Fast SixTrack Space Charge module

JB. Lagrange, H. Bartosik, R. De Maria,
K. Sjøbæk, F. Schmidt

Disclaimer

SixTrack

Code developed by F. Schmidt

(based on an earlier program from DESY)

Features:

- written in Fortran,
- Symplectic,
- written in the optics of tracking speed,
- used and developed by a wide community,
- conversion routine in MAD-X to create SixTrack input

Motivation

- MAD-X SC implemented, but limited parallel computing possibility (built for the adaptive mode).
- Need for study of very long term space charge effects in several rings at CERN (PS, SPS, LHC,…). Frozen space charge is enough in some studies.
- Frozen space charge in SixTrack could be heavily parallelised (~100 times faster).

→ Implementation of frozen SC module in SixTrack

Code versions

MAD-X SC 1.01.01 (based on MAD-X 5.02.07)

SixTrack 4.7.8

http://github.com/SixTrack/SixTrack

Beam-Beam & Space Charge

Horizontal 4D Beam-Beam kick

(opposite direction, proton-proton, on momentum, round beam):

$$\Delta x' = +\frac{2Nr_{cl}}{\gamma}\cdot\frac{x}{\rho^2}\cdot\frac{1+\beta^2}{2\beta^2}\left(1-e^{\frac{-\rho^2}{2\sigma_x^2}}\right)$$

colliding

if ultra-relativist,≃1

Horizontal 4D Space Charge kick:

$$\Delta x' = +\frac{2Nr_{cl}}{\gamma B_f}\cdot\frac{x}{\rho^2}\cdot\frac{1-\beta^2}{\beta^2}\left(1-e^{\frac{-\rho^2}{2\sigma_x^2}}\right)$$

$$= \frac{1}{\gamma^2\beta^2}$$

beam direction

MAD-X SC module

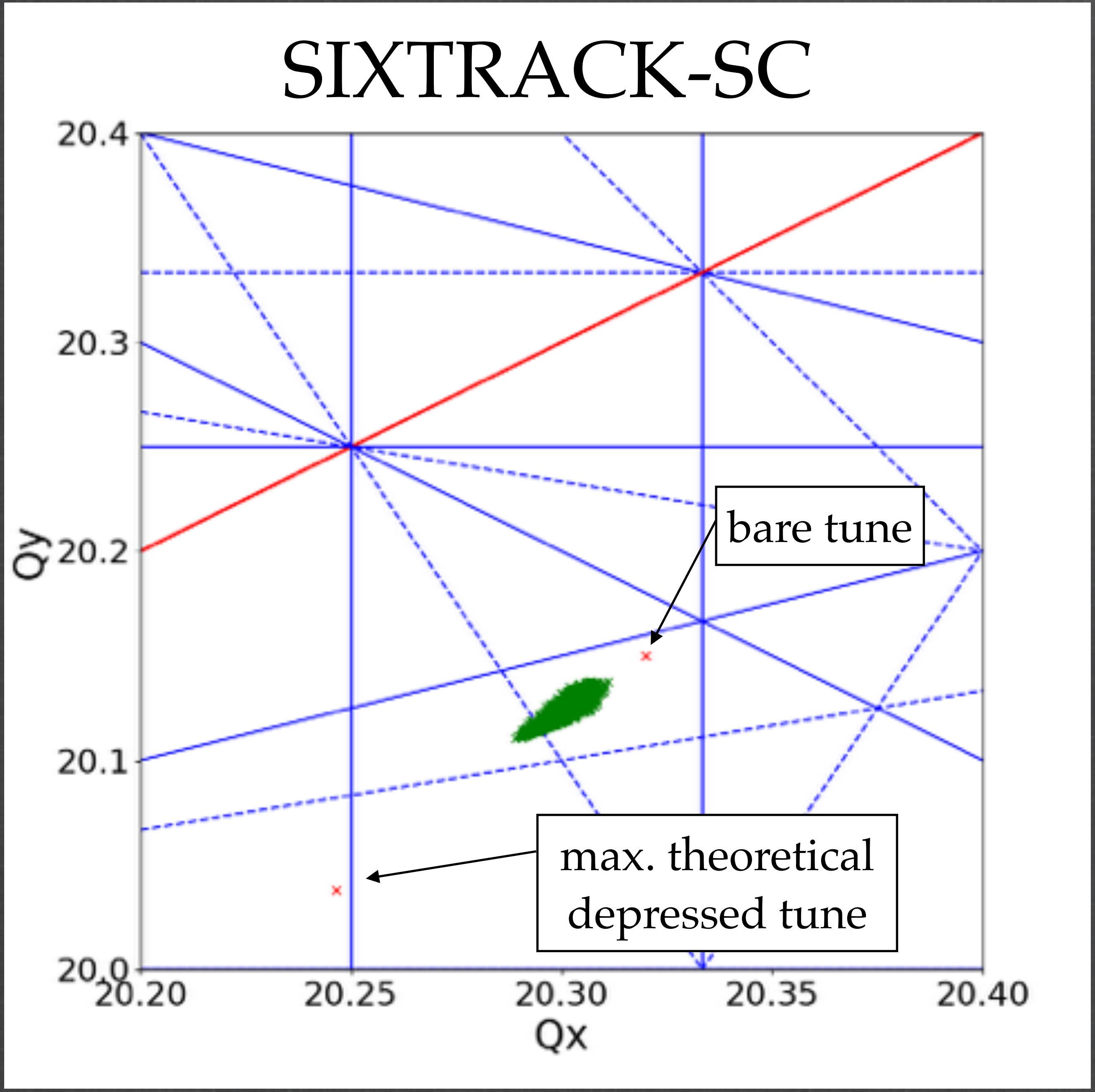
- Use of the MAD-X beam-beam module with specific preparation MAD-X macros and fortran executables.
  - Split the elements to add the SC kick in between.
  - Ultra-relativistic option (flag bb_ultra_relati=true).
  - the factor $\frac{1}{\gamma^2\beta^2}$ is added to the charge unit previously.
  - the bunch factor is added to the number of particles previously.

