FINAL REPORT

to the Accord

"**Software Development for Electron Cooling Calculation"**

between Brookhaven National Laboratory (BNL)
and Joint Institute for Nuclear Research (JINR)

I.N.Meshkov, A.O. Sidorin, A.V.Smirnov, G.V.Trubnikov
Joint Institute for Nuclear Research, Dubna, Russia

June, 2005

Abstract

This report summarizes results of the software development during the total term of the Accord realization (from 2002 to 2005). More detailed discussion is devoted to the code development performed in the framework of Attachment #2 to the Accord (2004 -2005).

The code benchmarking and support will continue during 1-year service term presume until April 1, 2006.

Contents

Introduction 3

I. Summary of code development 4

I.1. Initial status of the work 4
I.2. Results of the Accord completion 5
I.3. Development in the framework of Attachment #2 (2004-2005) 9

II. General results of the software development in the framework of Attachment #2. Physical description. Results of benchmarking. 11

II.1. Improvement of core-tail model 11
II.2. Algorithm for IBS simulation based on Bjorken-Mtingwa approach 12
II.3. Realization of G.Parzen model for bi-Gaussian distribution 16
II.3. Friction force for non-magnetized electrons 19
II.4. Simulation of ion-electron recombination in presence of undulator field 25
II.5. Format of the table for import of the friction force from external program 28
II.5.1. Expected shape of the friction force 28
II.5.2. Interpolation methods 30
II.5.3. Description of the numerical procedures and benchmarking results 31
II.6. Integration of BETACOOL under UAL framework 35
II.6.1. Library realization and adaptor scheme 35
II.6.2. User program and usage of adaptors and library 38

References 40

Introduction

In the framework of the **Accord on a Software development for Electron Cooling Calculation** JINR provided source-codes for RHIC electron cooling simulation based on BETACOOL program. New BETACOOL code is platform independent object oriented code written in the ANSI / ISO standard of C++ language and it is opened for further development. In the current version of BETACOOL the model of electron beam describes essential peculiarities of the RHIC cooling system. A library of analytical formulae for friction force calculation was developed. The intrabeam scattering process can be calculated using a few different models. The calculations are based on two algorithms: numerical solution of the differential equations for root mean square parameters of the ion distribution function and simulation of the distribution function evolution using modelling particles.

The Accord was extended till April 1, 2005 by second Attachment in order to improve models of IBS and electron cooling simulations, develop new models for these effects calculation and include a collision effects calculation (several models of luminosity and beam-beam parameter).

As result of this work several new models of intrabeam scattering were implemented into program: the so-called "core – tail" model, Bjorken-Mtingwa IBS theory model, G.Parsen "Bi-gaussian" IBS model. All existent models of IBS were carefully benchmarked and compared to the experimental data from RHIC. Library of the electron cooling models was extended and tested: non-magnetized friction force model, numerically calculated friction force which can be implemented as a table with arbitrary division on ion velocities, new model of electron beam is implemented.

Concerning effects of colliding beam – the luminosity calculation was developed for different models of the particle distribution, calculation of the beam-beam parameter evolution in time is also provided for different beam emittance representations.

BETACOOL code as a library was successfully integrated into Unified Accelerator Library (UAL) for dynamics simulation. Different models of the heating/cooling effects calculations developed in BETACOOL are made to be enabled in the frame of UAL.

The first part of this report briefly describes results of the software development during the total term of the Accord realization (from 2002 to 2005). More detailed attention is devoted to code development performed in the framework of Attachment #2 to the Accord (2004 -2005). In the second part the basic physical models developed during last year and results of their benchmarking are presented.

I. Summary of code development

I.1. Initial status of the work

The software for electron cooling simulation at RHIC is based on BETACOOL program, which is being developed in JINR in cooperation with RIKEN (Japan), FZJ (Germany) and ITEP (Russia) [1,2].

Initial version of the program was dedicated to simulation of long term beam dynamics in ion storage rings. It was based on solution of the equation system for second order momentum of the ion distribution function and particle number (so called "RMS dynamics"). Such a model based on assumption that the ion beam has Gaussian distribution over all degrees of freedom permits to take into account a few heating and cooling effects by uniform way: each effect involved into simulations is presented as a set of heating (or cooling) rates and rate of the particle loss. The obvious advantage of this model is its simplicity that makes calculations faster than in the case of multiparticle simulations. The program was developed for usage on PC under Windows operation system. The following effects were used in the simulations:

- Electron cooling,
- Stochastic cooling,
- Intrabeam scattering,
- Scattering on residual gas,
- Interaction with internal target,
- Injection of new portion of the ion beam,
- External heating of the ion beam.

For stochastic cooling and scattering on residual gas simplest analytical models were used. IBS effect was simulated in accordance with Piwinski [3] or Martini [4] model. Electron cooling simulations were based on a library of analytical formulae for friction force calculation.

However, the initial version of the BETACOOL program had a few disadvantages, which did not permit to use it directly for simulations of electron cooling process at RHIC.

First of all it is related to the model of electron cooling system. The BETACOOL program was aimed for electron cooling simulations in low energy cooling systems based on electrostatic acceleration of DC magnetized electron beam of the round shape cross-section and uniform electron density. The cooling section was treated as a thin lens, therefore the solenoid field errors can not be taken into account correctly.

The RHIC electron cooler is planned to use bunched electron beam accelerated by energy recovery linac, therefore one cannot describe the electron beam parameters in the frame of the described model. The model can be used only in the case, when electron bunch length is substantially longer than ion one and electron density distribution is uniform in radial direction. Simulations of the electron beam dynamics shows, that electron bunch length can be comparable or even shorter than ion one and shape of the radial density distribution is determined by parameters of the electron beam acceleration and transportation systems and, in principle, can be arbitrary.

Long length of the cooling section leads to necessity of integration of the ion motion equation inside the cooling section in order to estimate requirements to the magnetic field quality.

The second problem related to electron cooling simulation is the choice of the algorithm of the friction force calculation. The proposed RHIC cooling system parameters are unique comparing with existing cooling systems: temperature of transverse degree of freedom of the electron beam is higher about four orders of magnitude and magnetic field value in the cooling section is larger by one order of magnitude than in usual cooling systems. Therefore, it is necessary to perform numerical calculations of the friction force and to use obtained results for the cooling process simulation.

Concerning intrabeam scattering (IBS) calculations one should note that BETACOOL program was never used before for simulation of a bunched ion beam at energy above the transition energy of the ring. Therefore, this part of the program had to be accurately tested and corrected. It was necessary also to develop new models for IBS simulation.

More serious required improvement of the program was related to modification of its basic physical model. Preliminary estimations of the RHIC ion beam parameters evolution due to common action of electron cooling and IBS showed that one can expect an appearance of non-Gaussian tails of the particle distribution function. In this case the luminosity is determined mainly by the core of the ion distribution function and simulations of RMS beam parameters can not predict correctly gain in luminosity due to action of electron cooling.

Other disadvantages of initial state of the program were related to the code conditions complicated for further development:
- the code was written using C++ Builder compiler and operated only with Windows, syntax of the C++ Builder was used inside the physical part of the code,
- the code had complicated structure, lack of comments, absence of unification in the variable dimensions.

I.2. Results of the Accord completion

During realization of the Accord the initial software of the code was completely rebuild. The code was divided by two independent parts: physical code, which is made using only standard C++ syntax and interface part, which is an executable file working under Windows environment. Connection between two parts of the program is provided using three types of the files: input, output and file containing parameters for control of the calculation process. Such a structure on the one hand allows to use the program on PC, control and analyse results during simulations. From the other hand the physical part of the program can be compiled for UNIX operation system and used for calculations on supercomputer independently on interface. The interface part in this case is used for preparation of input files and post processing of the simulation results after completion of the calculations. UNIX version of the program was integrated into Unified Accelerator Library (UAL) framework as a library of effects. Format of the file of input parameters was modified to simplify usage of the program with command prompt interface. All input and output files are in the text format. The parameters in the input file are divided by groups in accordance with the structure of BETACOOL objects.

In the physical part of the code the following general modifications were done:
- for unification of the formulae presentation the new class of dimensional variables was developed and benchmarked,
- new algorithm for simulation of the ion distribution function evolution was developed. In this algorithm (called Model Beam algorithm) the ion beam is presented by array of modeling particles,
- the models of the ion ring and the ion beam were developed in such a way, that allows realizing both basic algorithms (RMS dynamics and Model Beam) at the same input parameters,
- the heating and cooling effects are realized on the basis of common standard and at the same parameters can be used in each algorithm,
- new models for IBS and electron cooling simulation were developed,
- new effects for particle loss simulation, for calculation of luminosity and beam-beam parameters, for simulation of external heating were developed.

The Model beam algorithm is based on solution of Langevin equation in the momentum space for each particle from array presenting the beam. The effects leading to variation of the ion distribution function are presented by friction and diffusion terms. The friction term lead to regular variation of the particle momentum, the diffusion – to random variation of the momentum components and it corresponds to so called “Langevin’s force”. In such a way, the heating and cooling effects involved into simulations lead to change of the particle momentum components and particle number, which is calculated in accordance with step of dynamics simulation over time. Each effect is located in some position of the ring characterizing by the ring lattice functions. Transformation of the beam inside the ring is provided using linear matrix at random phase advance between the effect locations. Results of the simulations can be presented both as evolution of the beam profile in time or as time dependencies of the beam emittance and particle number.

For IBS diffusion power calculation the real ion ring optics structure is necessary. The model of the ion ring realizes a few possibilities of initialization of the ring structure. The ring optics structure can be described in the input file of the text format like MAD input file. The lattice functions in this case can be calculated using internal interpreter or using MAD8 program, which can be launched from BETACOOL. The lattice function can be also imported from output file of the MAD program as it was realized in initial version of the program.

The ion beam model realizes a few algorithms for calculation of distribution function parameters of the particle array. The beam model calculates rms parameters of the distribution function, Full Width on Half Maximum of the beam profile in each degree of freedom or phase volume including given percent of the particles.

In the case of RMS dynamics simulation the change of the beam emittance due to IBS can be calculated on the basis of one of the following analytical models:

- Piwinski [3],

- Martini, or extended Piwinski [4],

- Bjorken-Mtingwa [5],

- two modifications of Jie Wei model [6],

- plasma relaxation formulae [7].

In the Model Beam algorithm the following simplest models for detailed IBS simulation are available:

- Burov's model [8],

- two modifications of the core-tail model [9],

- G.Parzen formulae for Bi-Gaussian distribution [10].

To solve all the problems related with the cooling process simulation mentioned in the chapter I.1 a hierarchy of objects was developed. Structure of the electron cooler presentation permits to extract procedures of different levels and to include them into calculation of the cooling process in other programs.

The cooling simulation is based on a friction force calculation in the particle rest frame. The friction force can be calculated in accordance with one of the analytical models from a library including:

- Budker's formula [11],

- Formulae for nonmagnetized electron beam with flattened velocity distribution based on asymptotic representation [12] or on numerical evaluation of the integrals over the electron distribution function,

- Parkhomchuk's formula [13],

- Derbenev-Skrinsky-Meshkov formulae [14, 15].

The friction force values can be also imported from external file calculated by another program.

The next layer of the simulation is related with a cooler representation as a map, transforming particle coordinates from entrance to the exit of the cooling section and calculating the ion loss

probability due to recombination with electrons. Calculation of the cooler map is based on a model of electron beam that provide transformation of the ion velocity to the frame related with the electron beam and takes into account real geometry of the cooler. Now in the BETACOOL five models of electron beam are available for simulations:

-uniform cylinder, that corresponds to electron beam geometry in usual electron cooling systems,

- Gaussian cylinder, which can be used when the electron bunch has a Gaussian distribution in transverse directions and the electron bunch length is sufficiently longer than the ion one,

- Gaussian bunch, having Gaussian distribution in all degrees of freedom,

- Uniform bunch, having Gaussian distribution in the longitudinal direction and uniform distribution in the transverse ones,

- Hollow beam, having different electron density in the central and outer parts.

The cooler model takes into account variation of the magnetic field in the cooling section. For this purpose the co-ordinates of the electron beam trajectory inside cooling section are taken from additional file and the ion motion equations are solved numerically inside the cooler.

On the basis of the cooler map the program calculates a kick of the ion momentum after crossing the cooling section that is necessary for simulation of the ion distribution evolution in the frame of the Model Beam algorithm. The map of the cooler is used also for the cooling rate calculation that is necessary for RMS dynamics simulation. The cooling rate calculation can be performed using two model of the ion beam – the cooling rates for “rms particle”, or cooling rates for the ion beam with Gaussian distribution in all degrees of freedom.

The luminosity and beam-beam parameter calculation in the frame of RMS dynamics simulation is based on analytical formulae. For Model Beam algorithm a few numerical procedures were developed.

The effects for particle loss simulation takes into account the particle losses due to electron capture in the cooling section, due to interaction with residual gas and due to reactions in the collision point. In the frame of Model Beam algorithm there are procedures for particle loss simulation due to acceptance and separatrix limitations. A few procedures for generation of new modeling particles in the case of losses were developed.

All the algorithms were benchmarked by comparison of results with other codes and with experimental data obtained at RHIC, CELSIUS, COSY, LEAR, ESR and AD rings.

The code benchmarking will continue during 1-year service term until April 1, 2006.

