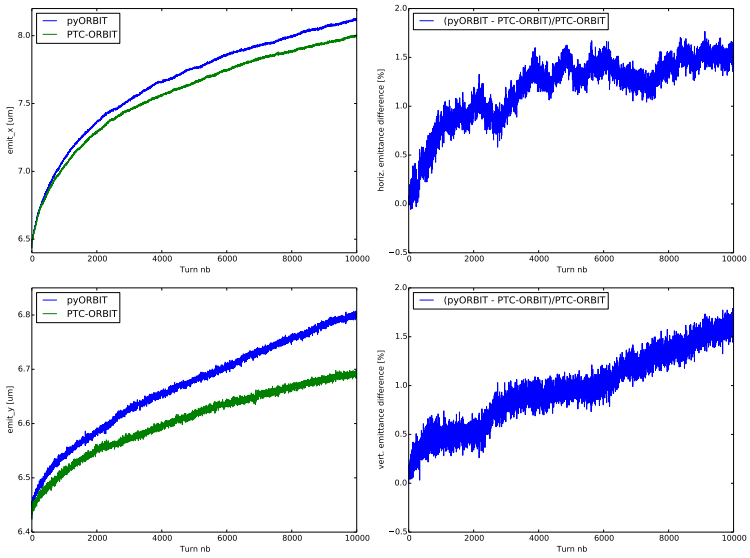

Figure 3: Top: horizontal emittance evolution with space charge as a function of turn number in PTC-ORBIT and pyORBIT (left) and their relative difference (right). Bottom: vertical emittance evolution with space charge as a function of turn number in PTC-ORBIT and pyORBIT (left) and their relative difference (right).

4 Position of the longitudinal node in PTC-ORBIT

In PTC-ORBIT, the longitudinal space charge (LSC) node is the place where the longitudinal binning is done for the 2.5D space charge calculation. In the PTC-ORBIT script, the LSC node was originally created with the index 115, which means it is put in the middle of the lattice. There is then no longitudinal binning in the first half turn in this case, and a coasting beam is simulated for this half first turn. To be sure to have a longitudinal binning from the beginning, the LSC node should be created with the index -10. The next simulations of PTC-ORBIT will be implemented this way.

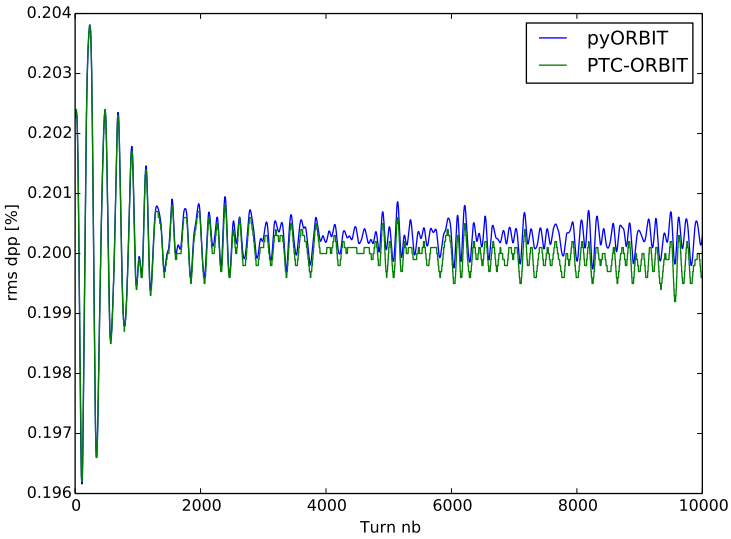

Figure 4: rms momentum spread in PTC-ORBIT and pyORBIT with space charge.

5 Number of longitudinal nodes and update of longitudinal binning

The code does not accept more than 1 longitudinal node per turn by default. In Sources/LSpaceCharge.cc, LSpaceCharge::addFFTLSpaceCharge there is an error triggered (tooManyFFTLongs) if more than 1 node is added. Furthermore, the longitudinal binning in PTC-ORBIT is updated only in the LSC node, and not in the transverse space charge (TSC) nodes. The update is then done only once per turn in PTC-ORBIT. In pyORBIT, the update is done at every TSC node.