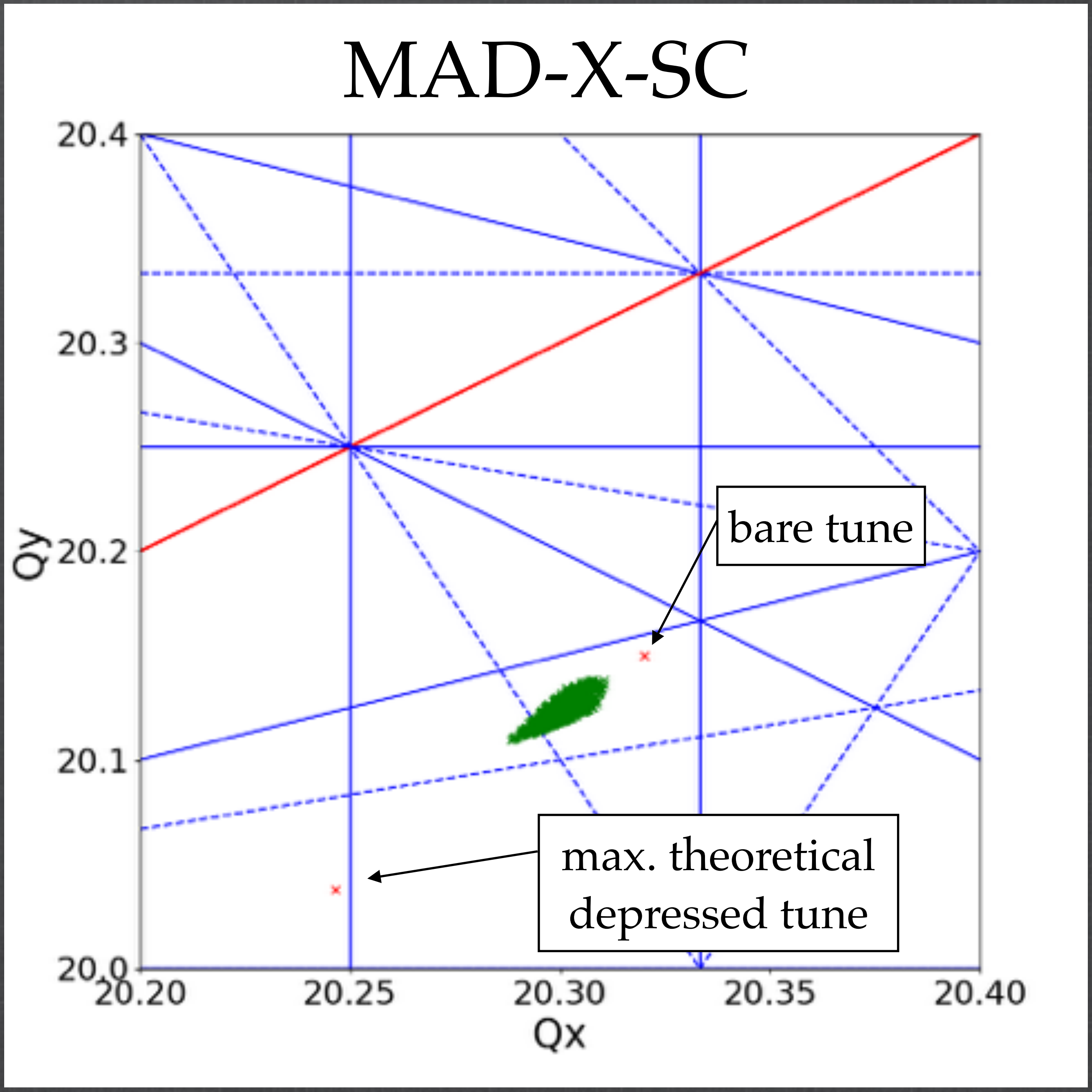
SixTrack beam-beam module & input from MAD-X