I.3. Development in the framework of Attachment #2 (2004-2005)

1. Two basic algorithms were improved: dynamics simulation of rms ion beam parameters and Model Beam algorithm. The following changes were made:

- the model of the ring was modified to allow usage of lattices which are calculated on-line with MAD. MAD application can be launched from BETACOOL,

- all effects were modified in the uniform form to be able to calculate momentum deviation (so-called Kick) from any effect,

- a possibility to use a particle beam with different species to use as a colliding beam is realized,

- numerical procedures for calculation of the particle array parameters were developed.

1.1. Modification of the ring model required, development of the following procedures and algorithms:

- interpreter of input MAD file,
- format of input BETACOOL file describing an ion ring optic structure, based on format of the input MAD file,

1.2. As a result of the effect structure modification in the present version each effect can be uniformly used in all basic algorithms.

2. Electron cooling calculation was improved in the following way:

- structure of the effect was rebuilt and unified,
- algorithm which uses for electron cooling a tabulated friction force values pre-calculated with another code was developed,
- algorithm for numerical calculation of the friction force for non-magnetized electrons was developed,
- algorithm which realizes simulation of the ion-electron recombination in presence of an undulator field was implemented and tested,
- new models of the electron beam was added – uniform bunch and hollow beam.

3. Development of intrabeam scattering simulation included:

- improvement of the core-tail model for detailed IBS simulation. The “core – tail” model was improved using bi-Gaussian approximation of the ion distribution function. In the frame of the “core-tail” model in the Model Beam algorithm the core parameters were calculated using FWHM characteristics of the beam profile, the tail was determined as an rms profile width. In general case the model particle distribution function can be presented as a sum of two Gaussian functions for core and tail particles,
- model of the IBS rates calculation in accordance with theory by Bjorken-Mtingwa,
- IBS effect calculation in the frame of G.Parzen assumption for Bi-Gaussian distribution of the ion beam.

4. For calculation of luminosity and particle loss in collision point new effect was created. For luminosity calculation in Model Beam algorithm the following procedures were developed:

- procedure presuming different models for luminosity calculation – via local density, coordinate ellipsoid, or profile density,
- calculation of beam-beam parameter in three ways of the beam emittance presentation – RMS, FWHM or emittance occupied by the definite percentage of particles.

The procedures are based on calculation of local ion beam density [particles/cm^2] along the ion trajectory.

For axial symmetry beam the luminosity can be calculated using particles co-ordinates or smoothed profiles.

For arbitrary distribution one can use the particle co-ordinates only, in this procedure estimation for hourglass effect is included.

5. Simulation of the effect of additional artificial heating was improved. This effect realizes the following new kinds of heating:

- constant rates,
- linear deviation of beam emittances,
- diffusion power,
- diffusion heating acting on beam emittances.

Development of the interface part and format of input file

Development of the interface part has a goal to make the interface structure reflecting the structure of the algorithms and to extend possibilities for user to work with file structure on hard disk.

The interface part was modified in accordance with change of the program structure: visual forms for input of algorithm parameters, beam-beam parameter, additional heating effects were created. Forms for existing effects: Electron cooling, IBS, particle losses, Lattice structure were modified.

Additional forms were developed for visualization of the particle array parameters and for definition of algorithm parameters. Every algorithm has its own form for parameterization, and there are common forms for saving and format the results. Functions for control of the graphics calculation are combined in special form.

UNIX version of the program and integration with other packages.

1. Prepared and benchmarked the version of BETACOOL for UNIX OS.
2. BETACOOL program was successfully compiled as an independent library of procedures. Special required adapter procedures for integration of BETACOOL under UAL framework were prepared.
3. Intrabeam scattering simulations in the frame of Model Beam algorithm were compared and tested with results of tracking simulation based on Molecular Dynamics technique.
4. Major cooling models developed in BETACOOL were tested and benchmarked in the frame of scheme when BETACOOL is implemented as a library into UAL framework.

II. Results of the software development in the framework of Attachmet #2. Physical description. Results of benchmarking

II.1. Improvement of core-tail model

In the first version of the "core-tail" model in the Model Beam algorithm the core parameters were calculated using FWHM characteristics of the beam profile, the tail was determined as an rms profile width. The "core – tail" model was later improved using fitting of core and tails with separate Gaussians. The modeled particle distribution function can be presented as a sum of two Gaussian functions for core and tail particles:

$$f(x) = f_{tail}(x) + f_{core}(x) = a_{tail} \exp\left[-\frac{1}{2}\left(\frac{x}{\sigma_{tail}}\right)^2\right] + a_{core} \exp\left[-\frac{1}{2}\left(\frac{x}{\sigma_{core}}\right)^2\right], \qquad \text{(II.1.1)}$$

where a_{tail}, a_{core}, σ_{tail}, σ_{core} – amplitudes and widths of beam profile. These parameters can be calculated using mean square method. This method minimizes the deviation between the model beam profile and Bi-Gaussian distribution:

$$\sum_{i=1}^{N} \left[y_i - f(x_i)\right]^2 \to Minimum\,, \qquad \text{(II.1.2)}$$

where (x_i, y_i) – points of the beam profile distribution, N is total number of profile histogram divisions. The optimization problem is solving in the program using Powell method. The same procedure is realized for all degrees of freedom: for momentum distribution, horizontal and vertical profiles. RMS parameters of the ion beam $\vec{E}_{RMS}\left[\varepsilon_x, \varepsilon_y, \Delta p / p\right]$ are calculated in the suggestion that distribution of model particles has Gaussian shape. Beam parameters (emittances, momentum spread) and particle number of Bi-Gaussian distribution are:

$$\vec{E}_{tail} = \vec{E}_{RMS}\sigma_{tail}^2, \quad \vec{E}_{core} = \vec{E}_{RMS}\sigma_{core}^2, \quad N_{core} = \left[1 + \frac{\sigma_{tail}}{\sigma_{core}} \cdot \frac{a_{tail}}{a_{core}}\right]^{-1}, \quad N_{tail} = 1 - N_{core}\,. \qquad \text{(II.1.3)}$$

The IBS growth rates for core and tail parts of Bi-Gaussian distribution are calculates with standard procedure on the base of choosing IBS model for Gaussian distribution:

$$\begin{aligned} \vec{R}_{tail} &= IBS.Rates(\vec{E}_{tail}, N) \\ \vec{R}_{core} &= IBS.Rates(\vec{E}_{core}, N) \end{aligned}. \qquad \text{(II.1.4)}$$

Model particles in the tail get the kick with standard procedure using of $\vec{R}_{tail}$ growth rates. Particles in the core get the kick from core and tail heating rates in accordance with:

$$\vec{R}_{core} N_{core}^2 + \vec{R}_{tail} N_{tail}^2 \frac{\vec{E}_{tail}}{\vec{E}_{core}}\,. \qquad \text{(II.1.5)}$$

II.2. Algorithm of IBS simulation based on Bjorken-Mtingwa approach

To take into account dispersion in vertical plane as well as in horizontal the IBS growth rates are calculated in accordance with [16]:

$$\begin{cases} \dfrac{1}{\tau_x} = \left\langle \dfrac{\mathrm{H}_x}{\varepsilon_x}\gamma_0^2 I_{zz} - 2\dfrac{\beta_x\phi_{Bx}}{\varepsilon_x}\gamma_0 I_{xz} + \dfrac{\beta_x}{\varepsilon_x} I_{xx} \right\rangle_s \\ \dfrac{1}{\tau_y} = \left\langle \dfrac{\mathrm{H}_y}{\varepsilon_y}\gamma_0^2 I_{zz} - 2\dfrac{\beta_y\phi_{By}}{\varepsilon_y}\gamma_0 I_{yz} + \dfrac{\beta_y}{\varepsilon_y} I_{yy} \right\rangle_s \\ \dfrac{1}{\tau_z} = \left\langle \dfrac{1}{2}\dfrac{\gamma_0^2}{\sigma_p^2} I_{zz} \right\rangle_s \end{cases} \qquad \text{(II.2.1)}$$

where $\phi_{Bi} = D_i^{/} + \alpha_i D_i / \beta_i$, and $\alpha_i, \beta_i, \gamma_i$ - lattice functions in the horizontal (i=x) and vertical (i=y) plane, $\varepsilon_{x,y}$ are the horizontal and vertical emittances, σ_p – rms momentum spread. Angular brackets mean averaging over the ring circumference. At zero vertical dispersion these formulae coincide with original Bjorken-Mtingwa theory. The diffusion coefficients I_{ij} are calculated in each position of the ring by numerical evaluation of the following integrals

$$I_{ij} = A\int_0^\infty d\lambda \frac{\lambda^{1/2}}{\sqrt{\det \Lambda}}(\delta_{ij} Tr\Lambda^{-1} - 3\Lambda_{ij}^{-1}), \qquad \text{(II.2.2)}$$

where the matrix $\Lambda = I\lambda + L$, I – unit matrix, and matrix L is calculated via beam rms parameters and ring lattice functions in accordance with:

$$L = \begin{bmatrix} \dfrac{\beta_x}{\varepsilon_x} & 0 & -\gamma_0\dfrac{\beta_x\phi_{Bx}}{\varepsilon_x} \\ 0 & \dfrac{\beta_y}{\varepsilon_y} & -\gamma_0\dfrac{\beta_y\phi_{By}}{\varepsilon_y} \\ -\gamma_0\dfrac{\beta_x\phi_{Bx}}{\varepsilon_x} & -\gamma_0\dfrac{\beta_y\phi_{By}}{\varepsilon_y} & \dfrac{\gamma_0^2\mathrm{H}_x}{\varepsilon_x} + \dfrac{\gamma_0^2\mathrm{H}_y}{\varepsilon_y} + \dfrac{\gamma_0^2}{\sigma_p^2} \end{bmatrix}, \qquad \text{(II.2.3)}$$

where $(i = x, y)$, $\mathrm{H}_i = \beta_i D'^2_i + 2\alpha_i D_i D_i^{/} + \gamma D_i^2$. The IBS constant A is determined as in other IBS models:

$$A = \frac{c r_i^2 N L_c}{8\pi\beta^3\gamma_0^2\varepsilon_x\varepsilon_y\sigma_p\sigma_s}. \qquad \text{(II.2.4)}$$

Here β and γ_0 are the Lorenz parameters, r_i is the ion classical radius, N is the ion number, L_c is the Coulomb logarithm, which is introduced as an input parameter.

For benchmarking of IBS calculation a comparison was provided between two base models of IBS calculation of Martini and newly developed model based on Bjorken-Mtingwa theory. Calculation

results for beam parameters evolution are in good agreement for different models (Figs. II.2.(1,3,5)). Simulations were performed for RHIC FODO lattice at standard beam parameters.

Examples of calculation of beam parameter evolution including electron cooling, IBS, collision point are presented on (Figs. II.2.(2,4,6)). Simulations were performed for the following parameters: RHIC FODO lattice at standard ion beam parameters, ECOOL: uniform cylinder model for electron beam, beam radius :0.1 cm, beam current – 50 A. Results for different models of IBS with other effects are in good agreement for all cases.

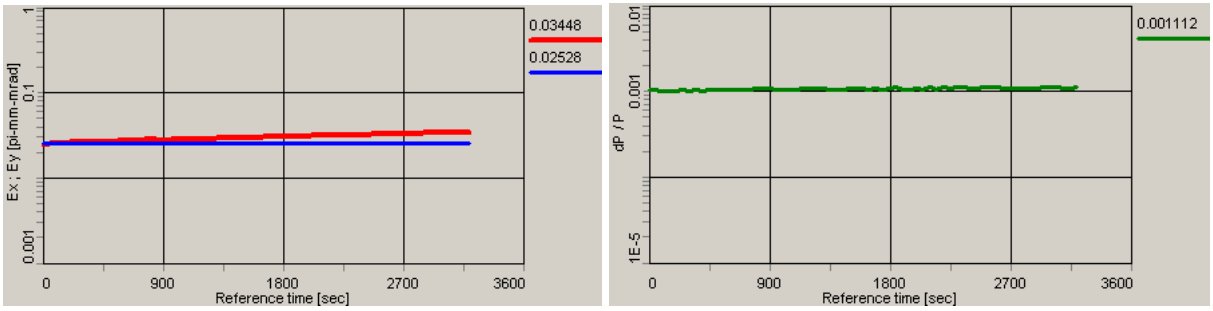


Fig. II.2.1. Left plot - emittance time dependence, right plot – dP/P time dependence. Bjorken - Mtingwa model for IBS, following effects are switched: IBS

