Python ORBIT in a Nutshell

Jeff Holmes
Oak Ridge National Laboratory
Spallation Neutron Source

Space Charge MiniWorkshp
CERN
May 20, 2014

Python ORBIT (py-orbit)

- py-orbit is a collection of computational beam dynamics models for accelerators, designed to work together in a common framework.
- It was started as a "friendly" version of the ORBIT Code, written using publically available supported software.
  - Users interface with code via Python scripts.
  - Computationally intensive code in C++ (mostly, PTC is in Fortran).
  - Wrappers make C++ routines available to users at Python level.
  - Uses MPI for multiprocessing.
- py-orbit source code is publically available via Google Codes:
  - svn checkout https://py-orbit.googlecode.com/svn/trunk py-orbit --username youraccount@gmail.com
  - It is not difficult to develop your own extensions to py-orbit. We welcome responsible participation in developing py-orbit models.
- py-orbit people at present:
  - Owners: Andrei Shishlo (ORNL), Sarah Cousineau (ORNL), Jeff Holmes (ORNL)
  - Committers: Sabrina Appel (GSI), Oliver Boine-Frankenheim (GSI), Hannes Bartosik (CERN), Timofey Gorlov (ORNL)
  - Additional user: Sasha Molodozhentsev (KEK)

What Does py-orbit Do?

- Most things that ORBIT does for rings and transfer lines
  - Single particle tracking
    - Native symplectic tracker
    - PTC tracking
    - 3D field tracker
    - Linear tracking with matrices (this summer)
  - Space charge
    - Longitudinal
    - 2D potential and direct force
    - Full 3D (not parallel) and 3D ellipses
    - 2.5D (this summer)
  - Impedances
    - Longitudinal
    - Transverse dipole (this summer)
  - Injection, painting, RF cavities, collimation, apertures
- Linac Modeling
  - RF cavities
  - Magnets
  - Full 3D space charge (not parallel) and 3D ellipses
- Laser Stripping, Nonlinear Optics, Electron Cloud

What's Good About py-orbit?

- Entire source code is available. Uses standard Python, C++, and Fortran (PTC). Extra libraries only for FFTs and PTC.
- User is free to develop specialized or extended models to suit his/her own needs.
- With permission of owners, users' models can be incorporated into public version.
- Many examples demonstrate use of models in scripts. Some documentation in Googlecode wikis.
- Bunch class is extendable:
  - Basic bunch has macroparticle coordinates in 6D.
  - User can add various properties:
    - Particle ID tag
    - Spin
    - Species, ionization number, excited state, etc.

What's Bad About py-orbit?

- Code is not yet complete. Missing:
  - Linear tracking with matrices (this summer)
  - 2.5D space charge (this summer)
  - Transverse dipole impedance (this summer)
  - Electron Cloud
- Documentation is incomplete:
  - Although lots of examples illustrate use of modules and making of scripts,
  - Some of models are documented in Googlecode wikis, but the rest needs to be done.
- No "professional" full-time support:
  - py-orbit development has been carried out by working accelerator physicists (owners) trying to model and solve problems.
  - Only one of us (Shishlo) is a true master of computer science.
  - We all have to work at our "day jobs".

Why Consider Using py-orbit?

- Standard current software.
- Wide range of accelerator modeling capabilities.
- Everything is benchmarked.
- ORBIT approach developed over more than a decade to solve practical, as well as idealized, accelerator problems. It’s (almost) all now in py-orbit.
- You can extend it for your own needs.
- We’ll try to help with your questions.
- There is interest at a number of facilities.
- We at SNS are migrating to it as our number one beam dynamics code. We will support it in the future.

Thank you