- Very similar than MAD-X beam-beam module.
- Ultra-relativistic hypothesis by default.
- Off-momentum behaviour in BB kick included.
- Input directly produced from MAD-X-SC (SIXTRACK command at the end of the macro)

Tune spread

SPS, 1.5 10^{11} protons, bunch length 0.22 m, 1000 particles





Summary & future plans

- SixTrack is a fast code with high capability of parallelised computation.
- In the same way than in MAD-X-SC, the beam-beam module can be used for a frozen space charge study.
- The input can be produced directly by MAD-X (bug for the SC kicks in dipoles, being investigated).
- Full benchmark with experiments and other codes (pyORBIT, MAD-X_SC) under way.

Benchmark of PTC-ORBIT and pyORBIT

JB. Lagrange, H. Bartosik, F. Schmidt

Motivation

PTC-ORBIT not maintained anymore.

$\longrightarrow$ switch to pyORBIT

However:

- pyORBIT never used for acceleration and injection
- all simulations for PS Booster done with PTC-ORBIT

Need for a proper benchmarking between the 2 codes.
M. Kowalska started a few months ago, I recently took over.

Simulation parameters used for benchmark

PTC-ORBIT:

/afs/cern.ch/project/LIUsc/space_charge/Codes/PTC22.10.2014-ORBIT10_SLC6_mpich2

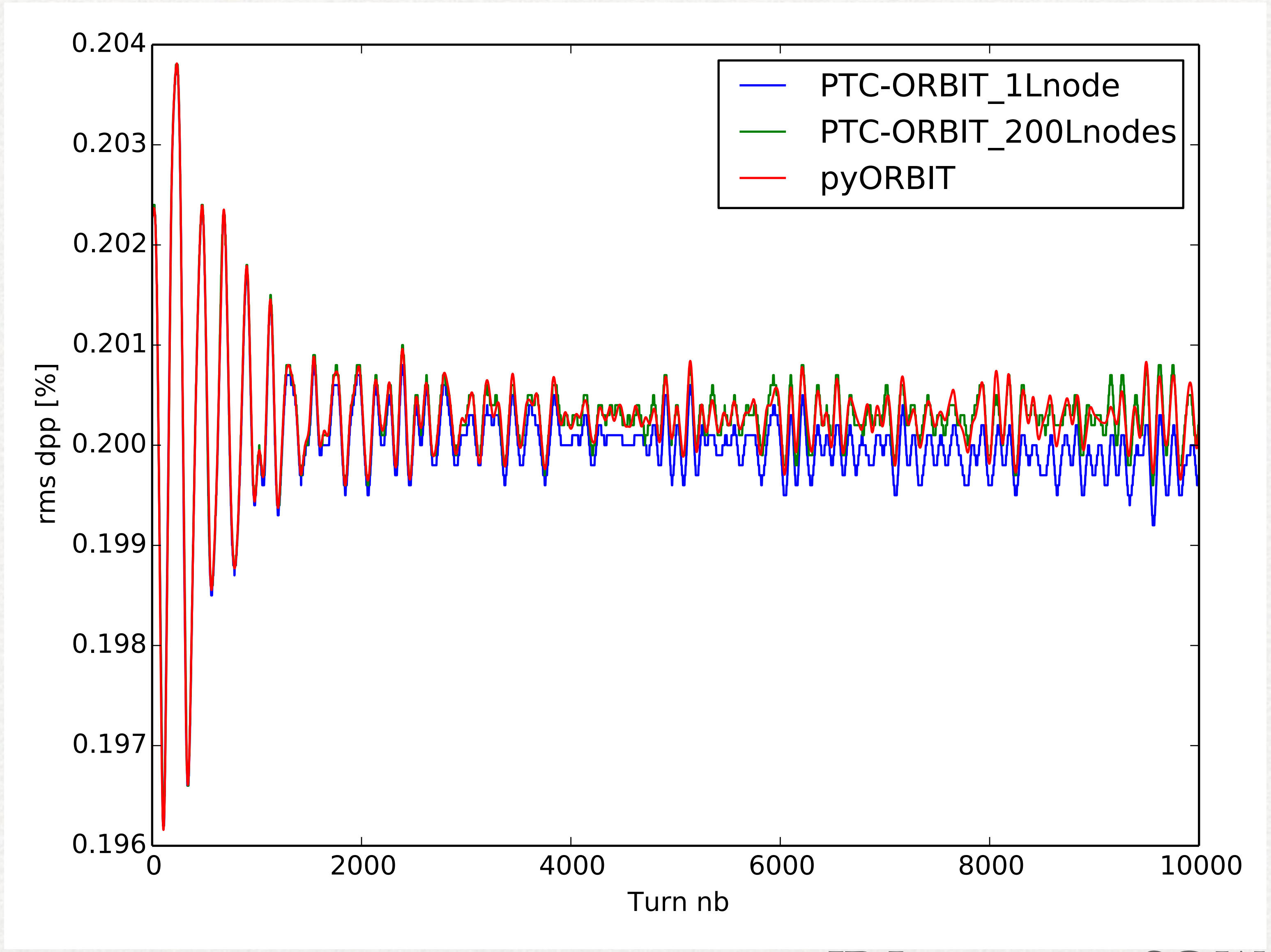
pyORBIT

/afs/cern.ch/project/LIUsc/space_charge/Codes/py-orbit_revison1291_dev_FrozenPIC

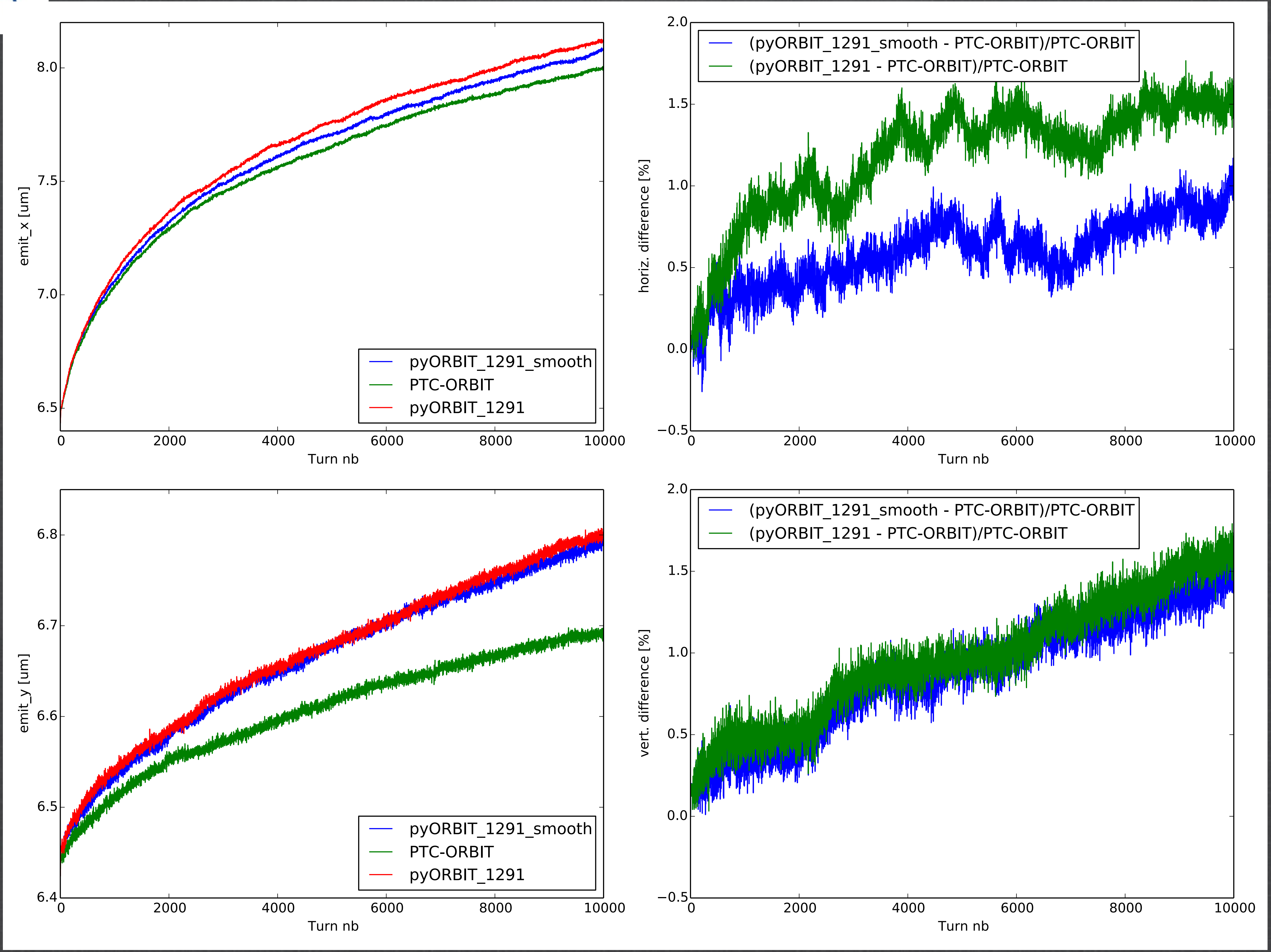
- 100 000 tracked particles,
- The 2.5D space charge (SC) method is used in both codes.
- 100 000 particles,
- 10 000 turns in the PS Booster (nominal case),
- SC grid 128x128x64

Longitudinal binning update

PTC-ORBIT updates longitudinal binning only in the longitudinal SC node, and accepts only 1 LSC in the lattice. If we put 200 LSC in PTC-ORBIT, the difference in momentum spread is corrected.



