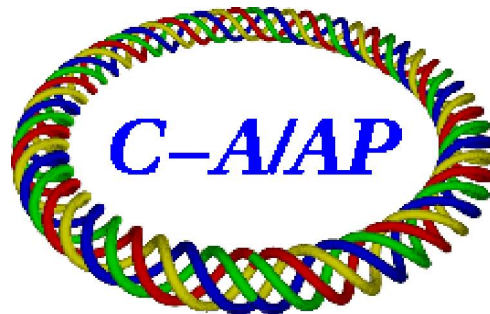# SimTrack: A compact c++ library
for particle orbit and spin tracking in accelerators

Y. Luo

Collider-Accelerator Department
Brookhaven National Laboratory
Upton, NY 11973

U.S. Department of Energy
Office of Science, Office of Nuclear Physics

SimTrack: a compact c++ library
for particle orbit and spin tracking in accelerators

Yun Luo

Brookhaven National Laboratory, Upton, NY 11973, USA

SimTrack is a compact c++ library of 6-d symplectic element-by-element particle tracking in accelerators originally designed for head-on beam-beam compensation simulation studies in the Relativistic Heavy Ion Collider (RHIC) at Brookhaven National Laboratory. It provides a 6-d symplectic orbit tracking with the 4th order symplectic integration for magnet elements and the 6-d symplectic synchro-beam map for beam-beam interaction. Since its inception in 2009, SimTrack has been intensively used for dynamic aperture calculations with beam-beam interaction for RHIC. Recently, proton spin tracking and electron energy loss due to synchrotron radiation were added. In this article, I will present the code architecture, physics models, and some selected examples of its applications to RHIC and a future electron-ion collider design eRHIC.

1 Introduction

The Relativistic Heavy Ion Collider (RHIC) at Brookhaven National Laboratory collides heavy ions and polarized protons. In proton-proton operation, the current working point is constrained between 2/3 and 7/10 in the tune space. To further increase the luminosity with increased proton bunch intensity, two electron lenses have been installed to compensate the large beam-beam tune spread and nonlinear beam-beam resonances [1, 2]. To understand the effects of electron lenses on the single proton beam dynamics and on the proton beam loss and emittance growth, massive numeric simulation studies are needed. SimTrack was originally designed to meet these requirements.

Back in 2008, some codes for proton tracking already existed, such as SixTrack [3], BBSIM [4], LIFE-TRAC [5]. SixTrack is a 6-d element-by-element tracking code with a fast computing speed. To achieve the fast speed, SixTrack concatenates so-called linear elements such as drifts, bends, and quadrupoles into linear blocks and sandwich zero-length (or thin-lens) nonlinear magnetic elements between blocks. SixTrack is written in Fortran 77 and is an executable which reads in several input files and dumps many output files. We found that it is not straight forward for us to create new types of elements, such as an electron lens, and it is not flexible enough to manipulate elements and correct optics before particle tracking. BBSIM and LIFETRAC are c++ and Fortran libraries for proton particle tracking respectively. Both of them do not calculate the linear optics. The thin-lens nonlinear magnetic elements are inserted between linear matrices which had to be imported from other sources such as MADX [6] or generated analytically with Twiss parameters.

SimTrack was written with the most powerful features of c++ , such as class definition, standard template library (STL) containers, and polymorphism. It is a compact and efficient library for 6-d symplectic element-by-element particle tracking in accelerators. SimTrack adopts the 4th order symplectic integration [7] for the optical transfer through magnetic elements. It accepts both thin and thick magnets. Exact or expanded Hamiltonian can be chosen. The symplectic 6-d synchro-beam map by Hirata [8] was adopted for beam-beam interaction. The first version of SimTrack was released in 2009 with about 6000 lines but it already met our requirements for particle tracking for head-on beam-beam compensation in RHIC. Over the years, linear optics calculation, some limited nonlinear optics calculation, tracking with exact Hamiltonian, proton tracking with spin, electron tracking with synchrotron radiation energy loss were added. Currently the source code has about 12000 lines.

In the following, I will first present the code architecture of SimTrack, followed by the physics models to guarantee the 6-d symplectic particle tracking. Then I will give a few selected examples of its application to RHIC and a future electron-ion collider design eRHIC.

2 Code Architecture

2.1 Element Definition

SimTrack supports a wide spectrum of element types. Each type of element is a derived class of a base element class. The base element class is an abstract class. It holds parameters common to all element types and defines the common interface and transfer functions.

The abstract element class defines the name, type, length, location, offset and tilt, and physical aperture of elements. It also defines the 6-d closed orbit, 3-d spin closed orbit, linear betatron Twiss and coupling parameters, 6×6 linear transfer map, linear one-turn map and so on.

For particle transferring through an element, we define three functions: Pass() to transfer 6-d orbit coordinates, DAPass() to transfer linear truncated power series (tps), and sPass() to transfer 9-d particle coordinates which includes 6-d orbit coordinates and 3-d spin vector. All the particle transfer functions defined in the abstract element are virtual, which means that their actual operations will be deferred in the definitions of derived classes. The particle transfer functions for different types of elements are normally different.

The definition of the abstract element class is :

```
class Element
{
 public:
  Element(string name);
  string  NAME, TYPE;
  double  S, L, DX, DY, DT;
  double  APx, APy;
  double  X[6], n0[3], T[36], M[36], A[16];
  double  Beta1, Alfa1, Beta2, Alfa2,  Mu1, Mu2,
          r, c11, c12, c21, c22;
  double  Etax, Etay, Etaxp,  Etayp;
  virtual void      SetP(const char *name, double value)=0;
  virtual double  GetP(const char *name)=0;
  virtual void      Pass(double x[6])=0;
  virtual void      DAPass(tps x[6])=0;
  virtual void      sPass(double x[9])=0;
};
```

The derived element class inherits all the parameters and functions of the abstract base element class. To create a derived element class or a new type of element, we can define additional private parameters only relevant to this type of element. For example, for the quadrupole class, we need to define its normal and skew focusing strengths $K1L$ and $K1SL$. We call them specific parameters of elements. We also need to explicitly define the particle transfer functions Pass(), DAPass(), and sPass(). Besides them, we explicitly define two interface functions SetP() and GetP() which allow us to set and access the specific parameters of elements.

