ORBIT: BEAM DYNAMICS CALCULATIONS FOR HIGH-INTENSITY RINGS

J.A. Holmes, V. Danilov, J. Galambos, A. Shishlo, ORNL, Oak Ridge, TN 37831, USA
S. Cousineau, Indiana University, Bloomington, IN 47405, USA
W. Chou, L. Michelotti, F. Ostiguy - FNAL, Batavia, IL 60510, USA
J. Wei, BNL, Upton, NY, 11973, USA

Abstract

We are developing a computer code, ORBIT, specifically for beam dynamics calculations in high-intensity rings. Our approach allows detailed simulation of realistic accelerator problems. ORBIT is a particle-in-cell tracking code that transports bunches of interacting particles through a series of nodes representing elements, effects, or diagnostics that occur in the accelerator lattice. At present, ORBIT contains detailed models for strip-foil injection including painting and foil scattering; rf focusing and acceleration; transport through various magnetic elements; longitudinal and transverse impedances; longitudinal, transverse, and three-dimensional space charge forces; collimation and limiting apertures; and the calculation of many useful diagnostic quantities. ORBIT is an object-oriented code, written in C++ and utilizing a scripting interface for the convenience of the user. Ongoing improvements include the addition of a library of accelerator maps, BEAMLINE/MXYZPTLK; the introduction of a treatment magnet errors and fringe fields; the conversion of the scripting interface to the standard scripting language, Python; and the parallelization of the computations using MPI. The ORBIT code is an open source, powerful, and convenient tool for studying beam dynamics in high-intensity rings.

1 OVERVIEW

High-intensity proton ring beams are characterized by low energy, high intensity, and low loss requirements. Satisfying the beam-loss requirements necessitates a detailed understanding of beam dynamics in this regime. At high intensity, collective effects due to space charge and wakefields strongly affect the beam behavior, and single particle models do not suffice. Also, because of the complexity of collective phenomena for bunched beams in high-intensity rings, a computational approach is productive for theoretical studies and indispensable in solving detailed design and engineering problems.

Recognizing this, the SNS Accelerator Physics Group at ORNL, with help from colleagues at BNL, undertook the development of an object-oriented general-purpose code, ORBIT [1,2]. ORBIT began as a C++ rewrite of ACCSIM [3], developed under the SuperCode driver shell [4], but has since undergone extensive independent development, which will be described here.

ORBIT is a particle tracking code in 6D phase space. Its basic classes are *herds*, which are groups of particles, and *nodes*, which operate on the herds. ORBIT has been designed to simulate real machines: it has detailed models for transport through various lattice elements; injection foil and painting; rf and acceleration; 2.5D and 3D space charge; longitudinal impedance and space charge; transverse impedance; apertures: collimation; and beam diagnostics.

Present work on ORBIT focuses on the exploitation of some of the more recent and computationally demanding physics modules and on the enhancement of overall capabilities. With the completion of the transverse impedance, 3D spacecharge, and collimation routines, the physics for new classes of problems is available. However, meaningful calculations will require high performance computing techniques, such as the implementation of parallel computing. Also, because ORBIT has developed with an emphasis on collective effects, single particle transport models are still rudimentary. In order to assess losses to the accuracy demanded in recent and future high intensity rings, high order maps and a comprehensive treatment of magnet errors are necessary. Finally, although SuperCode is an innovative tool, it is neither standard nor supported. Comparable tools, based on the Python programming language, now are standard and present an attractive option.

Consequently, we are now developing parallel algorithms for collective beam dynamics, including higher order single particle maps, creating magnet error routines, and planning to implement a new driver shell based on Python. We now describe the models in ORBIT.

2 COMPLETED ORBIT MODELS

The coordinate representation in ORBIT utilizes the usual accelerator expansion. The horizontal direction is represented by x, x', the vertical direction by y, y', and longitudinal phase space by the phase angle ϕ, and energy deviation, dE, from a specified closed orbit reference particle. The independent variable is the machine location s.

ORBIT can construct lattices by reading output files from MAD [5] or DIMAD [6], by direct specification, or by specifying a uniform focusing channel. Linear transport is carried out through symplectic matrix multiplication. Nonlinear elements can be evaluated in the thin lens approximation, and higher order single particle transport terms, such as chromaticity, can be evaluated using second order transport matrices. There is no specific facility for the treatment of errors or fringe fields. Field errors in straight lattice elements can be calculated in MAD and included in the MAD output.

ORBIT can inject particles turn-by-turn or utilize a complete distribution specified at the start. A variety of distributions can be generated internally. Any externally generated distribution can be read in. Injection painting is treated with user-defined time-dependent closed orbit bumps. ORBIT contains an injection foil model that keeps track of foil hits and applies transverse kicks based on multiple Coulomb scattering. Particles that miss the foil at injection are removed from the beam.