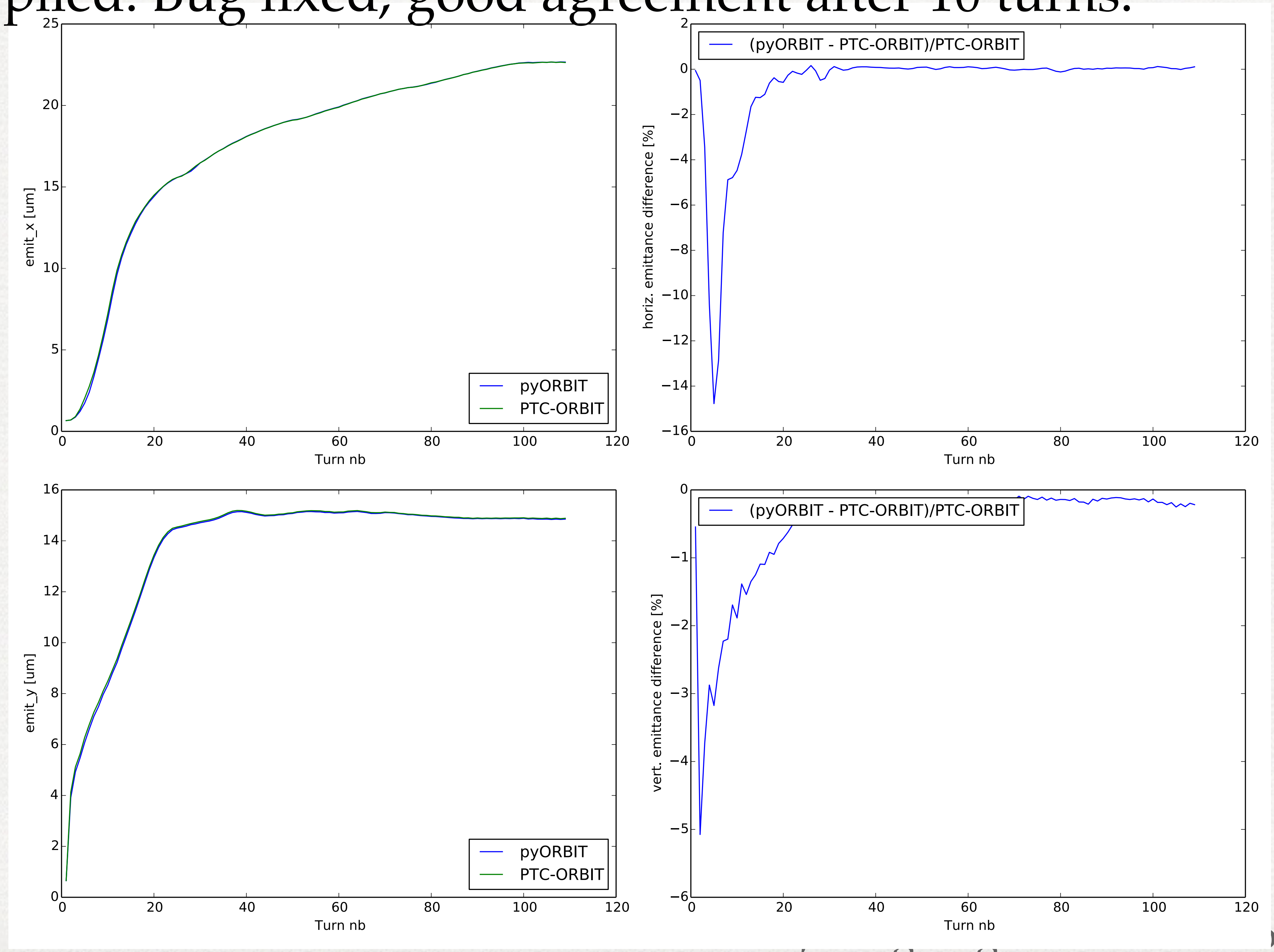
Smooth longitudinal binning



Injection and acceleration

(5 10^5 particles injected over 100 turns and tracked for 10 turns)

Bug found in pyORBIT, when Twiss parameters are updated in the presence of acceleration, the children nodes are reinitialized and no SC kick is applied. Bug fixed, good agreement after 10 turns.