Here I list the elements currently supported by SimTrack and their specific parameters:

```
DRIFT:
SBEND: ANGLE,E1,E2
QUAD : K1L,K1SL
SEXT : K2L,K2SL
SKEWQ: K2SL
OCT  : K3L,K3SL
MULT : KNL[11],KNSL[11]
GMULT: ANGLE,E1,E2, KNL[11],KNSL[11]
KICKER: HKICK,VKICK
HKICKER: HKICK
VKICKER: VKICK
HACDIP : TURNS,TURNE,HKICKMAX,NUD,PHID
VACDIP : TURNS,TURNE,VKICKMAX,NUD,PHID
ACMULT : NORDER,TTURNS,KL,PHIO
COOLING: ALPHA
DIFFUSION: DIFFX, DIFFY, DIFFZ
HBPM:
VBPM:
BPM:
MARKER:
RCOLL : RSIZE
ECOLL : XSIZE,YSIZE
SOLEN : KS
RFCAV : VRF, FRF, PHASEO
INSTR :
BEAMBEAM: NSLICE,NP,BBSCALE, TREATMENT,EMITX,EMITY,
          SIGMAL,BETAX, ALFAX, BETAY,ALFAY
LRBB:     NP,BBSCALE, TREATMENT,EMITX,EMITY,
          SIGMAL,BETAX, ALFAX, BETAY,ALFAY
ELENS:    NSLICE,NE,BBSCALE,BETAE,SIGMAX,SIGMAY,NSLICE
```

```
MATRIX:   M66[36],XCO_IN[6],XCO_OUT[6]
ELSEP:    EX, EY
```

2.2   Line Definition

Line is a sequence of elements, which can be a transfer line or a ring. In SimTrack, the class Line includes a vector container of elements and optics parameters relevant to lines. The number of elements of a line is not fixed. The parameters of the Line class include total length, element number, and transverse and longitudinal optics parameters. Since a line is defined as a vector container, it is straight forward to add, delete, or insert an element. Connection of two lines can be simply done with the "+" operator. Additional functions to split elements, to make thin, to concatenate neighboring drifts, and to replace zero-strength magnetic elements with drifts are also provided.

The definition of class Line is :

```
class Line
{
 public:
  Line();
  vector  <Element *> Cell;
  double Length;
  long    Ncell;
  double Tune1, Tune2, Tune3;
  double Chromx1, Chromy1, Chromx2, Chromy2, Chromx3, Chromy3;
  double frev0, Vrf_tot, Orbit_Length, Qs;
  double Alfa0, Alfa1, Alfa2, Gammat, Slip;
  double Bucket_length,Bucket_height,Bucket_area;
  double Bunch_length,Bunch_area,Bunch_height;
  double U0_SR;
};
```

To access the common parameters defined in the base element class, we use pointers. To set and access the specific parameters defined in the derived element classes, we call SetP() and GetP(). Here I give an example source code to demonstrate them. The following c++ source code is designed to increase the strengths of all skew quadrupoles with the name "SQ1" by an amount of 1.0 m^{-1}.

```
for(i=0;i<rhic.Ncell;i++){
  if( rhic.Cell[i]->TYPE==''SKEWQ'' and rhic.Cell[i]->NAME==''SQ1') ) {
    rhic.Cell[i]->SetP(''K1SL'', rhic.Cell[i]->GetP("K1SL")+1.0);
  }
}
```

In SimTrack, elements with the same name can have different strengths.

2.3   Polymorphism

Powerful c++'s polymorphism is used in SimTrack. For most types of magnetic elements, Pass() and DAPass() can call an identical template transfer function. For this purpose, all the basic math operators are redefined or overloaded to work with both regular numbers and truncated power series. A data type tps is defined for linear truncated power series in SimTrack, which includes 6-d closed orbit and 6×6 linear orbit coordinate derivatives. Usage of polymorphism greatly reduces the redundancy in coding and therefore reduces the size of source code. It also makes much easier for users to create a new type of element.

For example, to calculate the one-turn orbit transfer matrix, we first search the 6-d closed orbit. On top of it, we transfer a linear tps through the ring. The input linear tps at the beginning of a ring is the 6-d closed orbit plus a 6×6 identity matrix. To pass a linear tps, we call DAPass() of elements just as we call Pass() to transfer 6-d orbit coordinates.

The linear optics and coupling parameters are derived from eigenvectors of the one-turn matrix following Ref. [9]. The GNU Scientific Library (GSL) [10] is only used for eigenvector calculations in SimTrack. However, all the functions defined in GSL are available to SimTrack users. GSL is a numerical library for C and C++ programmers. It is a free software under the GNU General Public License. GSL provides a wide range of mathematical functions such as random number generators, special functions, and least-squares fitting and so on.

3 Physics Models

3.1 Symplectic Integration

For a particle motion governed by a Hamiltonian, the Jacobian transfer matrix $\mathbf{M}$ should meet the symplectic condition [11]

$$\mathbf{M}^t \mathbf{S} \mathbf{M} = \mathbf{S}. \tag{1}$$

The elements of the Jacobian matrix $\mathbf{M}$ are defined as

$$M_{ij} = \frac{\partial X_i}{\partial X_j}, \tag{2}$$

where X are the 6-d coordinates. In SimTrack, we use $X = (x, p_x, y, p_y, -c\Delta t, p_t = \frac{\Delta E}{cp_0})$, where (x, p_x), (y, p_y), and $(-c\Delta t, p_t)$ are the paired canonical positions and momenta. c is the speed of light, Δt is the difference of time-of-flight. p_0 is the momentum of the reference particle. The path length s of the reference particle is the independent variable. The 6-d matrix $\mathbf{S}$ is

$$\mathbf{S} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 \end{pmatrix}. \tag{3}$$

$\mathbf{S}^t$ is the transpose matrix of $\mathbf{S}$. Symplectic condition is crucial for long-term particle tracking especially for hadrons. A violation of the symplectic condition will produce unphysical tracking results and introduces artificial emittance growth.