The RF cavity model provides longitudinal kicks based on a specified time-dependent waveform with multiple harmonics. For nonaccelerating cases, the synchronous phase is assumed to be zero, and only the harmonics and time-dependent voltages need to be specified. For accelerating cases, the harmonics, voltages, and time-dependent dipole fields must be specified. The model uses this information to calculate the synchronous phase and the resulting kicks. The transverse phase space is also adjusted to conserve normalized emittance.

The 2.5D space charge model is implemented as a series of kicks separated by other transport operations. Particles are binned in a 2D rectangular grid using a second order distribution scheme. The potential for the distributed charges is then solved on the transverse grid using a fast FFT solver. Conducting wall (circular, elliptical, or rectangular beam pipe) boundary conditions are then imposed using a method described in Ref. [7]. Particle kicks are obtained from second order interpolation of the potential, completing what might be called a "quasi-symplectic" evaluation. Finally, the kicks are weighted by the local longitudinal density to account for bunch factor effects. This is the reason we call the model 2.5D. There is also an alternative direct force (momentum-conserving) solver without beam pipe that uses a method described in Ref. [8].

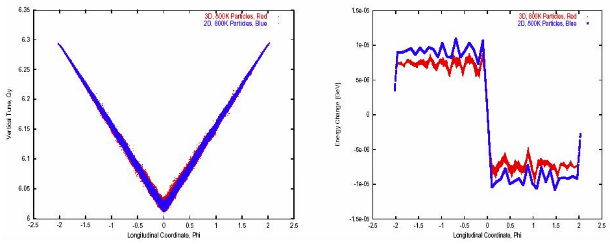

Figure 1. 3D space charge benchmark for triangular charge distribution: a) tune shifts vs. position, b) longitudinal force vs. position. 2.5D and 3D results agree with each other and with analytic calculations.

The 3D spacecharge model is a simple generalization of the 2.5D routine. Particles are distributed to a 3D rectangular grid using a second order scheme. Typically, for rings, the longitudinal spacing greatly exceeds the transverse spacing. The potential is solved as a 2D problem using the distributed charges and fast Fourier transforms on the transverse grid for each longitudinal slice. Conducting wall boundary conditions (circular, elliptical, or rectangular beam pipe) are used to "tie together" the transverse solutions into a 3D potential. Particle kicks are obtained by interpolating the potentials in 3D using a second order "quasi-symplectic" interpolation scheme.

ORBIT treats longitudinal impedances and/or space charge in a fashion similar to ESME [9]. The longitudinal impedance is represented as harmonics of the fundamental ring frequency. Particles are binned longitudinally and the binned distribution is Fourier transformed. The space charge contribution to the impedance is combined with the external impedance. The Fourier transformed distribution is multiplied by the impedance to give the longitudinal kicks to the particles. For many rings, the synchrotron period is much longer than a turn, and it is sufficient to evaluate the longitudinal impedance and space charge kicks once each turn. More frequent evaluations may be required for rings having higher synchrotron frequencies.

Transverse impedances are also treated as localized nodes in ORBIT. The validity of this approach requires the element length to be short compared to the betatron oscillation wavelength. Otherwise, multiple impedance nodes are required. As with the longitudinal impedance, the transverse impedance is represented by its Fourier components, but now the appropriate frequencies are the betatron sidebands of the ring frequency harmonics. For both impedance models, the formulation incorporates velocities below the speed of light. Particle kicks are obtained by convolution of the beam current dipole moment with the impedance. In evaluating the beam current it is assumed that the dipole moment evolves from the previous turn through a simple betatron oscillation.

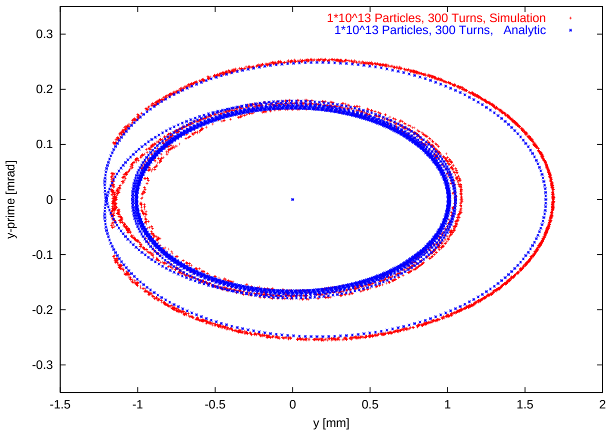

Figure 2. Phase space coordinates for benchmark of transverse impedance model with analytic calculation for pencil beam.

Circular, elliptical, rectangular, or racetrack apertures are defined in ORBIT. The apertures can be set either to allow particles to pass through and simply tabulate the hits, or to remove the particles from the beam and tabulate the loss locations.

