SIMULATION TOOLS FOR HIGH INTENSITY RINGS

S. Cousineau[*] , ORNL, Oak Ridge, TN 37830, USA

Abstract

Several codes have been developed specifically to simulate beam dynamics in high intensity rings. These codes contain detailed algorithms for modeling the relevant physics in the beam, from single particle transport to important collective effects, including space charge and impedances. Among the various codes, a number of different methods have been adopted for the treatment of space charge. The codes have been applied to a variety of problems in existing machines, as well as for the study, design, and optimization of future machines. A review of existing ring simulation codes is presented, with specific emphasis on the space charge implementation. A more in-depth description of the features and application histories of a few specific codes is given, as well as a summary of code benchmarks with experimental data.

INTRODUCTION

The low loss requirements for future high intensity rings will require detailed knowledge of beam dynamics in the machine. Large scale computational models provide an invaluable tool for simulating the multitude of processes that can contribute to beam loss. In particular, in the regime of high beam intensities and low beam energies, collective effects such as space charge and impedances have a significant effect on the beam behavior. In order to predict beam loss at the levels required by future machines, i.e., tiny fractions of the beam intensity, the simulations must realistically account for the entire transport of the beam in the ring, from mapping through external magnetic fields to modeling of collective effects. Due to the complexity of this task, a computational framework is natural. Simulations of this type are productive in the analysis of instability thresholds, halo development, and emittance dilution in existing machines, and are equivalently useful in the design and optimization of future machines.

In the past few years, a wealth of simulation tools geared specifically for application to high intensity rings has been developed. The majority of these tools began as small, in-house codes written with a specific problem in mind, but have subsequently been upgraded into more general packages applicable across a broad range of problems. Each package, or code, typically contains models for a substantial subset of the relevant physical process: injection, strip foil effects, RF capture and acceleration, transport through linear and nonlinear magnetic fields, field errors, fringe fields, space charge, impedances, boundaries and image charges, electron-proton interactions, apertures and collimation, modeling of non-standard hardware, diagnostic capabilities, etc.

A realistic simulation of a high intensity beam would ideally algorithms for all the relevant physics together with a large number of macro-particles to reduce numerical noise. However, because of the computational expense of many of the algorithms, especially the collective algorithms, one usually must limit either the number of models included, or the number of macro-particles. Typically, the most dominant effects on the beam are known apriori, and the algorithms representing these effects are prioritized over algorithms which contribute only marginally to the beam behavior. Recently, this problem has been alleviated somewhat by the implementation of parallel computing into many of the codes. The parallel framework allows for fast processing of large jobs, which may include multiple collective processes, as well as large numbers of particles ($\geq 10^6$). Though there is still progress to be made, many simulations can now realistically portray beam behavior in a high intensity ring.

With the capability to simulate real machines comes the opportunity to benchmark the codes against experimental data. Successful benchmarks with experiment are critical to establishing credibility of the models in the codes. Additionally, since codes are often used for the design of future machines, a successful benchmark with a current machine lends confidence to the predictions for future machines. Experimental studies at several existing machines have recently been undertaken, and effort is underway to involve a larger number of codes in more ambitious benchmark tests.

This paper presents a review of the current state-of-the-art in in simulation tools for high intensity ring beams. Aspects of the various treatments of space charge are highlighted. A brief summary of code capabilities for a representative sampling of codes is presented, and a discussion of specific problems to which the code has been applied is given. Summary tables of each code availability, space charge implementation, and code benchmark versus experiment are also shown.

SUMMARY OF AVAILABLE CODES

Information on many of the available ring space charge codes was presented at the ICFA Beam Dynamics Mini-Workshop on Space Charge Simulation [1]. Table 1 lists the codes and gives the details of code accessibility and documentation, programming language, appropriate platform for installation, parallelization status, and any additional external libraries required.

[*] 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.

Table 1: Summary of codes and portability.