Symplectic integration is one kind of numeric integrations to solve differential equations. Different from other methods, symplectic integration preserves symplectic condition while solving the Hamiltonian equations. Symplectic integration is pioneered by R. Ruth and generalized to higher orders by H. Yoshida [12]. Starting with a splittable Hamiltonian [7],

$$H(p, q) = H_1(p) + H_2(q), \tag{4}$$

where p, q are the canonical positions and momenta, the second order symplectic integrator is simply

$$e^{-:H:} = e^{:-LH_1/2:} e^{:-LH_2:} e^{:-LH_1/2:}, \tag{5}$$

which is actually the well-known drift-kick-drift model. The fourth order sympletcic integration is

$$\begin{aligned} e^{-:H:} = \; & e^{:-c_1 L H_1:} e^{:-d_1 L H_2:} e^{:-c_2 L H_1:} e^{:-d_2 L H_2:} \\ & e^{:-c_2 L H_1:} e^{:-d_1 L H_2:} e^{:-c_1 L H_1:} + O(L^5), \end{aligned} \tag{6}$$

where $c_{1,2}$ and $d_{1,2}$ are coefficients,

$$\begin{aligned} c_1 &= \frac{1}{2(2-2^{1/3})}, c_2 = \frac{1-2^{1/3}}{2(2-2^{1/3})}, \\ d_1 &= \frac{1}{2-2^{1/3}}, d_2 = -\frac{2^{1/3}}{2-2^{1/3}}. \end{aligned} \tag{7}$$

We notice that c_2 and d_2 are negative which will mathematically involve a negative length drift and nonlinear magnetic kicks with opposite strengths.

Symplectic integration has been used in modern optics design and particle tracking codes such as TRACY-II [13], PTC [14], LEGO [15] and so on. In SimTrack, fourth order symplectic integration is used. Compared to second order symplectic integration, fourth order symplectic integration needs fewer integration steps to have the calculated orbits converge.

3.2 Exact and Expanded Hamiltonian

The exact Hamiltonian for a particle motion in circular accelerators is [16, 11]

$$H_{exact} = \frac{p_t}{\beta_0} - \left(1 + \frac{x}{\rho}\right) \left[\sqrt{1 + \frac{2p_t}{\beta_0} + p_t^2 - p_x^2 - p_y^2} + \frac{q}{p_0} A_s(x, y) \right]. \tag{8}$$

Here q is the particle's charge, β_0 is the reference particle's velocity divided by the speed of light c, and $A_s(x,y)$ is the scalar potential of magnetic fields. In SimTrack, the local reference coordinate system is always built on the design orbit of a reference particle.

If $\beta = v/c \longrightarrow 1$, where v is the particle's velocity, and $p_{x,y} \ll 1$, the exact Hamiltonian is approximated to the so-called expanded Hamiltonian

$$H_{expanded} = \frac{p_x^2 + p_y^2}{2(1+\delta)} - \frac{q}{p_0} A_s(x,y). \tag{9}$$

The new canonical coordinates are $(x, p_x, y, p_y, -c\Delta t, \delta = \frac{\Delta p}{p_0})$.

The expanded Hamiltonian is a good approximation for most electron storage rings such as synchrotron light sources and ultra-relativistic hadron rings, including the Large Hadron Collider (LHC). For RHIC, the expanded Hamiltonian is used at top energies from 100 GeV/nucleon to 250 GeV/nucleon. However, for a low energy hadron accelerator like RHIC's injector Alternative Gradient Synchrotron (AGS) where the proton energy is between 2.5 GeV and 25 GeV, the exact Hamiltonian must be used.

The expanded Hamiltonian is splittable according to Eq. (4),

$$\begin{array}{rcl}
H_{expanded,1} & = & \frac{p_x^2 + p_y^2}{2(1+\delta)}, \\
H_{expanded,2} & = & -\frac{q}{p_0} A_s(x,y).
\end{array} \tag{10}$$

$H_{expanded,1}$ actually represents a drift and $H_{expanded,2}$ a multipole thin-lens kick. Both $H_{expanded,1}$ and $H_{expanded,2}$ are analytically solvable. To solve the expanded Hamiltonian with symplectic integration, only "+","-","×","÷" operations are involved and therefore it saves computing time. The expanded Hamiltonian has been adopted by the electron ring design codes TRACY-II, LEGO and so on.

For the exact Hamiltonian, we are able to split it into two solvable parts. For a straight section, we split it into

$$\begin{array}{rcl}
H_{exact,1} & = & \frac{p_t}{\beta_0} - \sqrt{1 + \frac{2p_t}{\beta_0} + p_t^2 - p_x^2 - p_y^2}, \\
H_{exact,2} & = & -\frac{q}{p_0} A_s(x,y).
\end{array} \tag{11}$$

Here $H_{exact,1}$ is the Hamiltonian for a pure drift and $H_{exact,2}$ is for a thin-lens kick. For a sector bending magnet with multipole field errors, we split the Hamiltonian as follows:

$$\begin{array}{rcl}
H_{exact,1} & = & \frac{p_t}{\beta_0} - (1 + \frac{x}{\rho})\sqrt{1 + \frac{2p_t}{\beta_0} + p_t^2 - p_x^2 - p_y^2} + \frac{x}{\rho} + \frac{x^2}{2\rho^2}, \\
H_{exact,2} & = & -\frac{q}{p_0} A_s(x,y) - \frac{x}{\rho} - \frac{x^2}{2\rho^2}.
\end{array} \tag{12}$$

Here $\frac{x}{\rho} + \frac{x^2}{2\rho^2}$ is the scalar potential for a sector magnet. $H_{exact,1}$ is the Hamiltonian for a pure sector bending magnet and $H_{exact,2}$ is for a thin-lens kick. As pointed out by Neri and others [17], symplectic integration is also valid for a Hamiltonian with two solvable parts.

Compared to the expanded Hamiltonian, particle tracking with exact Hamiltonian takes longer computing time since it involves trigonometric functions. However, it is a must for low energy proton and ion accelerators. In SimTrack, both expanded and exact Hamiltonian are implemented.

3.3 Beam-Beam Interaction

With the weak-strong beam-beam interaction model, the strong beam is assumed to be rigid and to have a 3-d Gaussian distribution. The charge distribution for one slice of the strong bunch is given by

$$\rho(x,y;z) = \frac{N^* e}{2\pi \sigma_x \sigma_y} \exp\left(-\frac{x^2}{2\sigma_x^2} - \frac{y^2}{2\sigma_y^2}\right), \tag{13}$$

where N^* is the number of particles, e the particle's charge state, $\sigma_{x,y}$ are the transverse rms beam sizes. After a test particle passes through this slice, the 4-d beam-beam transverse kicks are given by

$$\begin{array}{rcl}
\Delta y' + i\Delta x' & = & \frac{2N^* r_p}{\gamma} \sqrt{\frac{\pi}{2(\sigma_x^2 - \sigma_y^2)}} \\
& & \left[w\left(\frac{x+iy}{\sqrt{2(\sigma_x^2 - \sigma_y^2)}}\right) - \exp\left(-\frac{x^2}{2\sigma_x^2} - \frac{y^2}{2\sigma_y^2}\right) w\left(\frac{\frac{x\sigma_y}{\sigma_x} + i\frac{y\sigma_x}{\sigma_y}}{\sqrt{2(\sigma_x^2 - \sigma_y^2)}}\right) \right].
\end{array} \tag{14}$$

$w(x + iy)$ is the complex error function which is defined as $w(z) = e^{-z^2}\left(1 + (2i/\sqrt{\pi})\int_0^z dt e^{t^2}\right)$. Here we assumed proton-proton beam-beam interaction and $\sigma_x > \sigma_y$.