5.1 Switching off the error trigger in PTC-ORBIT

When the error trigger is switched off, we can add 200 longitudinal nodes to have a simulation similar than in pyORBIT. A simulation has been done to evaluate the effect of the number of longitudinal nodes in the ring. The simulation with 1 LSC node (PTC-ORBIT_1Lnode) and the simulation with 200 LSC nodes (PTC-ORBIT_200Lnodes) do 10^4 turns, with 10^5 macroparticles, a transverse space charge grid $128 \times 128 \times 64$ and longitudinal SC switched off. The results are presented in Fig. 5 and Fig. 6, and explain the origin of the difference in rms momentum spread between PTC-ORBIT_1Lnode and pyORBIT.

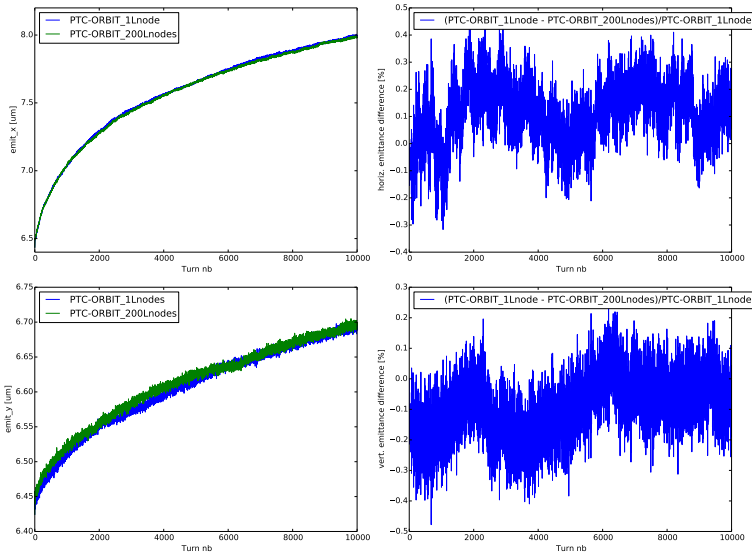

Figure 5: Top: horizontal emittance evolution with space charge as a function of turn number in PTC-ORBIT with 1 LSC node and PTC-ORBIT with 200 LSC nodes (left) and their relative difference (right). Bottom: vertical emittance evolution with space charge as a function of turn number in PTC-ORBIT with 1 LSC node and PTC-ORBIT with 200 LSC nodes (left) and their relative difference (right).

6 Integration Length

In PTC-ORBIT, the integration length for a transverse space charge node is computed as the average of the neighbouring PTC nodes lengths. In pyORBIT, the integration length is equal to the length of the associated PTC node, with the transverse space charge node placed before the PTC node.

Moreover, in PTC-ORBIT, the first and last space charge node of the lattice consider that the missing neighbouring node has no length, so the integration length of these space charge nodes are just half of the length of the existing neighbouring node.

The code PTC-ORBIT has been modified to connect the first and the last PTC node regarding the integration length (see Fig. 7).

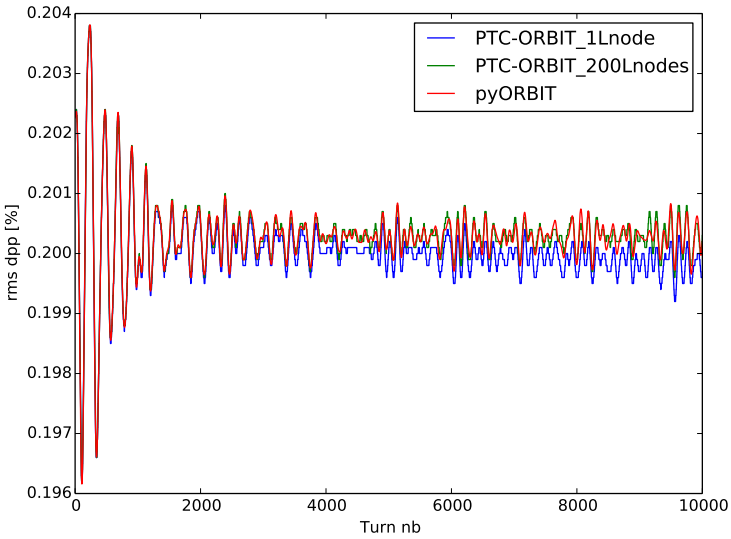

Figure 6: rms momentum spread in PTC-ORBIT with 1 LSC node, with 200 LSC nodes and pyORBIT.