CODE	Access Information	Language	Platform	Parallel Capability	External Libraries
ESME	www-ap.fnal.gov/ESME/	F77	Unix/Linux	Yes	None
LONG1D	www.triumf.ca/people/koscielniak/long1d.htm	F77	DEC/Linux	No	GPlot, CERN libs
Track 1D, 2D, 3D	c.prior@rl.ak.uk	F77	Any	No	None
SIMPSONS	shinji.machida@kek.jp	F77	Any	In progress	None
ACCSIM	www.triumf.ca/compserv/accsim.html	F77	Unix/Linux	No	CERN libs
ORBIT-ORNL	www.sns.gov/APGroup/Codes/orbit.htm	C++	Unix/Linux	Yes	MPI, SuperCode, MXYZTPLK, FFTW
ORBIT-BNL	luccio@bnl.gov	C++	Unix/Linux	Yes	MPI
ML/IMPACT	rdryne@lbl.gov	F90	Unix/Linux	Yes	MPI
GenTrackE	andreas.adelmann@psi.ch	C++	Unix/Linux	Yes	MPI
Best	hongqin@princeton.edu	F90	Unix/Linux	Yes	MPI, OpenMP
Synergia	spentz@fnal.gov	F90/C+	Unix/Linux	Yes	MPI

An important component of any code simulating high intensity ring beams is the space charge algorithm. Historically, space charge effects were considered important mainly in linear accelerators, where the peak density of particles is much greater than in rings. Several space charge codes exist for linacs, many of which have been extensively applied to space charge studies in current and future machines [2]. Recently, space charge has been recognized as an important phenomenon for rings as well. Even for very high intensity ring beams, the space charge force is small compared to that in linac beams. However, due to the periodic nature of rings and the long duration time spent there, the ring beam is sensitive to resonant excitations induced by space charge, and a relatively small space charge force can significantly alter the beam behavior. Implementation of space charge routines for rings began as early as 1980 [3]. The effort has increased dramatically in recent years, stimulated by projects such as the SNS machine [4] and the prospect of future high intensity proton drivers.

A variety of approaches to solving Poisson's equation for the ring beam have been developed, each with its own merits and drawbacks. One problem that all space charge algorithms must contend with is the computation expense of the problem, which normally occupies the largest single portion of CPU run time. This expense is rooted in the large number of macro-particles, the finely-spaced meshes, and the small integration lengths required by space charge solvers to provide accurate results with low numerical noise. To overcome this hurdle, many codes contain fast solvers which rely on a few approximations, or parallel processing capabilities, or both. Some of the more common approaches to solving Poisson's equation for the beam include various types of spectral methods (FFT convolutions in mode-space), finite element methods (FEM), or fast multipole method (FMM) techniques.

Table 2 gives a summary of routines and parameters relevant to the space charge implementation for the codes shown in Table 1 . For each code, the table lists the independent variable used for tracking, the dimension of the space charge model, the number of macro-particles that are typically tracked, the type of space charge algorithm, and whether any image or boundary charges can be included. Recall that the parallel capability of the codes is listed in Table 1.

DETAILS OF SOME SPECIFIC CODES

In this section, a brief discussion of some of the codes listed in Tables 1 and 2 is given. In addition to summarizing capabilities or highlighting key features, a few of the problems to which the codes have been applied are also presented. Unfortunately, due to space considerations, it is not possible to present separate reviews all of the codes listed in Tables 1 and 2. The discussion below is intended as representative sampling of codes and some of their applications, not a comprehensive survey.

ESME

The ESME code is designed for tracking in longitudinal phase space of a beam [5], and for other aspects of a proton synchrotron that are governed by RF. The code follows the beam distribution in energy and azimuth on a turn-by-turn basis, where the independent variable for tracking is the revolution time of the synchrotron. The code began development in the early 1980's, and has been extended considerably since this time, with implementation of the space charge algorithm in 1986. ESME now contains comprehensive capabilities for modeling the physics relevant to the longitudinal dynamics of the beam, including, but not limited to,

- Energy ramps: linear, parabolic, cubic, biased sinusoid, user defined
- Arbitrary phase and voltage curves, multiple RF sources, exact sinusoidal, non-sinusoidal
- Lattice nonlinearity via $\Delta p/p$ expansion

Table 2: Summary of routines and parameters relevant to space charge implementation.