When the bunch's rms length is comparable to the β^*s at the collision point, the 6-d symplectic synchro-beam map by Hirata is adopted in SimTrack. After the test particle passes through a strong bunch, its coordinate changes are [8]

$$
\begin{aligned}
x^{new} &= x + S(z, z*)f_x(X, Y; Z), \\
p_x^{new} &= p_x - f_x(X, Y; Z), \\
y^{new} &= y + S(z, z*)f_Y(X, Y; Z), \\
p_y^{new} &= p_y - f_Y(X, Y; Z), \\
z^{new} &= z, \\
\delta^{new} &= \delta - \tfrac{1}{2}f_X(X, Y; Z)[p_x - \tfrac{1}{2}f_X(X, Y; Z)] \\
&\quad - \tfrac{1}{2}f_Y(X, Y; Z)[p_y - \tfrac{1}{2}f_Y(X, Y; Z)] - g(X, Y; Z).
\end{aligned}
\tag{15}
$$

Here $X = x + p_x S(z, z_*)$, $Y = y + p_y S(z, z_*)$ are the test particle's coordinates at the actual colliding location $S = (z - z_*)/2$, z and z_* are the test particle and the strong slice's longitudinal positions w.r.t their bunch centers. $-f_{X,Y}$ are the 4-d transverse beam-beam kicks at S, which are given by Eq. (14).

In SimTrack, long-range beam-beam interaction, beam-beam interaction with crossing angle are also implemented. To compensate head-on beam-beam interaction, electron lens is added too.

3.4 Proton Spin Tracking

Currently there are two codes for proton spin simulation studies at Brookhaven National Laboratory: one is SPINK [18] developed by A. Luccio, and the other is Zgoubi [19] developed by F. Meot. Since the spin motion is related to the magnetic fields the particles feel on their orbits, accurate modeling of orbit motion is required. SPINK does not have its own orbit motion tracker. It uses the first and second order matrices generated with MADX. Zgoubi is a ray-searching code, which tracks particle's orbit motion by directly solving the Lorentz equation. All the physical quantities are expanded to 5th or even higher Taylor power series. Both codes are not symplectic. Zgoubi tracks spin in the same manner as ray-searching of an orbit, which integrates the spin vector along the path length in tiny steps.

The Thomas-BMT equation [20] describes the particle's spin motion,

$$
\frac{d\mathbf{S}}{dt} = \mathbf{S} \times \mathbf{\Omega},
\tag{16}
$$

$$
\mathbf{\Omega} = \frac{e}{\gamma m}\left[(1 + G\gamma)\mathbf{B}_\perp + (1 + G)\mathbf{B}_{||}\right].
\tag{17}
$$

$\mathbf{S}$ is the 3-d spin vector in the particle's frame. e and m are the charge and rest mass of the particle. γ is the Lorentz factor. G is the anomalous factor, and for protons, $G = 1.7928474$. $\mathbf{B}_\perp$ and $\mathbf{B}_{||}$ are the magnetic fields perpendicular and parallel to the particle velocity respectively. $\mathbf{B}$ is defined in the laboratory frame. In Eq. (17) we neglected the electric field's effect.

It is preferable to use the path length s of the reference particle as the the independent variable rather than the time t. We rewrite Eq. (5) as

$$
\frac{d\mathbf{S}}{ds} = \mathbf{S} \times \mathbf{F},
\tag{18}
$$

$$
\mathbf{F} = \frac{\sqrt{(1 + \frac{x}{\rho})^2 + x'^2 + y'^2}}{1 + \delta}\frac{\mathbf{\Omega}}{(B\rho)_0},
\tag{19}
$$

where ρ is the curvature radius for the reference particle on which the local coordinate system is built. In SimTrack, we integrate the particle orbit motion step-by-step through each magnet. For each integration step, we evaluate both $\mathbf{B}_\perp$ and $\mathbf{B}_{||}$ at the middle of this step.

Knowing the magnetic fields, we solve the spin motion Eq. (18) as a rotation motion in the 3-d space. The rotation axis is given by $\mathbf{F}$ and the absolute rotation angle is $|\mathbf{F}|$. Keep in mind that the spin vector rotates clockwise with the axis. At each integration of orbit motion, we have

$$
\mathbf{S}_2 = \mathbf{R}\mathbf{S}_1,
\tag{20}
$$

6

where $\mathbf{R}$ is a 3-d rotation matrix,

$$\mathbf{R} = \begin{pmatrix} \cos\theta + n_x^2(1-\cos\theta) & n_x n_y(1-\cos\theta) - n_z\sin\theta & n_x n_z(1-\cos\theta) + n_y\sin\theta \\ n_y n_x(1-\cos\theta) + n_z\sin\theta & \cos\theta + n_y^2(1-\cos\theta) & n_y n_z(1-\cos\theta) - n_x\sin\theta \\ n_z n_x(1-\cos\theta) - n_y\sin\theta & n_z n_y(1-\cos\theta) + n_x\sin\theta & \cos\theta + n_z^2(1-\cos\theta) \end{pmatrix}, \tag{21}$$

with the rotation angle θ and the rotation axis $\mathbf{n} = (n_x, n_y, n_z)$, $|\mathbf{n}| = 1$. Since $|\mathbf{R}| = 1$, we have $|\mathbf{R}_2| = |\mathbf{R}_1|$ and thus the spin transfer in SimTrack is orthogonal.

4   Applications

4.1   Short-term Tracking

Short-term single particle orbit tracking is used to generate turn-by-turn (TBT) beam position monitor (BPM) data for further beam dynamics analysis. To study the long-term stability of particle motion, Sim-Track provides functions to calculate the tune footprint, tune diffusion, action diffusion, Lyapunov exponent and so on.

Tune diffusion analysis, also called frequency map analysis (FMA), has been introduced into accelerator physics by J. Laskar [21]. In SimTrack, the tune diffusion over 2048 turns is defined as

$$|\Delta Q| = \sqrt{|\Delta Q_x|^2 + |\Delta Q_y|^2}, \tag{22}$$

where $|\Delta Q_x|$ and $|\Delta Q_y|$ are the differences of horizontal and vertical betatron tunes between the first and second 1024 turns. The betatron tunes can be determined from TBT BPM data with interpolated fast Fourier transformation (FFT) and Hanning windows or direct Fourier transformation. Tune diffusion can be plotted in the tune space (Q_x, Q_y) or in the amplitude space $(x/\sigma_x, y/\sigma_y)$.