7 Space charge kick

The space charge kick divided by the integration length has been computed in both codes. The results are shown in Fig. 8. The difference between the 2 codes is computed in Fig. 9.

8 Linear/smooth longitudinal binning

In PTC-ORBIT, the longitudinal binning is done smoothly (quadratically), while in pyORBIT, the binning is done linearly (See Fig. 10).

8.1 Smooth binning in pyORBIT

The code pyORBIT has been modified to implement a smooth longitudinal binning (See Fig. 11), by changing in src/spacecharge/SpaceChargeCalc2p5D.cc, line 120, the function getValueSmoothed is used and line 232, the function binBunchSmoothed is used.

8.2 Smooth binning in pyORBIT with the same longitudinal length

The longitudinal binning in pyORBIT is done over a fluctuating length delimitated by the extrema in longitudinal position, while it is done over a constant length in PTC-ORBIT (actually, it is done in phase, so always

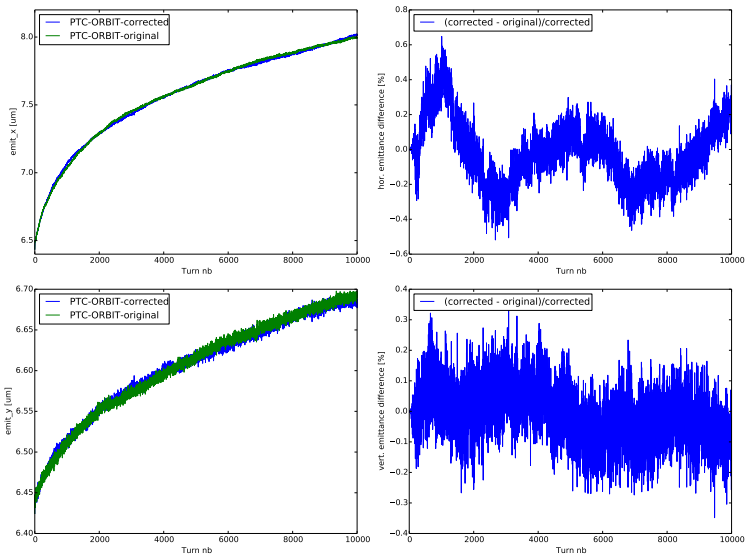

Figure 7: Top: horizontal emittance evolution with space charge as a function of turn number in original PTC-ORBIT and PTC-ORBIT with integration length corrected (left) and their relative difference (right). Bottom: vertical emittance evolution with space charge as a function of turn number in original PTC-ORBIT and PTC-ORBIT with integration length corrected (left) and their relative difference (right).

over 2π). The code pyORBIT has been modified to force the same length as in PTC-ORBIT, and the result is shown in Fig. 12.

The results of the emittance evolution for the different binning options over 10^4 turns are presented in Fig. 13.

9 Simulation with transverse and longitudinal space charge

In pyORBIT, the attribute "macrosize" can be addressed in two different ways, either globally, associated to the whole bunch, or individually, associated to every macro-particle. The longitudinal space charge kick in pyORBIT was taking into account only the global macrosize attribute, and not the individual macrosize attribute, putting to zero all longitudinal space charge kicks if the later was used. The bug has been corrected, and both options are handled in the longitudinal kick.

