# Manual of FASTION code

Chao Li*

(Dated: December 2019)

## I. INTRODUCTION

The code FASTION was developed to study the beam-ion interaction in electron rings. In electron accelertors, ions could be repeated generated due to collision ionization when electron bunches passed by. The ions generated would interact with the tail bunch leading to a "head-tail" coupled beam motion. Beside the beam-ion interaction, several potential APIs are also designed to keep possible potential to include more physical mechanisms into account, as impedance and wakefield. This manual will give basic models and equations used in the code, and setting to used the code are also explained.

## II. BEAM-ION INTERACTION

Denoting $P$ and $T$ as the vacuum pressure and temperature, the molecules density $n$ in the accelerator can be obtained from the general gas equation,

$$PN_A = nRT, \tag{1}$$

where $R$ and $N_A$ are the ideal gas constant and the Avogadro number. Denote $\sum$ as the ionization cross-section, $N_b$ as the number of electron particles passing by, the number of ionization ions per unit length is

$$\lambda = \sum nN_b. \tag{2}$$

For simplicity, the interaction between ions and beam is assumed taking place at lumped interaction locations, and the ions are assumed not to move longitudinally. When beam passes through interaction points bunch by bunch, the ions generated are randomly distributed in the same range as the size of the electron bunch passing by. At the interaction point, the accumulated ions are kicked by the passing bunched electron particles and then drift freely until next electron bunch comes. The motion equations of the $i$th accumulated ion $\vec{X}_i$ and the $k$th electron particle in the $j$th bunch $\vec{x}_{k;j}$ can be expressed as

$$\begin{aligned}
&\frac{d^2\vec{X}_i}{dt^2} + K_i(s)\vec{x}_{k;j} + \sum_{k=0}^{N_j}\vec{F}_C(\vec{X}_i - \vec{x}_{k;j}) = 0 \\
&\frac{d^2\vec{x}_{k;j}}{ds^2} + K_e(s)\vec{x}_{k;j} + \sum_{i=0}^{N_i}\vec{F}_C(\vec{x}_{k;j} - \vec{X}_i) = 0,
\end{aligned} \tag{3}$$

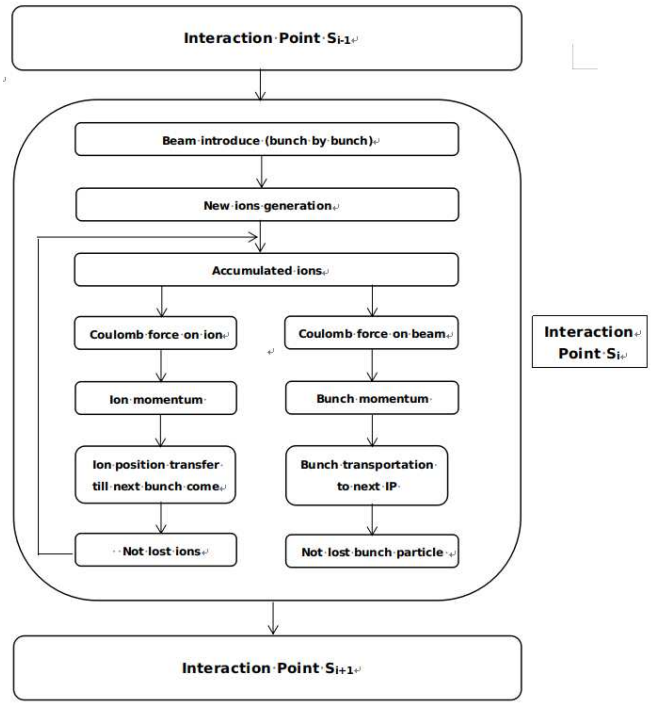


FIG. 1. Logic flow of the beam-ion interaction in simulation at the interaction points.

where $\vec{F}_C$ is the Coulomb force between the ions and electron particles, $K_i(s)$ and $K_e(s)$ represent the lattice focusing strength on ion and beam particle.

In the FASTION code, the 2D Bassetti-Erskine formula is use to get the space charge potential of the electron bunch, since the Gaussian profile assumption is rather good for electron bunch. At the interaction point $s_i$,

$$\begin{aligned}
E_{C,y}(\vec{x}) + IE_{C,x}(\vec{x}) = &\frac{n_b}{2\epsilon_0\sqrt{2\pi(\sigma_x^2 - \sigma_y^2)}}\{w(\frac{x + Iy}{\sqrt{2(\sigma_x^2 - \sigma_y^2)}}) \\
&- \exp(-\frac{x^2}{2\sigma_x^2} - \frac{y^2}{2\sigma_y^2}) \\
&+ w(\frac{x\frac{\sigma_y}{\sigma_x} + Iy\frac{\sigma_x}{\sigma_y}}{\sqrt{2(\sigma_x^2 - \sigma_y^2)}})\}\delta(s_i),
\end{aligned} \tag{4}$$

where $n_b$ is the line density of the electron beam, $w(z)$ is the complex error function, $\sigma_x$ and $\sigma_y$ are the beam rms size in horizontal and vertical direction, $x$ and $y$ are the distance from ions to the bunch centroid, $I$ is the complex unit. Substituting Eq. 4 into Eq. 3, the explicit