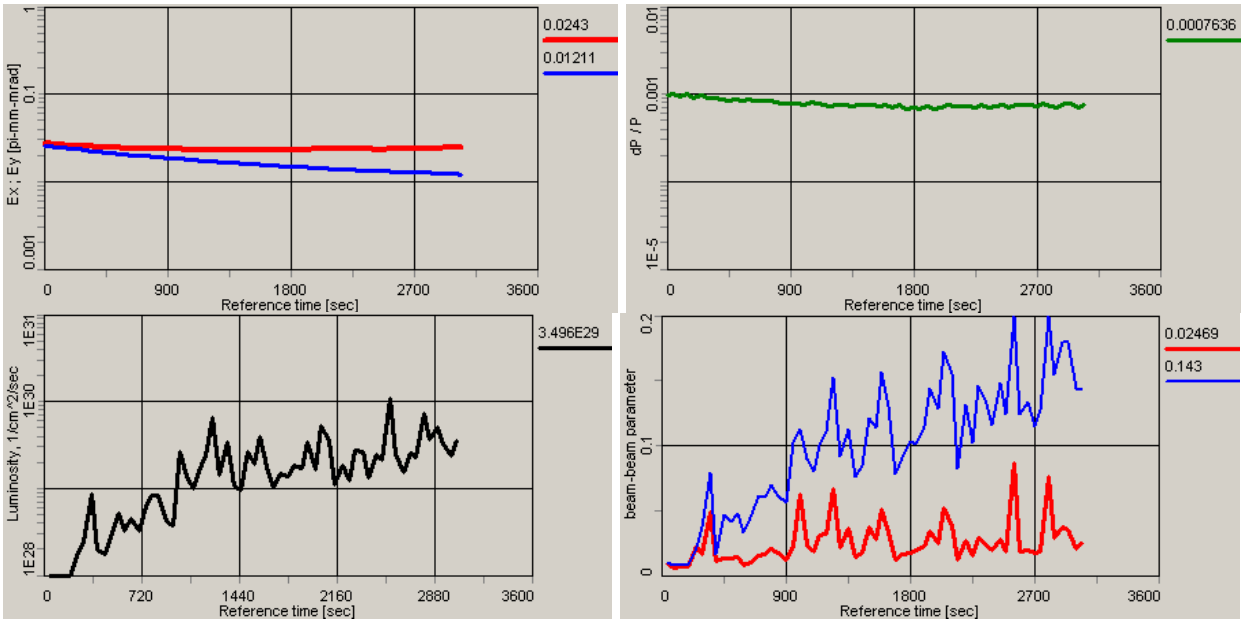


Fig. II.2.2. Left up plot - emittance time dependence, right up plot – dP/P time dependence, left bottom plot – luminosity time dependence, right bottom plot – beam-beam parameter time dependence. Bjorken - Mtingwa model for IBS, Following effects are switched: IBS + ECOOL + COLLISION POINT

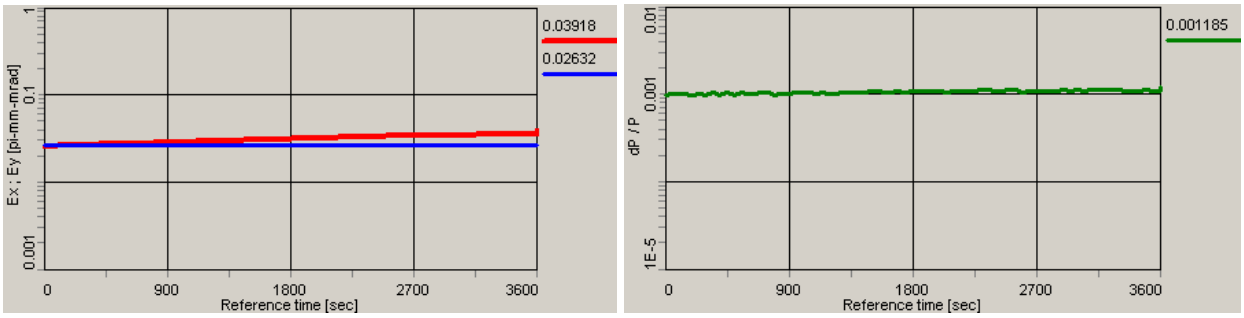


Fig. II.2.3. Left plot - emittance time dependence, right plot – dP/P time dependence. Martini model for IBS, Following effects are switched: IBS

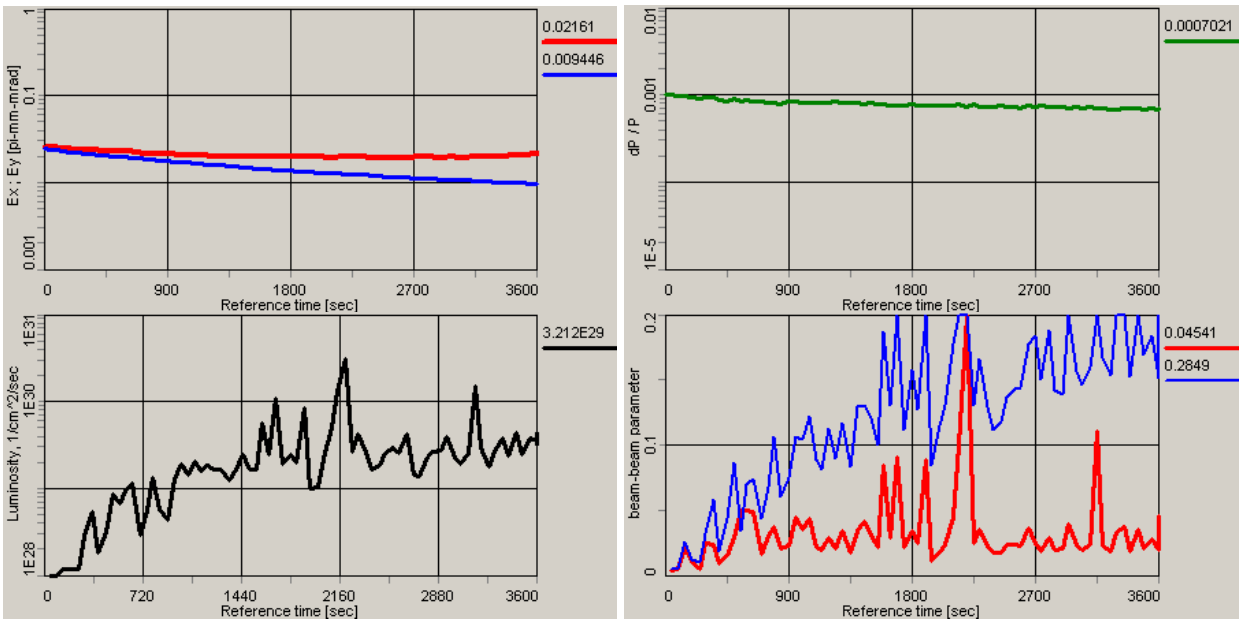


Fig. II.2.4. Left up plot - emittance time dependence, right up plot – dP/P time dependence, left bottom plot – luminosity time dependence, right bottom plot – beam-beam parameter time dependence. Martini model for IBS, Following effects are switched: IBS + ECOOL + COLLISION POINT

II.3. Realization of Parzen model for bi-Gaussian distribution

The model for IBS growth rates calculation proposed by Parzen [10] is based on presentation of the ion distribution function as a sum of two Gaussian distributions. The major difference of this analysis from the "core-tail" model is that it describes rms evolution of the distribution. While the "core-tail" model is an approximate approach to describe diffusion coefficients for individual particles which becomes very important when one has other effects, like electron cooling, acting on beam distribution. $N\,f(x,p)$ gives the number of particles in d^3xd^3p, where N is the number of particles in a bunch. For a bi-gaussian distribution, $f(x,p)$ is given by

$$f(x,p)=\frac{N_c}{N}\frac{1}{\Gamma_c}\exp\left[-S_c(x,p)\right]+\frac{N_t}{N}\frac{1}{\Gamma_t}\exp\left[-S_t(x,p)\right]. \qquad \text{(II.3.1)}$$

Here N_c and N_t are the particle number in the first (corresponding tu the core of total distribution) and second Gaussian (describing the tail of the total distribution). $\Gamma_{c,t}=\int\exp\left(-S_{c,t}(x,p)\right)d^3xd^3p$ are the corresponding normalization factors,

$$S=S_x+S_y+S_s, \qquad \text{(II.3.2)}$$

where

$$S_i=\frac{I_i}{\varepsilon_i},\ i=x,\ y,\ s \qquad \text{(II.3.3)}$$

is the ratio of the particle invariant of the motion I to 2-sigma emittance ε in the corresponding plane.

Because of $N_c + N_t = N$ the distribution (II.3.1) is described by 7 independent parameters: $\varepsilon_{x,c}, \varepsilon_{y,c}, \varepsilon_{s,c}, \varepsilon_{x,t}, \varepsilon_{y,t}, \varepsilon_{s,t}$ and N_c. To find these parameters for arbitrary array of particles one needs to solve 7-D optimization problem, which looks unrealistic. To avoid this difficulty the 7-D optimization problem was replaced by solution of 3 independent 3-D optimization problem. For each degree of freedom from smoothed beam profile the program calculates rms emittances for core and tail of distribution function using Powel method as described in chapter II.1. As result of the calculations one has a set of 9 parameters:

$$\varepsilon_{x,c}, \varepsilon_{x,t}, N_{x,c},$$
$$\varepsilon_{y,c}, \varepsilon_{y,t}, N_{y,c},$$
$$\varepsilon_{s,c}, \varepsilon_{s,t}, N_{s,c},$$

the emittances are used as corresponding parameters of Bi-Gaussian distribution, and intensity of core Gaussian is calculated as:

$$N_c=\frac{N_{x,c}+N_{y,c}+N_{s,c}}{3}. \qquad \text{(II.3.4)}$$

Thereafter the program calculates additional four parameters, required for evaluation of growth rates for total beam:

$$\varepsilon_{x,ct} = \frac{\varepsilon_{x,c}\varepsilon_{x,t}}{\varepsilon_{x,c}+\varepsilon_{x,t}},\ \varepsilon_{y,ct} = \frac{\varepsilon_{y,c}\varepsilon_{y,t}}{\varepsilon_{y,c}+\varepsilon_{y,t}},\ \varepsilon_{s,ct} = \frac{\varepsilon_{s,c}\varepsilon_{s,t}}{\varepsilon_{s,c}+\varepsilon_{s,t}} \text{ and } N_{ct} = \frac{N_c N_t}{N^2}. \qquad \text{(II.3.5)}$$

The diffusion coefficients for core and tail Gaussians are calculated in accordance with Bjorken-Mtingwa model, as described in chapter II.2:

$$I_{ij,c} = A_c \int_0^\infty d\lambda \frac{\lambda^{1/2}}{\sqrt{\det \Lambda_c}} (\delta_{ij} Tr\Lambda_c^{-1} - 3\Lambda_{ij,c}^{-1}), \qquad \text{(II.3.6)}$$

$$I_{ij,t} = A_t \int_0^\infty d\lambda \frac{\lambda^{1/2}}{\sqrt{\det \Lambda_t}} (\delta_{ij} Tr\Lambda_t^{-1} - 3\Lambda_{ij,t}^{-1}), \qquad \text{(II.3.7)}$$

$$I_{ij,ct} = A_{ct} \int_0^\infty d\lambda \frac{\lambda^{1/2}}{\sqrt{\det \Lambda_{ct}}} (\delta_{ij} Tr\Lambda_{ct}^{-1} - 3\Lambda_{ij,ct}^{-1}), \qquad \text{(II.3.8)}$$

where matria $\Lambda_c, \Lambda_t, \Lambda_{ct}$ are calculated in accordance with (II.2.3) with substitution of corresponding emittances. The IBS constants are calculated as usual:

$$A_i = \frac{c r_i^2 N_i L_c}{8\pi\beta^3\gamma_0^2 \varepsilon_{x,i}\varepsilon_{y,i}\sigma_{p,i}\sigma_{s,i}}. \qquad \text{(II.3.9)}$$

Diffusion coefficients for the total beam are calculated as:

$$I_{ij} = I_{ij,c}\frac{N_c}{N} + I_{ij,t}\frac{N_t}{N} + 2I_{ij,ct}. \qquad \text{(II.3.10)}$$

And finally the heating rates are calculated with the same formula (II.2.1). This procedure is repeated in each optic element of the ring and resulting heating rates are averaged over the ring circumference.

The described algorithm does not coincide exactly with original paper, however it uses the same final formula (II.3.10) and based on the same modification of the Bjorken-Mtingwa model as described in chapter II.2, which simplifies benchmarking of the model.

Without electron cooling (when the distribution function is closed to Gaussian) the results of simulations with G.Parzen model coincide within a few percents with results with Bjorken-Mtingwa model. For example beam parameters evolution simulated with G.Parzen model is presented in Figs. II.3.1 and II.2.3. Simulations were performed for RHIC FODO lattice at standard ion beam parameters.

In the case of electron cooling (when distribution deviates from Gaussian) simulation results based on Parzen model are very far from the results of other models. Examples of calculation of beam

parameters evolution including electron cooling, IBS, collision point are presented on Fig. II.3.2. Simulations were performed for the following parameters: RHIC FODO lattice at standard ion beam parameters, ECOOL: uniform cylinder model for electron beam, beam radius is 0.1 cm, beam current – 50 A. One can see, that instead of cooling in horizontal and vertical longitudinal degrees of freedom there is a fast heating. Emittance decreases only in vertical plane and as result the luminosity stay almos constant. This can be explained by the realized method of approximation the real particle distribution by two Gausses. The intensity of "core" Gaussian is calculated as intensity averaged between degrees of freedom (formula II.3.4). When formation of the core in one degree of freedom is faster then in others the diffusion coefficients increase in all degrees of freedom. Therefore the diffusion increases in degrees of freedom where core is not formed yet.

This problem can be solved by calculation of core intensity making averaging in accordance with weights of degrees of freedom. This procedure is under development now.