Simulations of the 2 codes have been done with transverse and longitudinal space charge. The parameters of the simulations are 10^5 particles, 10^4 turns in the PSB nominal case at 62 GeV without acceleration, with a space charge grid $128 \times 128 \times 64$. The results for the different longitudinal binning

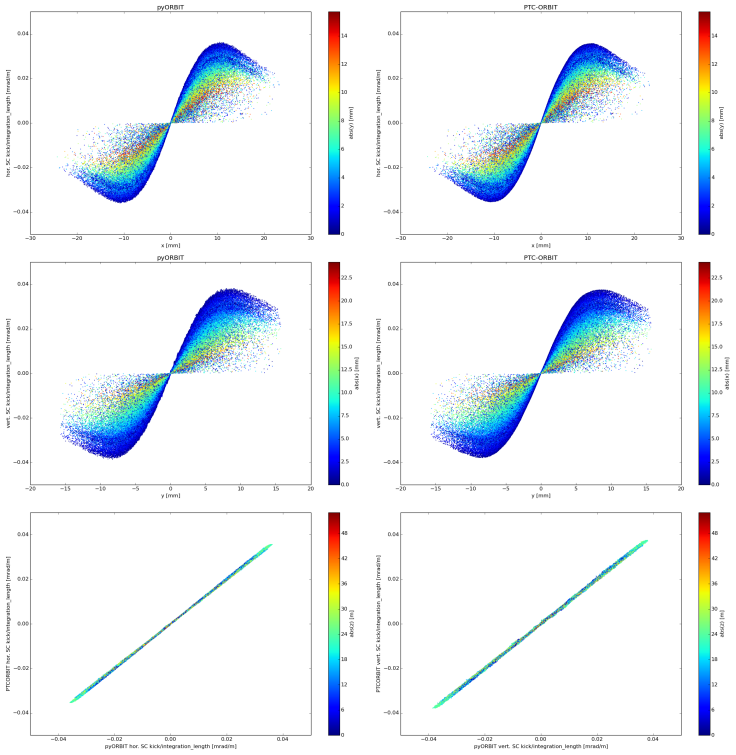

Figure 8: Horizontal (top) and vertical (middle) space charge kick divided by integration length in pyORBIT (left) and PTC-ORBIT (right). Bottom: Horizontal (left) and vertical (right) SC kick in pyORBIT, as a function of SC kick in PTC-ORBIT.

options are presented in Fig. 14.

10 Simulations with injection and acceleration

Simulations of the 2 codes have been done with transverse space charge, with the longitudinal space charge turned off. The parameters of the simulations are $5 \cdot 10^5$ particles injected turn by turn over 100 turns, then tracked over 10 turns in the PSB LIU case at 150 GeV with acceleration, with a space charge grid $128 \times 128 \times 64$. Simulations without longitudinal space charge have been performed. The results are shown in Fig. 15. There is a good agreement except for the first 10 turns.

The difference comes from the computation of the emittance. In pyORBIT, the emittance comes from the beam parameters only, while in PTC-ORBIT, the emittance is computed with the ring dispersion for dealing with the momentum spread in horizontal. It can lead to a difference, especially if the beam is not matched to the lattice. To confirm it, the data of the first

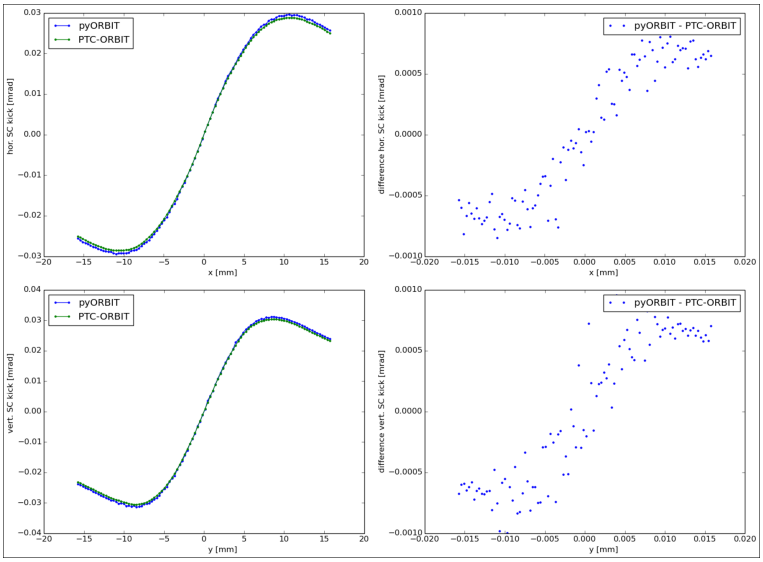

Figure 9: Top: horizontal space charge kick divided by integration length in pyORBIT and PTC-ORBIT (left) and their difference (right). Bottom: vertical space charge kick divided by integration length in pyORBIT and PTC-ORBIT (left) and their difference (right).