As an example, Figure 1 shows the tune diffusion in the tune space with beam-beam interaction and with half beam-beam compensation for RHIC. The lattice tunes are $(28.685, 29.695)$. The proton bunch intensity is 2.5×10^{11}. The protons collide at IP6 and IP8. The electron lens is installed at IP10. Half beam-beam compensation compensates half of beam-beam tune spread. In these plots, different colors represent different tune diffusion ranges. The left plot shows that there is not enough tune space between 2/3 and 7/10 to hold the beam-beam tune spread with a bunch intensity 2.5×10^{11}. With half head-on beam-beam compensation, the beam-beam tune spread is reduced and can be fitted into the available tune space. The resonance lines can be clearly identified in the tune diffusion map. Also we notice foldings of the tune footprints with half beam-beam compensation at larger betatron amplitudes. Full beam-beam compensation introduces more nonlinearities into the ring than half beam-beam compensation and the tune footprint will be heavily deformed.

The Lyapunov exponent is another indicator for long-term stability of single particle motion [22]. Two neighboring particles are launched with a small initial distance d_0 in the phase space. The Lyapunov exponent
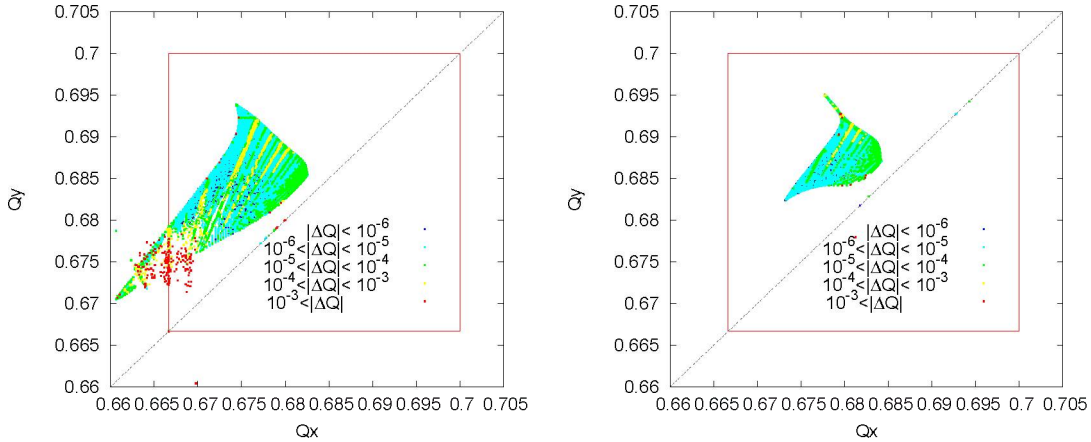

Figure 1: Left: Tune diffusion with beam-beam interaction. Right: Tune diffusion with half head-on beam-beam compensation. The lattice tunes are $(28.685, 29.695)$. The proton bunch intensity is 2.5×10^{11}.
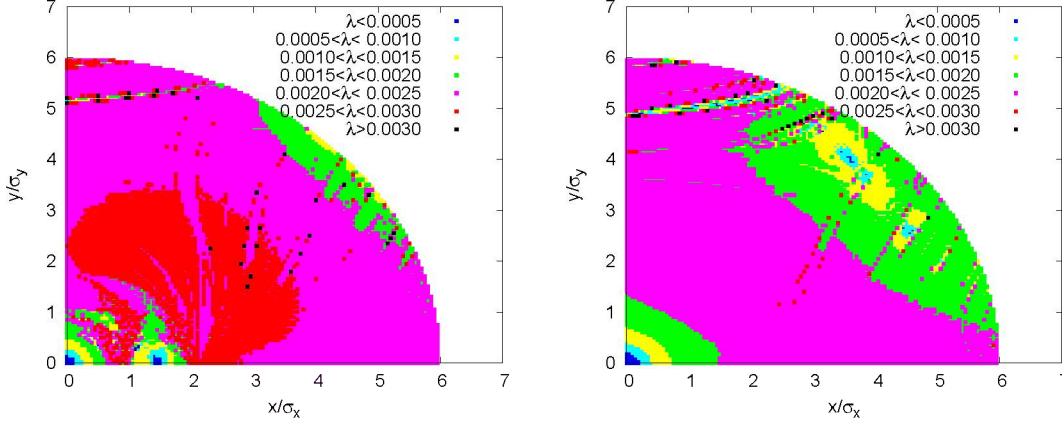
Figure 2: Left: The Lyapunov exponent map map with beam-beam interaction. Right: The Lyapunov exponent with half beam-beam interaction. The lattice tunes are (28.685, 29.695). The proton bunch intensity is 2.5×10^{11}.

is defined as

$$\lambda(n) = \frac{1}{n} \ln \frac{|\mathbf{X_2}(n) - \mathbf{X_1}(0)|}{d_0}. \tag{23}$$

where n is the tracking turn, $\mathbf{X_i}(n)$, i=1,2, are the coordinate vectors of these two particles. If the particle motion is regular, the distance of these two particles will grow linearly with tracking turns and $\lambda(n)$ tends to be zero. If the motion is chaotic, the distance will grow exponentially and $\lambda(n)$ converges to a positive value.

Figure 2 shows the Lyapunov exponent in the $(x/\sigma_x, y/\sigma_y)$ plane without and with beam-beam compensation from 2048 turn tracking. The lattice tunes are (28.685, 29.695). The proton bunch intensity is 2.5×10^{11}. Different colors represent the amplitude ranges of the Lyapunov exponent. The smaller the Lyapunov exponent is, the more stable the particle motion is. These plots demonstrate that with half head-on beam-beam compensation, the particles in the bunch core become more stable than without compensation. The reason is that beam-beam compensation moves the tunes of particles in the bunch core away from the third order resonances at $Q_{x,y} = 2/3$. It is interesting that the Lyapunov exponent map also discloses some resonance structures similar to the tune diffusion map.

4.2 Dynamic Aperture Calculation

The dynamic aperture is defined as the maximum betatron amplitude within which particles are not lost in a given turn of single particle tracking. It is often given in units of the rms transverse amplitude σ. The long-term dynamic aperture converges to the boundary between the regular and chaotic motion [22]. In RHIC dynamic aperture is frequently used to evaluate the stability of lattices and the effect of beam-beam interaction and so on.