CODE	Independent Variable	Dimension of SC Model	Number of Particles	Space Charge Model	Boundaries & Images
ESME	revolution time	1D long.	10^7	smoothed $\frac{d\lambda}{dz}$	Circular conducting wall
LONG1D	revolution time	1D long.	10^6	smoothed $\frac{d\lambda}{dz}$	Conducting wall
Track 1D, 2D, 3D	revolution time; s; s	1D; 2D; 3D	10^6	smoothed $\frac{d\lambda}{dz}$; FEM; FEM	Variable g-factor; Automatic treatment; Automatic treatment
SIMPSONS	t	2D, 3D	$\geq 10^4$	FFT in cylin. coords.	Circular conducting wall
ACCSIM	s	2.5D	$\geq 10^5$	Hybrid FMM	Conducting wall
ORBIT-ORNL	s	1D, 2.5D, 3D	$\geq 10^6$	FFT, force or potential	Conducting wall
ORBIT-BNL	s	1D, 2D, 3D	10^6	SU + LOR	Conducting wall, automatic images
ML/IMPACT	s	3D	$\geq 10^6$	Spectral	None
GenTrackE	t/s	3D	$\geq 10^9$	FEM, Multi-grid	Periodic, Dirichlet or Neumann, mixed
Best	t	3D	$\geq 10^6$	Spectral, FD	Circular conducting wall
Synergia	s	3D	$\geq 10^6$	Spectral	Open, periodic, rectangular, circular

- Momentum aperture checking
- Machine to machine transfers
- γ_t jump and RF phase reversal
- Imaginary γ_t lattices
- Voltage and phase feedback
- Space charge
- Wall impedances
- Graphics and diagnostics.

Having been in use for quite some time now, there is an exhaustive list of problems to which ESME has been applied. For example, it has been used at the Fermilab Main Ring and Injector to study bunch coalescing, phase lock, and slip stacking; at the Fermilab Booster for studies of transition crossing, γ_t jump, and beam instabilities; at the SNS in benchmark studies with the ORBIT code and for bunch shape manipulation in the SNS accumulator ring; and for a multitude of studies at other machines, including the CERN PS and SPS, the Pimms Medical Synchrotron, Petra (for protons), and a future proton driver machine.

TRACK 1D, 2D, and 3D

The Track codes consist of three separate codes for tracking in the longitudinal dimension (Track1D), in the transverse dimensions (Track2D), and in all three dimensions (Track3D) [6]. Like the ESME code, Track2D was developed in the early 1980's, and has a long history of additions, revisions, and applications. It was possibly the first particle-pushing code to perform multi-turn injection with full nonlinear space charge, this work having been performed for fusion studies in 1980 [3]. An essential feature of the space charge implementation is that it is based on a finite element approach, which allows for easy handling of complicated geometries. The algorithms also provides automatic treatment of image charges for open, periodic, elliptical, polygonal, and lossy boundaries.

As main in-house codes at the Rutherford-Appleton Laboratory, the Track codes have been extensively applied in upgrade and development studies of the ISIS machine. These studies have resulted in upgrade of the injection system and the design and installation of a new dual-harmonic RF system, due to come on line in the next year. Additionally, the Track codes have been used for studies of injection, funnel optimization, and chopper optimization in the ESS design, for studies at CERN machines (PS, PS Booster, and SPS), for simulations of a future proton driver at Fermilab, and for studies of a future neutrino factory.

ACCSIM

The ACCSIM code is unique from many of its predecessors in that it was specifically developed as a "ready-to-run" package for general use in the study, design, and optimization of accelerator rings and transport lines [7]. The code is comparably well-documented and a detailed User's Manual is available [8]. Not only is ACCSIM attractive from the standpoint of portability and user-friendliness, it also has a comprehensive suite of physics models. Some of these models are:

- Injection: internal distribution generators, painting,

foil effects
- First order tracking and thin element capabilities
- RF and acceleration, barrier RF
- Apertures for any element, collimators and targets
- Pick-up and damper systems
- Space charge: longitudinal, 2.5D transverse
- Image charges
- Many diagnostics, built in graphics.

The space charge routine is a hybrid FMM routine which yields computational times scaling with N_g, the number of grid points. This is a significant improvement over the scaling of the pure FMM routine, which depends on the number of particles, N_p [7]. The reference to 2.5D in the space charge routine refers to the fact that the transverse and longitudinal directions are quasi-independent, with coupling induced from length terms and longitudinal density factors. A new improvement to ACCSIM is the inclusion of a MAD parser into the program. This is currently available in the newest beta-release version, ACCSIM 4.