10 turns of the previous simulation in PTC-ORBIT are used to compute the emittance in the pyORBIT way. The results are presented in Fig. 16.

11 Latest version of pyORBIT

Simulations have been done with the latest version of pyORBIT, with some bugs fixed:

- ◇ The parameter "length_" is initialised in the constructor Grid1D(int zSize) (in src/spacecharge/Grid1D.cc), while the warning in getIndAndWZSmoothed (in src/spacecharge/Grid1D.cc) about zero lattice length with smooth longitudinal binning has been moved to the initialisation (in src/spacecharge/SpaceChargeCalc2p5D.cc) to avoid the repetition of the warning at every step.
- ◇ The function updateParamsPTC (in py/ext/ptc_orbit/ptc_orbit.py) is modified to solve a bug to change the parameters in the different nodes without reinitialising the node and removing the children nodes.
- ◇ The attribute "macrosize" assigned to each macro-particle is now taken into account in longitudinal space charge computation (in src/spacecharge/LSpaceChargeCalc.cc

Some changes have also been applied to meet the needs of the PS Booster simulations:

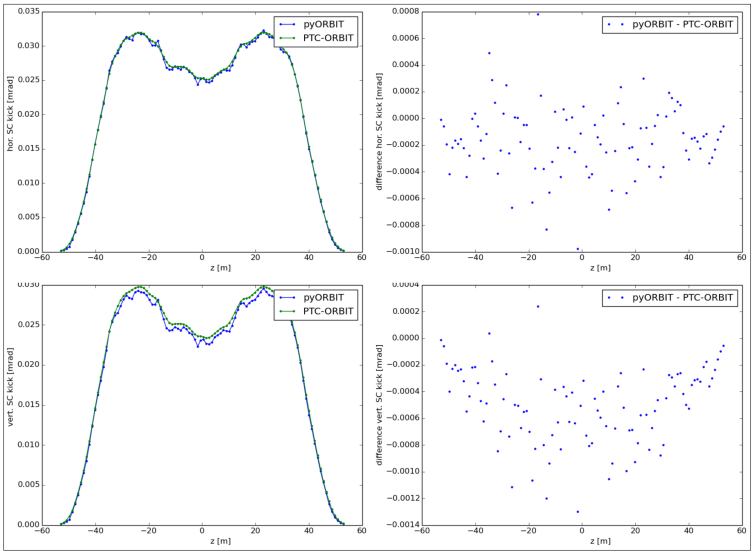

Figure 10: Top: horizontal space charge kick as a function of longitudinal coordinate in pyORBIT and PTC-ORBIT (left), and their difference (right). Bottom: vertical space charge kick as a function of longitudinal coordinate in pyORBIT and PTC-ORBIT (left), and their difference (right). This plot is obtained by adding in the input file 100 particles distributed along the longitudinal bunch with a horizontal amplitude of 15 mm (top) and a vertical amplitude of 9 mm (bottom), and plotting the angle difference for these particles before and after the TSC node.

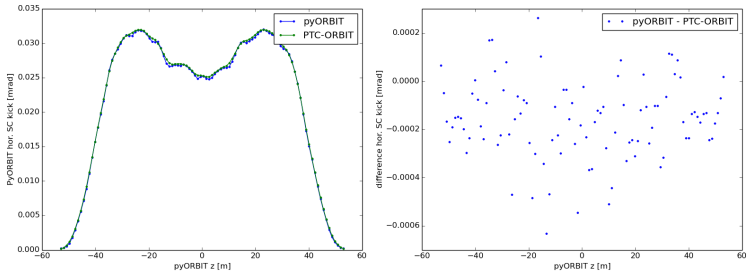

Figure 11: Horizontal space charge kick as a function of longitudinal coordinate in pyORBIT with smooth binning and PTC-ORBIT (left), and their difference (right). This plot is obtained by adding in the input file 100 particles distributed along the longitudinal bunch with a horizontal amplitude of 15 mm, and plotting the difference for these particles of horizontal angle before and after the TSC node.