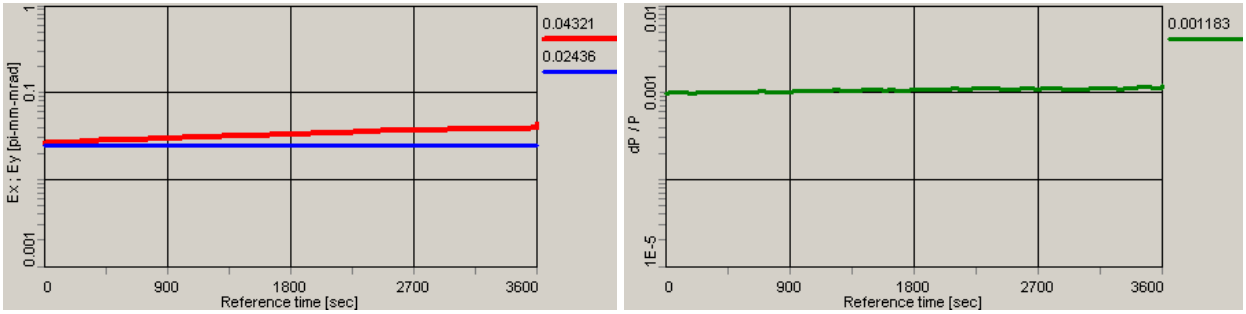


Fig. II.3.1. Left plot - emittance time dependence, right plot – dP/P time dependence Following effects are switched: IBS - G.Parzen theory.

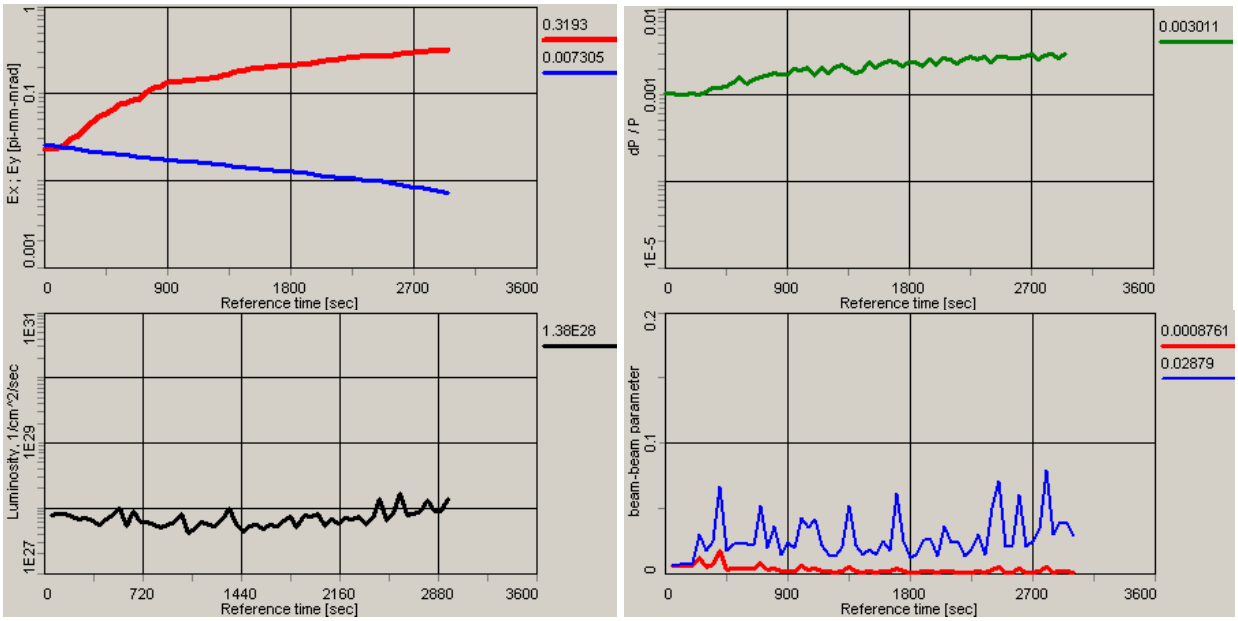


Fig. II.3.2. Left up plot - emittance time dependence, right up plot – dP/P time dependence, left bottom plot – luminosity time dependence, right bottom plot – beam-beam parameter time dependence. Following effects are switched: IBS (G.Parzen) + ECOOL + COLLISION POINT.

II.3. Friction force for non-magnetized electrons

In absence of magnetic field a binary collision is described as Rutherford scattering of ion on a free electron. In this case the friction force acting on the ion inside the electron beam with velocity distribution function $f(v_e)$ can be calculated with usual formula [12]:

$$\vec{F} = -\frac{4\pi n_e e^4 Z^2}{m} \int L \frac{\vec{V}_i - \vec{v}_e}{\left|\vec{V}_i - \vec{v}_e\right|^3} f(v_e) d^3 v_e , \qquad \text{(II.3.1)}$$

where n_e is electron density in the Particle Rest Frame (PRF), v_e, V_i are the electron and ion velocity, L – Coulomb logarithm:

$$L = \ln \frac{\rho_{\max}}{\rho_{\min}} . \qquad \text{(II.3.2)}$$

The maximum impact parameter ρ_{max} is calculated as a minimum from three values:

$$\rho_{\max} = \min\left\{\rho_{sh}, V_i \tau, a\right\}. \qquad \text{(II.3.3)}$$

The first term is a radius of the dynamic shielding sphere. In the case of uniform non-magnetized electron beam it can be calculated as:

$$\rho_{sh} = \frac{\sqrt{V_i^2 + \Delta_e^2}}{\omega_p} , \qquad \text{(II.3.4)}$$

that coincides with Debay radius when the ion velocity is less than electron velocity spread. The plasma frequency ω_p is equal to

$$\omega_p = \sqrt{\frac{4\pi n_e e^2}{m_e}} . \qquad \text{(II.3.5)}$$

Here

$$\Delta_e = \sqrt{\Delta_\perp^2 + \Delta_\parallel^2} \qquad \text{(II.3.6)}$$

is rms electron velocity spread, when the electron beam temperature is different in transverse and longitudinal degrees of freedom $T_\perp = m\Delta_\perp^{\,2}$, $T_\parallel = m\Delta_\parallel^{\,2}$.

The second term describes the distance, which the ion passes inside the electron beam. Here τ is the ion time of flight the cooling section in the PRF:

$$\tau = \frac{l_{cool}}{\beta\gamma c} . \qquad \text{(II.3.7)}$$

and the last term a is the electron beam radius (in the case of Gaussian distribution of electrons in radial direction this value is not used).

The minimum impact parameter ρ_{min} is determined by relative ion and electron velocity:

$$\rho_{\min} = \frac{Ze^2}{m_e} \frac{1}{\left|\vec{V}_i - \vec{v}_e\right|^2}. \qquad \text{(II.3.8)}$$

In general case the velocity distribution function can be approximated by Maxwellian distribution with different temperatures of longitudinal and transverse degrees of freedom:

$$f(v)d^3v = \left(\frac{m}{2\pi}\right)^{3/2} \frac{1}{T_\perp \sqrt{T_{II}}} e^{-mv^2{}_\perp / 2T_\perp - mv_{II}^2 / 2T_{II}} \, 2\pi v_\perp dv_\perp dv_{II}, \qquad \text{(II.3.9)}$$

In the case of RF electron beam acceleration the temperatures of the transverse and longitudinal degrees of freedom can be calculated from electron beam parameters as follows:

$$T_\perp = mc^2\beta^2\gamma^2\theta_{rms}^2,$$
$$T_\parallel = mc^2\beta^2\left(\frac{\Delta p}{p}\right)^2, \qquad \text{(II.3.10)}$$

where θ_{rms} is r.m.s. angular spread and $\Delta p/p$ – r.m.s. momentum spread of electrons in the cooling section.

In the previous version of the BETACOOL program for non-magnetized electron beam the asymptotic formulae obtained with Coulomb analogy were used [12]. In the case, when transverse velocity spread of electrons is substantially larger than longitudinal one the friction force can be approximated in three ranges of the ion velocity.

I. High velocity $v_i >> \Delta_\perp$, here longitudinal and transverse components of the friction force are equal:

$$\vec{F} = -\frac{4\pi Z^2 e^4 n_e L}{m} \cdot \frac{\vec{v}_i}{v^3}, \qquad \text{(II.3.11)}$$

and in this range the friction force shape coincides with simplest Budker's formula.

II. Low velocity $\Delta_\parallel << v_i << \Delta_\perp$. Here the transverse component of the friction force is given by the following expression:

$$\vec{F}_\perp = -\frac{4\pi Z^2 e^4 n_e L}{m} \cdot \frac{v_{i,\perp}}{\Delta_\perp^3}, \qquad \text{(II.3.12)}$$

and longitudinal one:

$$F_{\|} = -\frac{4\pi Z^2 e^4 n_e L}{m} \frac{v_{\|}}{|v_{\|}|\Delta_{\perp}^2}. \qquad \text{(II.3.13)}$$

III. Superlow velocity $v_i << \Delta_{\|}$. Here the transverse component of the friction force is equal to zero, the longitudinal component is given by:

$$F_{\|} = -\frac{4\pi Z^2 e^4 n_e L}{m} \frac{v_{\|}}{\Delta_{\|}\Delta_{\perp}^2}. \qquad \text{(II.3.14)}$$

The minimal impact parameter in the Coulomb logarithm is equal to:

$$\rho_{\min} = \frac{Ze^2}{m_e} \frac{1}{v_i^2 + \Delta_e^2}, \qquad \text{(II.3.15)}$$

where

$$\Delta_e = \sqrt{\Delta_{\perp}^2 + \Delta_{\|}^2}. \qquad \text{(II.3.16)}$$

This value of the electron velocity spread is used also for calculation of the dynamic shielding radius in the formula (II.3.4).

For more accurate calculation of the friction force the algorithm for numerical integration of formula (II.3.1) with electron distribution function (II.3.9) was introduced. Transverse and longitudinal components of the vector formula (II.3.1) can be written in the following form adapted to the numerical integration:

$$F_{\perp} = -\frac{4\pi Z^2 e^4 n_e}{m \cdot Int} \int_0^{3\Delta_{\perp}} \int_{-3\Delta_{\|}}^{3\Delta_{\|}} \int_0^{\pi} \ln\left(\frac{\rho_{\max}}{\rho_{\min}}\right) \frac{(V_{\perp} - v_{\perp}\cos\varphi)\exp\left(-\frac{v_{\perp}^2}{2\Delta_{\perp}^2} - \frac{v_{\|}^2}{2\Delta_{\|}^2}\right)}{\left((V_{\|} - v_{\|})^2 + (V_{\perp} - v_{\perp}\cos\varphi)^2 + v_{\perp}^2 \sin^2\varphi\right)^{3/2}} v_{\perp} d\varphi dv_{\|} dv_{\perp}$$

$$F_{\|} = -\frac{4\pi Z^2 e^4 n_e}{m \cdot Int} \int_0^{3\Delta_{\perp}} \int_{-3\Delta_{\|}}^{3\Delta_{\|}} \int_0^{\pi} \ln\left(\frac{\rho_{\max}}{\rho_{\min}}\right) \frac{(V_{\|} - v_{\|})\exp\left(-\frac{v_{\perp}^2}{2\Delta_{\perp}^2} - \frac{v_{\|}^2}{2\Delta_{\|}^2}\right)}{\left((V_{\|} - v_{\|})^2 + (V_{\perp} - v_{\perp}\cos\varphi)^2 + v_{\perp}^2 \sin^2\varphi\right)^{3/2}} v_{\perp} d\varphi dv_{\|} dv_{\perp} \qquad \text{(II.3.17)}$$

here the normalization factor is calculated in accordance with:

$$Int = \int_0^{3\Delta_{\perp}} \int_{-3\Delta_{\|}}^{3\Delta_{\|}} \int_0^{\pi} \exp\left(-\frac{v_{\perp}^2}{2\Delta_{\perp}^2} - \frac{v_{\|}^2}{2\Delta_{\|}^2}\right) v_{\perp} d\varphi dv_{\|} dv_{\perp}, \qquad \text{(II.3.18)}$$

the capital V is related to the ion velocity components, $v_{\|}$ and $v_{\perp}$ are the integration variables. The Coulomb logarithm is kept under the integral because of the minimal impact parameter is the following function of the electron velocity components:

$$\rho_{\min} = \frac{Ze^2}{m_e} \frac{1}{\left(V_{\parallel} - v_{\parallel}\right)^2 + \left(V_{\perp} - v_{\perp}\cos\varphi\right)^2 + v_{\perp}^2 \sin^2\varphi}. \qquad \text{(II.3.19)}$$

The dynamic shielding radius required for maximal impact parameter calculation is determined by parameters of the electron distribution function:

$$\rho_{sh} = \frac{\sqrt{V_{\perp}^2 + V_{\parallel}^2 + \Delta_{\perp}^2 + \Delta_{\parallel}^2}}{\omega_p}. \qquad \text{(II.3.20)}$$

Upper limit of the integration over φ is chosen to be equal π due to symmetry of the task. General problem in the numerical integration is to avoid division by zero in the formulae (II.3.17, II.3.19). Simplest solution is to choice a step of integration over polar angle so that $v_{\perp}\sin\varphi \neq 0$ at all possible values of φ, or $\varphi_i \neq 0, \pi$ at all i.