ORBIT contains a detailed collimation model. Collimator shapes include the aperture shapes, single or multiple edges at arbitrary angles, and also a rectangular plate collimator that can be used for beam windows or foils. Physics includes multiple Coulomb scattering, ionization energy loss, nuclear elastic and inelastic scattering, and Rutherford scattering for a number of materials. Monte Carlo algorithms are used for particle transport inside the collimator, and step sizes are carefully adjusted near collimator boundaries.

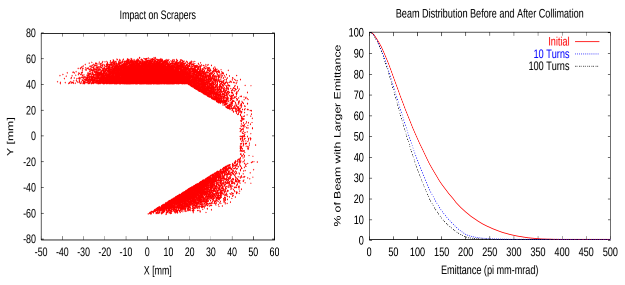

Figure 3. Results from SNS collimation study: a) particle impacts on beam scrapers, b) effects of collimation on emittance distribution.

A list of useful diagnostics in ORBIT includes the following: dumps of particle coordinates, tunes, or emittances at any point/node in the ring; histograms of particle distributions in x, y, phi, and emittance; rms emittances or beam moments versus turn or versus position; statistical calculation of beta functions; and longitudinal harmonics of the beam centroid. Because of the recent development of 3D models in ORBIT, we are extending our diagnostics to include more functions of longitudinal position.

3 ONGOING DEVELOPMENTS

The present emphasis in ORBIT development is on improvement of both physics and computational capabilities. In the physics category, we are significantly extending the single particle transport model by interfacing ORBIT and the BEAMLINE/MXYZPTLK library for maps and tracking [10]. In addition, we are including a comprehensive set of magnet error routines and a hard-edge fringe field model. At this time, the ORBIT - BEAMLINE/MXYZPTLK interface has been partly completed with some options working, and the errors will be included this summer.

The most urgent computational improvement to ORBIT is the introduction of parallel processing algorithms. We are doing this using the MPI libraries, and at present the parallelization of the 2.5D space charge routines has been carried out and parallel versions of the 3D space charge routines are being developed. We anticipate near-term completion of this work, to be followed by creation of parallel versions of the diagnostic routines.

In the slightly longer term, we plan to convert the ORBIT user interface and driver shell from SuperCode to Python. The feasibility of and methods for doing this have been examined, but the work remains to be carried out.

4 OPEN CODE

Finally, we emphasize that ORBIT is an open code. The source code can be obtained by downloading from http://www.sns.gov//APGroup/Codes/Codes.htm and there are also a user manual, instructions for building ORBIT, and examples at that web site.

5 ACKNOWLEDGMENT

SNS is managed by UT-Battelle, LLC, under contract DE-AC05-00OR22725 for the U.S. Department of Energy. SNS is a partnership of six national laboratories: Argonne, Brookhaven, Jefferson, Lawrence Berkeley, Los Alamos, and Oak Ridge.

REFERENCES

[1] J. Galambos, J. Holmes, D. Olsen, A. Luccio, and J. Beebe-Wang, *ORBIT Users Manual*, http://www.sns.gov//APGroup/Codes/Codes.htm
[2] J. Galambos, S. Danilov, D. Jeon, J. Holmes, D. Olsen, J. Beebe-Wang, and A. Luccio, in *Proceedings of the 1999 Particle Accelerator Conference*, (New York, 1999) 3143.
[3] F. Jones, *Users' Guide to ACCSIM*, TRIUMF Design Note, TRI-DN-90-17 (1990), http://www.triumf.ca/compserv/accsim.html
[4] S. W. Haney, *Using and Programming the SuperCode*, http://www.sns.gov/APGroup/Codes/Codes.htm
[5] H. Grote and F. Christoph Iselin, *The Mad Program, Version 8.19, User's Reference Manual*, CERN/SL/90-13, (Geneva, 1996).
[6]See ftp://csftp.triumf.ca/pub/CompServ/dimad/
[7] F. W. Jones, in *Proceedings of the 2000 European Particle Accelerator Conference*, (Vienna, 2000) 1381.
[8] R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles*, Institute of Physics Publishing (Bristol: 1988).
[9] J. A. MacLachlan, Longitudinal Phase Space Tracking with Space Charge and Wall Coupling Impedance, Fermi National Accelerator Laboratory, FN-446, (1987).
[10] L. Michelotti, *MXYZPTLK and BEAMLINE: C++ Objects for Beam Physics*, AIP Conf. Proc. No. 255 (Proc. Advanced Beam Dynamics Workshop on Effects of Errors in Accelerators, Their Diagnosis and Correction, Corpus Christi, Texas, 199