- ◇ a flag in the space charge 2.5D method is set (setSmoothBinning(True)) to use smooth longitudinal binning instead of linear longitudinal binning. By default the flag is set to false.

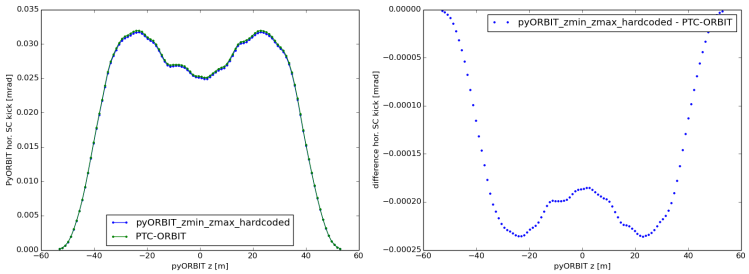

Figure 12: Horizontal space charge kick as a function of longitudinal coordinate in pyORBIT with smooth binning and a fixed longitudinal length and PTC-ORBIT (left), and their difference (right). This plot is obtained by adding in the input file 100 particles distributed along the longitudinal bunch with a horizontal amplitude of 15 mm, and plotting the difference for these particles of horizontal angle before and after the TSC node.

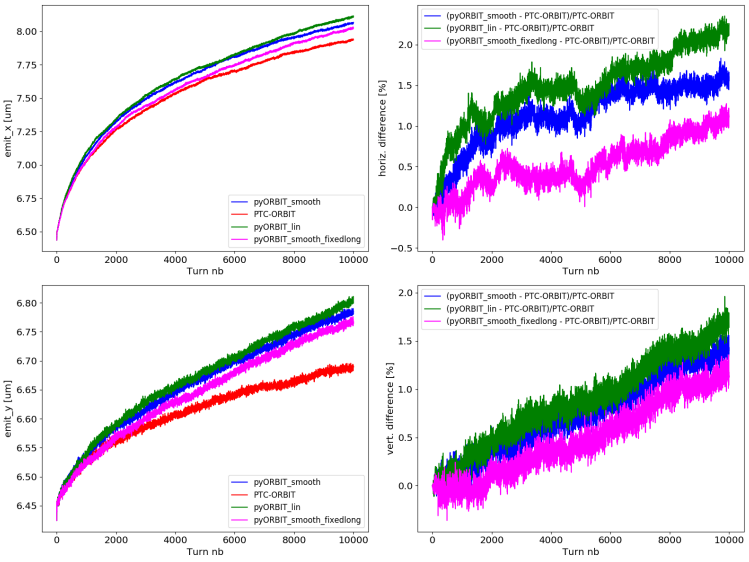

Figure 13: Top: horizontal emittance evolution with transverse space charge as a function of turn number (left) and their relative difference (right). Bottom: vertical emittance evolution with space charge as a function of turn number (left) and their relative difference (right).

◇ a flag ("part_id") has been added in the function orbit_to_pyorbit (py/orbit/utils/orbit_mpi_utils/bunch_orbit_to_pyorbit.py) to read from the input file the identification number of the particle. By default the flag is set to false.