The formulae (II.3.17) are applicable at all axial symmetrical electron distribution functions and coincide with Budker's formula:

$$\vec{F}(\vec{v}_i) = -\frac{\vec{v}_i}{v_i^3} \frac{4\pi n_e e^4 Z^2 L}{m} \varphi\left(\frac{v_i}{\Delta_e}\right), \qquad \text{(II.3.21)}$$

$$\varphi(x) = \sqrt{\frac{2}{\pi}} \int_0^x e^{-y^2/2} dy - \sqrt{\frac{2}{\pi}} x e^{-x^2/2}, \qquad \text{(II.3.22)}$$

in the case when $\Delta_{\parallel} = \Delta_{\perp}$. In BETACOOL program the friction force components in accordance with Budker's formula are calculated by numerical integration of (II.3.21) using Simpson method with a variable step. The relative mistake in the integral calculation is less than 10^{-4} in the total range of the ion velocity variation.

For benchmarking the code results of numerical evaluation of the integrals (II.3.17) was compared with Budker's formula. In the Fig. II.3.1 results of calculations at $T_{\parallel} = T_{\perp} = 5$ eV are presented. The friction force components were calculated in the centre of the electron bunch at geometry dimensions listed in the Table II.1.

Table II.1. Ion and electron beam parameters in simulations

Ion beam emittance, π·mm·mrad	0.025
Ion momentum spread	10^{-3}
Ion number	10^9
Beta functions in the cooling section, m	200
Electron cooler length, m	100
Electron beam emittance, π·mm·mrad	2
Electron momentum spread	10^{-4}
Number of electrons in bunch	$3\cdot10^{10}$
Transverse rms dimensions of the electron bunch, mm	1
Electron bunch length, cm	30

The number of the integration steps in (II.3.17) were 15 over angle, and 26 over each velocity components. The three dimensional integral is calculated during a few seconds at Pentium-IV 2.8 GHz. At such a step numbers the integral (II.3.17) underestimate the friction force value by about 5 %. The velocity correspondent to the position of the force maximum coincides with prediction by Budker's formula within an accuracy better than 5%. At smaller step the accuracy is better.

Calculation speed at asymptotic representation is about two orders of magnitude larger than at numerical integration and optimum choice of the algorithm for the friction force calculation can be perform to provide maximum calculation speed and required accuracy of characteristic time calculation. The asymptotic representation is valid at flattened distribution only. At expected parameters of electron bunch the temperature of transverse degree of freedom is about a few eV, the longitudinal one – 0.05 eV.

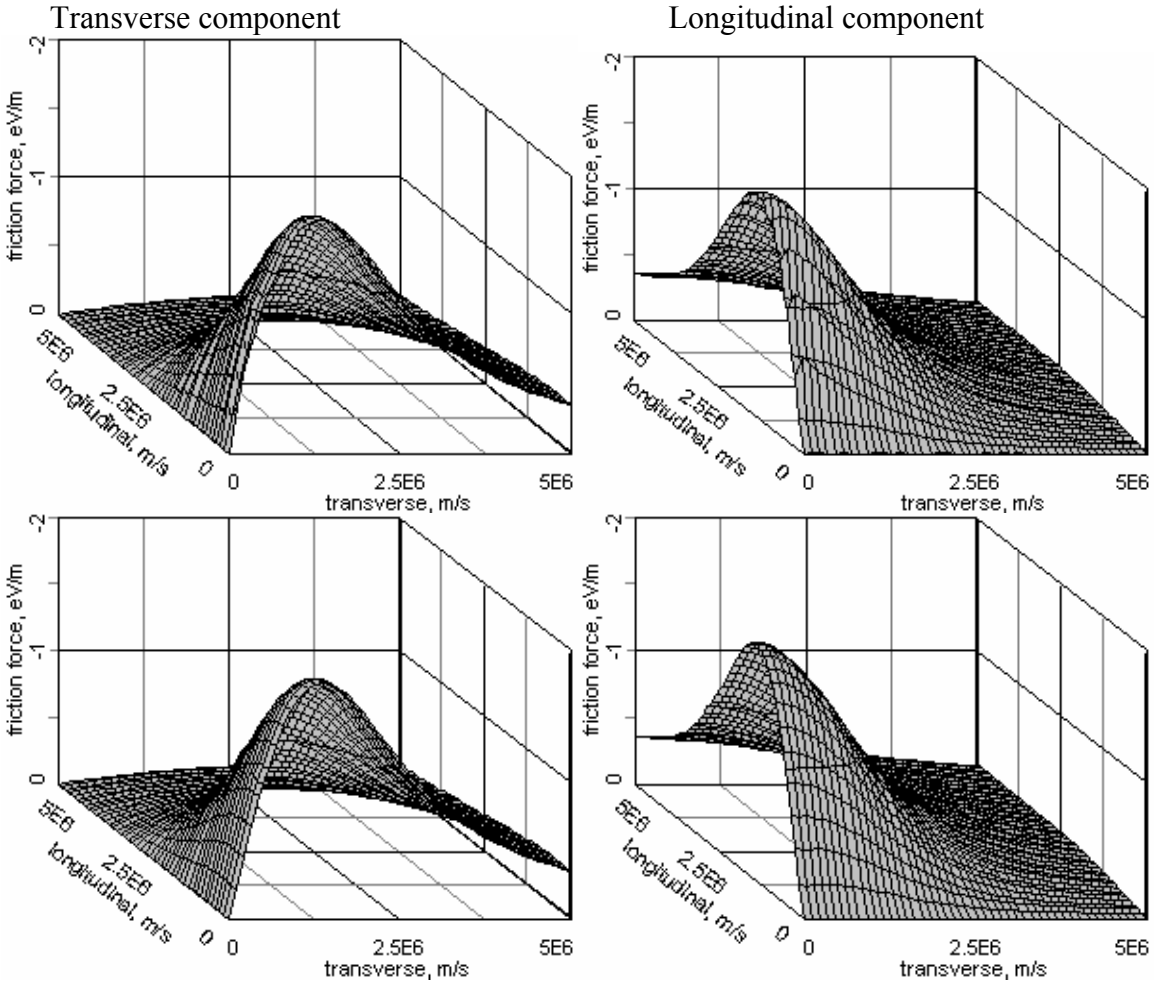


Fig. II.3.1. Components of the friction force as functions of the ion velocity components. Upper pictures are calculated by numerical integration over electron velocity distribution, lower pictures – in accordance with Budker's formula.

Shape of the numerically calculated friction force components and asymptotic representation are presented in the Fig. II.3.2. The calculations were performed at the same cooler and electron beam parameters as in the previous paragraph. The electron temperature were chosen $T_{\|}$ = 0.005 eV, $T_{\perp}$ = 5 eV.

The asymptotic representation overestimates the force value by the factor of about 2.5 in at the ion velocity in the vicinity of the electron velocity spread in transverse degree of freedom. In the region of high ion velocity results of the friction force calculation coincide practically for both methods. Correspondingly the asymptotic representation of the friction force can be used in simulations when the ion rms velocity is sufficiently larger than the electron velocity spread.

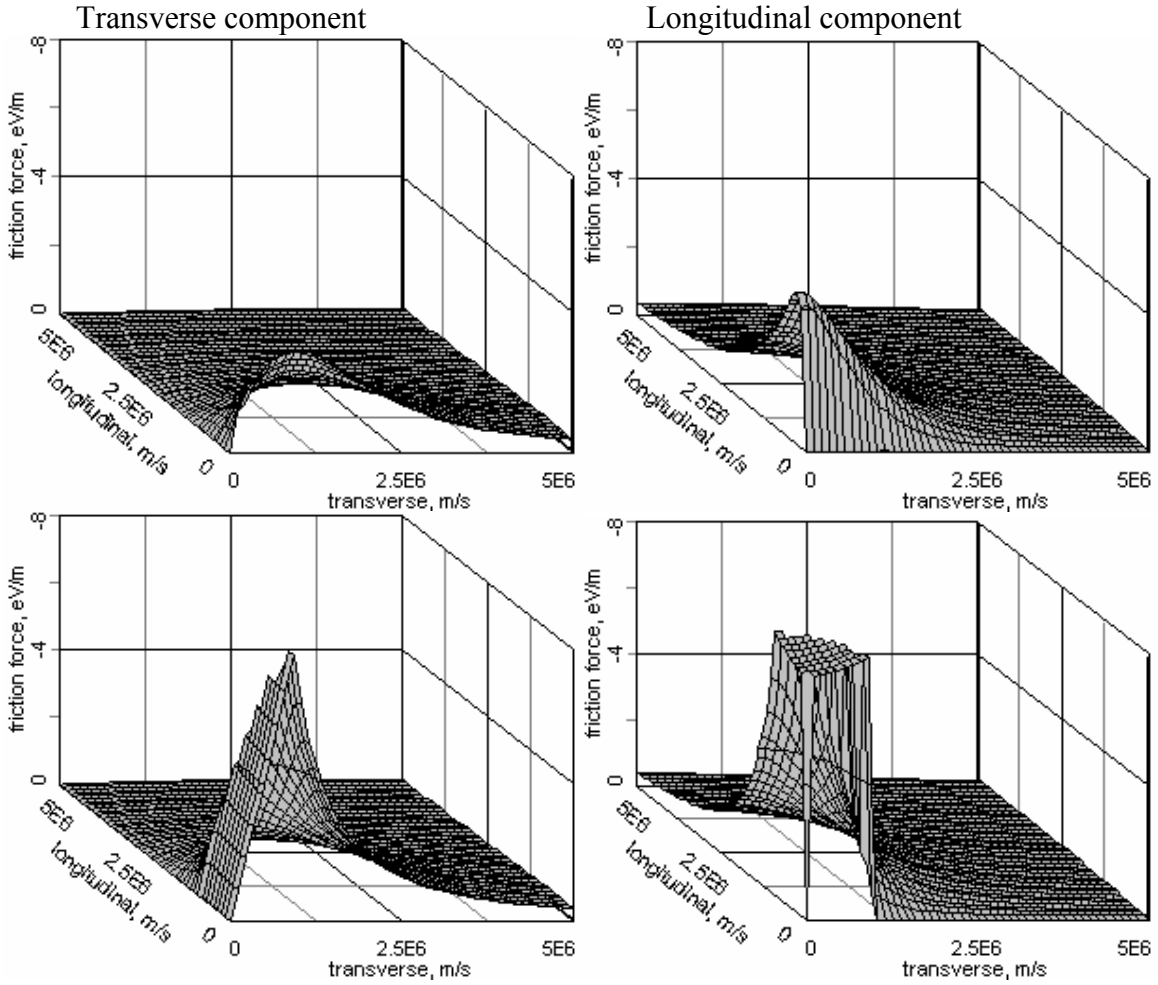


Fig. II.3.2. Components of the friction force as functions of the ion velocity components. Upper pictures are calculated by numerical integration over electron velocity distribution, lower pictures – asymptotic representation.

II.4. Simulation of ion-electron recombination in presence of undulator field

Electron cooling at RHIC using non-magnetized electron beam sufficiently simplifies the cooler design. Generation and acceleration of the electron bunch without longitudinal magnetic field permits to reach low value of emittance in the cooling section. General problem of such a scheme is high recombination rate at low electron temperature. Suppression of the ion recombination with electrons in the cooling section using undulator field was proposed for RHIC in [17]. In presence of the undulator field, trajectories of all electrons have the same coherent azimuth angle θ, determined by the undulator period λ and field value B at the axis:

$$\theta = \frac{eB\lambda}{2\pi pc}, \qquad \text{(II.4.1)}$$

where p is the electron momentum. Since the recombination cross section is approximately inversely proportional to the electron energy in the ion rest frame, the ion beam life time can be sufficiently improved.

One can expect that at impact parameters significantly larger than electron rotation radius

$$r_0 = \frac{\theta\lambda}{2\pi} = \frac{eB\lambda^2}{4\pi^2 pc} \qquad \text{(II.4.2)}$$

kinematics of a binary collision will be similar to Rutherford scattering of free electron. Thus the minimum impact parameter ρ_{min} in presence of the undulator field has to be replaced by r_0 value. At larger impact parameters the friction force can be calculated using formula (II.4.1) without tacking into account coherent electron velocity.

