IBSimu Particle Diagnostic

Working Draft

Duccio Marco Gasparri

2020-11-30

1 variables

- m [kg] particle mass (provided in u)
- q [J] charge of beam particle (provided in multiples of e)
- J [A/m$\hat{2}$] beam current density
- E [J] mean energy (provided in eV)
- Tp [J] parallel temperature (provided in eV)
- Tt [J] transverse temperature (provided in eV)
- $(x_1, r_1), (x_2, r_2)$ [m] beam emission line vectors
- N number of particles
- IQ [A] (A/m?) particle current
- v [m/s] from E and m

2 ParticleDataBaseCylImp::add_2d_beam_with_energy

Function ParticleDataBaseCylImp::add_2d_beam_with_energy (file: *particle-databaseimp.cpp*, line: 968) is used to add a beam of N particles with average energy E to a cylindrical geometry.

The charge q is provided by the user and is set constant for all the particles.

The beam emission line norm s [m] is defined:

$$s = \sqrt{(x_2 - x_1)^2 + (r_2 - r_1)^2} \tag{1}$$

The current IQ [A] is set for each particle as follows:

$$IQ = \frac{2\pi s J}{N}(r_1 + \frac{(r_2 - r_1)}{N}(n + 0.5)) \tag{2}$$

where $n \in [0, 1, ..., N-1]$.

The particles are distributed evenly spaced along the emission line defined by the vectors $(x_1, r_1), (x_2, r_2)$. The particle velocities v_x, v_r [m/s] are:

$$v_x = \frac{(x_2 - x_1)}{s}\sqrt{\frac{Tt}{m}}rnd_0 + \frac{(r_2 - r_1)}{s}\sqrt{\frac{2E}{m} + (\sqrt{\frac{Tp}{m}}rnd_1)^2} \tag{3}$$

$$v_r = \frac{(r_2 - r_1)}{s}\sqrt{\frac{Tt}{m}}rnd_0 + \frac{-(x_2 - x_1)}{s}\sqrt{\frac{2E}{m} + (\sqrt{\frac{Tp}{m}}rnd_1)^2} \tag{4}$$

and

$$w = \frac{d\theta}{dt} = \frac{\sqrt{\frac{Tt}{m}}rnd_2}{r_1 + \frac{(r_2 - r_1)}{N}(n + 0.5)} \tag{5}$$

with rnd_0, rnd_1 and rnd_2 normally distributed random variables.

3 Particle Diagnostic

The relevant functions are in files gtkparticlediagdialog.cpp (the GTK dialog file) and particlediagplot.cpp (does the actual plotting). It has the following methods:

- ParticleDiagPlot::build_data(): the function extracts the data from the ParticleDatabase
- ParticleDiagPlot::build_plot(): calls build_data(). the function extracts the data from the ParticleDatabase, set the decorations and add the graph to the _frame

4 ToDo

Particle Types

template¡class PP¿ class Particle { std::vector¡PP¿ _trajectory; ¡- trajectories PP _x; ¡- current position } Typedef Particle¡ParticleP2D¿ Particle2D; Typedef Particle¡ParticlePCyl¿ ParticleCyl; Typedef Particle¡ParticleP3D¿ Particle3D;

Particle Types in Beam

ParticleDataBaseCylImp::add_*_beam -¿ ParticlePCyl-¿ParticleCyl ParticleDataBase2DImp::add_*_beam-¿ParticleP2D-¿Particle2D ParticleDataBase3DImp::add_*_beam-¿ParticleP3D-¿Particle3D

ParticleDataBasePPImp¡ PP ¿::trajectories_at_plane

Int ParticleP2D::trajectory_intersections_at_plane (NON CONST std::vector¡ ParticleP2D ¿ & intsc) Return the number of trajectory intersections with plane Intersection points are appended to vector intsc. Int TrajectoryRep1D::solve() Returns solutions found [Linear [0,1], Quadratic[0,1,2], Cubic [0,1,2,3]]