◇ Examples of simulation scripts for the CERN PS Booster with fixed

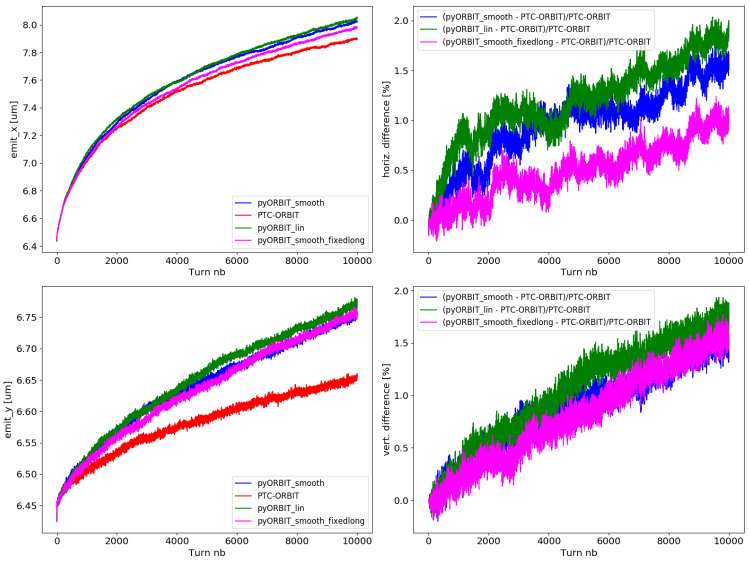

Figure 14: Top: horizontal emittance evolution with longitudinal and transverse space charge as a function of turn number (left) and their relative difference (right). Bottom: vertical emittance evolution with space charge as a function of turn number (left) and their relative difference (right).

energy and with multi-turn injection and acceleration have been added in examples/.

12 Simulations close to the horizontal integer resonance

A final benchmark has been performed with a simulation close to the horizontal integer resonance over 100 turns. The results without longitudinal space charge are presented in Fig. 17, and with longitudinal space charge in Fig. 18. They confirm the good agreement between the two codes.

13 Conclusion

- ◇ Good agreement between PTC-ORBIT and pyORBIT (1.5% emittance difference after 10^4 turns.)
- ◇ There is only 1 longitudinal space charge node per turn in PTC-ORBIT, and longitudinal binning update is done only in this node, while the update is done for every space charge kick in pyORBIT, creating a difference in rms momentum spread between the 2 codes.

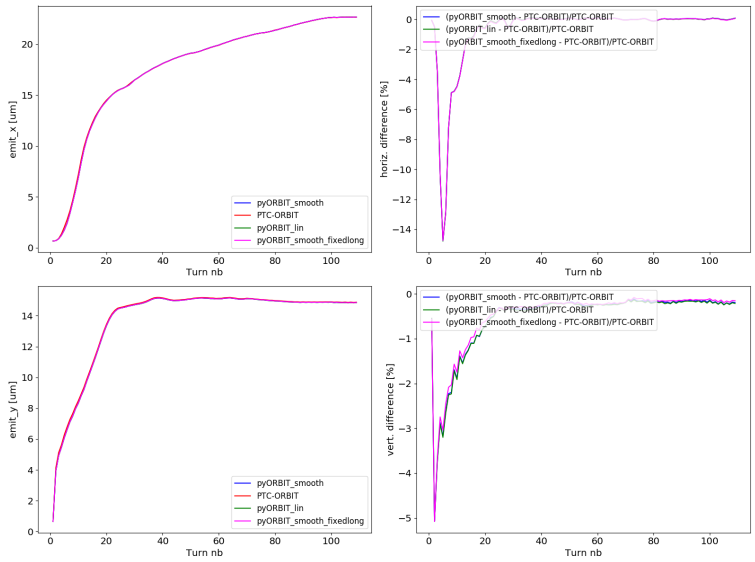

Figure 15: Top: horizontal emittance evolution with longitudinal and transverse space charge as a function of turn number (left) and their relative difference (right). Bottom: vertical emittance evolution with space charge as a function of turn number (left) and their relative difference (right).

- ◇ The longitudinal binning should be done smoothly in pyORBIT to have a better agreement between the two codes.
- ◇ Integration length is wrongly implemented in PTC-ORBIT, but it has a marginal effect on the result.