In difference with the friction force the recombination coefficient, determined via recombination cross section σ as follows

$$\alpha_r = \int (V_i - v_e)\sigma(V_i - v_e) f(v_e) d^3 v_e, \qquad \text{(II.4.3)}$$

has to be calculated taking into account the coherent transverse electron velocity. For the recombination rate calculation one should to use a distribution function

$$f(v)d^3v = \left(\frac{m}{2\pi}\right)^{3/2} \frac{1}{T_\perp \sqrt{T_{II}}} e^{-m(v_\perp + v_{und})^2/2T_\perp - mv_{II}^2/2T_{II}} 2\pi v_\perp dv_\perp dv_{II}, \qquad \text{(II.4.4)}$$

where v_{und} is the electron azimuth velocity due to rotation in the undulator field:

$$v_{und} = c\beta\gamma\theta. \qquad \text{(II.4.5)}$$

The ion beam life time due to recombination in the cooling section is calculated via recombination coefficient α_r by the following formula:

$$\frac{1}{N}\frac{dN}{dt} = -\frac{\alpha_r n_e}{\gamma^2}\frac{l_{cool}}{C}, \qquad \text{(II.4.6)}$$

here C is the ring circumference. Under assumption that ion velocity in PRF is substantially less than electron one the α_r is calculated in PRF by averaging of the recombination cross section over electron distribution function:

$$\alpha_r = \langle v\sigma(v)\rangle \qquad \text{(II.4.7)}$$

The recombination cross section can be calculated with good accuracy using the following formula [18]:

$$\sigma = A\left(\frac{h\nu_0}{E}\right)\left(\ln\sqrt{\frac{h\nu_0}{E}} + 0.1402 + 0.525\left(\frac{E}{h\nu_0}\right)^{1/3}\right), \qquad \text{(II.4.8)}$$

where $A = 2^4 3^{-3/2} he^2/\left(m_e^3 c^2\right) = 2.11\times10^{-22}$ cm^2, $h\nu_0 = 13.6\cdot Z^2$ eV is the ion ground state binding energy. The electron kinetic energy $E = \frac{m_e v_e^2}{2}$. In presence of the undulator field it has to be calculated as:

$$E = \frac{m}{2}\left(\left(v_\perp + v_{und}\right)^2 + v_\parallel^2\right), \qquad \text{(II.4.9)}$$

The formula (II.3.7) at flattened distribution (II.4.4) can be rewritten in the form adopted for numerical integration:

$$\alpha_r = \frac{1}{Int}\int_0^{3\Delta_\perp}\int_{-3\Delta_\parallel}^{3\Delta_\parallel}\sigma(E)\sqrt{\left(v_\perp + v_{und}\right)^2 + v_\parallel^2}\exp\left(-\frac{\left(v_\perp + v_{und}\right)^2}{2\Delta_\perp^2} - \frac{v_\parallel^2}{2\Delta_\parallel^2}\right)v_\perp dv_\parallel dv_\perp . \qquad \text{(II.4.10)}$$

The normalization factor is calculated as:

$$Int = \int_0^{3\Delta_\perp}\int_{-3\Delta_\parallel}^{3\Delta_\parallel}\exp\left(-\frac{\left(v_\perp + v_{und}\right)^2}{2\Delta_\perp^2} - \frac{v_\parallel^2}{2\Delta_\parallel^2}\right)v_\perp dv_\parallel dv_\perp . \qquad \text{(II.4.11)}$$

To avoid overflow in calculation of the exponents, at $v_{und} > 6\cdot\Delta_\perp$ the recombination coefficient is calculated directly from the coherent velocity:

$$\alpha_r = v_{und}\sigma\left(v_{und}\right), \qquad \text{(II.4.12)}$$

that corresponds to electron distribution in the form of delta function.

In absents of the undulator field the recombination coefficient α_r calculated at flattened electron velocity distribution is given by the formula [19]:

$$\alpha_r = B \cdot Z^2 \sqrt{\frac{1}{T_\perp}} \left[\ln\left(\frac{11.32Z}{\sqrt{T_\perp}} \right) + 0.14 \left(\frac{T_\perp}{Z^2} \right)^{1/3} \right], \; B = 3.02 \cdot 10^{-13} \frac{cm^3}{s} \qquad \text{(II.4.13)}$$

here T is in eV. This formula presumes electron distribution in londitudinal degree of freedom as delta function. Correspondingly, at small undulator magnetic field and electron temperature of longitudinal degree of freedom the numerical solution (II.4.7) has to coincide with (II.4.3). At large value of the coherent velocity the integral (II.4.7) has to coincide with solution (II.4.12).

At electron temperature of 5 eV the formula (II.4.13) gives for recombination coefficient the value of 5.061×10^{-9} cm^3/s. The integral (II.4.11), calculated numerically at undulator field of 10^{-3} G and electron temperature of $T_\parallel$ = 0.005 eV, $T_\perp$ = 5 eV, is equal to 5.173×10^{-9} cm^3/s, which coincides with (II.4.13) within an accuracy of 2%. (Undulator period is 12.6 cm.)

Comparison between values of the recombination coefficient calculated numerically and in accordance with asymptotic (II.4.12) is presented on the Fig. II.4.1. At undulator field value larger than 10 G numerical and asymptotic values of the recombination coefficient are coincide practically. At magnetic field of 50 G ($v_{und} \sim 7 \cdot \Delta_\perp$) relative difference is less than 10^{-4}.

It should be noted, that all the formulae are derived under assumption that ion velocity in PRF is substantially less than electron one, however at the parameters listed in the Table II.3.1 the rms ion velocity is equal to $5 \cdot 10^5$ m/s: $V_i^2 = c^2 \beta^2 \left(\gamma^2 \frac{\varepsilon}{\beta_{x,y}} + \sigma_p^2 \right)$. This value is comparable with electron velocity spread at temperature of a few eV and with coherent electron velocity in undulator field. At parameters from the Table 1 the coherent velocity is equal to $v_{und} = 3.5 \cdot 10^5 \cdot$B[G] m/s. Therefore, the algorithm described here will overestimate recombination rate, and more accurate calculations have to include dependence of the recombination coefficient on the ion velocity.

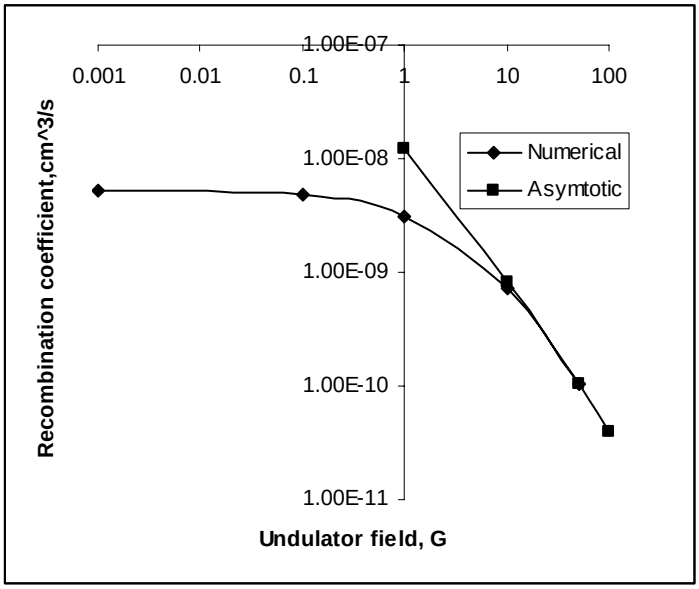


Fig. II.4.1. Recombination coefficient as function of the undulator magnetic field.

II.5. Format of the table for import of the friction force from external program

As soon as no any existing model for friction force calculation from the electron cooling effect satisfies and truly describes influence of cooling electrons on ions, the model for the electron cooling with direct calculation of binary collisions between real particles was proposed. In this case the result of simulation is a table of friction force values for different ion transverse and longitudinal velocities. The initial parameters for the modeling must be number of particles, range of studied velocities and step over velocities. Calculation of such table is planned with VORPAL code developed for electron cooling studies by Tech-X company [20]. This code is intended to be calculated in large computer cluster.

In order to optimize the simulation process it was proposed to use BETACOOL code for the definition of the table of friction force values and the fastest interpolation method.

Scheme of investigation was the following: with a known analytical model for friction force the table was generated and it was assumed as a tabulated one obtained from the direct binary collision calculation. Then this table saved to file was used in BETACOOL as one of the model of friction force calculation and was compared in rms dynamics simulation to the original analytical one. Result obviously depends on the velocity range and number of steps over velocity. So the goal of optimization was obtaining the similar results with tabulated and analytical models varying with range (or number of ranges) and splitting number in every range, and kind of interpolation over the table, simultaneously to minimizing calculation recourses.

II.5.1. Expected shape of the friction force

General problem of the cooling process simulation using analytical friction force formulae can be simply illustrated on example of a semi-empirical formula by Parkhomchuk. In accordance with this formula the friction force in magnetized electron beam has a symmetry form for both longitudinal and transverse components and can be presented by the following vector equation:

$$\vec{F} = -\vec{v}_i \frac{4Z^2 e^4 n_e L_P}{m} \frac{1}{\left(v_i^2 + \Delta_{e,eff}^2\right)^{3/2}}, \qquad \text{(II.5.1)}$$

where n_e is the electron density, Δ_{eff} is the effective electron velocity spread with taking into account variations of the magnetic field line position in the transverse direction. The ion velocity v_i has two components – along and across the magnetic field line:

$$v_i = \sqrt{v_\perp^2 + v_\parallel^2} \qquad \text{(II.5.2)}$$

The Coulomb logarithm L_p is given by the expression:

$$L_P = \ln\left(\frac{\rho_{\max} + \rho_{\min} + \langle\rho_\perp\rangle}{\rho_{\min} + \langle\rho_\perp\rangle}\right). \qquad \text{(II.5.3)}$$

Where the minimum impact parameter is calculated in accordance with

$$\rho_{\min} = \frac{Ze^2}{m_e} \frac{1}{v_i^2 + \Delta_{e,eff}^2}, \tag{II.5.4}$$

$$\langle \rho_\perp \rangle = \frac{\Delta_\perp m_e c}{eB} \tag{II.5.5}$$

is the Larmor radius of electron rotation around the magnetic field line, where $\Delta_\perp$ is the rms transverse velocity of the electrons.

The maximum impact parameter is calculated as a minimum from three values:

$$\rho_{\max} = \min\left\{ \max\left(\rho_{sh}, \sqrt[3]{\frac{3Z}{n_e}} \right), v_i \tau, a \right\}. \tag{II.5.6}$$

The dynamic shielding radius is calculated using same formula:

$$\rho_{sh} = \frac{\sqrt{v_i^2 + \Delta_e^2}}{\omega_p}, \tag{II.5.7}$$

Second term describes the distance, which the ion passes inside the electron beam. Here τ is the ion time of flight the cooling section in the PRF:

$$\tau = \frac{l_{cool}}{\beta\gamma c}. \tag{II.5.8}$$

and the last term is the electron beam radius.

The formula (II.5.1) well describes the friction force components when the Coulomb logarithm has a value of about 10, which corresponds to “good” magnetization. The maximum impact parameter is determined by the ion velocity, and minimum one – by the transverse emittance of the electron beam and the magnetic field value. When the ratio between maximum and minimum impact parameters is about unity, the formula (II.5.1) as the other formulae deduced in the logarithmic approximation can not predict the friction force shape correctly. In this case one needs to provide numerical evaluation of the friction force value. To minimize the time of the numerical simulation one needs to optimize the mesh parameters in the velocity space.

For instance, the transverse component of the friction force:

$$F_\perp = -v_\perp \frac{4Z^2 e^4 n_e L_P}{m} \frac{1}{\left(v_\perp^2 + v_\parallel^2 + \Delta_{e,eff}^2 \right)^{3/2}} \tag{II.5.9}$$

does not depend on longitudinal component of the ion velocity if

$$v_\parallel << v_\perp \tag{II.5.10}$$

and depends of $v_{\|}$ as

$$F_{\perp} \sim \frac{1}{v_{\|}^3} \qquad \text{(II.5.11)}$$

in the opposite case. Correspondingly, the mesh step over the longitudinal ion velocity can be larger in the region (II.5.10) and smaller in the other space. In the region (II.5.10) the friction force dependence on the transverse velocity component has three characteristic regions:

- linear in the range of small velocity,
- the maximum when the ion velocity is closed to effective electron velocity spread,
- $F_{\perp} \sim \frac{1}{v_{\perp}^2}$ in the region of high ion velocity.

Optimum step of the mesh along the transverse component of the ion velocity is different in all these intervals. The smallest step value has to be chosen in the range where one can expect inclination of real dependence from theoretically predicted.

In principle one can expect peculiarity of the transverse friction force component in the region of small ion velocity predicted by Derbenev-Skrinsky formula:

$$F_{\perp} = -v_{\perp} \frac{2\pi Z^2 e^4 n_e L_M}{m v^3} \frac{v_{\perp}^2 - 2v_{\|}^2}{v^2}, \qquad \text{(II.5.12)}$$

(here $L_M = \ln \frac{\rho_{\max}}{\langle \rho_{\perp} \rangle}$) the force can change a sign when $v_{\perp} < \sqrt{2} v_{\|}$. Correspondingly, step of the mesh has to be reduced in this region.

The procedure for the mesh generation described in this report was developed as a first step of the mesh structure optimization and requires further development, which will be provided in the second stage of the work.

II.5.2. Interpolation methods

As the simplest interpolation method was chosen linear interpolation when the nearest node for the given value is chosen. On the one hand this way is fast enough but it is very rough and gives acceptable results when the size of table is huge and very difficult to be processed by program.

The second method is 2D interpolation – so called bilinear [21]:
In multidimensional interpolation, we seek an estimate of y(x1, x2, . . . , xn) from an n-dimensional grid of tabulated values y and n one-dimensional vectors giving the tabulated values of each of the independent variables x 1, x2, . . . , xn. We will not here consider the problem of interpolating on a mesh that is not Cartesian, i.e., has tabulated function values at “random” points in n-dimensional space rather than at the vertices of a rectangular array. For clarity, we will consider explicitly only the case of two dimensions, the cases of three or more dimensions being analogous in every way. In two dimensions, we imagine that we are given a matrix of functional values ya[1..m][1..n].We are

also given an array x1a[1..m], and an array x2a[1..n]. The relation of these input quantities to an underlying function y(x 1, x2) is

ya[j][k] = y(x1a[j], x2a[k])

We want to estimate, by interpolation, the function y at some untabulated point (x1, x2).
An important concept is that of the grid square in which the point (x1, x2) falls, that is, the four tabulated points that surround the desired interior point. For convenience, we will number these points from 1 to 4, counterclockwise starting from the lower left. More precisely, if

x1a[j] ≤ x1 ≤ x1a[j+1]
x2a[k] ≤ x2 ≤ x2a[k+1]

defines j and k, then

y1 ≡ ya[j][k]
y2 ≡ ya[j+1][k]
y3 ≡ ya[j+1][k+1]
y4 ≡ ya[j][k+1]

The simplest interpolation in two dimensions is bilinear interpolation on the grid square. Its formulas are:

t ≡ (x1 - x1a[j])/(x1a[j+1]- x1a[j])
u ≡ (x2 - x2a[k])/(x2a[k+1]- x2a[k])

(so that t and u each lie between 0 and 1), and

y(x1, x2) = (1 - t)(1 - u)y1 + t(1 - u)y2 + tuy3 + (1 - t)uy4

Bilinear interpolation is frequently “close enough for government work.” As the interpolating point wanders from grid square to grid square, the interpolated function value changes continuously. However, the gradient of the interpolated function changes discontinuously at the boundaries of each grid square.