For RHIC, we normally track single proton particles up to 10^6 turns with SimTrack. For each tracking condition, we search the dynamic aperture in 10 equally spaced phase angles in the first quadrant in the $(x/\sigma_x, y/\sigma_y)$ plane. The initial $p_{x,y}$ are set to zero. The initial relative off-momentum deviation is chosen to be about 3 times the rms momentum spread of the beam. The dynamic aperture as a function of momentum deviation can be calculated as well. For comparison between different cases, we typically focus on the minimum dynamic apertures among the 10 phase angles.

In SimTrack, the bisection searching method is used for a quick dynamic aperture calculation. For each amplitude, we track twin particles separated by ± 0.025 σ. Only when both particles survive after 10^6 turns, the particle motion at this amplitude is considered stable. The minimum step of dynamic aperture calculation in SimTrack is set to 0.1 σ.

As an example, Figure 3 shows the dynamic apertures versus the proton bunch intensities for the 250 GeV RHIC proton operation with half beam-beam compensation in RHIC. Different cases are studied: with beam-beam interaction, with half beam-beam compensation, and with $k\pi$ phase advances between IP8 and the center of the electron lenses and the global second order chromaticity correction. This plot shows that when
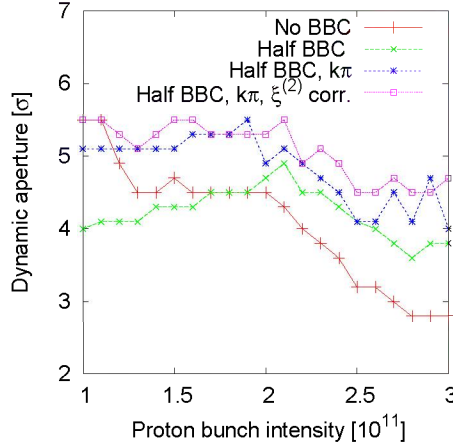
Figure 3: Calculated dynamic apertures with half head-on beam-beam compensation. $k\pi$ phase advances between IP8 and electron lenses and global second order chromaticity correction further improve the dynamic aperture with half beam-beam compensation.
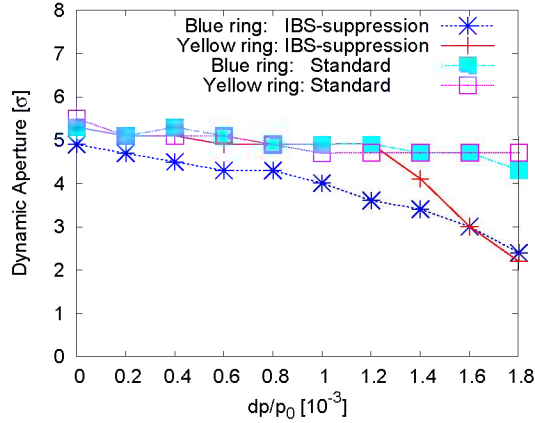

Figure 4: Calculated off-momentum dynamic apertures for the IBS-suppression lattices and standard lattices.

the proton bunch intensity is larger than 2×10^{11}, the dynamic aperture with beam-beam interaction drops. The reason is that the beam-beam tune spread is too large to be fitted between the 2/3 and 7/10 resonances. With half beam-beam compensation with an electron lens, the dynamic aperture increases when the proton bunch intensity is larger than 2×10^{11}. For a proton bunch intensity lower than 2×10^{11}, beam-beam compensation is not needed. In addition, the $k\pi$ phase advances between IP8 and the electron lenses and the global second order chromaticity correction further improve the dynamic aperture with half beam-beam compensation.

Intra-beam scattering (IBS) is the main limiting factor to the luminosity increase for the RHIC heavy ion operation. To counteract IBS, stochastic cooling has been implemented since 2007. Before the installation of full 3-d cooling, we explored a way to reduce the transverse IBS growth rates by reducing $< \frac{H_X}{\beta_x} >$ of the lattices. However, with the so called IBS-suppression lattices in the 2010 and 2011 Au-Au runs, we observed a large beam loss with longitudinal and vertical cooling.

Figure 4 shows the calculated off-momentum dynamic apertures for the IBS-suppression and the previous standard lattices for both rings. The IBS-suppression lattices give a smaller off-momentum dynamic aperture for large momentum deviations. Based on this simulation study, we decided to adopt the standard lattices for the 2012 uranium-uranium run [23]. Together with the full 3-d stochastic cooling, a burn-off dominated uranium beam lifetime was achieved in that run. More than 90% of particle loss went to luminosity production, which was observed in colliders for the first time.
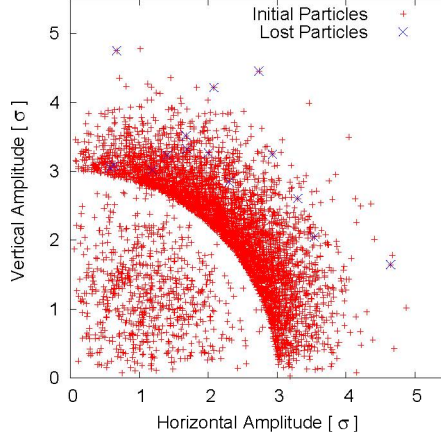
Figure 5: Initial transverse amplitudes of all and the lost macro-particles with an initial hollow Gaussian distribution. There are particles whose transverse amplitudes are less than 3σ. Those macro-particles are tracked because their longitudinal amplitudes are larger than 3σ.

4.3 Particle Loss Rate Calculation

In the operation of accelerators, the beam intensity and its loss rate can be directly measured. Although dynamic aperture is a strong indicator for the long-term stability of particle motion, there is no obvious calibration between the dynamic aperture and the beam lifetime. It is preferable to employ long-term multi-particle tracking to predict the beam loss and its locations along the ring.

For example, we calculate the beam loss rate with half beam-beam compensation in RHIC. The beam energy is set to 250 GeV and the β^*s at the 2 collision points are 0.5 m. The bunch intensity is 2.5×10^{11}. We track 4800 macro-particles up to 2×10^6 turns, which corresponds to 24 seconds in the RHIC. The initial coordinates of macro-particles are sampled from a 6-d Gaussian distribution. After 2×10^6 turns, only 1 macro-particle is lost. To reduce the statistical error in the calculation of beam lifetime, a large number of macro-particles and therefore a large amount of computing time are needed.