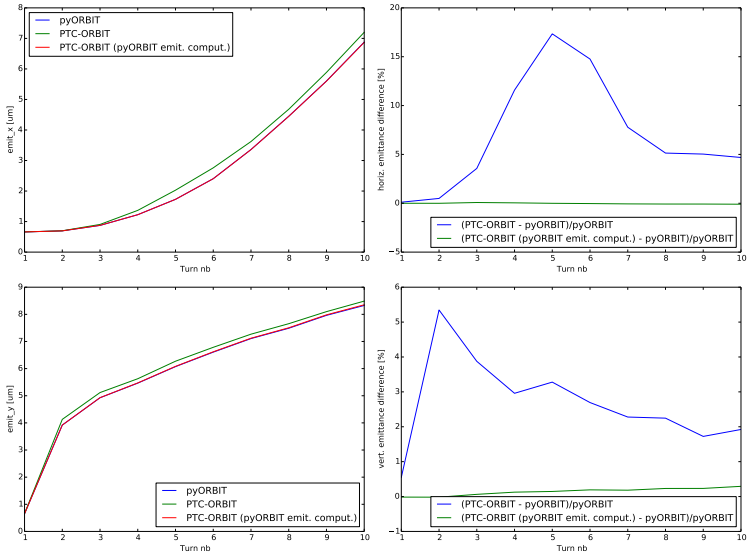

Figure 16: Top: horizontal emittance evolution with longitudinal and transverse space charge as a function of turn number (left) and their relative difference (right). Bottom: vertical emittance evolution with space charge as a function of turn number (left) and their relative difference (right).

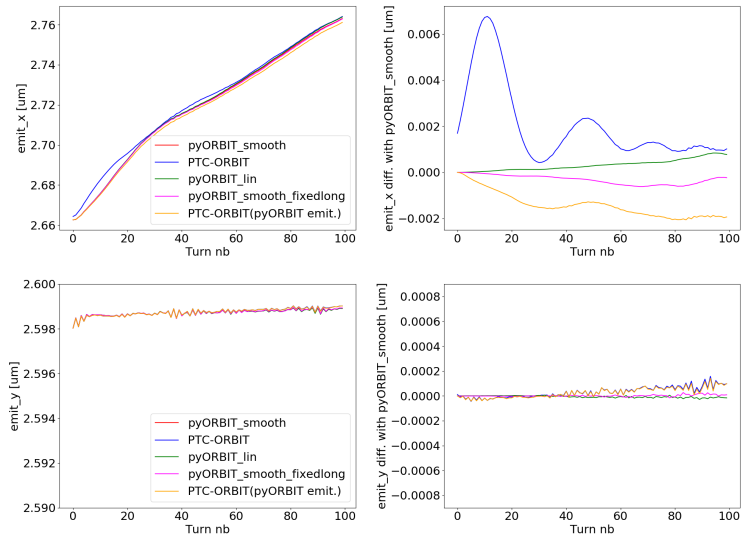

Figure 17: Top: horizontal emittance evolution with transverse space charge as a function of turn number (left) and their relative difference (right). Bottom: vertical emittance evolution with space charge as a function of turn number (left) and their relative difference (right).

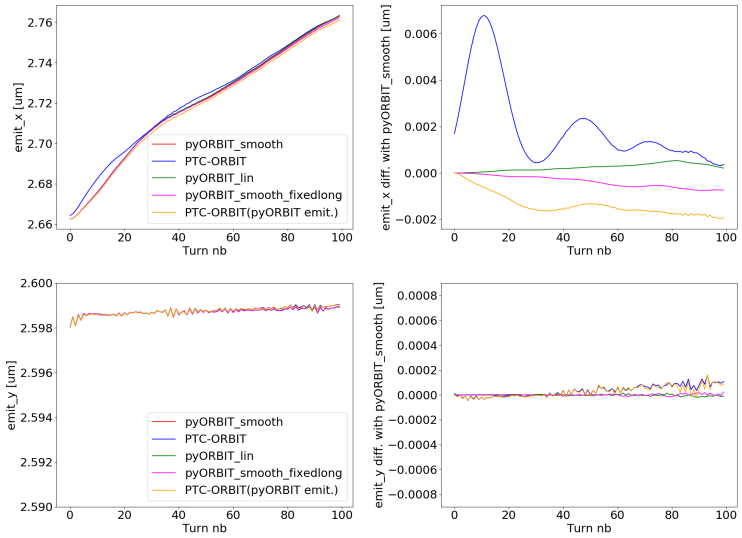

Figure 18: Top: horizontal emittance evolution with longitudinal and transverse space charge as a function of turn number (left) and their relative difference (right). Bottom: vertical emittance evolution with space charge as a function of turn number (left) and their relative difference (right).