II.5.3. Description of the numerical procedures and benchmarking results

For testing and comparison of all the models of calculations a special form in the BETACOOL code was created where user have a possibility to choose a model for the friction force calculation, type of interpolation, and variants of ion velocity range definition (Fig. II.5.1).

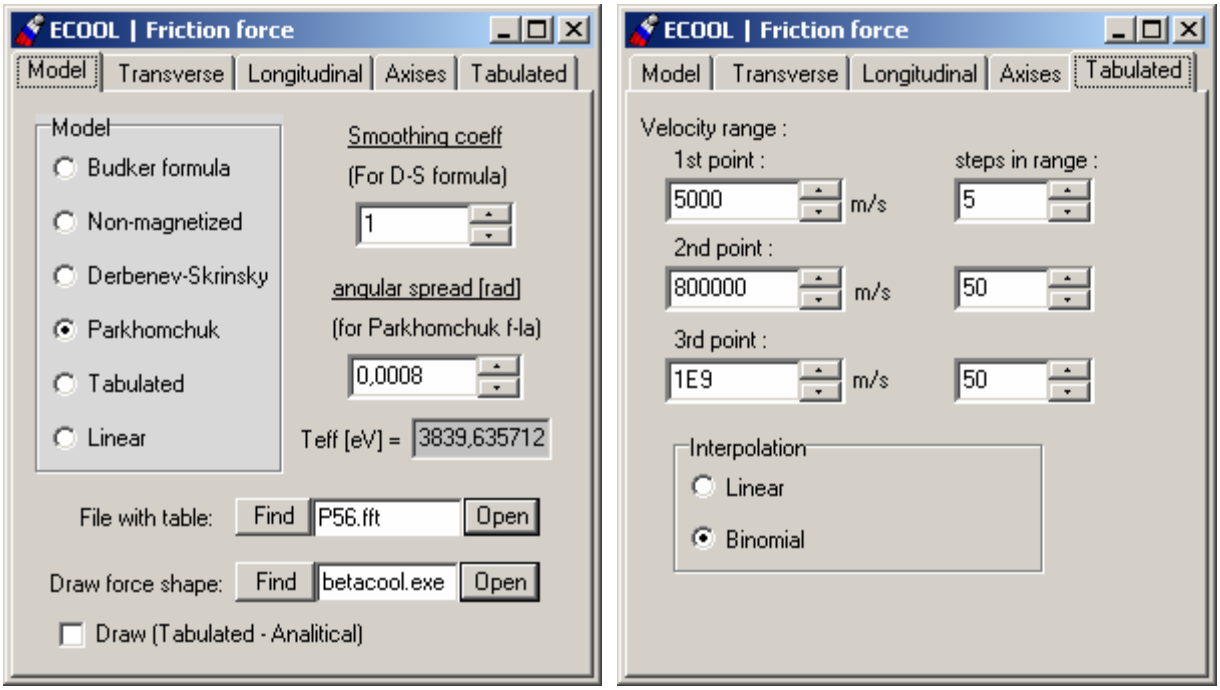


Fig. II.5.1. Friction force Form.

File with tabulated force values is saved in specialized format. User can define borders for several velocity ranges: 3 point mean 3 ranges – from 0 [m/s] to the 1st point, from the 1st point to the 2nd value of velocity, and from the 2nd to the 3rd one (Fig. II.5.2). Inside every range the number of steps over velocity can be defined. As soon as force dependence is symmetrical of zero value we can take into account only positive range of velocities.

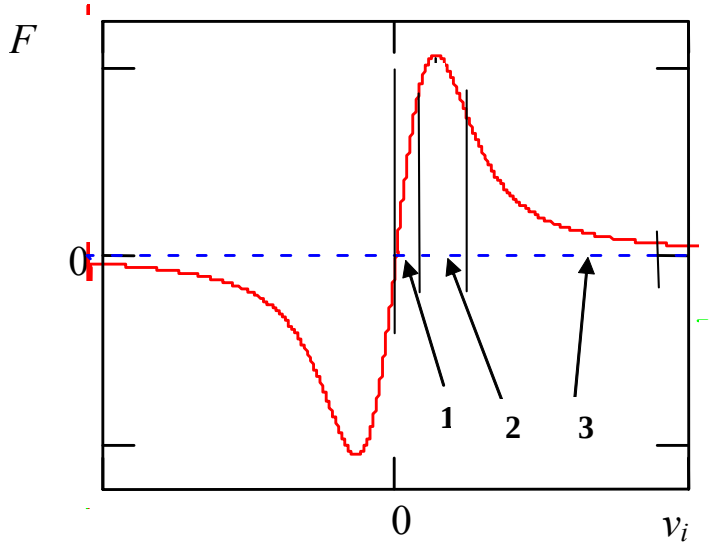


Fig. II.5.2. Analytical friction force dependence and defined velocity ranges.

At first such a splitting was intended to make a detailed analysis and splitting in the range 2, where there is a peak, which can be incorrectly interpreted with interpolation. This peak theoretically corresponds to the so-called effective temperature o the ions and can be easily interpreted in velocity units [m/sec]. For the RHIC parameters this va ue is about $7 \cdot 10^5$ m/sec. The dependence for the range 1 (as V) and range 3 (as $1/V^2$) were well kno vn. In this way we tried to make about 5-10

splits in the 1st range (from 0 – to $5 \cdot 10^5$ m/s) and ($9 \cdot 10^5$ – to the maximal possible ion velocity ~ $3 \cdot 10^8$ m/s). But for the second range maximal possible splitting was proposed (from 50 to 500 – depends on the PC power).

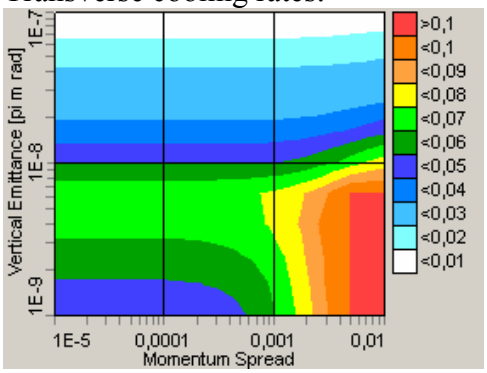
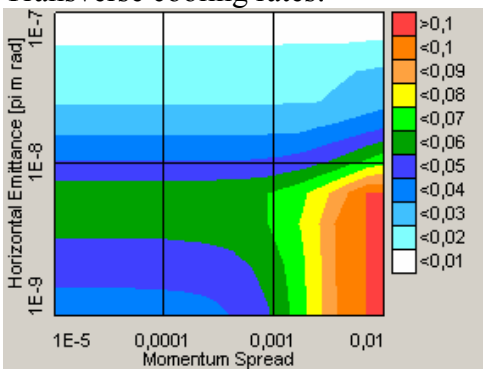
After some investigations the correct tendency for the approach was found: maximal splitting must be not in the "uncomfortable" region of the peak, but in the range where the most ion velocity are found for the machine parameters, namely transverse ion velocities. For the investigated RHIC parameters this region for transverse ion velocity is about $1 \cdot 10^6$ m/s. Here the contribution of the velocities is largest to the total friction force value. As soon as transverse velocities are larger than longitudinal (~ by two orders of magnitude) they dominate in the final friction force value (both velocities contribute to the friction force formula with the same weights).

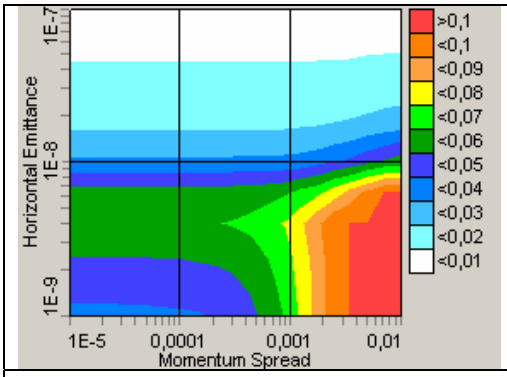
Criteria for good coincidence of friction force values obtained with interpolation from table and analytically calculated one can use comparison of 3D diagrams for beam r.m.s. growth rate dependence in the working range of transverse and longitudinal emittances for RHIC parameters.

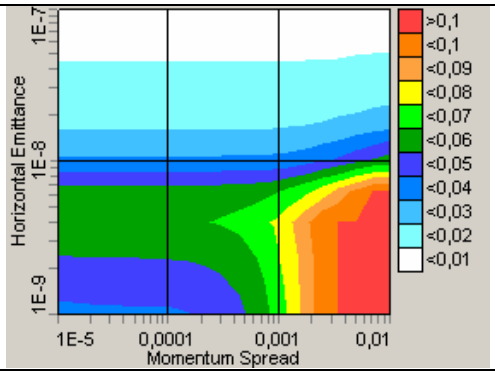
During optimization table splitting and velocity range were minimized. As a result reasonable conditions for the direct simulation of friction force table were achieved:

1st range of velocities: from 0 to $5 \cdot 10^3$ m/sec with 5 splits
2nd range of velocities: from $5 \cdot 10^3$ to $8 \cdot 10^5$ m/sec with 20 splits
3rd range of velocities: from $8 \cdot 10^5$ to $5 \cdot 10^6$ m/sec with 50 splits.

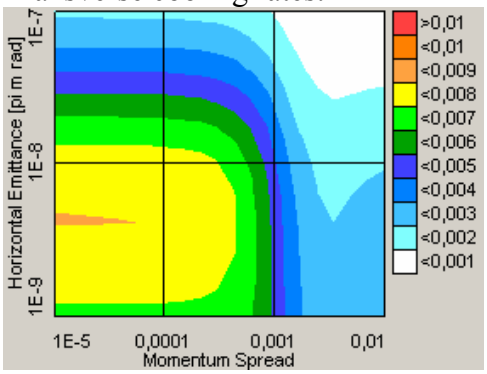
At least for the tested parameters, this suggests that the friction force can be accurately represented by approximately 70 x 70 numerical arrays. In other words, about 5000 velocity point calculation will be needed with the VOPRAL codes for each set of parameters. To decrease this number one needs to develop more appropriate algorithm of the mesh formation. First of all the number of divisions and step have to be different in longitudinal and transverse degrees of freedom.Mentioned above 3D plots for comparison are presented in the Fig. II.5.3.

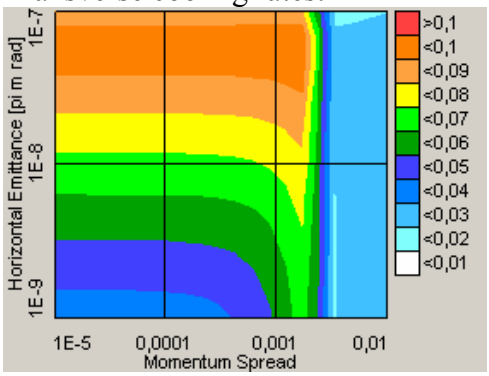
Analytical formula (Parchomchuk)	Bilinear interpolation from the table (par31.fft)
Transverse cooling rates: 	Transverse cooling rates: 
Longitudinal cooling rates:	Longitudinal cooling rates:



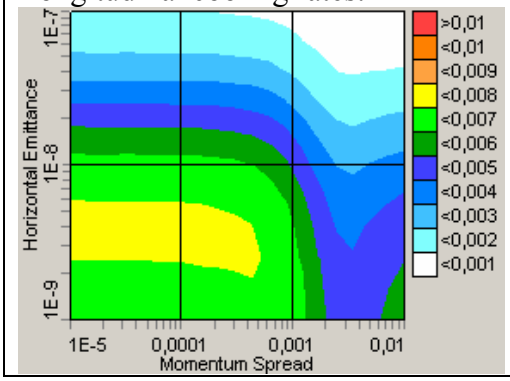


Gaussian bunch	
Analytical formula (Parchomchuk)	Bilinear interpolation from the table
Transverse cooling rates:	Transverse cooling rates:





Longitudinal cooling rates:	Longitudinal cooling rates:



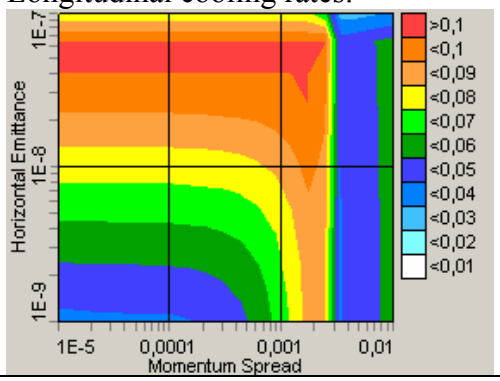


Fig. II.5.3. Comparison of cooling rates calculated using analytical formula and interpolation of tabulated values calculated with the same formula. (Uniform cylinder model).