One approach is to track macro-particles initially with a hollow Gaussian distribution [24, 25]. The validity of this approach is based on the assumption that the particles in the bunch core will not be lost over the total number of tracked turns. Thus, for the same number of macro-particles, a hollow Gaussian distribution has more particles in the bunch tail. The boundary between the stable core and the unstable bunch tail needs to be carefully chosen. Normally we first calculate the long-term dynamic aperture and then set the boundary well below it. After multi-particle tracking, we also need to double check if there are particles lost close to the boundary. If so, we have to reduce the boundary.

As a comparison, with the same simulation parameters, we track 4800 macro-particles whose initial transverse or longitudinal amplitudes are larger than 3σ. After 2×10^6 turns, there are 16 macro-particles lost. Figure 5 shows the transverse amplitudes of all and the lost macro-particles with hollow Gaussian tracking. In this example, the 4800 macro-particles actually represent 66269 particles of a 6-d Gaussian distribution. Although the computing time is the same for the Gaussian distribution, the statistical error in the particle loss rate calculation is reduced.

Another approach is to track macro-particles initially with a 6-d weighted Gaussian distribution [5]. In this approach, we designate a larger weight to the macro-particles in the bunch core while giving a smaller weight to the macro-particles in the bunch tail. Therefore, with the same number of macro-particles, there will be more macro-particles in the bunch tail and fewer macro-particles in the bunch core. For this method, 4800 macro-particles of a weighted Gaussian actually represented 70108 particles of a 6-d Gaussian distribution. After 2×10^6 turns, there are 20 macro-particles lost. Table 1 lists the number of lost macro-particles and the calculated beam loss rates from long-term multi-particle tracking. Both hollow and weighted initial distributions reduce the statistical error in the beam loss rate calculation.

4.4 Proton Spin Tracking

Spin tracking with SimTrack was benchmarked with Zgoubi for both RHIC and AGS lattices. We compared the spin vector evolution through single elements, such as sector bend, quadrupole, sextupole, RF cavity,

Table 1: Particle losses in long-term multi-particle tracking with different initial distributions

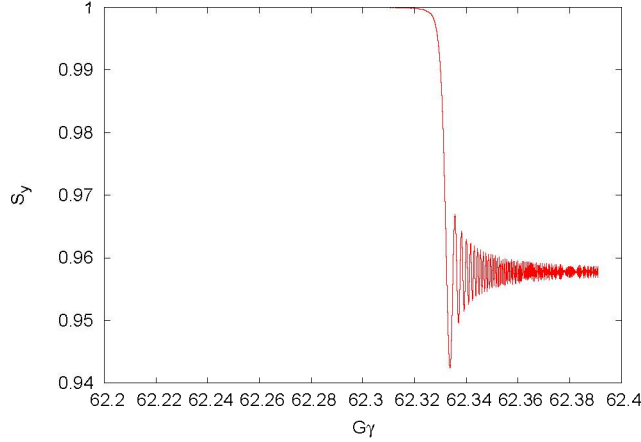| Case	$N_{represent}$	N_{lost}	Beam loss rate
Solid Gaussian	4800	1	2.9%/hr
Hollow Gaussian	66269	16	3.4%/hr
Weighted Gaussian	70108	20	4.0%/hr

Figure 6: An example of spin vector evolution through a spin resonance crossing.

and so on. The results from both codes agree well in short-term spin tracking.

Another direct test for spin tracking codes is to compute the polarization loss after a single spin resonance crossing. The well-known Froissart-Stora formula analytically predicts the final polarization after a single resonance crossing, which is given by

$$S_y = 2e^{-\frac{\pi|\epsilon|^2}{2\alpha}} - 1,\tag{24}$$

where $\alpha = dG\gamma/d\theta$ is the resonance crossing speed, and θ the bending angle. $|\epsilon|$ is the strength of the spin resonance.

Figure 6 shows the vertical spin vector S_y from SimTrack during an intrinsic spin resonance crossing in RHIC. The vertical fractional lattice tune is 0.67 and $\alpha = 2.7343 \times 10^{-6}$. The spin resonance at $G\gamma = 62.33$ is crossed. The initial spin points in the vertical direction. Since the spin resonance is so strong, we intentionally reduced the initial vertical emittance by a factor of 1×10^5 from the normal normalized emittance 10 π mm.mrad. With Eq. (24), and with the initial and final spin components in the vertical direction from particle tracking, we derived a spin resonance strength at $G\gamma = 62.33$ of 0.192, which is close to its analytical estimate of 0.195.

Next we demonstrate a long-term tracking with both orbit and spin motions on the energy acceleration ramp in RHIC. The injection energy is 25 GeV and the top energy 100 GeV. The exact Hamiltonian has been used. The actual ramp takes about 100 seconds or 7.8×10^6 turns. Protons will cross more than 100 spin intrinsic resonances on the way. The distance between neighboring resonances is only about 500 MeV. To overcome these spin resonances, two Serbian snake magnets have been installed in RHIC. Snake magnets flip the vertical spin vectors and maintain the spin tune constant at 0.5 to avoid spin resonances. However, snake magnets themselves introduce the snake resonances. Figure 7 shows the vertical spin component versus the particle energy. The vertical normalized emittance is 10 π mm.mrad.

4.5 Application for eRHIC Design

One optional design for the future electron-ion collider eRHIC at Brookhaven National Laboratory is to adopt energy recovery linac (ERL) based non-scaling fixed field alternative focusing (FFAG) rings for the electron beams. Combined function permanent magnets are proposed. Two separate FFAG rings will accelerate the electrons from 1 GeV up to 20 GeV in more than 10 passes.