ACCSIM has been used at many laboratories for a variety of ring machine studies. Benchmarks with experimental data have been performed for the CERN PS Booster and the KEK 12 GeV PS, both of which yielded successful results. Additionally, ACCSIM has been used for studies of injection at the Tsukub Hitachi medical synchrotron, for studies of injection and space charge in the future SNS 1 GeV accumulator ring, and for studies of injection and collimation in the J-PARC 3 GeV ring, to name just a few.

ORBIT

Historically, ORBIT began as a C++ rewrite of the F77-based ACCSIM code [9]. The motivation for the rewrite was two-fold: to provide a modular structure for programming, and to incorporate a driver shell for "on-the-fly" scripting. SuperCode was the driver shell chosen at the time and is still in use in the Oak Ridge version of ORBIT, ORBIT-ORNL. However, an effort is underway to replace SuperCode with a more standard, well-documented Python interface; the completion of this work is imminent.

Owing mainly to the modular structure of the code, ORBIT-ORNL has been developed as a collaborative effort between many authors. The modularity allows for the easy addition of new source code (a module), which acts independently from other modules. This structure helps a developer isolate and identify errors, and allows a user to pick and choose among existing modules to formulate a combination appropriate to the problem at hand. Among the available models in the ORBIT-ORNL code are:

- Injection: internal distribution generators, painting, foil effects
- Single particle transport: First or second order tracking, symplectic TEAPOT-type maps, or Mxyzptlk library. library.
- RF and acceleration
- Magnet errors and fringe fields
- closed orbit calculation and correction
- Apertures and collimators
- Space charge: 1D longitudinal, 2.5D force or potential, or 3D potential
- Image charges
- Transverse and longitudinal impedances
- Feedback and stabilization
- Diagnostics

ORBIT-ORNL has been the main code in use for simulations of the future SNS accumulator ring beam. It has been used for injection optimization, for collimation system design and optimization, for studies of halo formation from space charge, impedances, and resonance crossing, for development of a feedback and stabilization system, and for studies of loss distributions around the ring. Outside of the SNS project, ORBIT-ORNL was the main tool applied in a comprehensive study of space charge effects in the PSR ring [10], and is now in use for space charge studies of the Fermilab Booster ring.

The ORBIT-ORNL version presented above is housed at Oak Ridge National Laboratory and is different than the version housed at Brookhaven National Laboratory, ORBIT-BNL [11]. Both versions originated from the first ORBIT rewrite of ACCSIM, but eventually two different codes emerged under separate developers. ORBIT-BNL incorporates many of the same capabilities as the ORBIT-ORNL, including full 3D space charge, parallel capabilities, and impedances, but it no longer makes use of the driver shell format. Additionally, the algorithms for many of the routines differ, as demonstrated by Table 2. Though ORBIT-BNL uses position (s) as the independent variable for tracking, the space charge algorithm is formulated to remove the resulting approximation in the time domain. ORBIT-BNL also been used for machine studies, including simulations of the SNS accumulator ring, of the AGS, the AGS Booster, and the RHIC machine at Brookhaven, and also for simulations of SIS at the GSI.

New Directions: IMPACT-MaryLie, GenTrackE, UAL

A few codes that are either new to space charge simulation in rings or will be entering the scene soon are IMPACT-MaryLie, GenTrackE, and UAL.

The linear accelerator code IMPACT has recently been merged with the MaryLie package to provide particle tracking in rings. The new code, IMPACT-MaryLie [12], contains relevant modules from the original IMPACT code, as well as two new space charge computation methods: a Hermite-Gaussian expansion model, and a cellular analytic convolution model. Both of these models are designed to minimize the field error induced when solving Poisson's equation for beams with extreme aspect ratios.

GenTrackE is a framework for particle tracking applications which aims to provide good scalability for very large parallel processing jobs. The code features either truncated

Table 3: Summary of code benchmarks with experimental data.

Code - Machine	Quantity Compared	Agreement Level
ESME - Fermilab Booster	Longitudinal parameters	Good
ESME - CERN PS	Longitudinal parameters	Good
Track1D - ISIS	Longitudinal parameters	Good
ACCSIM - CERN PS	Beam profiles	Good
ACCSIM - KEK PS	Beam profiles	Good
ORBIT-ORNL - PSR	Beam profiles	Good
SIMPSON - KEK Booster	Emittance exchange	Good
ORBIT-ORNL - Fermilab Booster	Emittance growth	Under study
Best - PSR	E-cloud effects	Fair

power series maps or time integration methods for tracking, and an finite-element approach to space charge with semi-unstructured griding. Currently, the code has be run with up to 7×10^9 grid points. One application making use the GenTrackE framework is a parallelized, self-consistent electron-cloud model [13], adapted from M. Furman and M. Pivi's probabilistic model [14].