At chosen parameters of the mesh the coincidence is good for the electron beam model as an uniform cylinder. Difference in the cooling rates at Gaussian bunch reflects the fact, that the friction force is the function of transverse co-ordinates in this case. The friction force table was calculated at electron density in the centre of the electron bunch, but the density averaged over the ion betatron and synchrotron oscillations is sufficiently less. To avoid overestimation of the cooling force one needs to calculate the friction force table at some effective electron density, which can be estimated by comparison of the results calculated at different densities with the analytical formulae.

II.6. Integration of BETACOOL under UAL framework

The Unified Accelerator Libraries (UAL) provides a modularized environment for applying diverse accelerator simulation codes. At this time the object-oriented programs included in UAL are: PAC (Platform for Accelerator Codes), ZLIB (Numerical Library for Differential Algebra), TEAPOT (Thin Element Program for Optics and Tracking), ACCSIM (Accelerator Simulation Code), ORBIT (Objective Ring Beam Injection and Tracking Code). Modules that are partially supported and are under active development are ICE (Incoherent and Coherent effects), AIM (Accelerator Instrumentation Module), SPINK (tracks polarized particles in circular accelerator), and TIBETAN (longitudinal phase space tracking). The Application Programming Interface (API), written in Perl, provides a universal shell for integrating and managing all project extensions.

BETACOOL as a program calculating evolution of r.m.s beam parameters in presence of heating and cooling effects was successfully implemented into UAL. As soon as BETACOOL has a wide kit of different models for heating effects (intrabeam scattering, interaction of two beams, scattering on residual gas, etc) and cooling effects (electron cooling, stochastic cooling) all these effects have rather complex structure. The task was to organize a simple way to call any model in the frame of UAL and to integrate and transfer all internal parameters which BETACOOL operates with including initial parameters into UAL global scope.

BETACOOL uses a lot of parameters for any effect model, they are initial parameters of the beam(s), ring structure, lattices, parameters for the effect calculation itself, etc., and a kit of internal parameters which are temporary calculated. It is necessary to integrate them into UAL or initialize some of them from UAL variables.

The obvious advantage of BETACOOL implementation is object oriented method of the program code. In order to use some procedures and functions from BETACOOL effects calculation was proposed library-like scheme. Program code is compiled and linked to the library and special program-adaptor must be created using UAL interface which will contain so called “bridge” functions which let to use BETACOOL possibilities for effects simulation in the UAL framework.

II.6.1. Library realization and adaptor scheme

BETACOOL library is created as shared library ***libUalBetacool.so*** under Linux OS with a help of gcc compiler. This library is created in a standard way, we used a make utility.

To use all the BETACOOL functions it is necessary to make a linkage to BETACOOL header files where all the objects and functions are declared (here ***#include*** is to the folder where all BETACOOL headers are):

```
// Betacool classes
#include "..\include\xdynamic.h"
```

we made ***#include*** only to one BETACOOL header file because all others are included from it.

Two adaptors are created for now. One of them is declared and described in files ***Ring.hh, Ring.cc***.

This adaptor connects all the BETACOOL parameters with UAL internal variables. It creates an object of specially declared class Ring. When this object is created it uses a standard BETACOOL

input file (of *.bld type) as a parameters of initialization, where all the simulation parameters are described. User must edit this file and set parameters in accordance with simulation task.

```
// Constructor
BETACOOL::Ring::Ring(std::string& fileName)
{
  char fn[120];
  strcpy(fn, fileName.c_str());
  xData::Get(fn); // read file of parameters and initialization of Betacool objects
  xData::Set(fn); // more initialization of Betacool objects and save file of parameters
}
```

This constructor uses standard BETACOOL functions ***xData::Set()*** and ***xData::Get()*** for setting internal parameters. These functions load input file, organize a special data array inside BETACOOL framework and set its elements in accordance with a parameter file, where the coincidence between BETACOOL data array and parameters name is specialized.

Calling of any BETACOOL parameters is possible via global BETACOOL objects, which are declared in ***xdynamic.h*** file. Here is a list of these objects:

iDraw
iBeam
iRing
iTime
iTaskRates
iDynamics
iEcool
iForce
iEbeam
iIBS
iLosses
iHeat
iTarget
iRestGas
iColl
iStochastic

A special function is foreseen in the described adaptor file, inside class ***Ring***. It lets to fill the ring lattices in the BETACOOL array of variables with lattices from UAL which are calculated by one of the program in the UAL framework. This function creates an array of optic elements from the optic structure calculated in UAL.

```
/* Calculates twiss parameters and sets Ring containers */
   void build(const char* latticeName);
```

When tracking of lattices is completed by UAL tool and they are stored in dedicated array (here in example ***vtwiss***), they can be transferred to BETACOOL ring elements setting the kit of lattices for every element in accordance with the lattices calculated by another library of UAL (for example TEAPOT):

```
// Start the initialization of Betacool_Ring

  iRing.Number(vtwiss.size()-1);
  iRing.Arc = 0;
  for(int j = 0; j < vtwiss.size()-1; j++){
   iRing[j].EL_LATTICE = true;
   iRing[j].Length = m_lattice[j+1].getLength();
   iRing.Arc += iRing[j].Length;
   iRing[j].Lattice.dist  = iRing.Arc;
   iRing[j].Lattice.betax = vtwiss[j].beta(0);
   iRing[j].Lattice.betay = vtwiss[j].beta(1);
   iRing[j].Lattice.alfax = vtwiss[j].alpha(0);
   iRing[j].Lattice.alfay = vtwiss[j].alpha(1);
   iRing[j].Lattice.Dx = vtwiss[j].d(0);
   iRing[j].Lattice.Dy = vtwiss[j].d(1);
   iRing[j].Lattice.Dpx = vtwiss[j].dp(0);
   iRing[j].Lattice.Dpy = vtwiss[j].dp(1);
```

The second adaptor file (***CompositeTracker.cc, CompositeTracker.h***) contains description of few functions, which let to change beam parameters due to active effects influence using BETACOOL simulation models.

Here special class ***CompositeTracker*** is organized. The main function described here makes tracking of the particle beam through effects:

```
void BETACOOL::CompositeTracker::propagate(PAC::Bunch& bunch)
{
  readBunch(bunch);
  iBeam.Emit = iBeam.Get_Emit(iRing.LATTICE);
  iDynamics.Drawing(iRing.LATTICE);
  xLattice Lattice2 = iRing.LATTICE;
  for (int i = 0; i < xEffect::ACount; i++)
  {
   if (xEffect::AItems[i]->Use)
   {
     transRotate(Lattice2, xEffect::AItems[i]->Lattice);
     addKick(i);
     Lattice2 = xEffect::AItems[i]->Lattice;
   }
  }
  transRotate(Lattice2, iRing.LATTICE);
  longRotate();
  ++iTime;
  writeBunch(bunch);
}
```

It uses the following functions, which are created in adaptor file and use internal BETACOOL procedures:

- ***readBunch()*** – creates a bunch of particles (BETACOOL object) as an array, in accordance with number of model particles, and fills coordinates of every particles from real coordinates, calculated in and taken from UAL;

- ***iBeam.Get_Emit*** – calculates beam emittance accordingly to obtained coordinates and matched to the current element lattices;
- ***addKick()*** – calculates momentum deviation from the active effect (if such) in accordance with BETACOOL algorithm of Model Beam;

- ***transRotate()*** – propagates particles through optic element – changes particle transverse coordinates with the help of the element transformation matrix. Transformation matrix is calculated in the area between lattices, which are parameters of the function;

- ***longRotate()*** – rotates bunch in longitudinal phase space;

- ***write bunch ()*** – takes back changed array of particles to the UAL framework.

Two additional functions are also declared in adaptor file.
First of them allows setting lattices for any point of "BETACOOL ring" from the UAL array of lattices:

ModelBeamTracker.setLattice(twiss);

The second function has two parameters: name of effect, and lattices in the point where this effect is located. Function checks if such an effect is considered in BETACOOL initial parameters and if found sets lattices there in accordance with UAL calculation:

ModelBeamTracker.registerEffect("XADDHEAT", "clock8");

Here in adaptor file user can add in accordance with class ***CompositeTracker*** syntax any other function where can be used BETACOOL tools.

II.6.2. User program and usage of adaptors and library

To use adaptor functions with BETACOOL library user must follow next 2 steps:

1. Compile and link his own code with include of path to the library: ***libUalBetacool.so*** and BETACOOL header files. It can be done during the assembly of the executable user file with the following instructions in Makefile:

Include library instruction: -L ../lib/ -lUalBetacool
Include instruction: -I. -I../lib/

2. In user's own program code it is necessary to make #include instructions to BETACOOL headers and to the header file of adaptor:

```
#include "CompositeTracker.hh"      // Adaptor reference
#include "Ring.hh"                  // Adaptor reference
```

```
#include "xdynamic.h"                    // Betacool reference
```

Then the usage of adaptor and library to calculate effects from any of BETACOOL models is simply calling for functions:

```
//------------------------------------------------------------------------
  std::cout << "START BETACOOL " << std::endl;
//------------------------------------------------------------------------

// read BETACOOL input file
BETACOOL::Ring& theRing = BETACOOL::Ring::getInstance("rhicfodo.bld");

// set BETACOOL Lattices from UAL
 theRing.build(ring.c_str());
 std::cout << "Lattice Number = " << iRing.Number() << ", Circumference = " << iRing.Circ()
<< std::endl;

// calculate growth rates from BETACOOL Effects
vectorU Rates = xEffect::Summary(iTime, iBeam, iRing);
 std::cout << "Rates [Ex, Ey, dP/P, N] = " << Rates[0]() << ", " << Rates[1]() << ", " <<
Rates[2]() << ", " << Rates[3]() << std::endl;

// Constructor for beam propagation object
BETACOOL::CompositeTracker ModelBeamTracker;

// set time step
ModelBeamTracker.setTimeStep(iDynamics.IntegrationStep());

// set lattice in start point
ModelBeamTracker.setLattice(twiss);

// reset Lattice of Betacool Effect from UAL twiss
ModelBeamTracker.registerEffect("XADDHEAT", "clock8");

// Overwrite UAL bunch with the Gaussian distribution from Betacool
iDynamics.Distribution(iRing.LATTICE, false); ModelBeamTracker.writeBunch(bunch);

int istep = 0;

while(true)
  {
  // set time step
  ModelBeamTracker.setTimeStep(iDynamics.IntegrationStep());

// kicks from Betacool Effects (IBS, Ecool, ...)
  ModelBeamTracker.propagate(bunch);
}
```

REFERENCES

[1] A. Lavrentev, I.Meshkov “The computation of electron cooling process in a storage ring”, preprint JINR E9-96-347, 1996.
[2] I .N.Meshkov, A.O.Sidorin, A.V.Smirnov, E.M.Syresin, G.V.Trubnikov, P.R.Zenkevich," SIMULATION OF ELECTRON COOLING PROCESS IN STORAGE RINGS USING BETACOOL PROGRAM" , proceedings of Beam Cooling and Related Topics, Bad Honnef, Germany, 2001.
[3] A.Piwinski, Proc. 9th Int. Conf. on High Energy Accelerators, p. 105, 1974.
[4] M. Martini “Intrabeam scattering in the ACOOL-AA machines”, CERN PS/84-9 AA, Geneva, May 1984.
[5] J.D. Bjorken, S.K. Mtingwa, "Intrabeam scattering", Particle Accelerators, Vol. 13, p.115, 1983.
[6] Jei Wei “Evolution of Hadron Beams under Intrabeam Scattering”, Proc. of PAC’1993, p.3651.
[7] A.Fedotov, To simplified treatment of IBS process, BNL
[8] A.Burov, FERMILAB-TM-2213 (2003).
[9] A. Fedotov, I. Ben-Zvi, Yu. Eidelman, V. Litvinenko, G. Parzen, A. Sidorin, A. Smirnov, G. Trubnikov,
IBS for ion distribution under electron cooling, Proceeding of PAC’05 (2005).
[10] G.Parzen, Inrabeam scattering growth rates for a bi-gaussian distribution, BNL, C-A/AP#169, September 2004.
[11] G.Budker, Proc. Intern. Symp. Electron and Positron Storage Rings, Saclay, 1966.
[12] I.Meshkov, Electron cooling: Status and Perspectives, Phys. Part. Nucl., 25, (6), 1994, 631
[13] V.Parkhomchuk, New insights in the theory of electron cooling, NIM A 441 (2000) 9-17.
[14] Derbenev Ya.S., Skrinsky A.N., Fyzika plasmy, 1978. v.4, p.492
[15] I.Meshkov, Physics and Technique of Electron Cooling, RIKEN-AF-AC-2, 1997
[16] M.Venturini, Study of intrabeam scattering in low-energy electron rings, PAC-2001, pp. 2961-2963.
[17] Ya. Derbenev, “Electron cooling of heavy ions in solenoid with undulator”, TJNAL, 13.02.2001, unpublished.
[18] M.Bell, J.S.Bell, Capture of cooling electrons by cool protons, Particle accelerators, 1982 Vol.12 pp.49-52.
[19] A.Wolf, et al., Recombination in electron coolers, NIM A 441(2000), 183.
[20] D. Bruhwiler et al, AIP conference proceedings 773 (Bensheim, 2004), p. 394.
[21] William H. Press et al., Numerical Recipes in C, Cambridge university press.