For one intermediate design of the high energy FFAG ring [26], the lowest and highest energies are spanning from 6.6 GeV to 17.0 GeV. The reference orbit is defined with the highest energy. The maximum
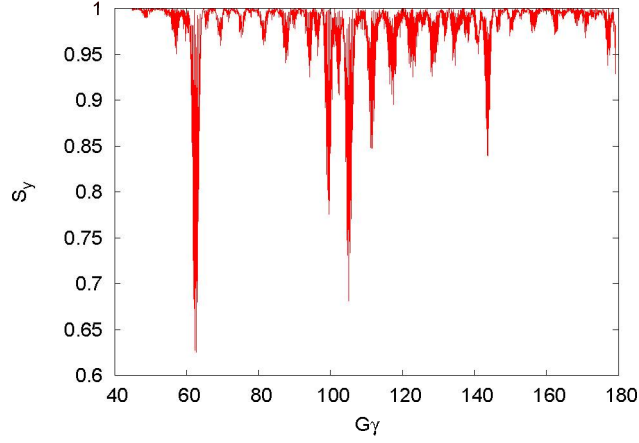
Figure 7: An example of spin vector evolution during an energy acceleration ramp in RHIC.
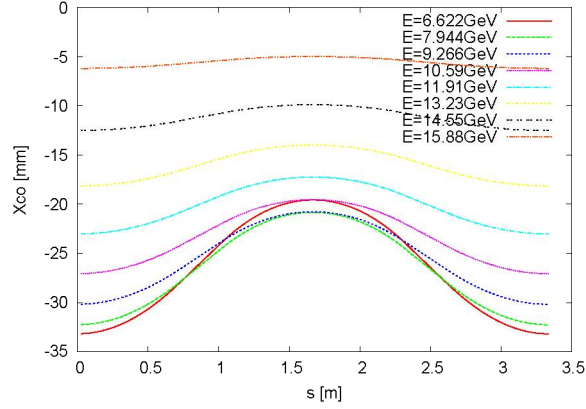

Figure 8: Closed orbits of one FODO cell in the high energy electron ring of the FFAG based eRHIC design.

relative energy deviation is -0.64%. To calculate the orbits, tunes, and Twiss parameters correctly with such a large momentum deviation, the exact Hamiltonian has to be used. Figure 8 shows the horizontal closed orbits for one FODO cell in the arcs. The maximum orbit shift between the lowest and largest energies are 33 mm. Figure 9 shows the transverse tunes of one FODO cell. The integer resonance will be crossed during acceleration with the non-scaling FFAG based design. Ref. [27] points out that large chromaticity may cause emittance growth.
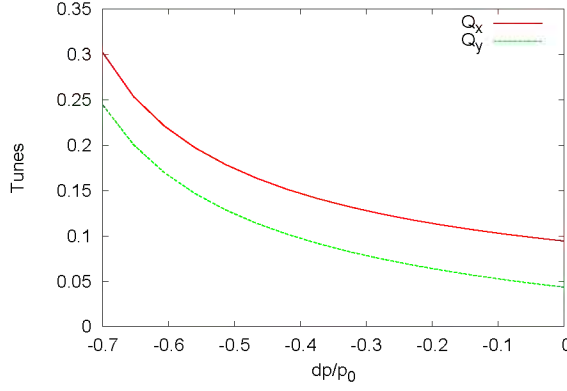
Figure 9: Transverse tunes of one FODO cell versus relative momentum deviation dp/p_0 in the high energy electron ring of the FFAG based eRHIC design.

5 Summary and Outlook

SimTrack is a compact c++ library for 6-d symplectic element-by-element particle tracking in accelerators. It adopts the 4th order symplectic integration for magnet elements and the synchro-beam map for beam-beam interactions. Benefiting from c++'s class definition, STL containers, and polymorphism, SimTrack is efficient to manage elements and to correct optics before launching particle tracking.

Since its inception in 2009, SimTrack has been extensively used to calculate the dynamic aperture with beam-beam interaction and beam-beam compensation in RHIC. Recently, proton spin tracking and electron energy loss due to synchrotron radiation were added. We plan to use SimTrack to do multi-particle spin tracking of a proton bunch from injection up to store energy in RHIC. For the design of an ERL-FFAG based eRHIC, SimTrack is also the preferable choice for particle tracking studies since it is able to track particles with large momentum deviations. To study the instabilities in the ERL-FFAG based eRHIC design, we plan to include wake fields and impedances in SimTrack.

6 Acknowledgments

References

[1] W. Fischer, *Beam-beam and BTF*, 2006 RHIC Accelerator Physics Experiments Workshop, November 2-3, 2005, BNL. Available at http://www.c-ad.bnl.gov/APEX/APEXWorkshop2006/agenda.htm.

[2] Y. Luo, et al., Phys. Rev. ST Accel. Beams 15, 051004 (2012).

[3] SixTrack homepage: http://sixtrack.web.cern.ch/SixTrack/

[4] H.J, Kim, T. Sen, Nucl. Instr. and Meth. A 642 (2011) 25-35.

[5] D. Shatilov, in Proceedings of 2005 Particle Accelerator Conference, Knoxville, TN, USA.

[6] MADX homepage: http://mad.web.cern.ch/mad/

[7] R. Ruth, IEEE Trans. Nucl. Sci. NS-30(4), pp.2669-2671, 1983.

[8] K. Hirata, et al., Particle Accel. 40, pp. 205-228, 1993

[9] Y. Luo, Phys. Rev. ST Accel. Beams 7, 124001 (2004).

[10] GNU Scientific Library, http://www.gnu.org/software/gsl/.

[11] E.D. Courant, H.S. Snyder, Annals of Physics, Vol. 3, pp.1-48, 1958.

[12] H. Yashida, Phys. Lett. A 150 (1990), pp. 262-268.

[13] J. Bengtsson, private communication, 2015.

[14] P.K. Skowronski, F. Schmidt, E. Forest, in the Proceedings of 2007 Particle Accelerator Conference, Albuquerque, New Mexico, USA.

[15] Yunhai Cai, Nucl. Instr. and Meth. A 645 (2011) 168-174.

[16] H. Goldstein, *Classical Mechanics*, Addison-Wesley, 1980.

[17] E. Forest, *Beam Dynamics: A New Attitude and Framework*, Harwood Academic Publishers, Amterdam, The Netherlands, 1997.

[18] A.U. Luccio, BNL C-AD/AP-Note 283, 2007 (unpublished).

[19] F. Meot, Nucl. Instr. and Meth. A 767 (2014) 112-125.

[20] M. Foroissart, R. Stora, Nucl. Instr. and Meth. 7 (1960), p.297.

[21] J. Laskar, Physica **67 D**, 257 (1993).

[22] M. Giovannozzi, W. Scandale and E. Todesco, Part. Accel. Vol 56, pp.195-225, 1997.

[23] Y. Luo, et al., Phys. Rev. ST Accel. Beams 17, 081003 (2014).

[24] Y. Luo, et al., In Proceedings of the 46th ICFA Advanced Beam Dynamics Workshop on High-Intensity and High-Brightness Hadron Beams, Morschach, Switzerland, 2010.

[25] H.J. Kim, et al., Phys. Rev. ST Accel. Beams 12, 031001 (2009)

[26] D. Trbojevic, private communication, 2015.

[27] V. N. Litvinenko, Phys. Rev. ST Accel. Beams 15, 074401 (2012).