The Unified Accelerator Libraries, or UAL, is not a particle-tracking code by itself, but is rather a collection of libraries that address diverse accelerator tasks [15]. Methods included in the libraries are derived from other codes, such as ACCSIM, TEAPOT, Zlib, etc. The official version of UAL does not currently contain a space charge implementation, but a non-official local version at BNL has been modified to provide the ORBIT-ORNL space charge routines. The modification has been successfully used for studies of the SNS beam, and future plans for UAL include adding this capability to the official version.

SUMMARY OF BENCHMARKS WITH EXPERIMENT

An important test of the validity of the models incorporated into any simulation tool is whether the model can be successfully benchmarked with experiment. Unfortunately, only a limited number of beam parameters can be evaluated through experimental measurement, and the experimental data is often complicated by effects that are not easily separated from the physical process under study. The task in any benchmark is to extract meaningful measurements where either a single physical phenomenon is isolated, or in which the effects of other phenomena can be easily identified. Many of these experiments have been performed and the data has been used for benchmarking of codes. Table 3 summarizes a number of these benchmarks.

Overall, the success rate for benchmark of simple quantities, such as beam profiles, is high. However, there is a distinct lack of benchmarks with more complicated quantities, such as emittances and coherent tunes. Toward this end, a plan was formulated at the ICFA Beam Dynamics Mini-Workshop on Space charge Simulation for a multiple code benchmark (ORBIT, ACCSIM, IMPACT-MaryLie, GenTrackE, and Best) with existing experimental data from the CERN PS [16, 17]. Two data sets have been identified for the study: 1) Emittance measurements while crossing the integer and half-integer resonances, and 2) measurement of emittance exchange while crossing the fourth-order Montague resonance, $2Q_x - 2Q_y = 0$.

REFERENCES

[1] http://www.isis.rl.ac.uk/AcceleratorTheory/workshop/workshop.htm

[2] R. Ryne *et al*, *Proceedings of the 2001 Particle Accelerator Conference*, (Chicago, 2001) 264.

[3] C. Prior, private communication.

[4] N. Holtkamp *et al*, *Proceedings of the 2002 European Particle Accelerator Conference*, (Paris, 2002) 164.

[5] http://www-ap.fnal.gov/ESME/

[6] C. Prior, RAL Technical Report, RAL-TR-1998-048.

[7] F.W. Jones, *AIP Conference Proceedings*, **448**, 359 (AIP, NY, 1998).

[8] F. Jones, Users' Guide to ACCSIM, TRI-UMF Design Notes, TRI-DN-90-17 (1990), http://www.triumf.ca/compserv/accsim.html.

[9] J. D. Galambos *et al*, The ORBIT User's Manual, http://www.sns.gov//APGroup/Codes/Codes.htm.

[10] S. Cousineau, *Understanding Space Charge and Controlling Beam Loss in High Intensity Synchrotrons*, PhD Thesis, Indiana University, 2003, unpublished; "Studies of Resonant Beam Behavior in the PSR," PRST-AB, submitted.

[11] A. Luccio *et al*, *AIP Conference Proceedings*, **448** 390 (Shelter Island, 1998).

[12] R. Ryne, "MaryLie/IMPACT: A Parallel 5th Order Beam Optics Code with Space Charge," *Proceedings of the 2003 Particle Accelerator Conference*, (Portland, 2003).

[13] M. A. Furman and A. Adelman, "Self-Consistent Parallel 3D Electron-Cloud Simulation in Arbitrary External Fields," *Proceedings of the 2003 Particle Accelerator Conference*, (Portland, 2003).

[14] M.A. Furman and M.T.F. Pivi, PRST-AB **5** 124404 (2002).

[15] www.ual.bnl.gov

[16] E. Metral, "Measurements of Transverse Space Charge Effects in the CERN Proton Synchrotron," *Proceedings of the 2003 Particle Accelerator Conference*, (Portland, 2003).

[17] I. Hofmann, "Nonlinear Resonance Benchmarking Experiment at the CERN Proton Synchrotron *Proceedings of the 2003 Particle Accelerator Conference*, (Portland, 2003)