* supperli.imp@gmail.com

momentum change of ions at the interaction point is

$$\Delta p_{i,y} + I\Delta p_{i,x} = \frac{2n_b r_e m_e c}{\gamma_e}(E_{C,y} + IE_{C,x}), \quad (5)$$

where $r_e$ is the classical electron radius, $m_e$ is the electron mass, $c$ is the speed of light, $\gamma_e$ is the relativistic factor of electron beam. Since the ions are much heavier than the electron, the lattice focusing $K_i(s)$ can be ignored.

As to the space charge potential well generated by the ions, since the ions distribution is usually not a Gaussian type and the ion particles almost occupy the whole pipe, in principle the Bassetti-Erskine formula is not suitable anymore. A self-consistent particle-in-cell (PIC) solver or ion density profile fitting is required. In the FASTION code, a compromise approach is applied. The ions distribution is truncated at 10 rms bunch size. The rms and centroid information of the truncated ion distribution are substituted in the Bassetti-Erskine formula to get the Coulomb potential. When one electron bunch particles passes by, the transverse momentum and position of the accumulated ions are updated according to the time interval until the next bunch comes. As to the bunched electron particles, after the momentum kicks induced by the accumulated ions, they are transferred to the next interaction point by applying the linear transport matrix.

## III. BUNCH-BY-BUNCH FEEDBACK

The bunch-by-bunch feedback based on the FIR filter is an effective way to cure the coupled bunched instability. It detects transverse or longitudinal centroid positions of beam bunches, processes the positions data to create kicker signals, and adds transverse or longitudinal kicks to the beam particles to damp its oscillations. Eq. 6 is the general form of a FIR filter

$$\Theta_n = \sum_{k=0}^{N} a_k x_{n-k}, \quad (6)$$

where $a_k$ represents the filter coefficient, $x_{n-k}$ and $\Theta_n$ are the input and output of the filter, corresponding to beam position data at the $(n-k)$th turn and kick strength on the beam at the $n$th turn. The number of the input data $N+1$ is defined as taps. Following the approaches shown in Ref. [1], the time domain least square fitting (TDLSF) method is used to get the filters coefficients $a_k$.

In the FASTION code, the beam momentum change by the bunch-by-bunch feedback at the $n$th turn is modeled as

$$\Theta_{x,n} = K_x \sum_{k=0}^{N} a_{k,x} x_{n-k},$$
$$\Theta_{y,n} = K_y \sum_{k=0}^{N} a_{k,y} y_{n-k}, \quad (7)$$

The beam motion transfer function in one turn including feedback is

$$\left[\begin{pmatrix} x_{n+1} \\ x'_{n+1} \\ y_{n+1} \\ y'_{n+1} \end{pmatrix}\right] = M_0 \left[\begin{pmatrix} x_n \\ x'_n \\ y_n \\ y'_n \end{pmatrix} + \begin{pmatrix} 0 \\ \Theta_{x,n} \\ 0 \\ \Theta_{y,n} \end{pmatrix}\right], \quad (8)$$

where $M_0$ is the one turn matrix at the kicker.

## IV. SYNCHRONTRON RADIATION DAMPING AND EXCITATION

Following approaches used in Hiratas BBC code and Yuan Zhang's Beam-beam code , the transportation through the arc consists of the following maps

1. From accelerator coordinates to normalized coordinates;
2. Transportation with synchrotron radiation;
3. From normalized coordinates to accelerator coordinates.

Basically, in the first setp, it follows,

$$X = BRHx \quad (9)$$

where H is the dispersion matrix, R is the Teng matrix, B is the Twiss matrix. At the second step, with synchrotron radiation, the arc transportation in the normalized coordinates is

$$\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \lambda_u m_u \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} + \sqrt{\epsilon_x(1-\lambda_u^2)} \begin{pmatrix} \hat{r}_1 \\ \hat{r}_2 \end{pmatrix},$$
$$\begin{pmatrix} X_3 \\ X_4 \end{pmatrix} = \lambda_\nu m_\nu \begin{pmatrix} X_3 \\ X_4 \end{pmatrix} + \sqrt{\epsilon_y(1-\lambda_\nu^2)} \begin{pmatrix} \hat{r}_3 \\ \hat{r}_4 \end{pmatrix},$$
$$\begin{pmatrix} X_5 \\ X_6 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \lambda_w^2 \end{pmatrix} m_w \begin{pmatrix} X_5 \\ X_6 \end{pmatrix} + \sqrt{\epsilon_z(1-\lambda_w^4)} \begin{pmatrix} 0 \\ \hat{r}_5 \end{pmatrix} \quad (10)$$

Here $\hat{r}$ are are independent Gaussian random variables with unit variance, $\lambda_i$=exp$(-1/T_i)$ with $T_i$ the damping time in unit of the number of turns, and $\epsilon_z$ is calculated as $\sigma_z\sigma_e$. At the last step,

$$x = H^{-1}R^{-1}B^{-1}X \quad (11)$$

## V. SETTINGS OF THE FASTION

FASTION is developed with C++. It is an open source tool which can be download with links https://github.com/ChaoLiIHEP/FASTION. With the make file, the code can be can be complied if all of the source scripts and head files are located in the same directory. Only the GSL [2] library are required.