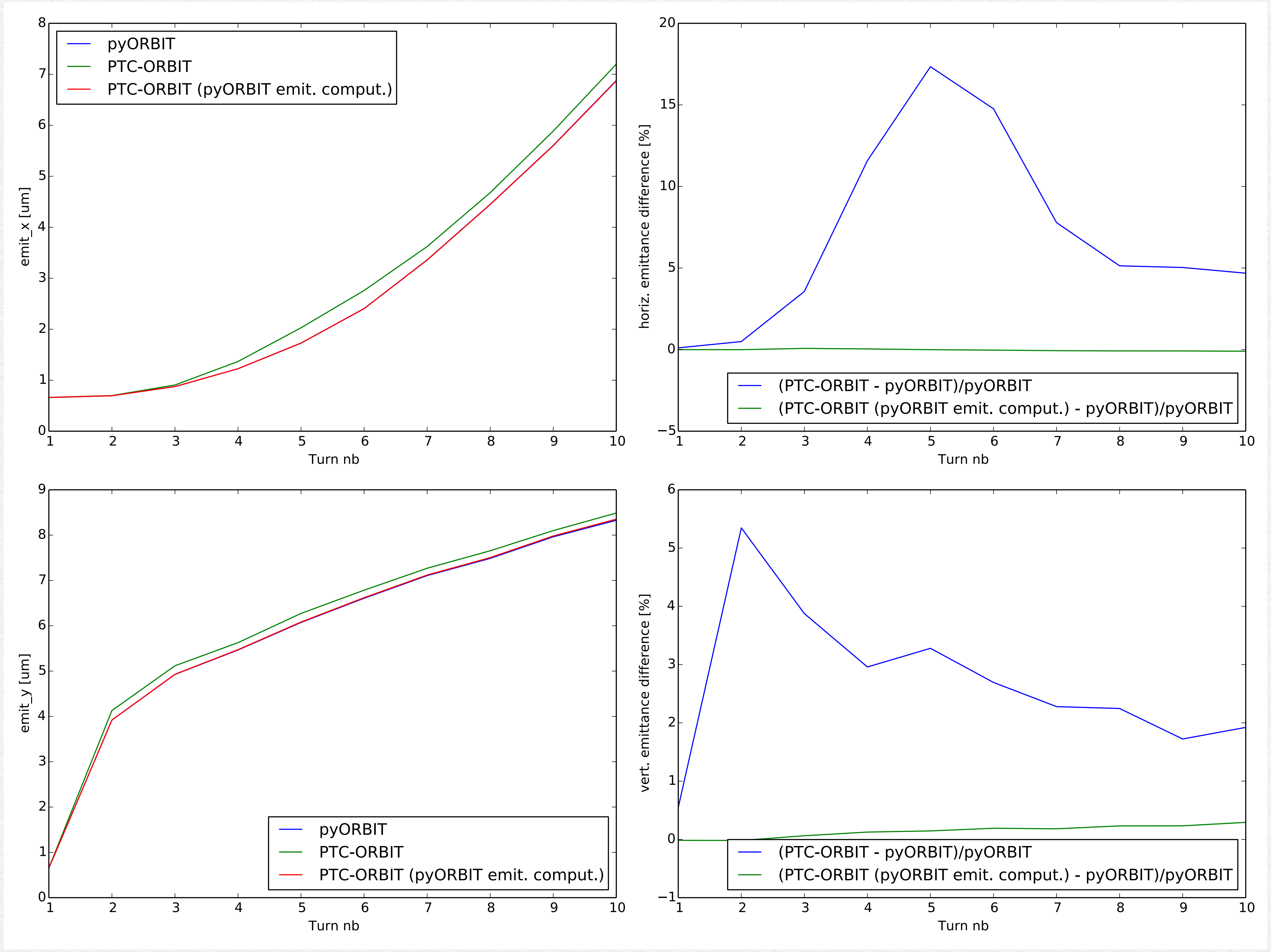
Difference in emittance computation

in pyORBIT, the emittance is computed only from the beam parameters:

$$\varepsilon_x = \sqrt{\left(cov(x,x) - \frac{cov(x,\Delta E)^2}{cov(\Delta E,\Delta E)}\right)\left(cov(x',x') - \frac{cov(x',\Delta E)^2}{cov(\Delta E,\Delta E)}\right) - \left(cov(x,x') - \frac{cov(x,\Delta E)cov(x',\Delta E)}{cov(\Delta E,\Delta E)}\right)^2}$$

in PTC-ORBIT, the emittance is computed with the ring dispersion for dealing with the momentum spread in horizontal.

Injection and acceleration (10 first turns)



JB Lagrange - SC Workshop 2017

Summary

- Good agreement between the 2 codes
- Some bugs and inconsistencies found in both codes and corrected in pyORBIT