### A. settings in input.dat

- CircRing [m]: circumference of ring;
- WorkQx: workiing point in x direction;
- WorkQy: workiing point in y direction;
- WorkQz: workiing point in z direction; It is a dummy parameter in beam-ion interaction simulation;
- RFBaseFrequency [Hz]: fundamental frequency of the RF cavity;
- CorssSectionEI [$m^2$]: ionization cross-section; Currently, the CO is used as the default gas in simulation; If the other ions are required, the mass number 28 also have to be modified in bunch.cpp, line 708 and line 573;
- PipeAperatureX [m]: pipe aperture in x direction;
- PipeAperatureY [m]: pipe aperture in y direction;
- PipeAperatureR [m]: pipe aperture R; the spare boundaries with $PipeAperatureX$ and $PipeAperatureY$ are used as default beam and ions loss criteria;
- ionlossboundary: default value in unit of beam rms size used as ion loss criteria;
- ionMaxNumber: if accumulated ions number si larger than $ionMaxNumber$, program will be quit automatically;
- numberofIonBeamInterPoint: number of ionteraction points used in program. The program will read the data in $InterPointParameter.dat$ automatically, in which $numberofIonBeamInterPoint$ sets of local parameters have to be supplied;
- electronBeamEnergy: electron beam energy;
- rmsBunchLength [m]: initial rms bunch length;
- rmsEnergySpread [rad]: initial rms bunch energy spread; The bunch initial distribution is truncated within the ragne of 3 sigma.
- macroEleNumPerBunch: number of macro-particles used to represent the electron bunch;
- distributionType: flag for initial electron beam distribution in transverse direction; $distributionType = 1$: kV; $distributionType = 1$: Water Bag; $distributionType = 3$: Gaussion truncated within 3 sigma;
- initialDisDx [m]: Max initial displacement error of electron bunch in x direction; errors are randomly generated within the range $initialDisDx$
- initialDisDy [m]: Max initial displacement error of electron bunch in y direction; errors are randomly generated within the range $initialDisDy$;
- nTurns: tracking turns in simulations;
- macroIonNumberGeneratedPerIP: number of macro-ions generated for each collision at interaction point.
- trainNumber: train number at the ring;
- totBunchNumber: total bunch number in rings
- bunchNumberPerTrain: it is an array; $trainNumber$ valeus have to be supplied;
- trainGaps: it is an array; $trainNumber$ valeus have to be supplied;
- SynRadDampingFlag: Flag to turn on the and off the synchrontron radiation damping and excitation; $SynRadDampingFlag = 1$ the values $synchRadDampTimex$, $synchRadDampTimey$ and $synchRadDampTimez$ are used in simulation.
- synchRadDampTimex: damping time in x direction in the unit of turns. It is a dummy variabel if $SynRadDampingFlag = 0$;
- synchRadDampTimey: damping time in y direction in the unit of turns. It is a dummy variabel if $SynRadDampingFlag = 0$;
- synchRadDampTimez: damping time in z direction in the unit of turns. It is a dummy variabel if $SynRadDampingFlag = 0$;
- current: total beam current;
- emittanceX [m rad]: beam rms emittance in x direction;
- kappa: emittance couping factor; with $kappa$ the emittance in y direction is $emittanceX * kappa$
- bunchBinNumberZ: number of bins for each bunch. It is used to calculate the beam-impedance interaction. Curretly, it is a dummy variable and not used.
- fIRBunchByBunchFeedbackFlag: flags to take the fir filter into account. $fIRBunchByBunchFeedbackFlag = 0$ bunch-by-bunch is not taken into account; if $fIRBunchByBunchFeedbackFlag = 1$, the code will read the parameter in $FIR_input.dat$ to get the paramters of filter, kickers and pickups

- printInterval: interval in unit of turns to print the beam data; intevalofTurnsIonDataPrint: interval in unit of turns to print the ion data;
- calSettings: index for simulations settings; if $calSettings = 1$, the beam-ion effect is calculated; if $calSettings = 2$, single bunch - longitudinal impedance interaction is simulated.

---

[1] T. Nakamura, "single-loop muli-dimensional digital feedback by fir filters," http://acc-web.spring8.or.jp/~nakamura/reports/Feedback_with

[2] https://www.gnu.org/software/gsl/.